Mastering Session Management

Siva Ram
AppSec Consulting, Inc
siva@appsecconsulting.com
650 898 7482

OWASP
July 23, 2009

The OWASP Foundation
http://www.owasp.org

Agenda

- Overview of Sessions
- Threats to Sessions
- Securing Sessions

Overview of Sessions

The What, Why and How of Sessions

OWASP

Purpose of Sessions

- Sessions
  - Maintain context between requests
  - Compartmentalize different users

Implementing Sessions – Session Tokens

- Querystring parameters

  http://domain.com?sessionid=jsq9wqoqe

- Hidden fields

  <input type=hidden name=sessionid value=jsq9wqoqe>

- Cookies

  Set-cookie: sessionid=jsq9wqoqe;domain=domain.com;HTTPOnly

Implementing Sessions – Passing Session Data

- Pass all session data in parameters

http://domain.com?user=siva&account=231432&action=modify&role=admin

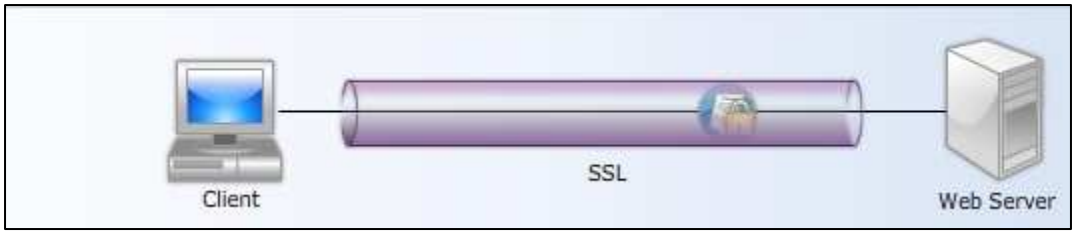
Detour - Cookies

- Session Cookies
  - Stored in memory
  - Cleared when browser is closed
  - Expires attribute left empty
- Persistent Cookies
  - Stored in the hard drive
  - Stays on client until expiry date
  - Expires attribute assigned a future date

Cookies – Other Attributes of Interest

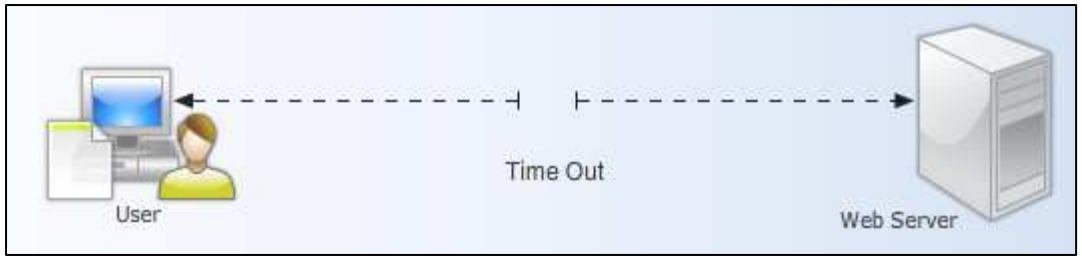
- HTTPOnly
  - Prevents client side scripts from accessing cookie

- Secure
  - Ensures cookie is transmitted over HTTPS
  - Does NOT encrypt cookie contents at rest



