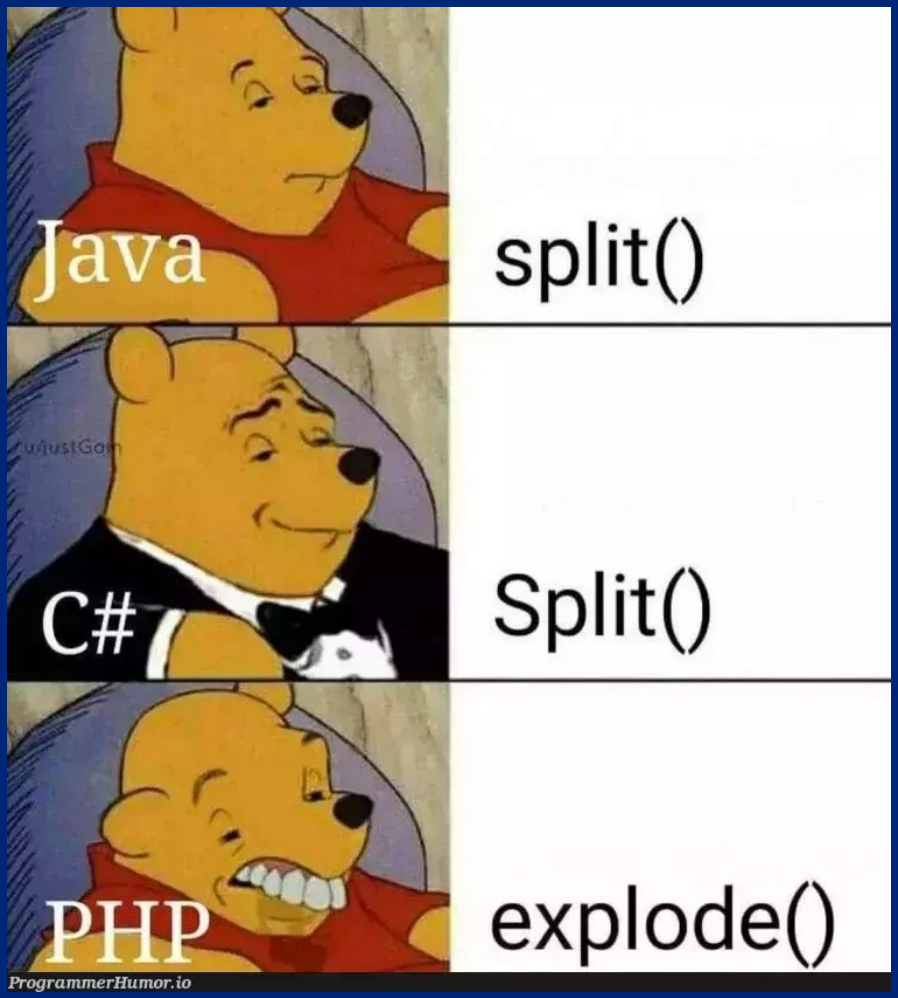
Università di Genova

PHP (3)



Marina Ribaudo, marina.ribaudo@unige.it

Access control

"You may need to introduce access control to your system for a few reasons. The first and most obvious reason is to **allow some people to see (or do) what you want them to see/do** while keeping the others out. However, you must also **know who did what and when**, so that they can be held accountable for their actions."

https://www.feistyduck.com/library/apache-security/online/apachesc-CHP-7.html

Access control

- **Identification**
  - L’utente presenta la sua identità
- **Authentication**
  - Verifica se l’utente può accedere al sistema
- **Authorization**
  - Verifica se l’utente può accedere ad una risorsa particolare
- **Accountability**
  - Capacità di dire chi ha avuto accesso ad un risorsa e quando e se la risorsa è stata modificata

Access control

- HTTP è stateless e nasce per lo scambio di risorse

- Problemi di **autenticazione e autorizzazione**
  - Basic authentication
  - Digest authentication
  - Form-based authentication

