COMP 175

System Administration and Security

SCRIPTING LANGUAGES

SCRIPTING SECURITY

So Many Choices: So Little Time

- Perl – designed for report processing
- PHP – purpose designed for web
- Ruby - influenced by Lisp, Smalltalk
- Python – influenced by **ALGOL**, Haskell, Lisp
- Java – Influenced by Ada, Pascal, Smalltalk

Most popular languages are based on ALGOL

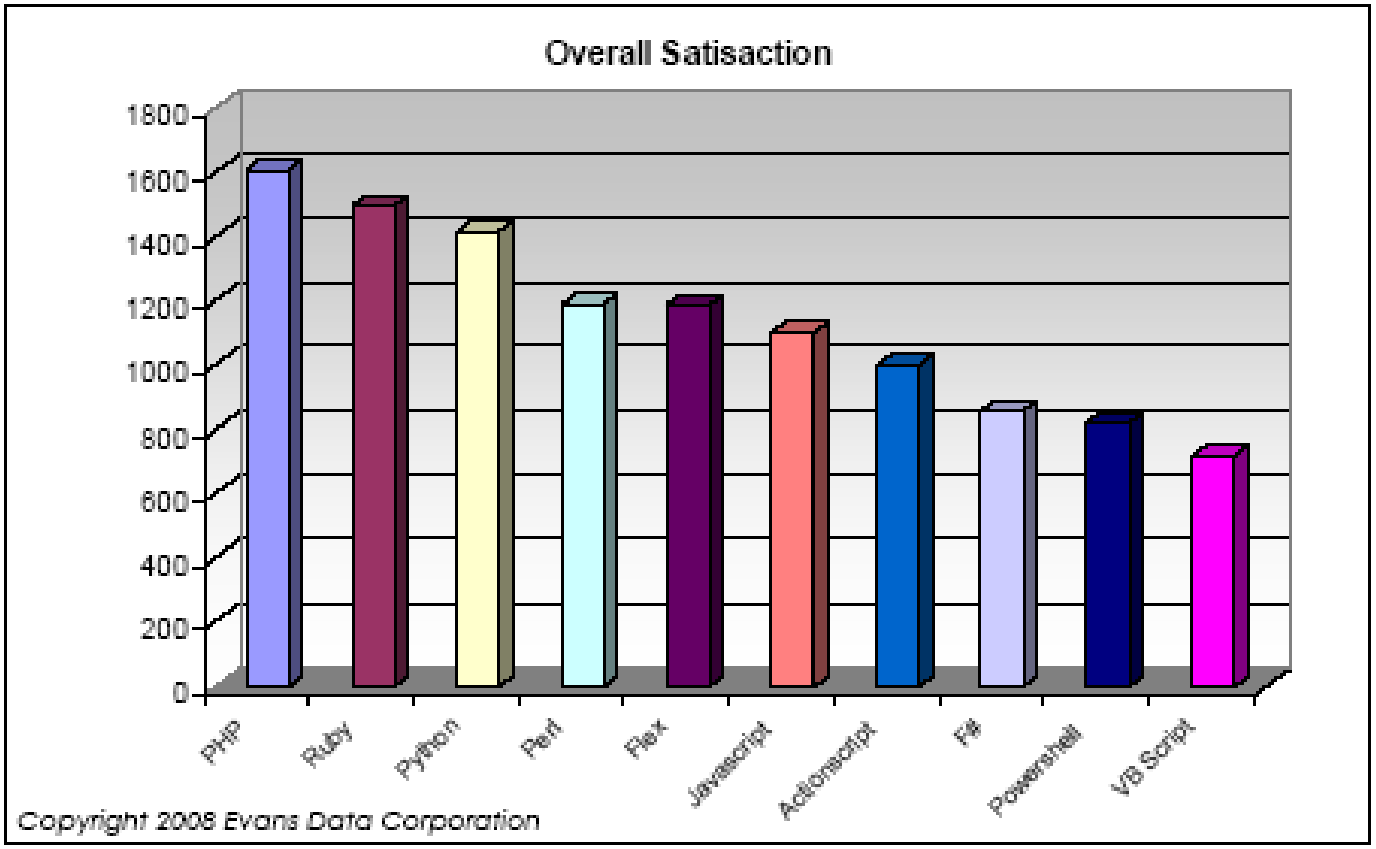
- hierarchical in structure
- environment nesting
- control structure nesting
- dynamic arrays
- reserved words
- user defined data types

