

COMP 4021
Internet Computing

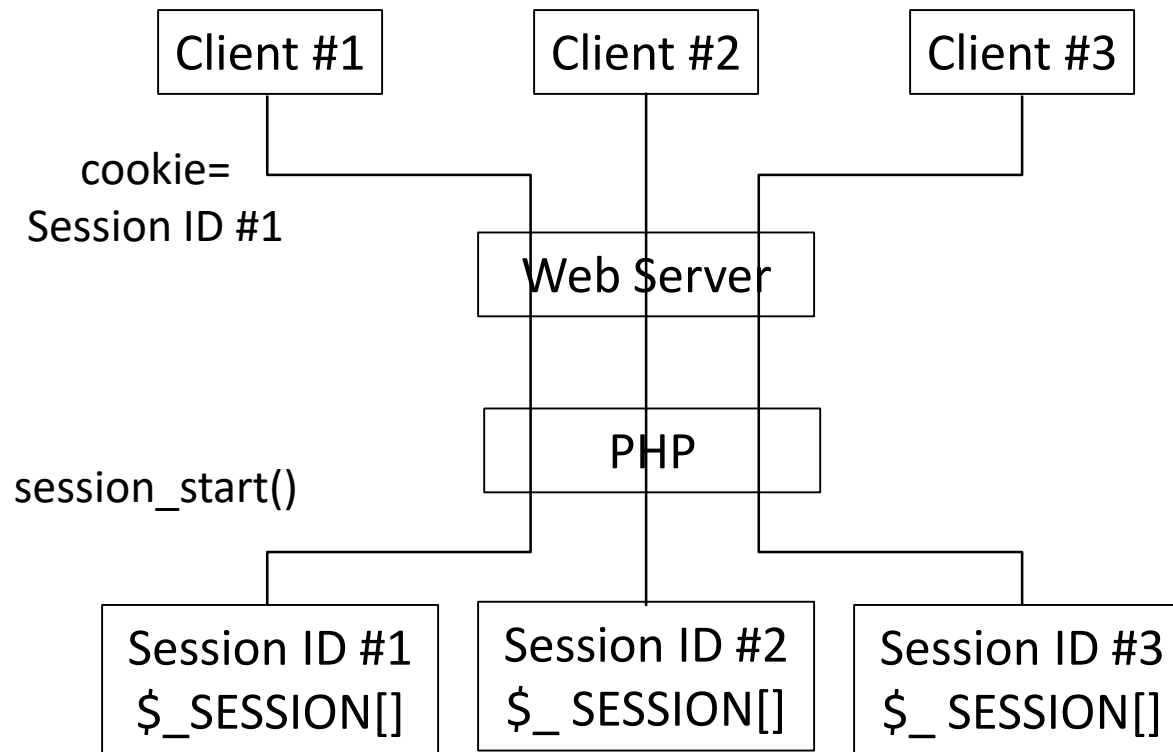
PHP Sessions

What is a 'Session'?

- ❑ HTTP is a **stateless** protocol
 - Requests/responses are independent
 - No 'memory' from one access to another
- ❑ If no state is maintained between accesses, then
 - To authenticate a user, the browser has to send authentication information (i.e. name and password) every time the user accesses a protected page
 - Even if user authentication is not needed (anonymous users), it is desirable to track if accesses to pages come from the same session (i.e., from the same browser in a time period)
 - Create variables that can be shared between web pages within the same session (e.g., "visit count" in the next slide)
- ❑ We mention "states" when we study cookies

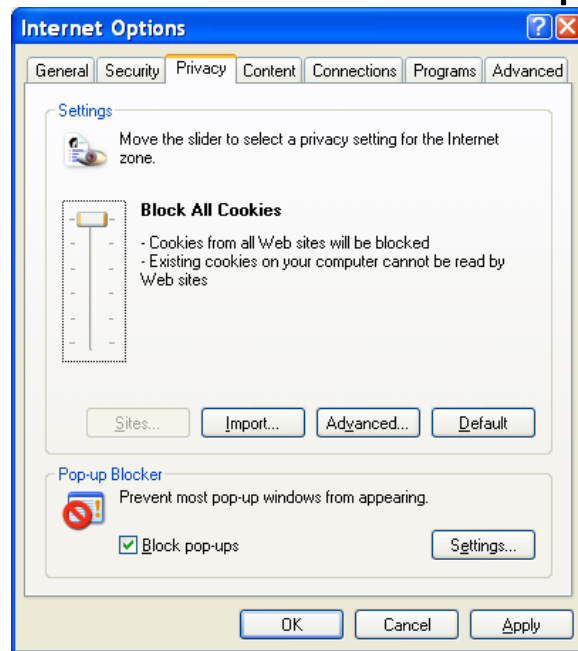
Session in PHP

- PHP supports session ID, session state and passing of session ID via cookie and URL parameter