Access control

- Altre tecniche, più o meno semplici da implementare, sono
  - URL nascoste
    // si usavano all'inizio del web, oggi non vanno più bene!
  - Controllo basato sull'indirizzo IP o sul nome di dominio
    // usato nelle intranet aziendali

Basic Authentication

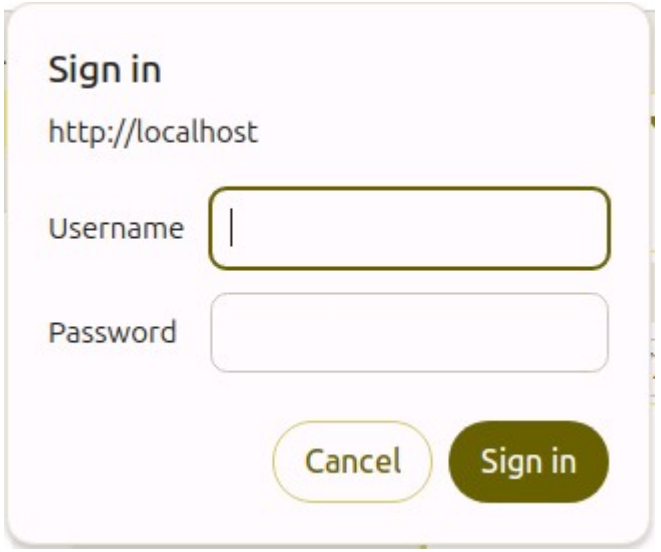
- I **controlli** basati sull'identità dell'utente possono essere **demandati al server web** sfruttando la **Basic Authentication di HTTP**
- Quando si cerca di accedere a informazioni protette con Basic Authentication
  - il browser visualizza una finestra di dialogo che richiede le credenziali all'utente
  - le credenziali vengono scambiate tra browser e server per tutta la durata dell'interazione

Basic Authentication

▼ Response Headers

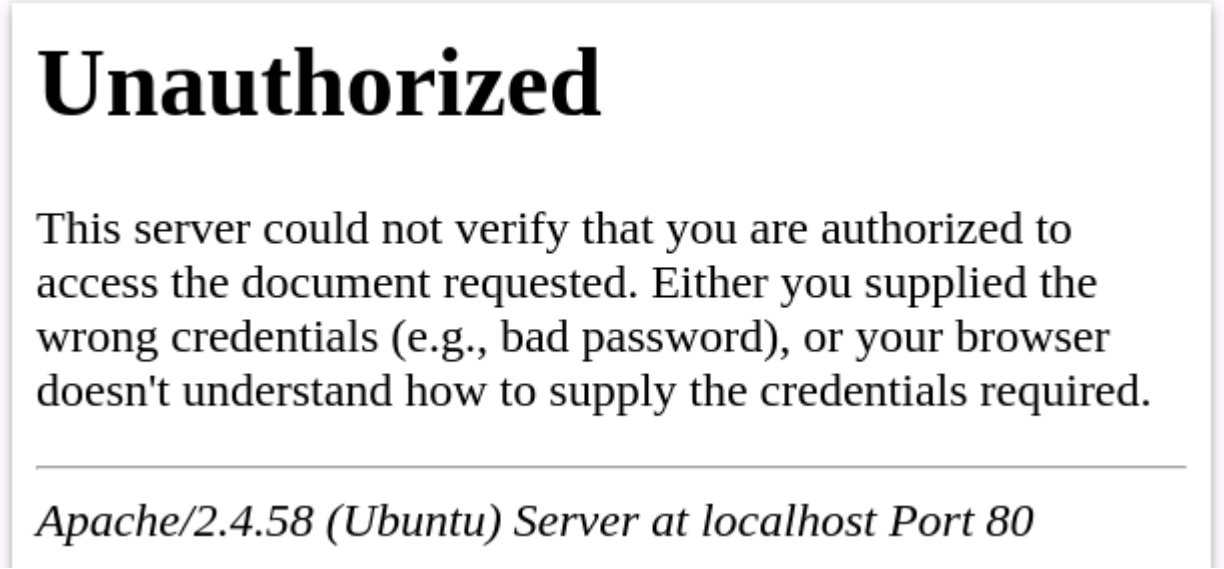
Raw

```
HTTP/1.1 401 Unauthorized
Date: Mon, 18 Nov 2024 10:48:00 GMT
Server: Apache/2.4.58 (Ubuntu)
WWW-Authenticate: Basic realm="Please provide username and pass
word"
Content-Length: 456
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=iso-8859-1
```



