



php



Ανάπτυξη Διαδικτυακών Συστημάτων &
Εφαρμογών

Τμ. Μηχανικών Πληροφορικής και
Ηλεκτρονικών Συστημάτων
ΔιΠαΕ

Αντώνης Σιδηρόπουλος

http = stateless



Login.html

```
<html><body>
<form action="login.php" method="GET">
user: <input name="username"><br/>
password: <input name="pass" type="password">
</form></body></html>
```

Login.php

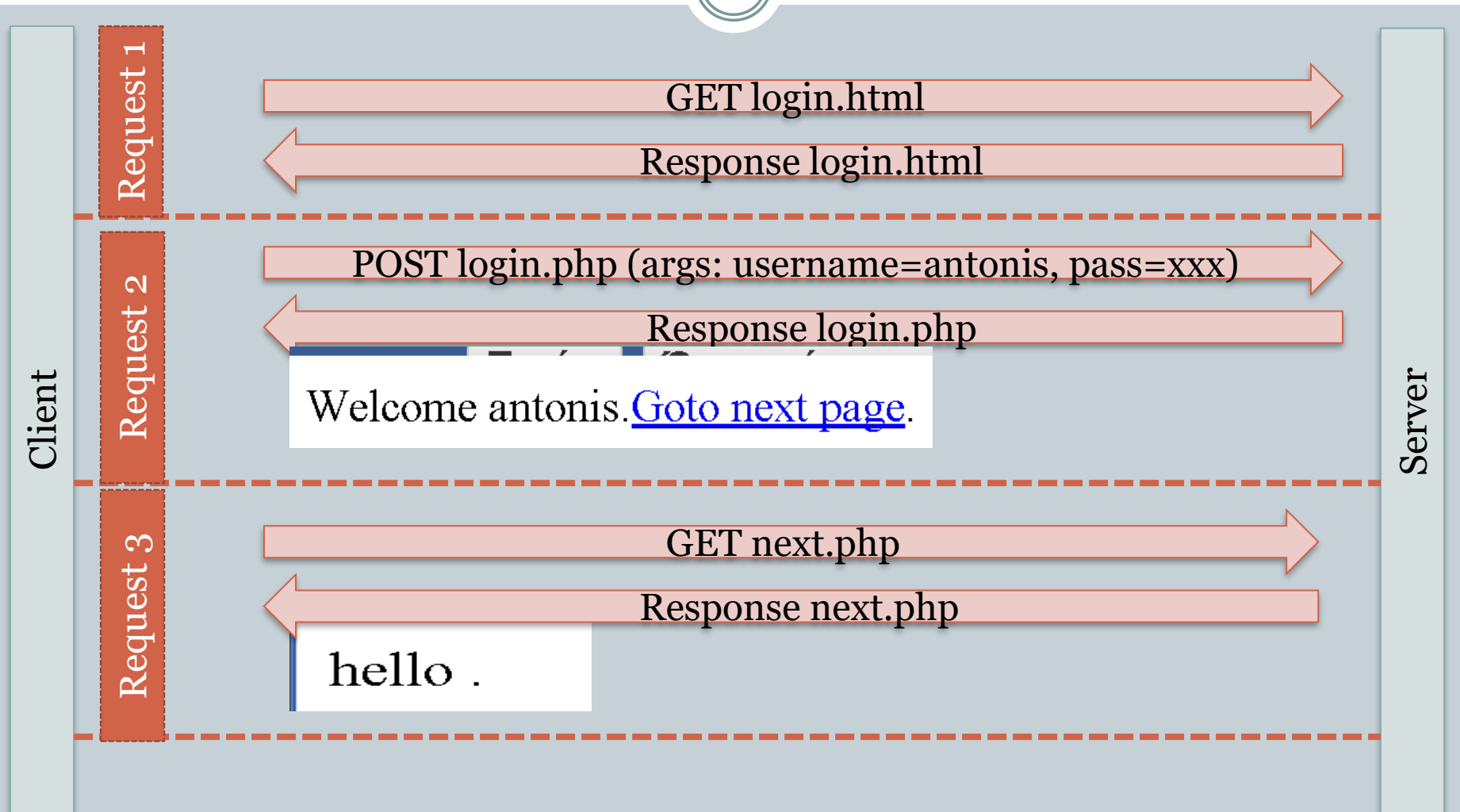
```
<?php
// check the $_REQUEST[username] & $_REQUEST[pass]
print "Welcome $_REQUEST[username].";
print "<a href=\"next.php\">Goto next page</a>.";
?>
```