So pick one

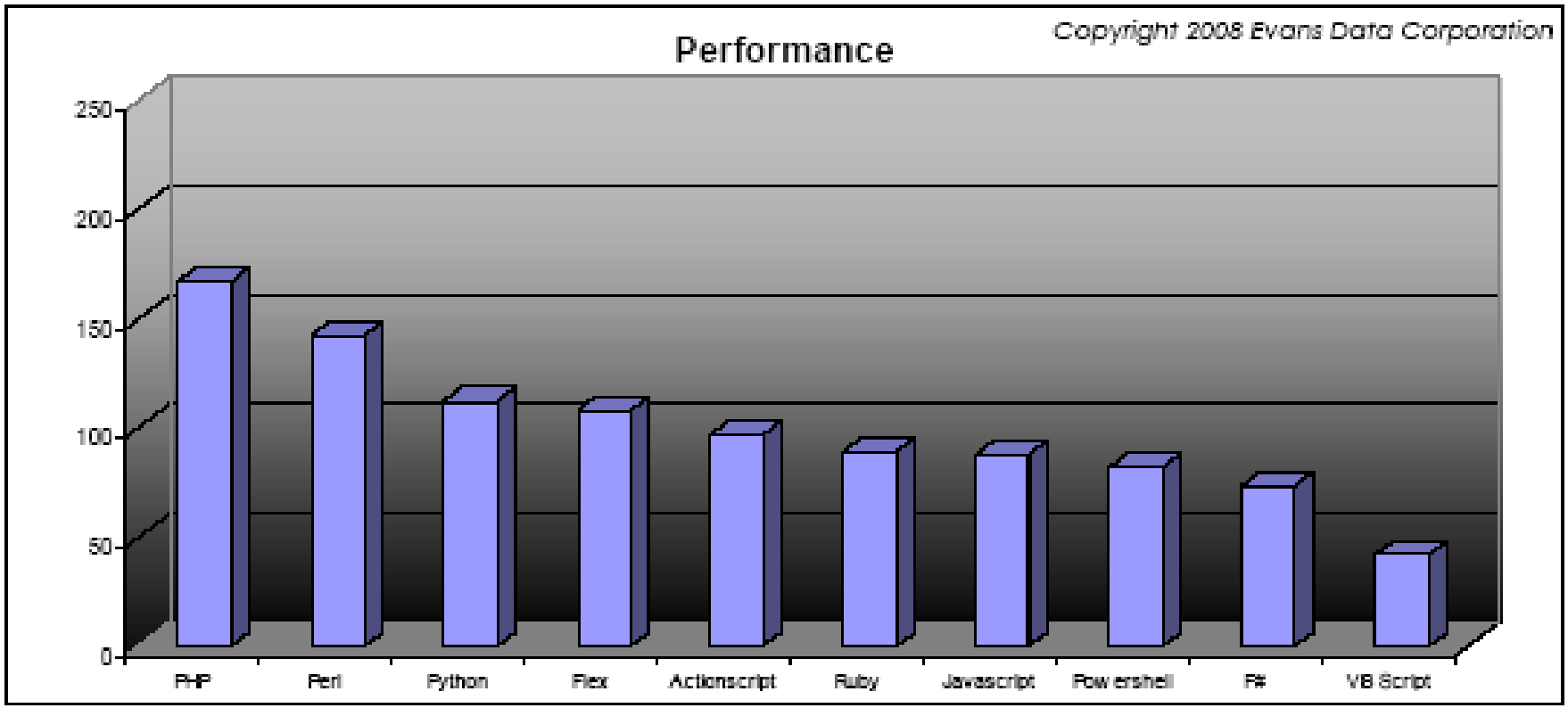
Regular Expressions- xkcd

Comparison: Satisfaction

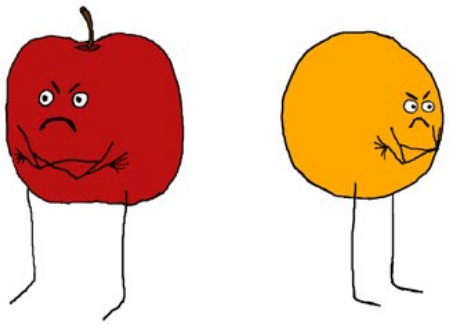
Maintainability / Readability
Cross-platform portability
Memory management
Client side scripting
Exception handling
Availability of tools
Quality of tools
Performance
Extensibility
Ease of Use
Community
Security



Comparison: Performance



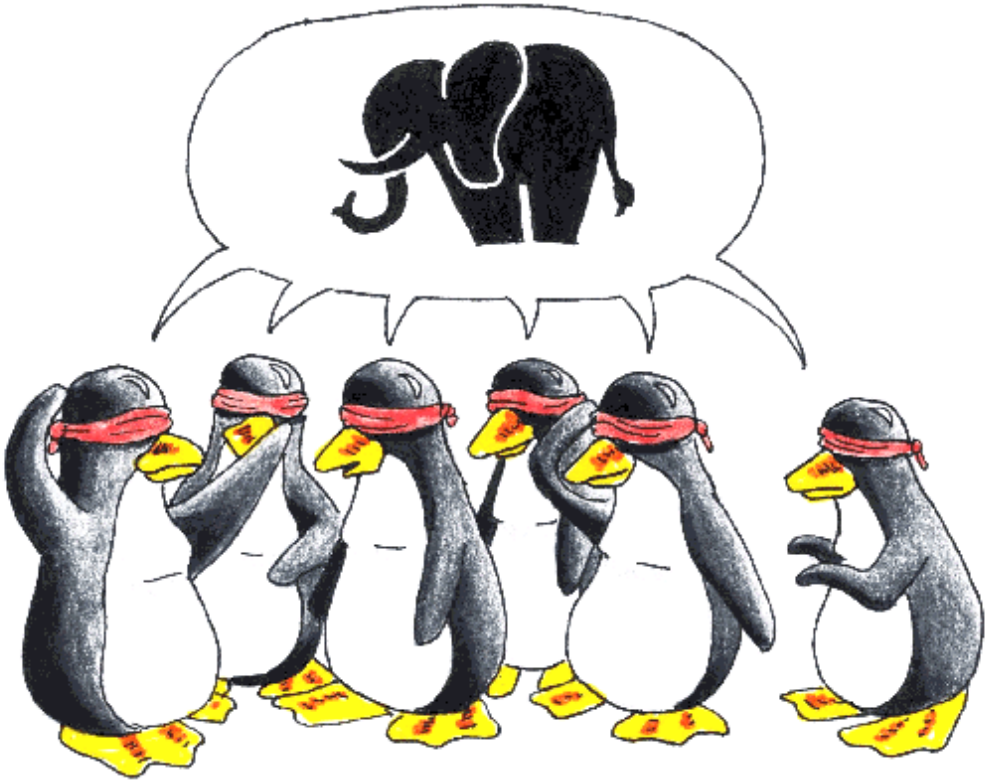
Caveats:
n=400
PHP compared to Powershell?
Client side vs. Server side?
F# ? B♮