Basic Authentication

HTTP/1.1 401 Unauthorized



Basic Authentication

Authorization: Basic bWFyaW5hOm1hcmluYQ==

Pagina ad accesso riservato, senza credenziali non si entra :)

Basic Authentication: How to

- Si deve creare un file di testo **.htaccess** nella directory che si vuole proteggere e specificare delle direttive

  Vedi: http://httpd.apache.org/docs/current/howto/auth.html

```
AuthType Basic
AuthName “Restricted Area for My Server”
AuthUserFile  /absolutepath/nomefilepwd
Require valid-user
```

** invece di scrivere valid-user, si possono elencare gli utenti che possono accedere, oppure un gruppo di utenti*

Basic Authentication: How to

- Le **credenziali** degli utenti possono essere **salvate in un file di testo**, specificando la direttiva
  - AuthBasicProvider **file**

- Con il file di testo ci sono problemi di efficienza (il file viene letto per ogni accesso alle risorse nell'area protetta) e si possono anche usare formati di storage come **dbm** (db chiave-valore) e **dbd** (db relazionale), o **LDAP**
  - AuthBasicProvider **dbm**
  - AuthBasicProvider **ldap**

Basic Authentication: How to

- Il file delle password si crea e aggiorna con il comando **htpasswd** (-c si deve usare solo la prima volta che si crea il file delle password)

```
htpasswd –c nomefilepwd nomeuser
New password: **********
Re-type new password: **********
```

Nota: Il file .htaccess e quello delle password devono essere **leggibili dal web server** (chmod 644)

Basic Authentication: How to

- Perché tutto funzioni bisogna **abilitare un modulo di Apache** (mod_auth) e **modificare la direttiva AllowOverride** nel file di configurazione di Apache
  Per chi fosse interessato/a
  https://www.keycdn.com/support/htaccess-not-working

- Il file .htaccess permette di specificare anche altre configurazioni per i server web che usano Apache
  Per chi fosse interessato/a
  https://www.keycdn.com/support/popular-htaccess-examples

Basic Authentication

- Basic authentication ha un certo numero di **svantaggi**
  - Le credenziali sono trasmesse in base64
  - Non esiste la possibilità di fare logout (si deve chiudere il browser)
  - Il form di login non può essere personalizzato
  - I proxy HTTP possono estrarre le credenziali dal traffico di rete

Basic Authentication

- Basic authentication ha un certo numero di **svantaggi**
  - Per ogni accesso a una pagina/risorsa protetta, il server deve leggere le credenziali dalla richiesta HTTP e poi accedere a
    - file .htaccess per le direttive
    - file/database/LDAP per la password
  - Se il numero degli utenti cresce, questo meccanismo di autenticazione diventa inefficiente

Digest Authentication

- Permette l’autenticazione senza inviare le credenziali in chiaro (in base64)
- Il server invia al client una challenge e il client risponde inviando un **hash** della soluzione della challenge e della password
- Il server verifica se il client possiede la password corretta

Digest Authentication

*"This module implements **HTTP Digest Authentication** (RFC2617), and provides an alternative to mod_auth_basic where the **password is not transmitted as cleartext**."*

It uses MD5… The MD5 calculations used in HTTP digest authentication is intended to be "one way", meaning that it should be difficult to determine the original input when only the output is known. If the password itself is too simple, however, then it may be possible to test all possible inputs and find a matching output (a brute-force attack) – perhaps aided by a dictionary or suitable look-up list, which for MD5 is readily available

*"Therefore, using **basic auth** and encrypting the whole connection using **mod_ssl** is a much better alternative."*

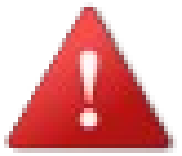
https://httpd.apache.org/docs/2.4/mod/mod_auth_digest.html