next.php

```
<?php
print "hello $_REQUEST[username].";
?>
```

http = stateless

3



workaround

4

- Θα θέλαμε όλες οι μετέπειτα αιτήσεις να περιλαμβάνουν την πληροφορία username.
 - Υποχρεωτικά και password (ώστε να γίνεται κάθε φορά επιβεβαίωση)
 - Θα πρέπει όλες οι φόρμες της εφαρμογής να περιλαμβάνουν τα πεδία αυτά (hidden και συμπληρωμένα)
 - Θα πρέπει όλα τα links της εφαρμογής να περιλαμβάνουν τα πεδία αυτά (υποχρεωτικά GET!!!! – oooooorrrrrssss!!!! – password στο GET)
- ΔΕΝ είναι εφικτά τα παραπάνω ή είναι πολύ κακή λύση.

Cookies

5

- Τα cookies είναι strings τα οποία στέλνει ο server μέσα από το http response προς τον client.
- Μετά ο client σε κάθε request που στέλνει στον server συμπεριλαμβάνει όλα τα cookies (του server) μέσα στο http request.
- Αποθηκεύονται στον client μέχρι να γίνουν expire (δεν αποθηκεύονται στον server)
 - Συνήθως ορίζεται να γίνουν expire με τον τερματισμό του browser
 - Ή κάνουν expire μετά από κάποιο χρονικό διάστημα: μερικά λεπτά ή μερικές ημέρες
 - Ή δεν κάνουν expire ποτέ.
- Χρησιμοποιούνται επίσης για cross-server user tracking. Ένας server ορίζει ένα cookie για άλλον server. (by default δεν επιτρέπεται άμεσα – χρησιμοποιείται όμως έμμεσα)

Cross domain user tracking

6

Οι σελίδες από το w3schools.com κάνουν “include” javascripts από .google.com

Αν επισκεφτούμε άλλο site που κάνει “include” javascripts από .google.com (google analytics) η google κάνει ταυτοποίηση χρήστη και καταγράφει το navigation history μας.

Το .google.com θέτει μόνιμα cookies, έχοντας κρατήσει την πληροφορία ότι έχουμε επισκεφτεί το w3schools

Περισσότερα:
<https://developers.google.com/analytics/devguides/collection/analyticsjs/cross-domain>

http = stateless

7

Login.php

```
<?php
// check the $_REQUEST[username] & $_REQUEST[pass]
print "Welcome $_REQUEST[username].";
setcookie('username', $_REQUEST['username'], time() + (86400 * 30), "/");
setcookie('pass', $_REQUEST['pass'], time() + (86400 * 30), "/");

print "<a href=\"next.php\">Goto next page</a>.";
?>
```

next.php

```
<?php
print "hello $_COOKIE[username].";
?>
```

ae0tos.it.teithe.gr/~asidirop/tmp/login.php?usernam

Εφαρμογές H.P. WEATHER Banks TEI GoogleMore Academic Asid LocalServices » Άλλοι σελιδοδείκτες

Welcome antonis. [Goto next page.](#)