www.sketchiesuniverse.com

Objectives

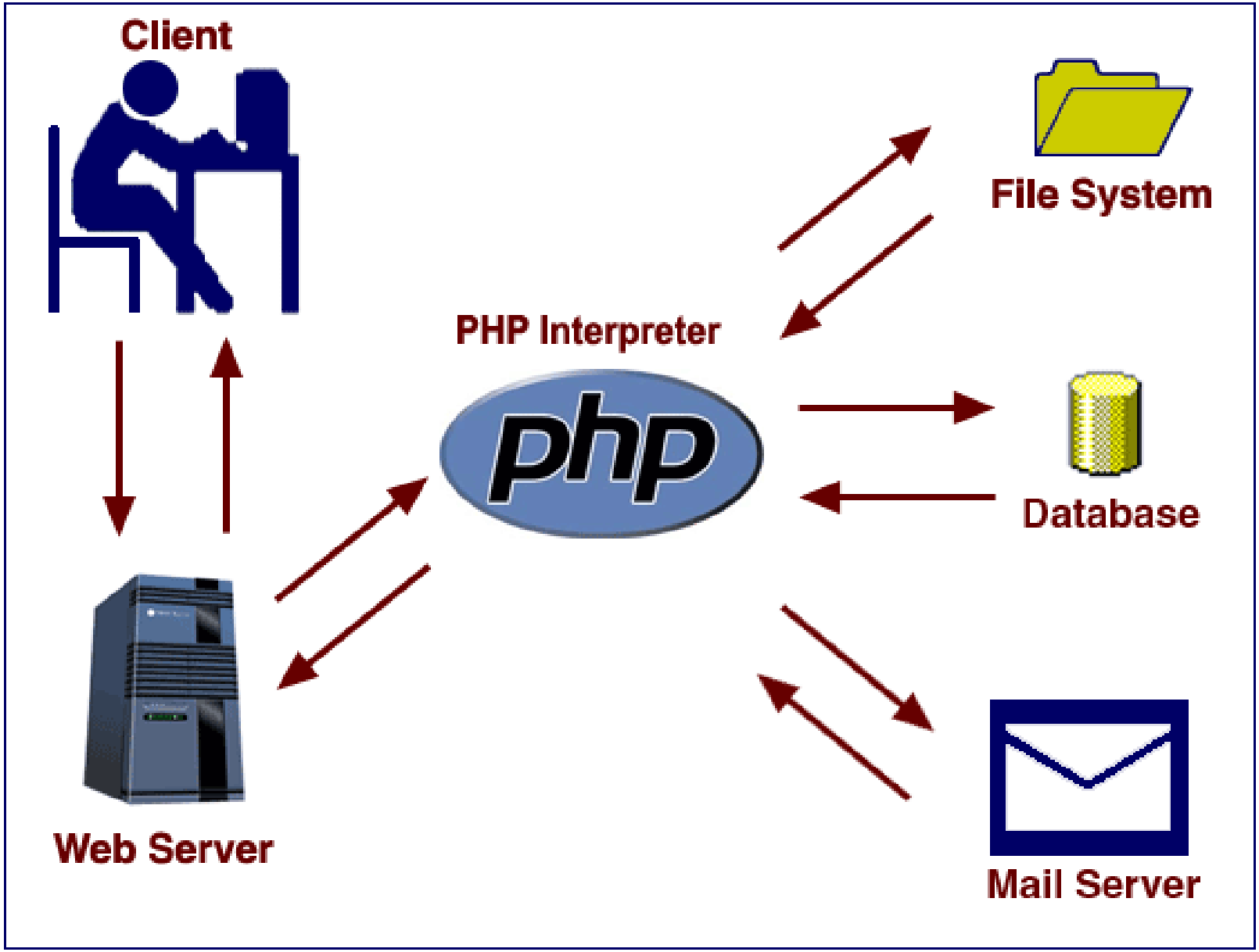
1. General understanding of how PHP works
2. To write a simple PHP page
3. Understand and work with simple PHP variables



PHP Overview

- A general-purpose server-side scripting language
- Originally designed for web development to produce dynamic web pages
  - PHP code embedded into HTML source
  - Interpreted by web server's PHP module
  - Which generates web page document
- Evolved to include a command-line interface
- Standalone graphical applications
- Competes w/ MS Active Server Pages (ASP)

Picture = 1,000 words



Language Features

- PHP language features such as:
  - control structures
  - operators
  - variable types
  - function declaration
  - class/object declaration

- Similar to other compiled or interpreted languages such as C or C++ *or Algol*
- PHP originally stood for Personal Home Page
- Now stands for PHP: Hypertext Preprocessor
  - Which is a recursive backronym

Be Warned


I pity the fool who use numerical operators as boolean evaluators!
Mr. T
AutoPuzzle.com

Very Brief History

- 1995 – Released - Personal Home Page tools
- 1997 – PHP: Hypertext Preprocessor PHP 3
- 1999 – Rewritten – Zend Engine
- 2000 – PHP 4
- 2002 – CLI added
- 2004 – PHP 5
- 2015 – Oct 29 7.0.0 RC 6 released *development*
- 2022 - Nov 24 8.2 to be released
- Unicode issues are delaying PHP 6? *Abandoned*
- Present on ~75% of all web servers

Unicode?

- www.unicode.org/charts/

Translations

ما هي الشفرة الموحدة "يونيكود" ؟ in Arabic

ইউনিকোড কী? in Bangla

什麼是Unicode(統一碼/標準萬國碼)? in Trad'l Chinese

Qu'est ce qu'Unicode? in French

Was ist Unicode? in German

Τι είναι το Unicode; in Greek (Monotonic)

?(Unicode) מה זה יוניקוד in Hebrew

यूनिकोड क्या है? in Hindi

Cos'è Unicode? in Italian

ユニコードとは何か? in Japanese

유니코드에 대해? in Korean

O que é Unicode? in Portuguese

Что такое Unicode? in Russian

¿Qué es Unicode? in Spanish

யூனிக்கோடு என்றால் என்ன?, in Tamil

PHP

now: cloud.google.com/php

- www.php.net The PHP project site
- www.zend.com Prebuilt PHP application stack
- http://www.apachefriends.org
  - Prebuilt package of Apache, MySQL, PHP, Perl
  - Plus assorted tools
  - Versions for Linux, Windows, OS X, Solaris

Cloud Use

- When PHP is installed and used in cloud environments, software development kits (SDKs) are provided for using cloud-specific features. For example:

- Amazon Web Services provides the AWS SDK for PHP
- Windows Azure can be used with the Windows Azure SDK for PHP

PHP Code Blocks

- Files typically end in .php .cgi
  - Determined by webserver configuration
- PHP code block is embedded within HTML
- When server encounters PHP tags it switches from HTML to PHP mode

- Four ways to embed the PHP code

```
<?php echo(“code”); ?>
<? echo(“code”); ?>
<SCRIPT Language=‘php’> echo(“code”); </SCRIPT>
<% echo(“code”); %>
```

PHP Scripts

- HTML tag <?php ?>
- End lines with a ";"
- Server processes embedded script
- Source is not visible, results passed to browser

```
<P>
<?php $myvar = "Hello Class!";
echo $myvar;
?>
</P>
```