Form-based authentication

- Invece di lavorare a livello di protocollo HTTP si lavora a livello di applicazione web
- Per le pagine ad accesso riservato, in risposta ad una richiesta da parte di un utente non ancora autenticato, l’applicazione restituisce il form per il login

Form-based authentication

“Applications are often given far less testing than the web server and potentially contain more security issues. Some files in the application, for example, may not be protected at all. Images are almost never protected. Often applications contain large amounts of code that are executed prior to authentication. The chances of an intruder finding a hole are much higher when application-level authentication is used.”



https://www.feistyduck.com/library/apache-security/online/apachesc-CHP-7.html

Università di Genova

PHP (4)

Marina Ribaudo, marina.ribaudo@unige.it

Session control

- Il controllo di sessione permette di **tener traccia dell'utente** durante la sua interazione con un sito web

“Suppose you are building one e-commerce site, to allow anyone to buy the product you must ask them to log-in with their user name and until they log out your system must track the user in every step, this concept is called as “session tracking”.

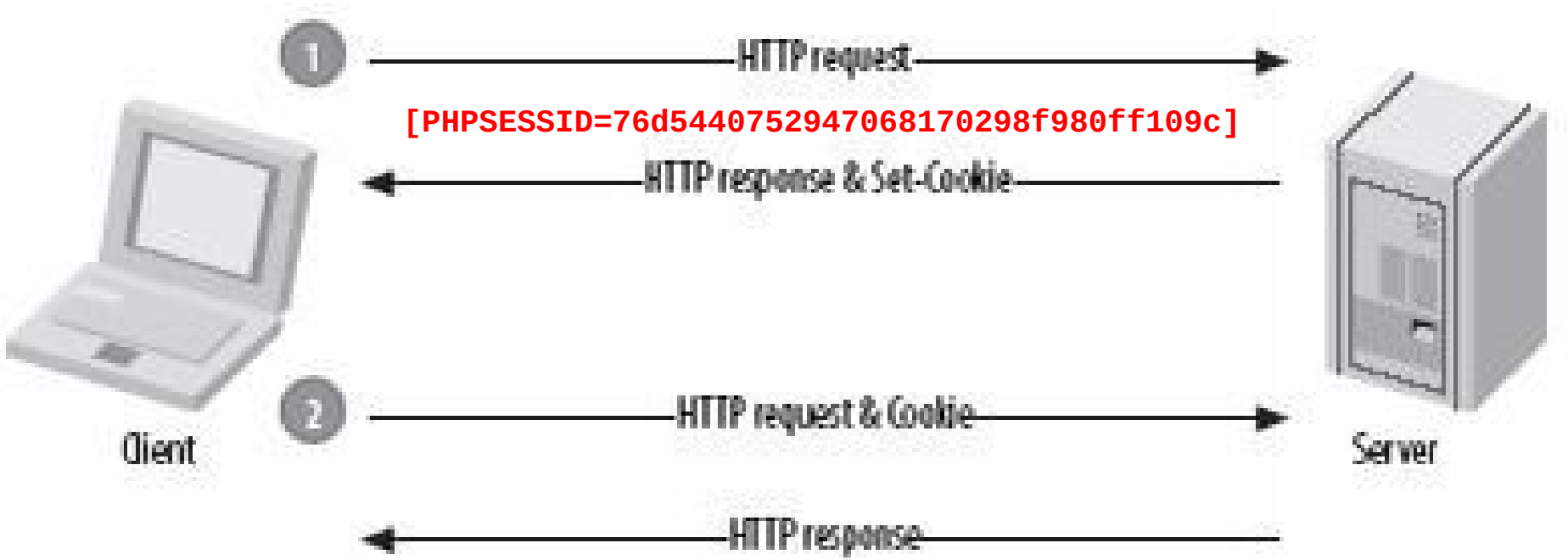
*Now why do we need to track the session, answer is very simple. **HTTP is stateless protocol**, and when you refresh the page, it lost everything, which your project should not!”*