Back on Track – Session Timeout

- Automatic termination of session
- Configurable on server or through code



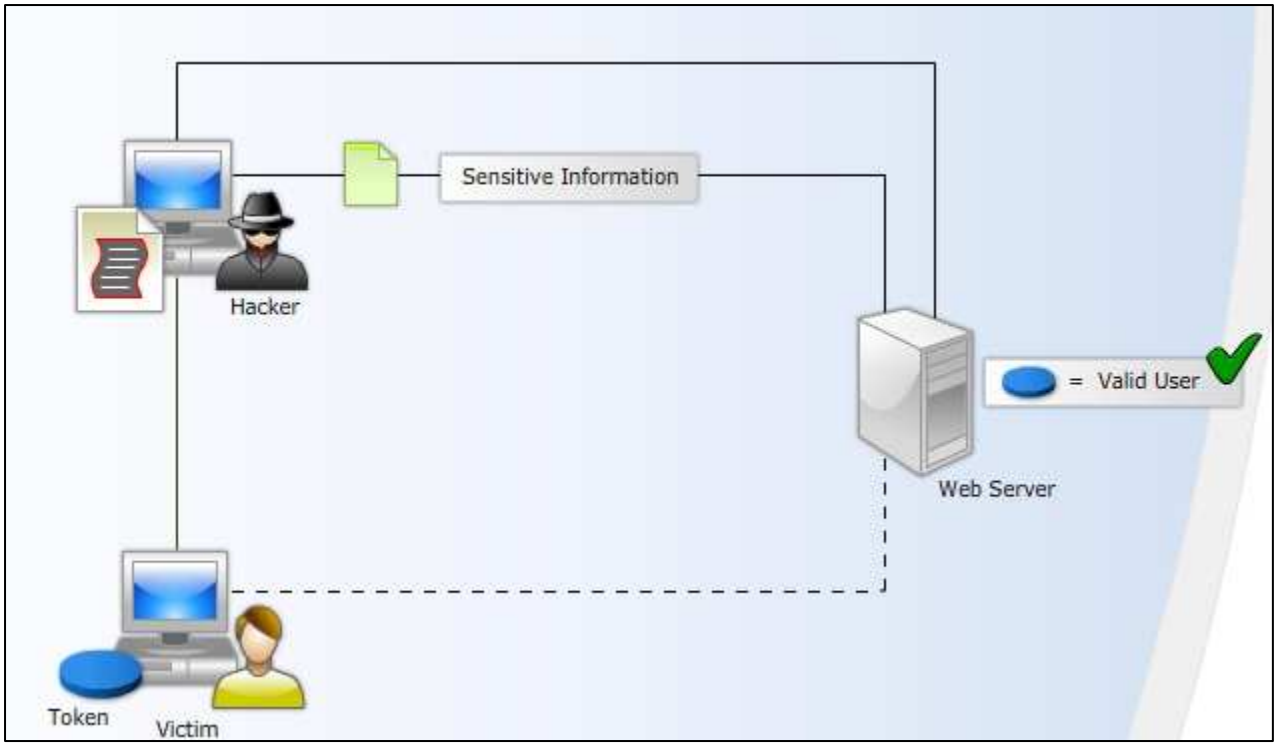
Threats to Sessions

The What and How of Session Attacks

OWASP

Session Hijacking

- Break into another user's session

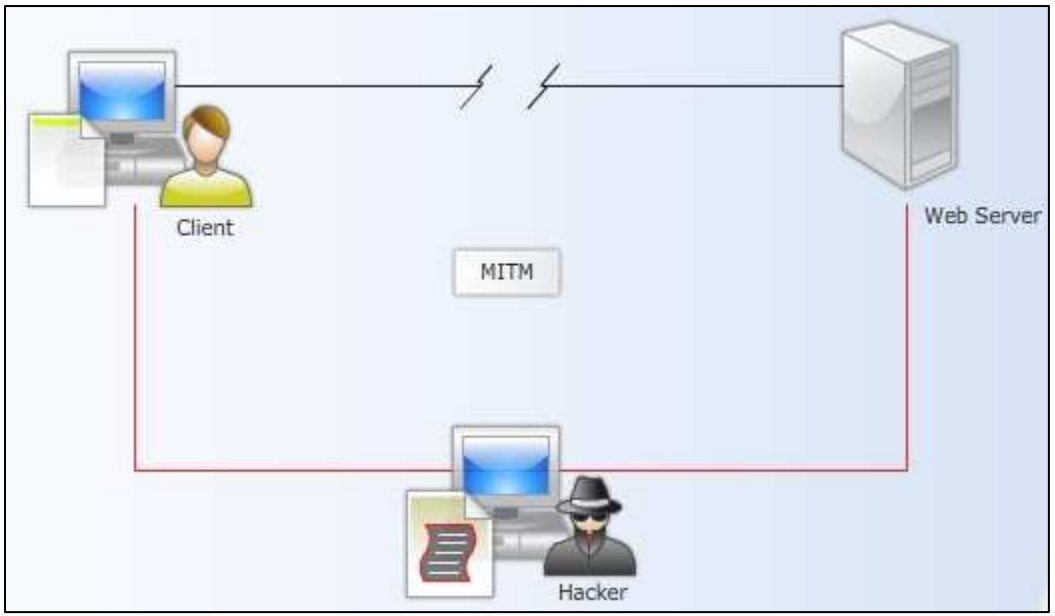