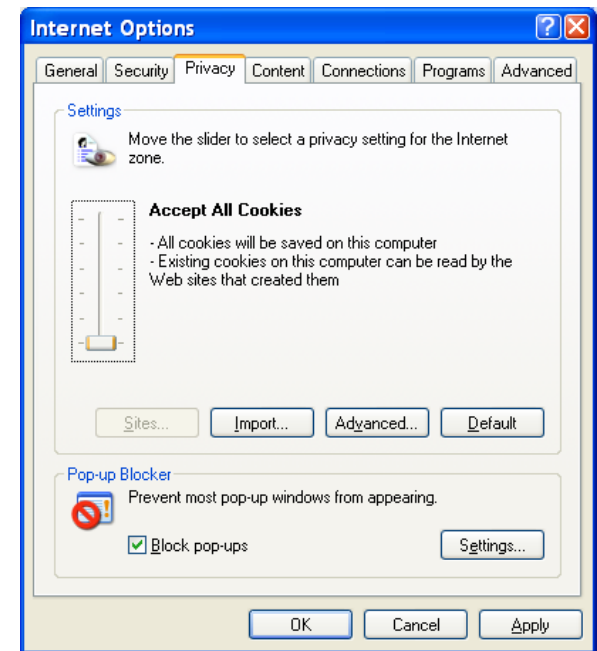


Using Cookies to Track Sessions

- One way to create states is to use cookies, but a user may block cookies on their computer



High 'privacy' – all cookies blocked



Low 'privacy' - all cookies accepted

- Cookies are stored on browsers and are open to attack

Create Session and Session Variables

- ❑ Session ID is transferred using cookies in session_start()
- ❑ If cookies are disabled, page always reports "... 0 times"

```
<?php
    session_start(); // create session ID and $_session global variable

    if ( !isset( $_SESSION['count'] ) ) { // If does not exist, create it
        $_SESSION['count'] = 0;
    } else {
        $_SESSION['count']++;           // Increment count by one
    }

    echo "You have visited here ".
        $_SESSION['count']." time(s)."; // "." is string concatenate
```

?>

[session_using_cookies](#)

session_start()

- ❑ `session_start()` creates a session; placed at the beginning of your PHP script, before PHP has done any printing
- ❑ After a session is created, `session_start()` creates:
 - A unique session ID stored on server, accessible by `session_id()`
 - A super global array `$_SESSION[]` storing session related data, e.g., `$_SESSION["count"]`, `$_SESSION["last_visit_time"]`, etc.
- ❑ `session_start()` must be called each time the page is loaded in order to "count" the page as part of a session
- ❑ If multiple pages are counted in the same session, e.g., counting visits to both `page_1.php` and `page_2.php`, then `session_start()` must be executed in both pages

By Cookies or GET Parameter

- ❑ When `session_start()` is called, it creates a new session or resumes an existing session
- ❑ When cookies are enabled, `session_start()` creates **PHPSESSID** session cookie to store the session ID , and sets **SID** to NULL
- ❑ When cookies are disabled, `session_start()` gets session ID from **PHPSESSID=...** parameter in URL; if not found, generate a new session ID
 - In both cases, sets **SID** to the string: **PHPSESSID=<...session ID...>**
- ❑ PHP session module supports both methods: Use cookies first, if unsuccessful, use GET parameter
 - The previous example does not make use of **SID** when cookies are disabled, so visit count is 0; see next example on how **SID** is used

Unregistering a Session Variable

```
<?php
    session_start();

    // Dump the session variable
    unset($_SESSION['count']) ;
?>
```