Session control

- Per gestire il controllo di sessione si usa l'**array superglobale $_SESSION**

- Inoltre in PHP "… a visitor accessing your web site is assigned an **unique id**, the so-called **session ID**. This is either stored in a cookie on the user side or is propagated in the URL …"

Session control



[dal libro Essential PHP security]

Session control: how to

- Iniziare la sessione
- Creare le variabili di sessione
- Usare le variabili di sessione
- Rilasciare le variabili di sessione
- Chiudere la sessione

Iniziare la sessione

session_start();

Verifica se l'utente ha già un identificatore di sessione

Se non lo trova, ne crea uno, altrimenti rende accessibili le variabili di sessione già create in precedenza per quell'utente

Quando si usano le sessioni è **obbligatorio inziare tutti gli script delle pagine riservate con session_start()**

https://www.php.net/manual/en/function.session-start.php

Creare le variabili di sessione

$_SESSION[“key”] = <value>;

La variabile di sessione viene creata assegnando una **chiave** all’array superglobale **$_SESSION** e un **valore** corrispondente

La variabile esiste e viene “tracciata” fino a quando non si termina la sessione

Usare le variabili di sessione (1)

Con isset() si verifica se la variabile di sessione esiste

```
if (isset($_SESSION[“key”]))  {
    <here session variable exists>
    <present content for logged users>
}
else {
    <redirect to login page>
}
```

Se sì, l'utente è autorizzato a proseguire, altrimenti lo si rimanda alla pagina di login

Usare le variabili di sessione (2)

Con isset() si verifica se la variabile di sessione **non** esiste

```
if (!isset($_SESSION[“key”]))  {
    <redirect to login page>
    exit();
}

<here session variable exists>
<present content for logged users>
```

Se la variabile di sessione non esiste, si rimanda alla pagina di login e si esce dallo script

Rilasciare le variabili di sessione

unset($_SESSION[“key”]);

La sessione esiste ancora, ma la variabile con chiave key non è più registrata come variabile di sessione

session_unset(); // deprecato

$_SESSION = array();

Chiudere la sessione

session_destroy();

Cancella l’identificatore di sessione

Identificatore di sessione


```
<?php
        session_start();
        echo "PHPSESSID: " . session_id();
?>
```

```
PHPSESSID: 0a777aa640f1bad7ac3f788065a99ba6
```