ae0tos.it.teithe.gr/~asidirop/tmp/next.php

Elements Network Sources Timeline Profiles Resources Audits Console

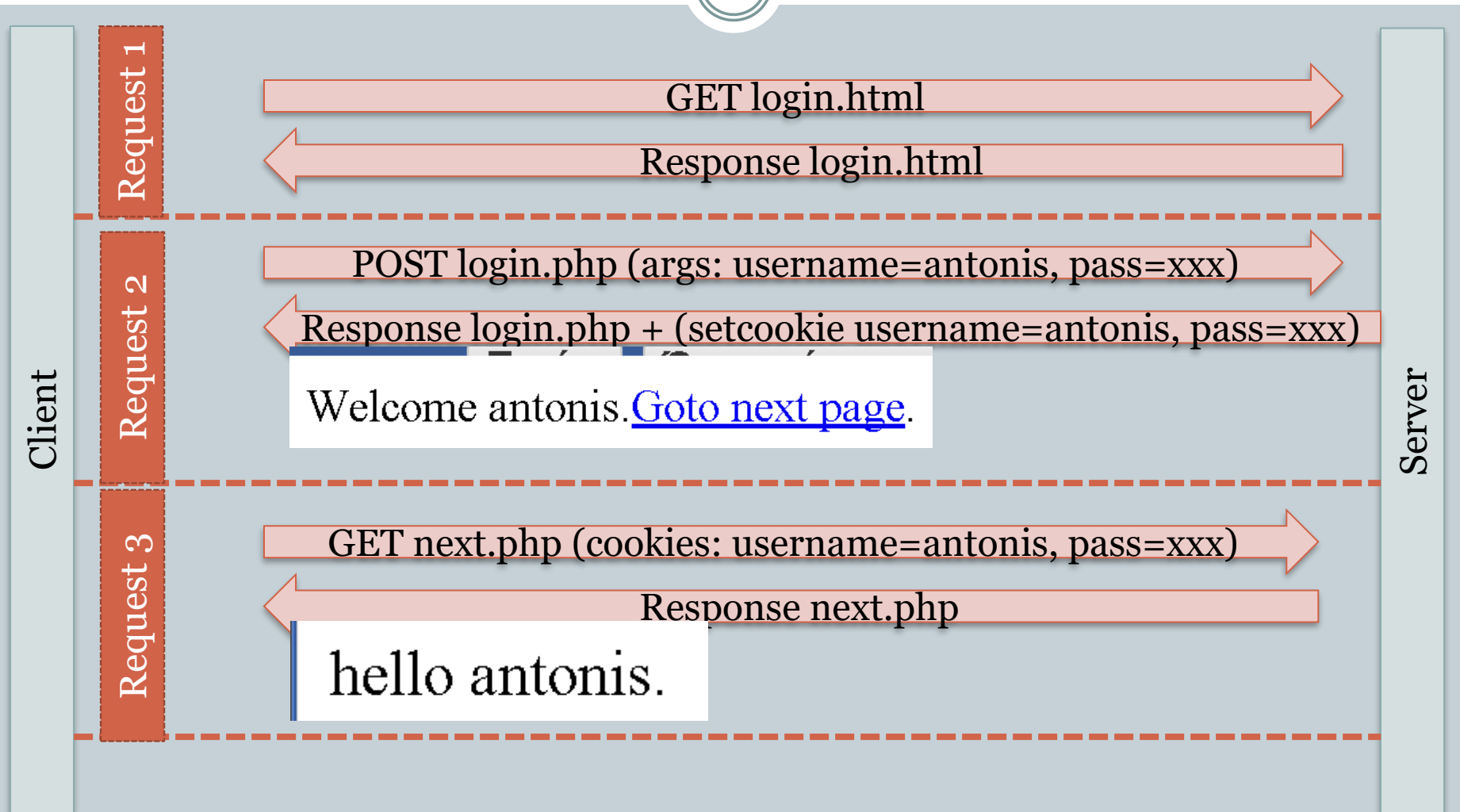
Name	Value	Domain	...	Expires / ...	S...	H	S...
PHPSESSID	rpkqba8nf465o9qs3r7jhgu...	ae0tos.it.teithe.gr	/	Session	35		
pass	123	ae0tos.it.teithe.gr	/	2014-11...	7		
style	pre_style1	ae0tos.it.teithe.gr	/	2015-08...	15		
username	antonis	ae0tos.it.teithe.gr	/	2014-11...	15		

03-PHP-MySQL.pdf 02b-PHP-Adv.pdf

Εμφάνιση όλων των λήψεων...

cookies

9



cookies

10

- Στο προηγούμενο παράδειγμα σε κάθε αίτηση του browser αποστέλλονται username+password
- Θα πρέπει σε κάθε αίτηση να γίνεται έλεγχος και ταυτοποίηση των username+password
- + Ο server δεν χρειάζεται να αποθηκεύσει καμία πληροφορία
- Αποστέλλεται πολλές φορές το password μέσω δικτύου (risky)
- Ο server κάθε φορά πρέπει να κάνει ταυτοποίηση (κατανάλωση πόρων)
- Εάν δεν βάζαμε το password σε cookie, τότε το σύστημα θα ήταν ευάλωτο σε fake logins.
- Λύση: SESSION!