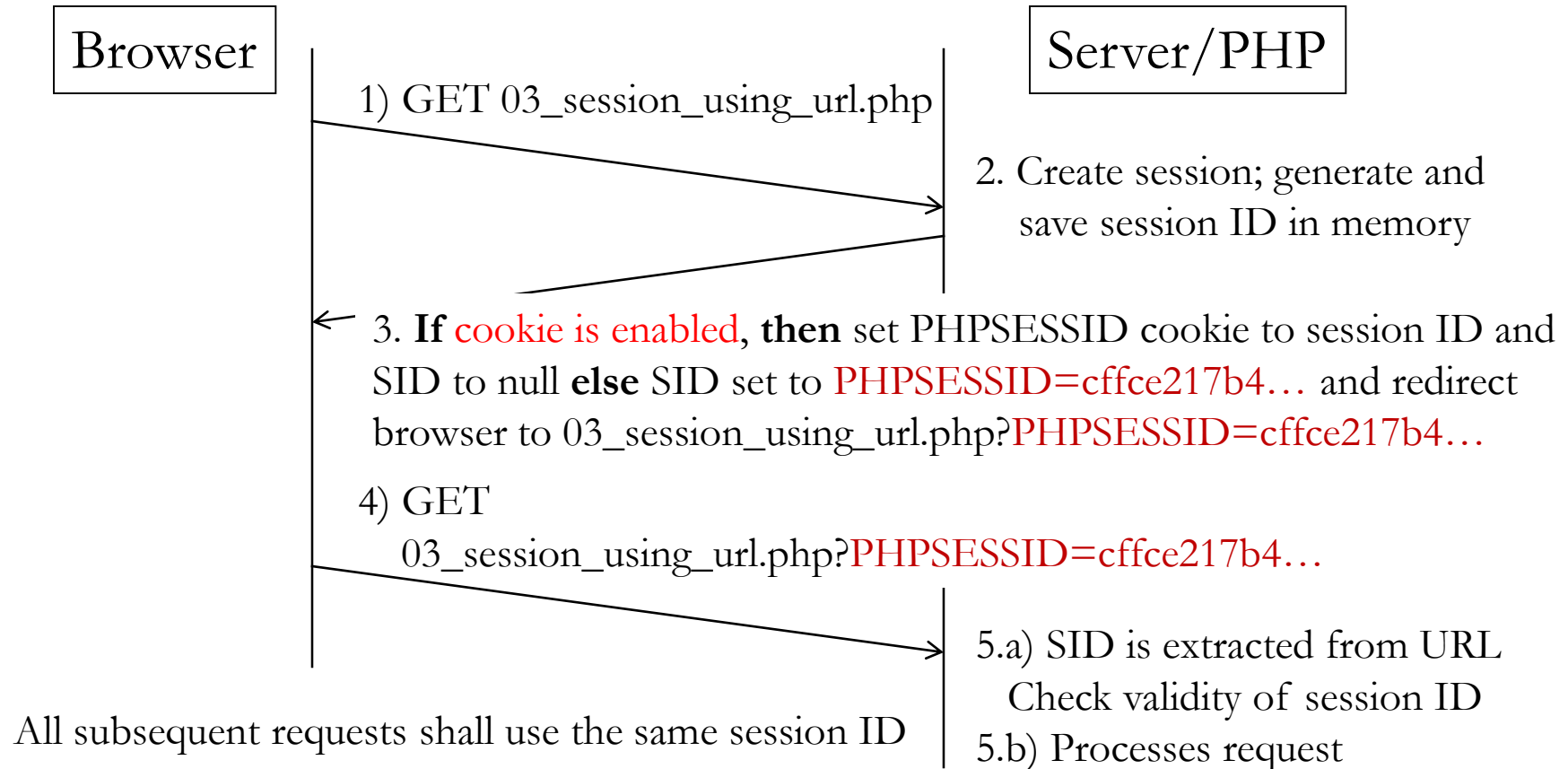
`session_unset()`: Erase all session variables and data

`session_destroy()`: Destroy session data without destroying session variables

session_start() and Session IDs

- A session ID must be at least 128 bits long (why so long?)
- What does session_start() do?
 - **If session has been created (session ID in browser cookie or in URL parameter)**
 - **then** load session data and continue
 - **else** generate session ID and set browser cookie, create \$_SESSION
 - **If session cookie does not exist** (i.e., cookie disabled; use URL redirect)
 - **then** PHP constant SID set to session ID (PHPSESSID=cffce217b4...),
 - **else** SID set to empty (no need because cookie can do the job)
- The first GET is: GET 03_session_using_url.php
- Depending on whether cookies or redirect is used, subsequent GETs are either
GET 03_session_using_url.php?
or
GET 03_session_using_url.php?PHPSESSID=cffce217b4...