Embedded PHP

- PHP embedded inside of the HTML

```
<body>
<P>
<?php $myvar = "Hello Class!";
echo $myvar;
?>
</P>
</body>
```

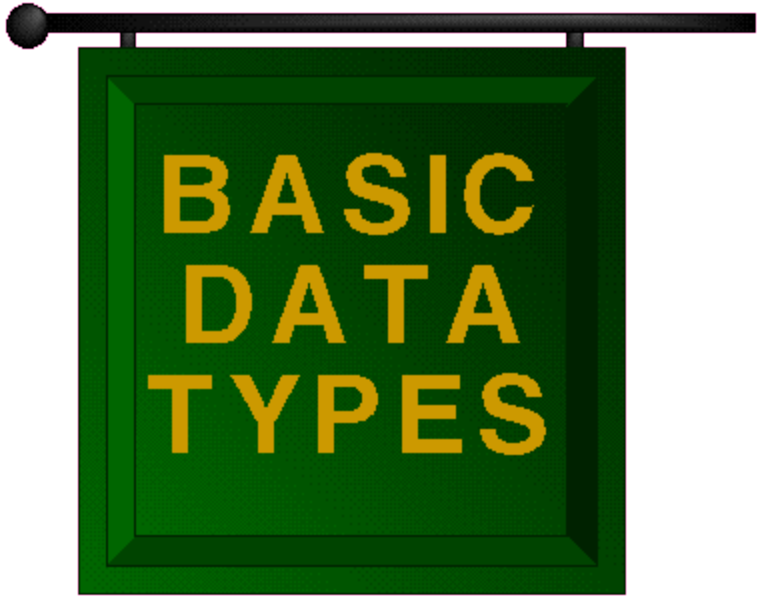
Calling HTML from PHP

- HTML called from within the PHP script

```
<?php
echo "<html><head>";
echo "<title>Yo</title>";
…
…
?>
```

PHP Data Types

- Three basic data types
  - Integer
  - Double
  - String
- Additional data types
  - Array
  - Object



- PHP is an untyped language
  - variables type can change on the fly

Constants

- Values that never change
- Defined in PHP by using the define() function
  - define(“CNL”, “Cisco Networking Lab”)
- defined() function says whether the constant exists or not

Variables

- Typed by context (but one can force type)
  - so it's loose
- Begin with "$"
- Assigned by value
  - $foo = "Bob"; $bar = $foo;
- Assigned by reference, this links vars
  - $bar = &$foo;

- gettype, settype, isset, unset, is_int, intval

Operators

- Arithmetic (+, -, *, /, %) and String (.)
- Assignment (=) and combined assignment

```
$a = 3;
$a += 5; // sets $a to 8;
$b = "Hello ";
$b .= "There!"; // sets $b to "Hello There!";
```

- Bitwise (&, |, ^, ~, <<, >>)

```
$a ^ $b  (Xor: Bits set in $a or $b but not both)
~ $a (Not: Bits set in $a are not set, and vice versa)
```

- Comparison (==, ===, !=, !==, <, >, <=, >=)

Operators

- Error Control (@)
- When this precedes a command, errors generated are ignored (allows custom messages)

- Execution (` is similar to shell_exec() function)
- You can pass a string to the shell for execution:
  $output = `ls -al`;
  $output = shell_exec("ls -al");

- This is one reason to be careful about user set variables!

Operators

- Logical

$a and $b	And	True if both $a and $b are true
$a or $b	Or	True if either $a or $b is true
$a xor $b	Xor	True if either $a or $b is true, but not both
! $a	Not	True if $a is not true.
$a && $b	And	True if both $a and $b are true.
$a || $b	Or	True if either $a or $b is true.

- The two ands and ors have different precedence rules, "and" and "or" are lower precedence than "&&" and "||"
- Use parentheses to resolve precedence problems
  - or just to be clearer

Control Structures

- if, else, elseif
- while, do-while, ~~doo-wop~~
- for, foreach
- break, continue, switch
- require, include, require_once, include_once


```
for($i=0;$i < 10;$++i) {
echo("the value is :". $i);
}
```

Alternative Syntax

```
for($i=0;$i < 10;$++i) :
// html code goes here
endfor;
```

PHP Statements

IF statement

```
if (<condition>) {
//php code goes here
}
else {
//php code goes here
}
```

Alternative Syntax

```
if(<condition>) :
//html code goes here
else :
//html code goes here
endif;
```

Looping

For loop

```
for($i=0;$i < 10;$++i) {
echo(“the value is :”. $i);
    }
```

Alternative Syntax

```
for($i=0;$i < 10;$++i) :
// html code goes here
endfor;
```

Switch

Examples comparing elseif and switch

```
if ($i == 0) {
    echo "i equals 0";
} elseif ($i == 1) {
    echo "i equals 1";
} elseif ($i == 2) {
    echo "i equals 2";
}
```

```
switch ($i) {
case 0:
    echo "i equals 0";
    break;
case 1:
    echo "i equals 1";
    break;
case 2:
    echo "i equals 2";
    break;
}
```

External Files

- To include an external file:
  - `require()`
  - `include()`
  - `include_once()`
  - `require_once()`
  - `readfile()`
    - remote files can be specified

- Use with caution close to user input--if a hacker can specify the file to be included, that file will execute within your script *and permissions*

Useful String Functions

```
str_replace()
trim(), ltrim(), rtrim()
implode(), explode()
addslashes(), stripslashes()
htmlentities(), html_entity_decode()
htmlspecialchars()
strip_tags()
```

Useful String Functions

htmlspecialchars

- Convert special characters to HTML entities
- Usage: string htmlspecialchars (string string [, int quote_style [, string charset]])
- Converts only:

  & => &

  " => " (small subset)