session

11

- Ο server αντί να θέσει ως cookies τα `username=antonis, password=123` θέτει `PHPSESSID=4324534AEDF643435ACDE43`
- Εσωτερικά διατηρεί μια βάση με τα sessionIDs και καταχωρεί στο `4324534AEDF643435ACDE4` την πληροφορία `username=antonis, password=123`
- Η βάση αυτή «καθαρίζεται» κατά τακτά χρονικά διαστήματα.
- + Δεν αποστέλλονται τα στοιχεία ταυτοποίησης.
- + Δεν είναι ευάλωτο το σύστημα σε fake-logins διότι ο 'hacker' θα πρέπει να «κλέψει» ένα sessionID (sessionID injection)
- + Δεν χρειάζεται να αποθηκευτούν passwords πουθενά.

sessions

12

Login.php

```
<?php
// check the $_REQUEST[username] & $_REQUEST[pass]
print "Welcome $_REQUEST[username].";
session_start();
$_SESSION['username'] = $_REQUEST['username'];

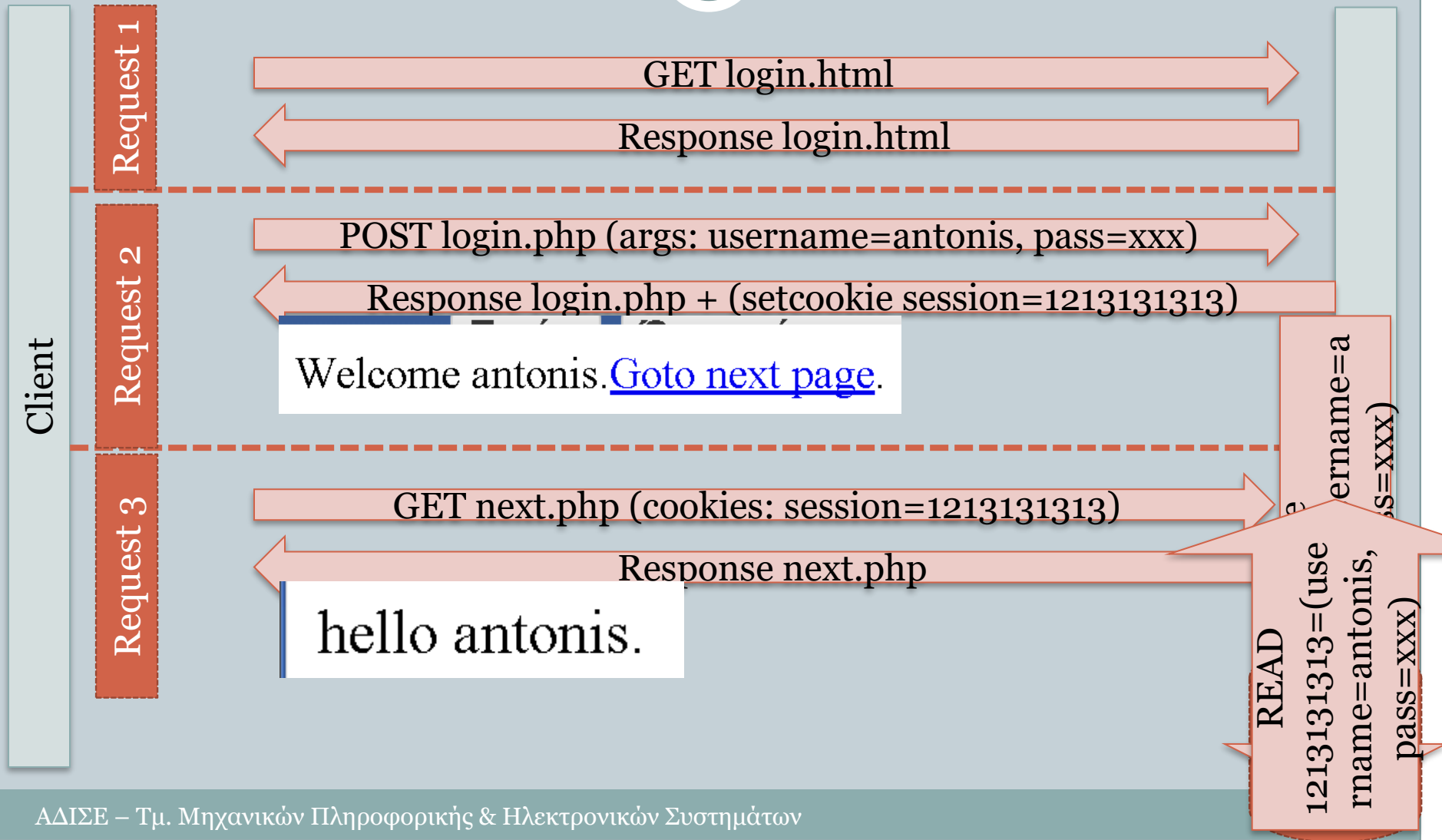
print "<a href=\"next.php\">Goto next page</a&gt.";
?>
```