Durata di una sessione

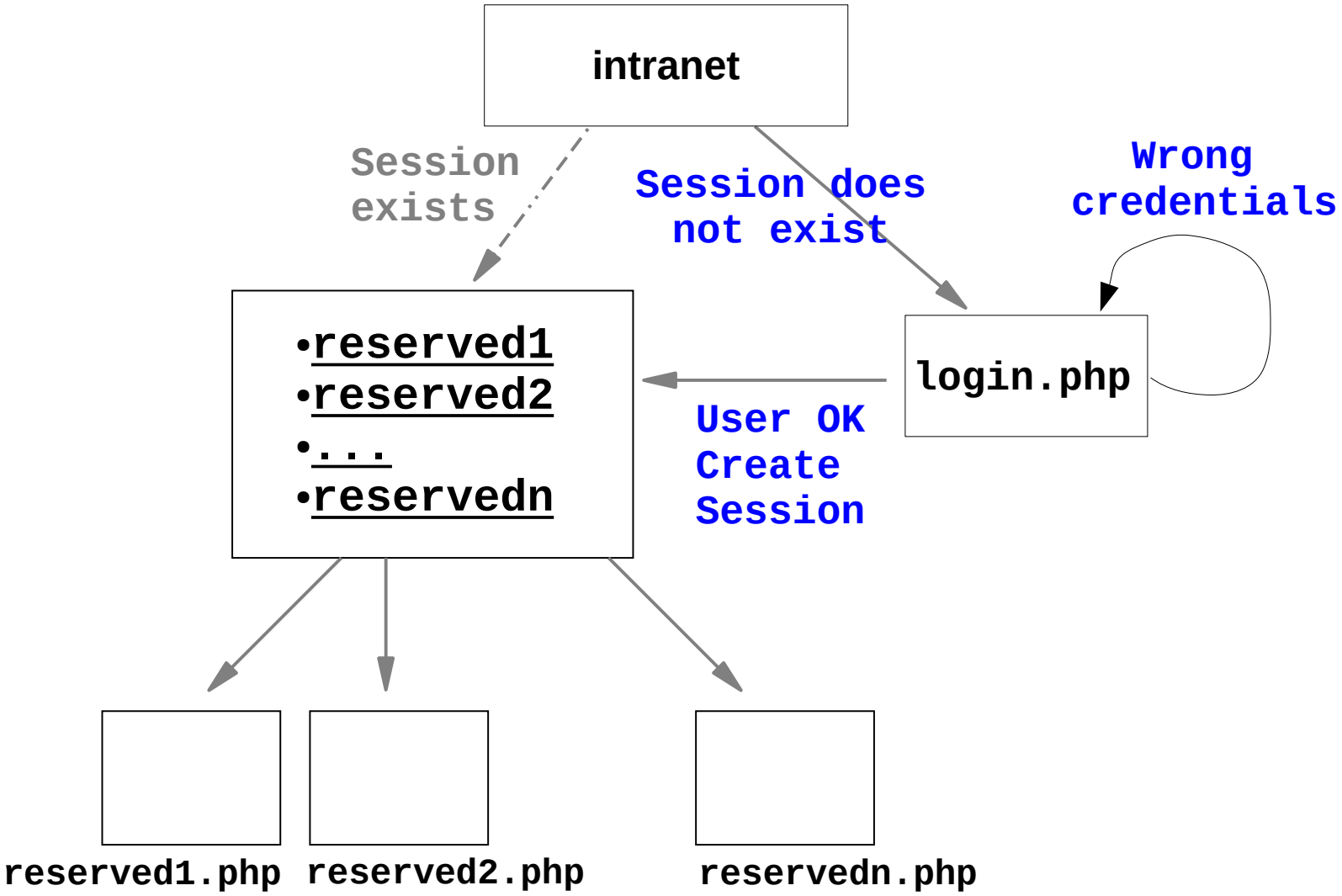
- Le sessioni in PHP hanno una durata che è definita dalla direttiva **session.gc_maxlifetime** nel file di configurazione **php.ini**
  - default 24 minuti
- Si può impostare un valore diverso per la durata di una sessione utilizzando la funzione **ini_set()**
  - ini_set('session.gc_maxlifetime', 3600);
- Una sessione può terminare anche prima della scadenza se l'utente **chiude il browser** o se il server web termina la sessione per motivi di sicurezza

Valori della sessione sul server

- Oltre all'indentificatore SID, nelle variabili di sessione si possono **memorizzare altre informazioni** che, per ogni utente autenticato, sono disponibili a tutti gli script del sito
- Queste informazioni (il contenuto dell'array $_SESSION) sono memorizzatie in un'area del server web non accessibile dall'esterno e riservata all'utente root
- Usando la funzione php_info() Si può capire dove sono memorizzate le informazioni di sessione

session.save_path	/var/lib/php/sessions

Session control: esempio



Session control: esempio

- Nella fase di autenticazione
  - Si verifica la correttezza delle credenziali dell'utente
  - Se le credenziali sono corrette
    - si creano una (o più) variabili di sessione
    - si presentano i servizi dell'area riservata
  - Altrimenti (credenziali errate)
    - si rimanda al login