Session Hijacking – How To

- Obtaining valid session id
  - Interception
  - Prediction
  - Fixation

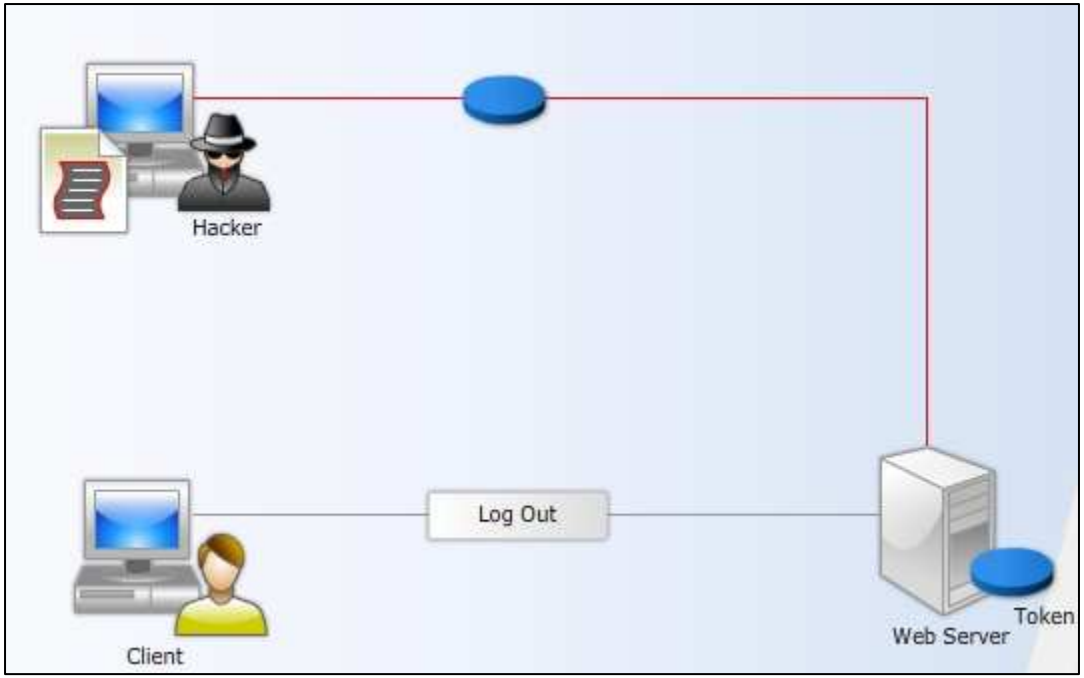
Interception - MITM

- Man in the middle



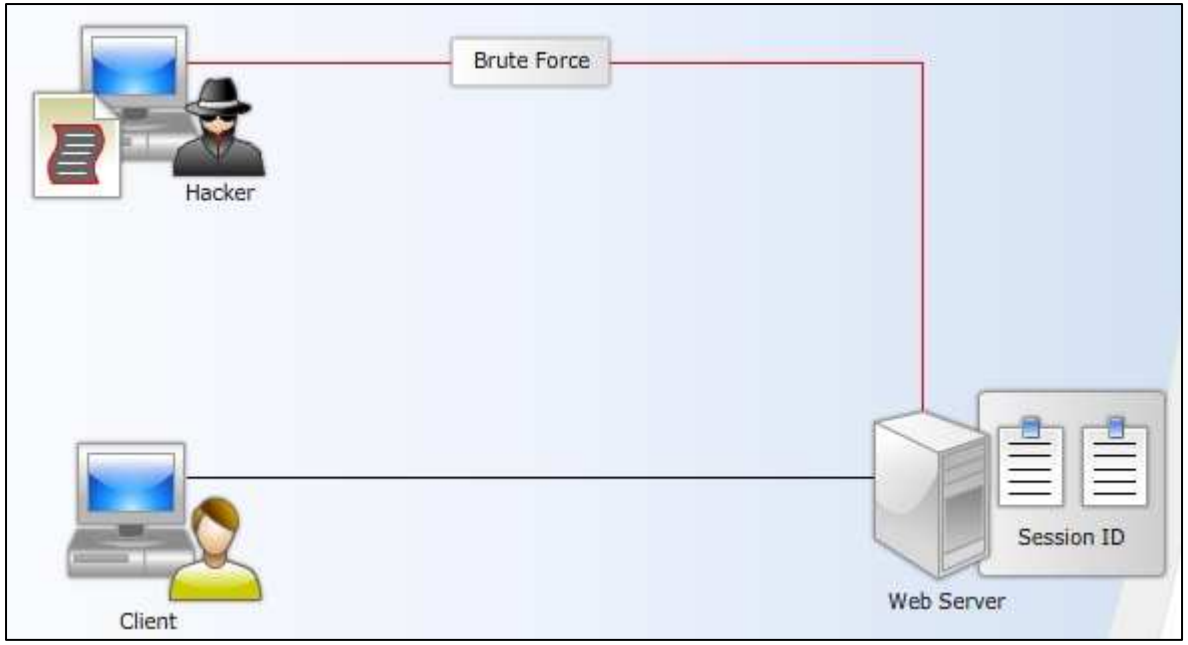
Interception - XSS

- Session ids can be obtained through XSS or other coding issues