next.php

```
<?php
session_start();
print "hello $_SESSION[username].";
?>
```

cookies

13



Security

14

- In 2013, 9% of all vulnerabilities listed by the National Vulnerability Database were linked to PHP.
- historically, about 30% of all vulnerabilities listed since 1996 in this database are linked to PHP.
- Technical security flaws of the language itself or of its core libraries **are not frequent** (22 in 2009, about 1% of the total although PHP applies to about 20% of programs listed).
- Recognizing that **programmers make mistakes**, some languages include taint checking to automatically detect the lack of input validation which induces many issues. Such a feature is being developed for PHP, but its inclusion into a release has been rejected several times in the past.

Session injection

15

- Για αποφυγή του sessionID injection μπορεί απλά να αποθηκεύεται μέσα στο session και το IP του χρήστη. Έτσι μπορεί να αναγνωριστεί αν χρησιμοποιηθεί το ίδιο sessionID από άλλο IP.
- Περισσότερα: <http://php.net/manual/en/session.security.php>

Login.php

```
<?php
// check the $_REQUEST[username] & $_REQUEST[pass]
print "Welcome $_REQUEST[username].";
session_start();
$_SESSION['username']= $_REQUEST['username'];
$_SESSION['IP']= $_ENV['REMOTE_ADDR'];
print "<a href=\"next.php\">Goto next page</a>.";
?>
```

next.php

```
<?php
session_start();
If ($_ENV['REMOTE_ADDR'] != $_SESSION['IP']) {
    die "You have logged in from another location";
}
print "hello $_SESSION[username].";
?>
```

Functions

16

Syntax:

```
function functionName(arguments) {  
    ....  
    code to be executed;  
    ....  
    return value;  
}
```


void function, no argument

17

- ```
<?php
function writeMsg() {
 echo "Hello world!";
}
```

```
writeMsg(); // call the function
?>
```

# void function, some arguments

18

```
<?php
function familyName($fname, $year) {
 echo "$fname Mcaffee. Born in $year

";
}

familyName("Hege", "1975");
familyName("Stale", "1978");
?>
```

# void function, some arguments, default values

19

```
<?php
function familyName($fname, $year=1950) {
 echo "$fname Mcaffee. Born in $year
\n";
}

familyName("Hege", "1975");
familyName("Stale");
?>
```




Hege Mcaffee. Born in 1975 <br>  
Stale Mcaffee. Born in 1950 <br>

# non-void function, some arguments

20

```
<?php
function sum2($x, $y) {
 $z = $x + 2 * $y;
 return $z;
}
$x = sum2(5, 10) ;
echo "sum2(5,10) = $x
\n";
echo "sum2(5,10) = " . sum2(5, 10) . "
";
?>
```



```
sum2(5,10) = 25

sum2(5,10) = 25

```

# array arguments

21

```
<?php
function my_sum($nums) {
 $s = 0;
 foreach($nums as $a) {
 $s += $a;
 }
 return $s;
}
print my_sum(array(1,5,2,3));
?>
```



11

# array argument as optional arguments

22

```
<?php
function test_params($a, $b, $arrOptionalParams = array()) {
 $c = 'sat'; // default value for c
 $d = 'mat'; // default value for d
 foreach($arrOptionalParams as $key => $value) { ${$key} = $value;}
 echo "$a $b $c on the $d";
}
test_params('The', 'dog', array('c' => 'stood', 'd' => 'donkey'));
test_params('The', 'cat', array('d' => 'donkey'));
test_params('A', 'dog', array('c' => 'stood'));
```



The dog stood on the donkey  
The cat sat on the donkey  
A dog stood on the mat

# Call by reference

23

```
<?php
function mymul(&$a, &$b) {
 $a *=2;
 $b /=2;
 return $a * $b;
}
$x=5;
$y=8;
print "x=$x, y=$y\n";
print "mymul(x,y) = " . mymul($x,$y) .
"\n";
print "x=$x, y=$y\n";
print "mymul(2,3) = " . mymul(2,3) .
"\n";
?>
```



```
x=5, y=8
mymul(x,y) = 40
x=10, y=4
```