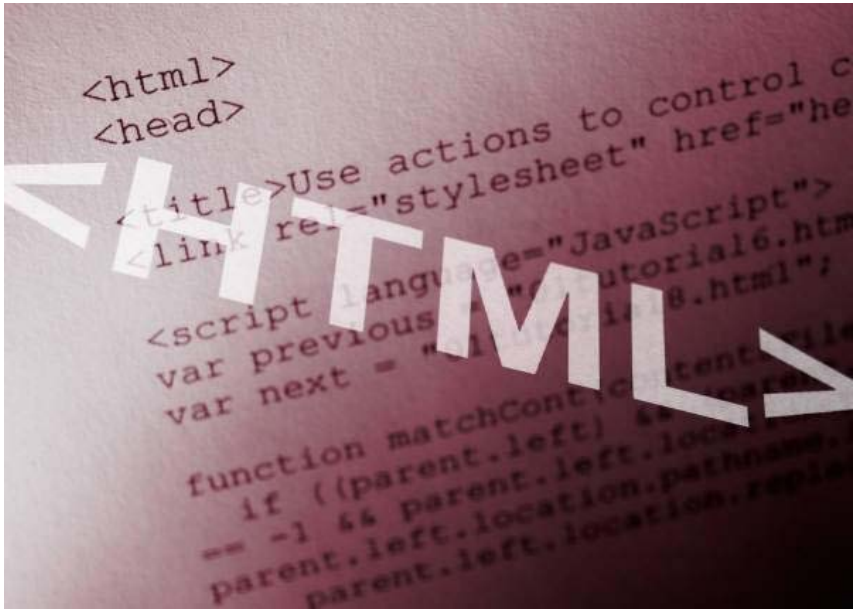
htmlentities

- Convert all applicable characters to HTML entities
- Converts every char to HTML applicable chars

Useful String Functions

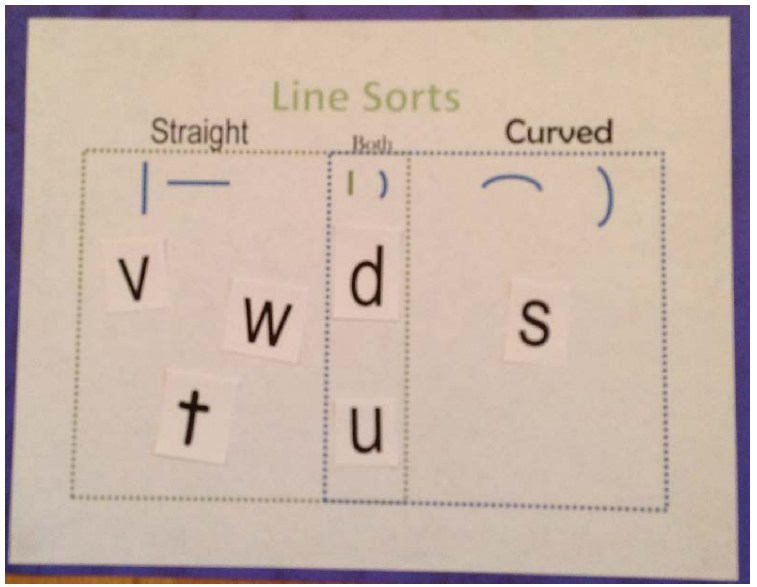
strip_tags

- Strip HTML and PHP tags from a string
- Can tell this function to ignore HTML that you consider harmless, or that you want to include



Sorting Functions

- `sort()` : sorts elements in numeric and alphabetical order
- `rsort()` : sorts elements in the reverse order
- `asort()` : sorts elements in array without changing the indices
- `ksort()` : sorts arrays by key



Application Program Interfaces

- PHP comes with myriad of options
  - i.e. supports several APIs and interfaces to other programming tools such as:
- Database connectivity
- LDAP
- XML
- Mail protocols such as IMAP, SMTP
- Image functions
- etc....

Image Creation & Modification

- PHP offers powerful set of functions for generating and manipulating images
- Uses the GD library for most image functionality
- GD library used for generating 2-D graphics
- PHP API's provides functions for:
  - Rendering regular geometric figures
  - Modifying images
  - Manipulate text, font and color
  - Pixels in the image

....creating the images on the fly

Mailing Functions

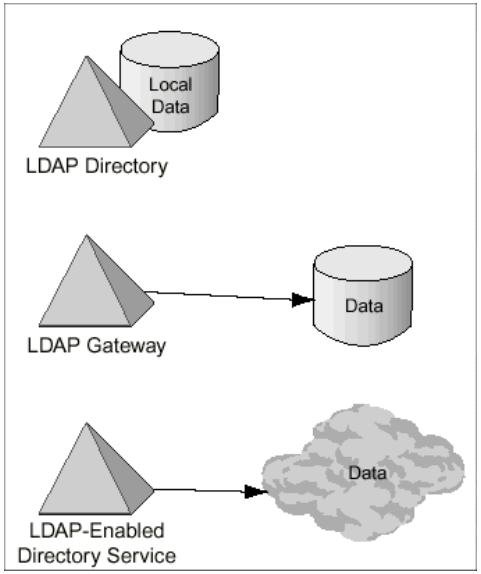
- Sending E-Mail
  - `mail()`
  - Used to send simple text messages
  - Depends on the local mail delivery system
    - Uses SMTP
  - Accepts e-mail for every recipient
  - Delivers e-mail
- Receiving E-Mail
  - PHP works well with IMAP
  - Rich set of support functions
  - `imap_open, imap_delete, imap_close, imap_mail_copy, imap_mail_move` etc…

PHP's Built-In Functions

```
$functions_list = get_defined_functions();
foreach($functions_list['internal'] as $function_name) {
    echo($function_name) . "\n";
}
```

LDAP Support

- PHP provides LDAP API's to create LDAP clients
- Provides transparent access to backend LDAP directory servers
- Examples:
  - Web based e-mail client
  - Telephone Directory



- LDAP can be used to implement Sign On (SSO)

Database Extensions

- XML Support
- Database Support
  - Oracle
  - PostgreSQL
  - mSQL
  - MySQL
  - IBM DB2
  - From CUBRID to Tokyo Tyrant

PHP Information

```
<html>
 <head>
   <title>PHP</title>
 </head>
 <body>
   <?php phpinfo(); ?>
 </body>
</html>
```