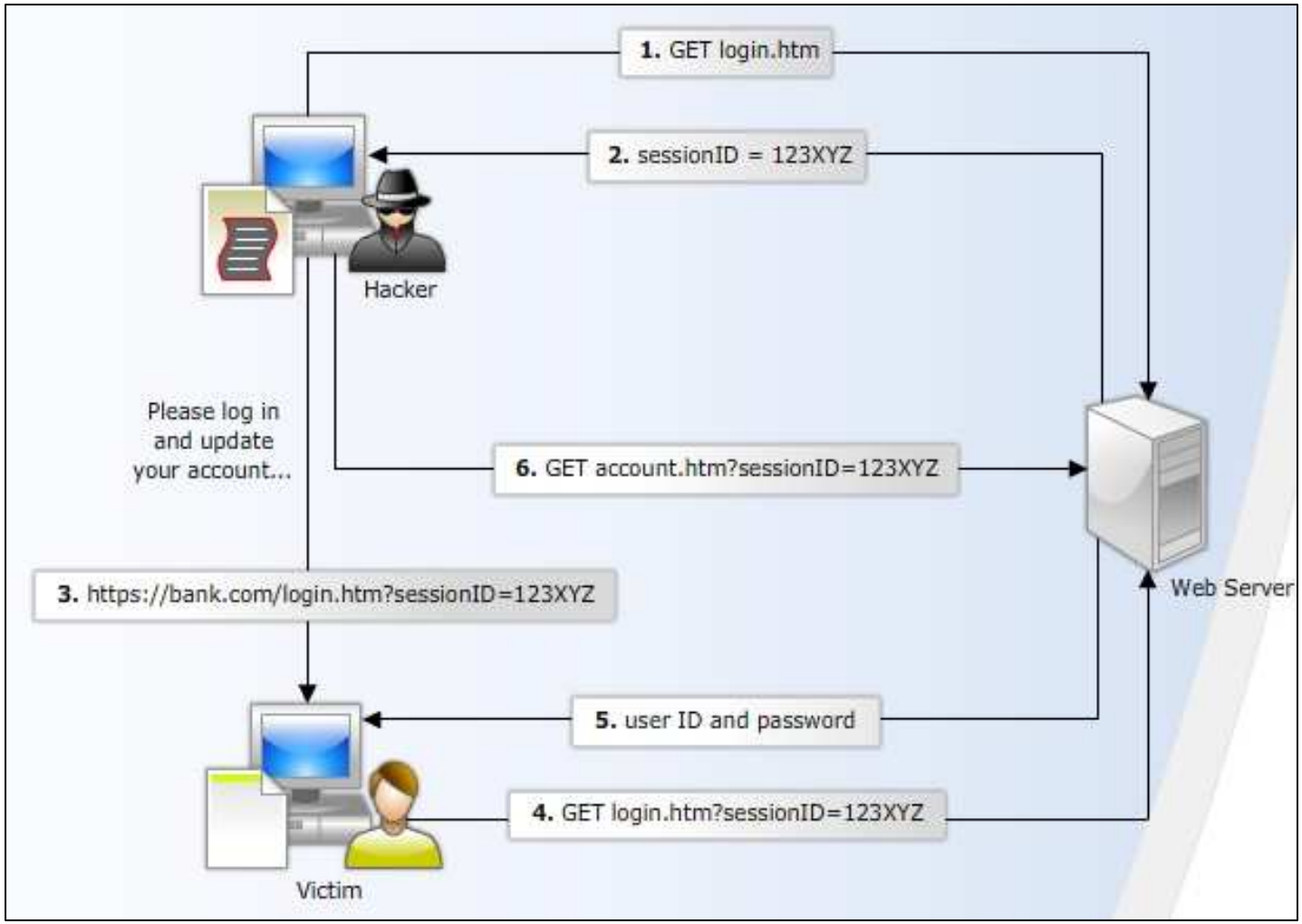


Prediction

- Sequential numbers
- Small character space
- Short session id



Session Fixation



Common Problems

- Persistent cookies
- HTTPOnly is not used
- Secure attribute not set
- XSS vulnerabilities
- Custom session id
- Improper session termination