PHP Fatal error:  
Only variables  
can be passed by  
reference in  
/home/asidirop/tm  
p/a.php on line  
14

## 4. Include & Require

24

- Οι `include()` & `require()` αντικαθιστούν τον εαυτό τους μ' ένα συγκεκριμένο αρχείο (όπως η `#include` στην C).
- Η `require()` όταν συμπεριληφθεί σε ένα βρόχο θα καλέσει μία φορά το `target` αρχείο.
- Είναι καλό να περιλαμβάνουμε την `include` πάντα σε block (`{...}`) έστω και εάν είναι το μόνο statement.



# require vs. include

25

- Use **require** when the file is required by the application.
- Use **include** when the file is not required and application should continue when file is not found.

# include

26

```
<html>
<body>

<h1>Welcome to my home page!</h1>
<?php include 'noFileExists.php';
echo "I have a $color $car.";
?>

</body>
</html>
```

**Welcome to my home page!**

I have a .

# require

27

```
<html>
<body>

<h1>Welcome to my home page!</h1>
<?php require 'noFileExists.php';
echo "I have a $color $car.";
?>

</body>
</html>
```

**Welcome to my home page!**

# Nesting Files

28

- `require()`, `include()`, `include_once()`, `require_once()` are used to bring in an external file
- This lets you use the same chunk of code in a number of pages, or read other kinds of files into your program
- Be VERY careful of using these anywhere close to user input--if a hacker can specify the file to be included, that file will execute within your script, with whatever rights your script has (`readfile` is a good alternative if you just want the file, but don't need to execute it)
- Yes, remote files can be specified

# Nesting Files

29

- ✓ require “the\_code.php”;
- ✓ require “../../the\_code.php”;
- ✓ require “/the\_path/the\_code.php”;



Αν μεταφέρω το site, ίσως να μην δουλεύουν σωστά.



require “<http://www.example.com/thecode.php.inc>”

Αν το example.com αλλάξει κάτι?



\$file = “<http://www.example.com/thecode.php.inc>”  
require \$file;



\$file = \$\_REQUEST[‘file’];  
require \$file;

- 2 Βασικές τακτικές:

1. ένα αρχικό αρχείο που «καλεί» τις υποσελίδες – το προτιμούμε όταν όλες οι σελίδες θα είναι πανομοιότυπες.
2. πολλές σελίδες που καλούν την βασική – το προτιμούμε όταν οι σελίδες δεν είναι πανομοιότυπες αλλά περιέχουν κοινά στοιχεία.

# ένα αρχικό αρχείο που «καλεί» τις υποσελίδες.

31

## index.php

```
...
<div id='menu'>

</div>
<div id='main'>
<?php
...

switch ($_REQUEST['page']) {
case 'shopinfo': require "pages/shopinfo.php";
 break;
case 'userinfo': require "pages/userinfo.php";
 break;
case 'login': require "pages/login.php";
 break;
default: require "pages/error.php";
}
</div>
<div id='footer'>.....</div>
...
>
```

## pages/shopinfo.php

```
This is the shop info page
...
<?php
print "hello !!!!!!!
";

>
```

## pages/login.php

```
<form action="" ...>
<input name='username'>
<input type='password' name='pass' />
</form>
```

## pages/error.php

```
The page $_REQUEST['page'] is not a
valid page. Goto to Start
Page
```

# πολλές σελίδες που καλούν την βασική.

32

## shopinfo.php

```
<?php
require_once "lib/menu.php";
require_once "lib/footer.php";

print_menu();
?>

<div id='main'>
This is the shop info
</div>

<?php
print_footer();
?>
```

## login.php

```
<?php
require_once "lib/menu.php";
require_once "lib/footer.php";

print_menu();
?>

<div id='main'>
<form action="" ...>
<input name='username'>
<input type='password' name='pass' />
</form>
</div>

<?php
print_footer();
?>
```

## lib/menu.php

```
<?php
function print_menu() {
 print "<div id='menu'>";
 print "...";

 print "</div>";
} ?>
```

## lib/footer.php

```
<?php
function print_footer() {
 print "<div id='footer'>";

 print "</div>";
} ?>
```