PHP Version 5.3.2

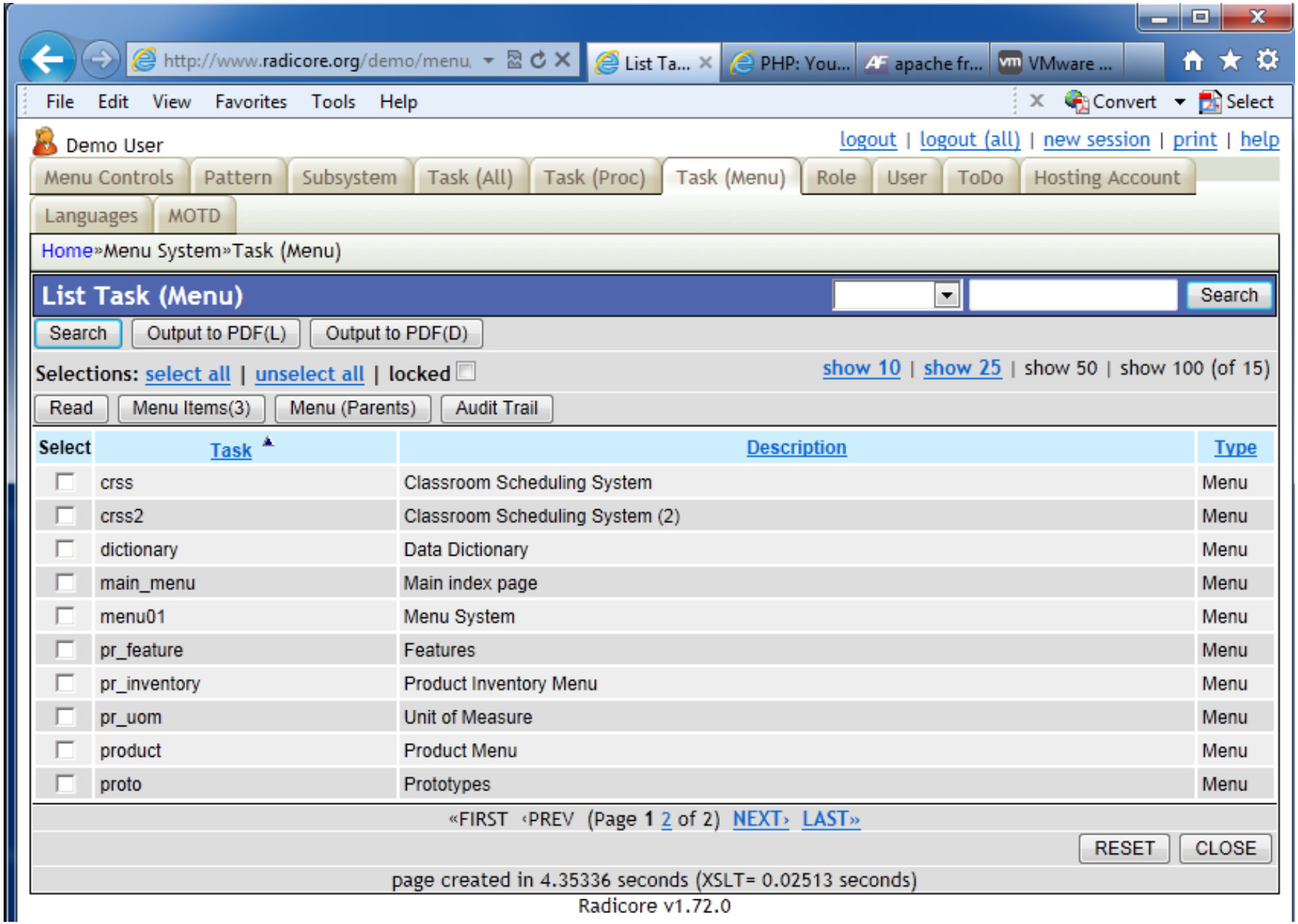
System	Linux rich 2.6.32.16-141.fc12.x86_64 #1 SMP Wed Jul 7 04:49:59 UTC 2010 x86_64
Build Date	Apr 27 2010 17:56:21
Configure Command	'./configure' '--build=x86_64-redhat-linux-gnu' '--host=x86_64-redhat-linux-gnu' '--target=x86_64-redhat-linux-gnu' '--program-prefix=' '--prefix=/usr' '--exec-prefix=/usr' '--bindir=/usr/bin' '--sbindir=/usr/sbin' '--sysconfdir=/etc' '--datadir=/usr/share' '--includedir=/usr/include' '--libdir=/usr/lib64' '--libexecdir=/usr/libexec' '--localstatedir=/var' '--sharedstatedir=/var/lib' '--mandir=/usr/share/man' '--infodir=/usr/share/info' '--cache-file=../config.cache' '--with-libdir=lib64' '--with-config-file-path=/etc' '--with-config-file-scan-dir=/etc/php.d' '--disable-debug' '--with-pic' '--disable-rpath' '--without-pear' '--with-bz2' '--with-exec-dir=/usr/bin' '--with-freetype-dir=/usr' '--with-png-dir=/usr' '--with-xpm-dir=/usr' '--enable-gd-native-ttf' '--with-t1lib=/usr' '--without-gdbm' '--with-gettext' '--with-gmp' '--with-iconv' '--with-jpeg-dir=/usr' '--with-openssl' '--with-pcre-regex=/usr' '--with-zlib' '--with-layout=GNU' '--enable-exif' '--enable-ftp' '--enable-magic-quotes' '--enable-sockets' '--enable-sysvsem' '--enable-sysvshm' '--enable-sysvmsg' '--with-kerberos' '--enable-ucd-snmp-hack' '--enable-shmop' '--enable-calendar' '--without-sqlite' '--with-libxml-dir=/usr' '--enable-xml' '--with-system-tzdata' '--with-apxs2=/usr/sbin/apxs' '--without-mysql' '--without-gd' '--disable-dom' '--disable-dba' '--without-unixODBC' '--disable-pdo' '--disable-xmlreader' '--disable-xmlwriter' '--without-sqlite3' '--disable-phar' '--disable-fileinfo' '--disable-json' '--without-pspell' '--disable-wddx' '--without-curl' '--disable-posix' '--disable-sysvmsg' '--disable-sysvshm' '--disable-sysvsem'
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc
Loaded Configuration File	/etc/php.ini
Scan this dir for additional .ini files	/etc/php.d
Additional .ini files parsed	/etc/php.d/curl.ini, /etc/php.d/fileinfo.ini, /etc/php.d/gd.ini, /etc/php.d/json.ini, /etc/php.d/mbstring.ini, /etc/php.d/mcrypt.ini, /etc/php.d/mysql.ini, /etc/php.d