Securing Sessions

Design and Coding Considerations for Securing Sessions

OWASP

Design Considerations - What

- What will be stored in session objects?
  - Sensitive information
  - Compliance requirements
  - Encryption

Design Considerations - How

- How will session be maintained?
  - QueryString (URL Rewriting)
  - Hidden fields
  - Cookies

Design Considerations - Where

- Where will session data be stored?
  - Single server
    - In memory
  - Server farm
    - Cluster
    - Common repository such as DB or file system

Design Considerations – IP Binding

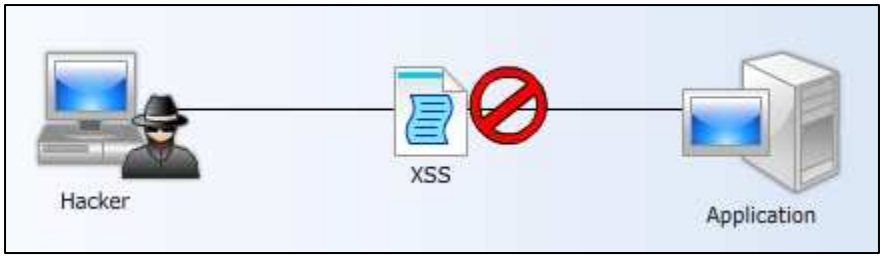
- Bind session to IP address
- Can be a problem if IP changes in the middle of a session

Design Considerations – Browser Fingerprinting

- Assign unique id to the client and track during session
- May not work where same image is used for all clients
- Can indicate an attack if fingerprint changes in the middle of a session

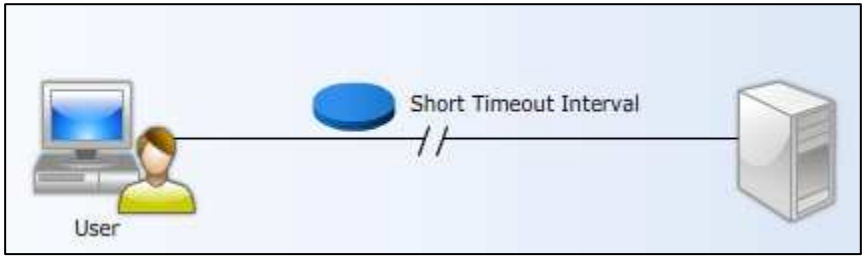
Coding Considerations – XSS and Others

- Protect against XSS and other vulnerabilities
- HTTPOnly attribute for cookies



Coding Considerations – Session Timeout

- Use short session timeouts



Coding Considerations - Logout

- Provide an explicit and prominent logout link

- Do not depend on session timeout

Coding Considerations – Session Termination

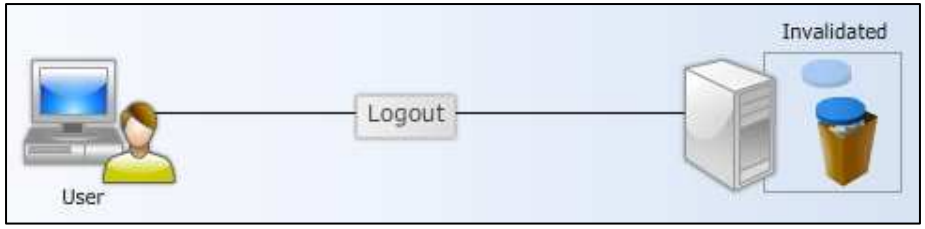
- Terminate sessions properly

```
J2EE

HttpSession session = request.getSession(true)
...
session.invalidate()

.Net

Session.Abandon()
```



- Session state is different from session id

Coding Considerations – Session Id Reuse

- App Servers reuse ids in certain cases
  - Domain level cookie
  - Different app on same server

Coding Considerations – New Session Id

- Regenerate session id on login or privilege change

PHP

session_regenerate_id(TRUE);

The 'TRUE' parameter value will ensure that the old session data is deleted on the server.

.Net

Session.Abandon()

Response.Cookies.Add(new HttpCookie("ASP.NET_SessionId", ""));

The second line will ensure that a new session id is generated, instead of just the session state being cleared and session id reused.

J2EE

session.invalidate()

session.putValue("User",strUserId);

Coding Considerations – Custom Session Id

- Use high entropy session id
  - SecureRandom vs Math.Random

Summary

- WAKE UP!
- Test your (and Siva's) memory
  - List all the major points

Still Got Time?

- Multiple logins with same login id
- Impact of SSO Cookies
- Cross-Site Request Forgery issues