Session control: esempio

Al login

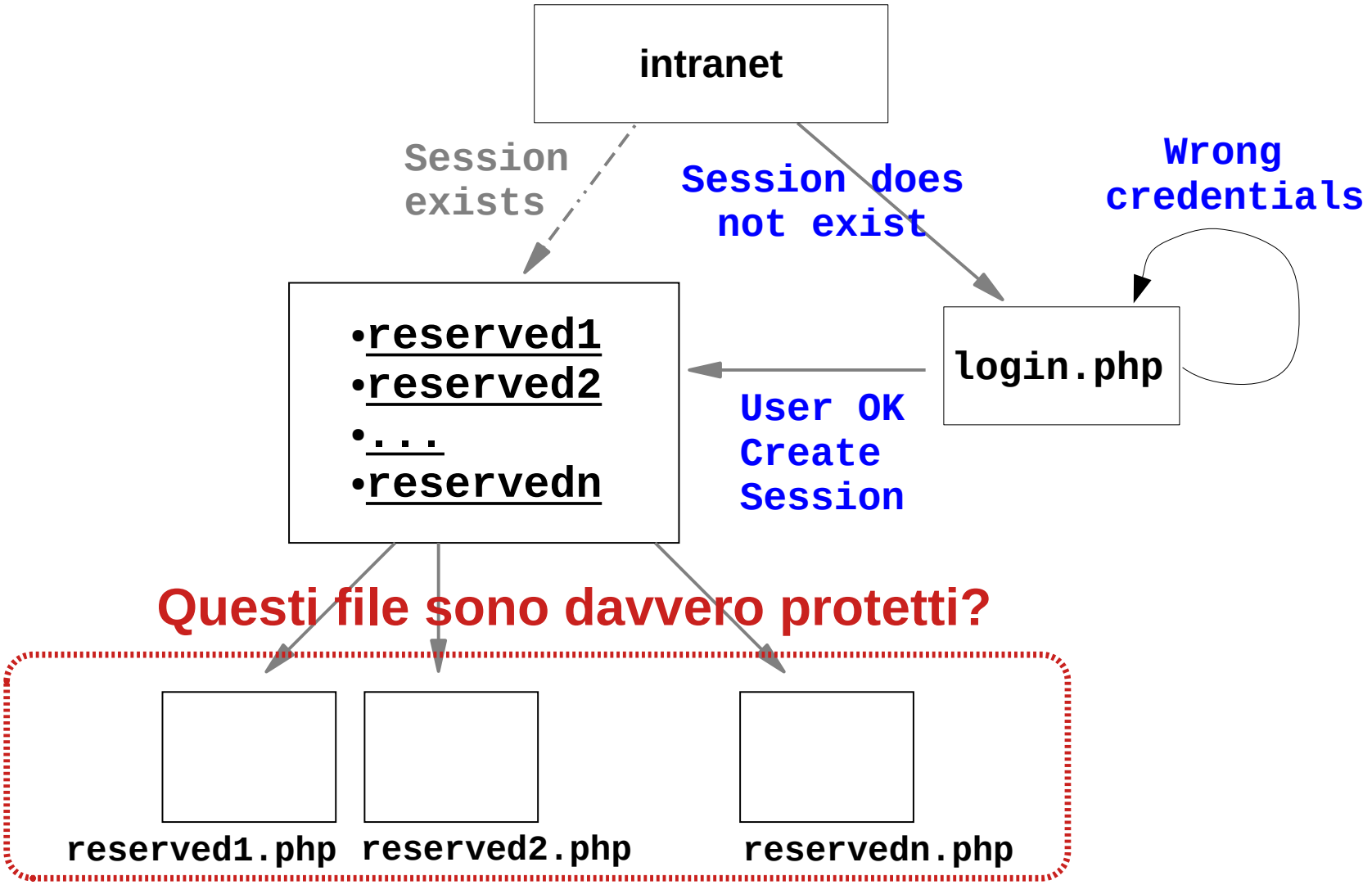
```
<?php
   session_start();

/* check if username and password are stored in a User table*/

if <YES> {
     $_SESSION[“key1”] = ...;
     $_SESSION[“key2”] = ...;
     $_SESSION[“key3”] = ...;
     /* present list of services */
   }
else {
      echo “<p class=’error’>Access denied, check your credentials</p>\n”;
   }

?>
```

Session control: esempio



Session control: esempio

- Durante la navigazione nell'area riservata
  - **Tutti i file** che offrono servizi **devono controllare le variabili di sessione**, non basta il controllo sul primo file!

Session control: esempio

- Durante la navigazione nell'area riservata

```
<?php
  session_start();

  If (!isset($_SESSION[“key1”]))  {
      <redirect to login page, header() function>
      exit();
  }

  <show reserved page>
```


Practice
Practice
Practice