Common Uses

- Guestbook
- E-mail form
- Hit counter
- Image Gallery
- Surveys
- Tests

- PHPmotion video sharing CMS (like YouTube)
- SugarCRM Customer Relationship Manager
- Nanoweb HTTP server

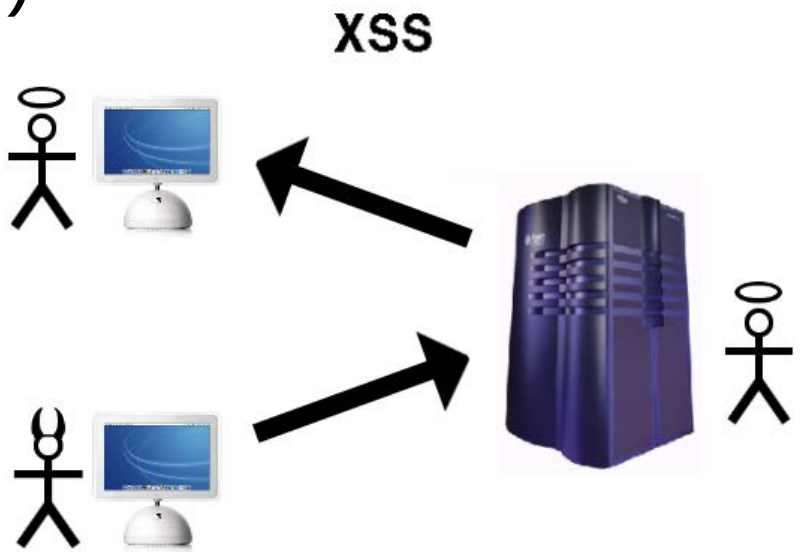
RAD in PHP


http://www.radicore.org/demo/menu
List Ta...
PHP: You...
apache fr...
VMware ...
File Edit View Favorites Tools Help
Convert
Select
Demo User
logout | logout (all) | new session | print | help
Menu Controls
Pattern
Subsystem
Task (All)
Task (Proc)
Task (Menu)
Role
User
ToDo
Hosting Account
Languages
MOTD
Home»Menu System»Task (Menu)
List Task (Menu)
Search
Search
Output to PDF(L)
Output to PDF(D)
Selections: select all | unselect all | locked
show 10 | show 25 | show 50 | show 100 (of 15)
Read
Menu Items(3)
Menu (Parents)
Audit Trail
Select
Task
Description
Type
crss
Classroom Scheduling System
Menu
crss2
Classroom Scheduling System (2)
Menu
dictionary
Data Dictionary
Menu
main_menu
Main index page
Menu
menu01
Menu System
Menu
pr_feature
Features
Menu
pr_inventory
Product Inventory Menu
Menu
pr_uom
Unit of Measure
Menu
product
Product Menu
Menu
proto
Prototypes
Menu
«FIRST ‹PREV (Page 1 2 of 2) NEXT› LAST»
RESET
CLOSE
page created in 4.35336 seconds (XSLT= 0.02513 seconds)
Radicore v1.72.0

PHP Security

Types of PHP Attacks

- Command execution and/or writing to filesystem
- SQL injection
- Session Hijacking
  - Cross Site Request Forgeries (CSRF)
  - Session reading/predicting
  - Cross Site Scripting (xss)



PHP Security

- Insecure php.ini

- display_errors = Off
- log_errors = On
- error_reporting = E_ALL
- register_globals = Off

URL= index.php?administrator=xyz

```
<?php
if (isset($administrator))
{
    $authorized = true;
}
?>
```

Develop Best Practices

- Develop with security and production in mind
- Form strict policies concerning how data is sanitized - and at what stage.
- $_GET, $_COOKIE, $_POST always sanitized based on where it's going not where it came from

  Mysql = mysql_real_escape_string()
  Postgres = pg_escape_string ()
  To browser = htmlentities () or strip_tags()
  To Shell = escapeshellcmd()

Develop Best Practices

Securing Includes

- Place them outside of document root

```
ini_set ("include_path",".:/home/user/libs");
```

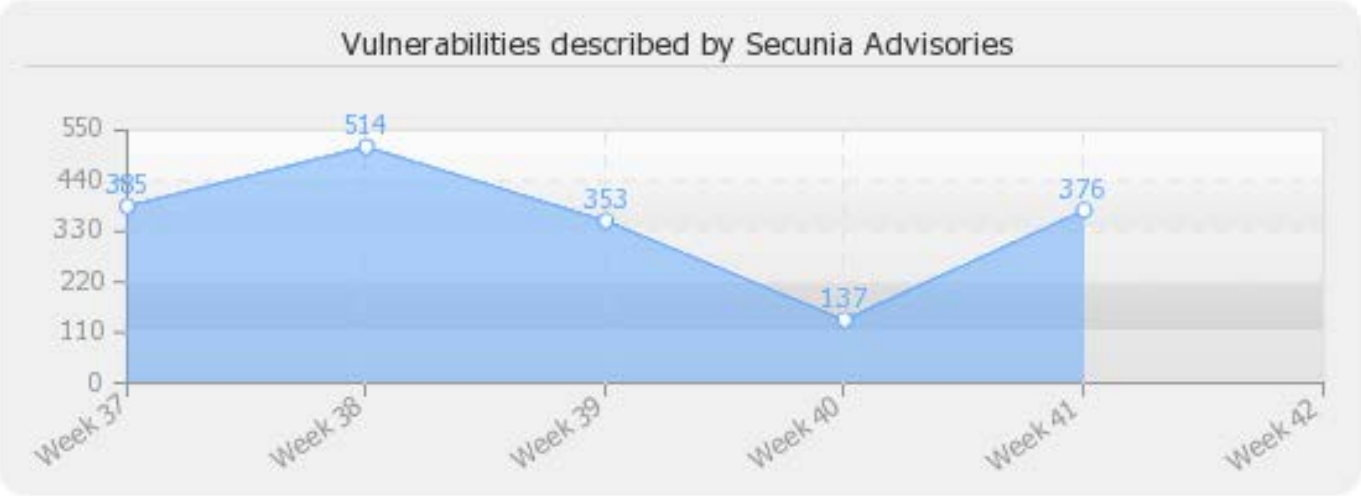
- But, if they must be placed in root...
  - End them in .php, so source not revealed
  - Example: database.inc.php

```
<Files ~ "\.inc$">
    Order allow,deny
    Deny from all
</Files>
```

