Penetration Testing

Lecture 10

Instructor: Dr. Cong Pu, Ph.D.

cong.pu@okstate.edu

Acknowledgment: Adapted partially from course materials from Dr. Wenliang Du at Syracuse University, Dr. Fengwei Zhang at Southern University of Science and Technology, and Dr. Steven M. Bellovin at Columbia University.

Testing for Insecure Direct Object References: Summary

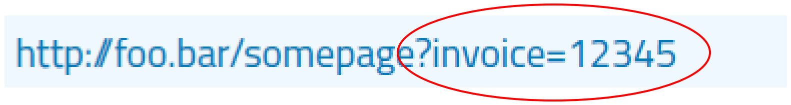
- Insecure Direct Object References:
  - application provides *direct access to objects* based on user input
  - consequences:
    - attackers can bypass authorization
      - access resources in the system directly, e.g., database records or files
      - access resources directly by *modifying* the value of a parameter used to directly point to an object
        - such resources can be database entries belonging to other users, files in the system, and more
  - why: application takes user input and uses it to retrieve an object without performing sufficient authorization checks

Testing for Insecure Direct Object References: How to Test

- To test for this vulnerability
  1. map out all locations in the application where user input is used to reference objects directly
     - for example, locations where user input is used to access a database row, a file, application pages and more
  2. modify the value of the parameter used to reference objects and assess
     - whether it is possible to retrieve objects belonging to other users or otherwise bypass authorization

Testing for Insecure Direct Object References: How to Test

- The value of a parameter is used directly to retrieve a database record
  - sample request:

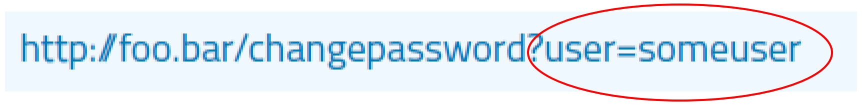


http://foo.bar/somepage?invoice=12345

  - in this case, the value of the invoice parameter is used as an index in an invoices table in the database.
  - the application takes the value of this parameter and uses it in a query to the database.
  - the application then returns the invoice information to the user.
  - since the value of invoice goes directly into the query, by modifying the value of the parameter it is possible to retrieve any invoice object, regardless of the user to whom the invoice belongs.

Testing for Insecure Direct Object References: How to Test

- The value of a parameter is used directly to perform an operation in the system
  - sample request:

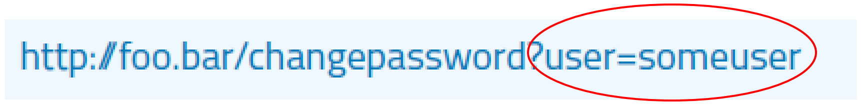


```
http://foo.bar/changepassword?user=someuser
```

  - in this case, the value of the user parameter is used to tell the application for which user it should change the password.
  - in many cases this step will be a part of a wizard, or a multi-step operation.
    - in the first step the application will get a request stating for which user's password is to be changed
    - in the next step the user will provide a new password (without asking for the current one)

Testing for Insecure Direct Object References: How to Test

- The value of a parameter is used directly to perform an operation in the system
  - sample request:

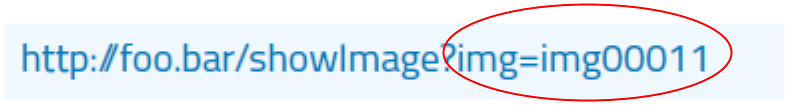


```
http://foo.bar/changepassword?user=someuser
```

  - the user parameter is used to directly reference the object of the user for whom the password change operation will be performed.
  - to test for this case the tester should attempt to provide a different test username than the one currently logged in, and check whether it is possible to modify the password of another user.

Testing for Insecure Direct Object References: How to Test

- The value of a parameter is used directly to retrieve a file system resource
  - sample request:



http://foo.bar/showImage?img=img00011

  - in this case, the value of the file parameter is used to tell the application what file the user intends to retrieve.
  - by providing the name or identifier of a different file (for example file=image00012. jpg) the attacker will be able to retrieve objects belonging to other users.

Testing for Insecure Direct Object References: How to Test

- The value of a parameter is used directly to access application functionality
  - sample request:

    `http://foo.bar/accessPage?menuitem=12`

  - in this case, the value of the menuitem parameter is used to tell the application which menu item (and therefore which application functionality) the user is attempting to access.
  - assume the user is supposed to be restricted and therefore has links available only to access to menu items 1, 2 and 3. By modifying the value of menuitem parameter it is possible to bypass authorization and access additional application functionality.
  - to test for this case the tester identifies a location where application functionality is determined by reference to a menu item, maps the values of menu items the given test user can access, and then attempts other menu items.