A Possible Interaction Scenario



Example Using URL

- Transfers session ID using PHPSESSID parameter in URL

```
<?php
// This will check for both cookie and URL methods
session_start(); // Step 2: Create new session or resume existing session

if (!isset($_SESSION['count'])) { // new session
    header("Location: 03_session_using_url.php?".SID); // Step 3: redirect if
                                                         session is new
    $_SESSION['count'] = 0; // now value is set, so won't execute next time
} else { // existing session; PHPSESSID parameter in URL is ignored
    echo "You have visited here ".$_SESSION['count']." time(s)";
    $_SESSION['count']++; }
?>
```

03_session_using_url.php
(Run with Firefox)

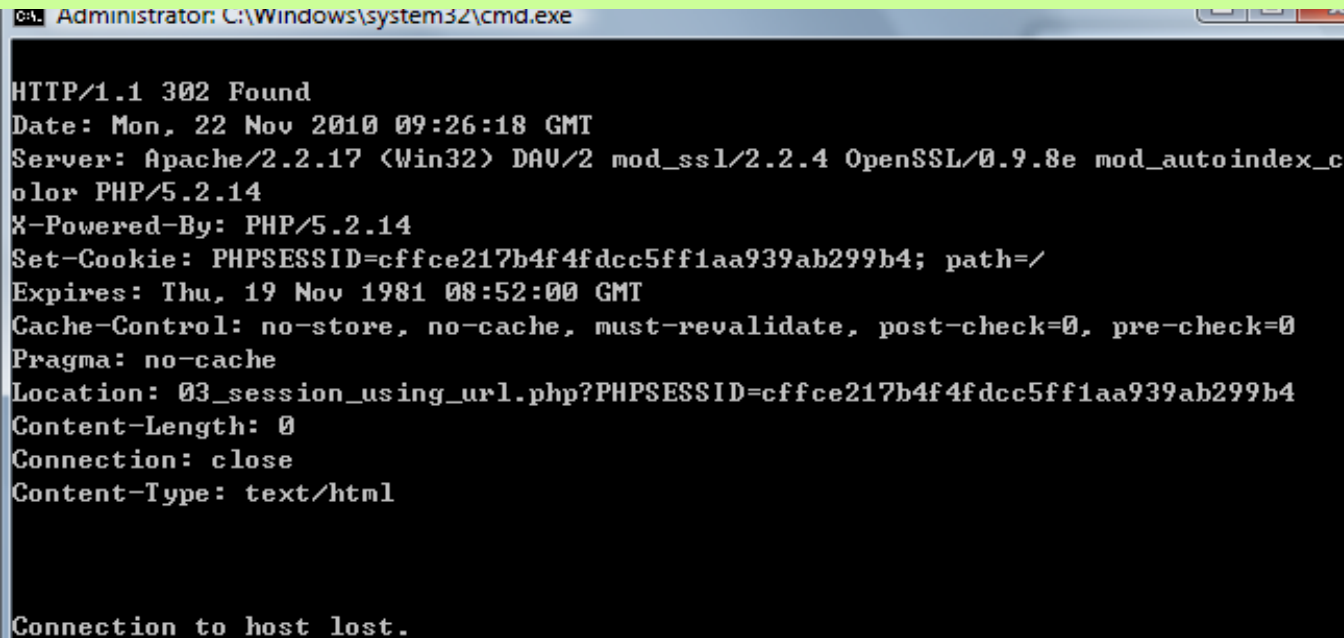
Using URL to pass Session ID

- An example of using URL to pass session ID in a website developed using Java



Example Output using PHP

Session ID is passed to browser by setting the PHPSESSID cookie



```
Administrator: C:\Windows\system32\cmd.exe

HTTP/1.1 302 Found
Date: Mon, 22 Nov 2010 09:26:18 GMT
Server: Apache/2.2.17 (Win32) DAV/2 mod_ssl/2.2.4 OpenSSL/0.9.8e mod_autoindex_color PHP/5.2.14
X-Powered-By: PHP/5.2.14
Set-Cookie: PHPSESSID=cffce217b4f4fdcc5ff1aa939ab299b4; path=/
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Location: 03_session_using_url.php?PHPSESSID=cffce217b4f4fdcc5ff1aa939ab299b4
Content-Length: 0
Connection: close
Content-Type: text/html

Connection to host lost.
```

Tell the browser to re-load the same program again, this time passing the Session ID to the program so that it can access and manipulate it

Take Home Message

- ❑ Session is important for security reason (time out a login)
- ❑ It is also important for web analytics, e.g., a session can be treated as a 'visit'
- ❑ Passing session ID in URL is insecure, and is considered bad practice today
- ❑ PHP provides functions for handling sessions (in particular, `session_start()` and `$_SESSION[]`)
- ❑ How PHP handles sessions can be configured in `php.ini` (detailed not discussed), so don't be surprised if PHP behaves differently in different websites