Secure PHP Applications

- When installing free web applications always be aware of security advisories
  - phpMyAdmin — google: phpmyadmin exploit
  - Gallery — google: gallery exploit
  - phpnuke — *giyf*
- Signup for US Cert, Distro advisories
- http://secunia.com/advisories/

Statistics - Past 5 Weeks



Secunia Advisories

2nd Dec, 2011 Advisories

- Hillstone Software HS TFTP Library Denial of Service Vulnerability
- Final Draft Script File Parsing Buffer Overflow Vulnerabilities
- GOM Player Playlist Parsing Buffer Overflow Vulnerability
- WikkaWiki Multiple Vulnerabilities
- JBoss AS Admin. Console Cross-Site Scripting and Request Forgery
- SUSE update for seamonkey
- Ipswitch TFTP Server Directory Traversal Vulnerability
- HP Device Access Manager Unspecified Code Execution Vulnerability
- Ariadne URL Cross-Site Scripting Vulnerability
- Hero Framework "month" Cross-Site Scripting Vulnerability
- SugarCRM Two SQL Injection Vulnerabilities
- Perl PAR Module Insecure Temporary File Security Issue
- Perl PAR-Packer Module Insecure Temporary File Security Issue
- HP-UX update for BIND

Scripting

The Dark Side

Scripting

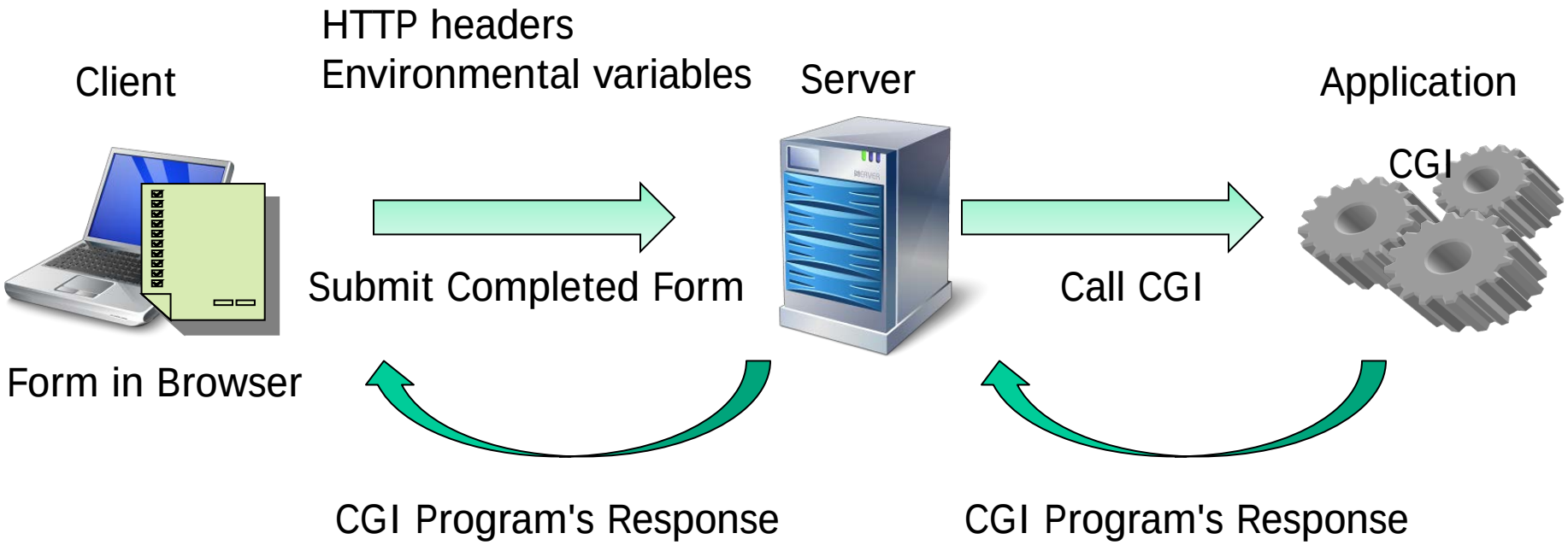
- Scripting makes the web richly interactive by accepting and responding to user entry
- Also introduces huge security problems
  - Were there not enough security issues already?
  - Web pages with embedded hostile scripts
  - HTML formatted e-mail
  - Attachments
  - Apps/applets
- Security issues introduced at both:
  - Server
  - Client

Common Gateway Interface

- Common Gateway Interface (CGI)
- Method for web server software to delegate the generation of web content to executable files
- Servers, browsers, programs can exchange data
- CGI Scripts (any language) use this standard
- Make the web dynamic and interactive
- CGI programs can be:
  - C++ and compiled to an executable
  - PERL, shell, etc. and stored as scripts
    - Interpreted on server or browser at run-time

CGI Scripts

- Server – CGI Interface
- Reside on the server side and executed as needed
- An open gateway allowing anyone anywhere to run an executable or their own apps on the server



CGI Scripts

- Due to programming complexity and lack of program development discipline, errors introduced into programs are difficult to find, especially in non-compiled scripts
- Imported scripts may introduce hostile code into the server
- The script can propagate into other server applications and other servers
- Resource intensive
- Remote execution possible

Compromised Scripts

- May allow an attacker access to the system's password file for decryption
- May allow mailing of a map of the system which gives the attacker more time offline to analyze the system's vulnerabilities
- May allow starting a login server on a high port and telneting in
- May allow a distributed denial of service attack against the server or other servers
- May allow erasing or altering the server's log files

Malicious Scripts

- Malicious code provided by one client for another client: This can happen, for example, in sites that host discussion groups where one client can embed malicious HTML tags in a message intended for another client.
- Scripting languages such as PERL, PHP, and the Bourne shell pass information needed to perform tasks through command line statements which are then executed by an interpreter. This can be very dangerous.
- Clients may use special characters in input strings to confuse other clients, servers, or scripts.

Build Secure Web Apps

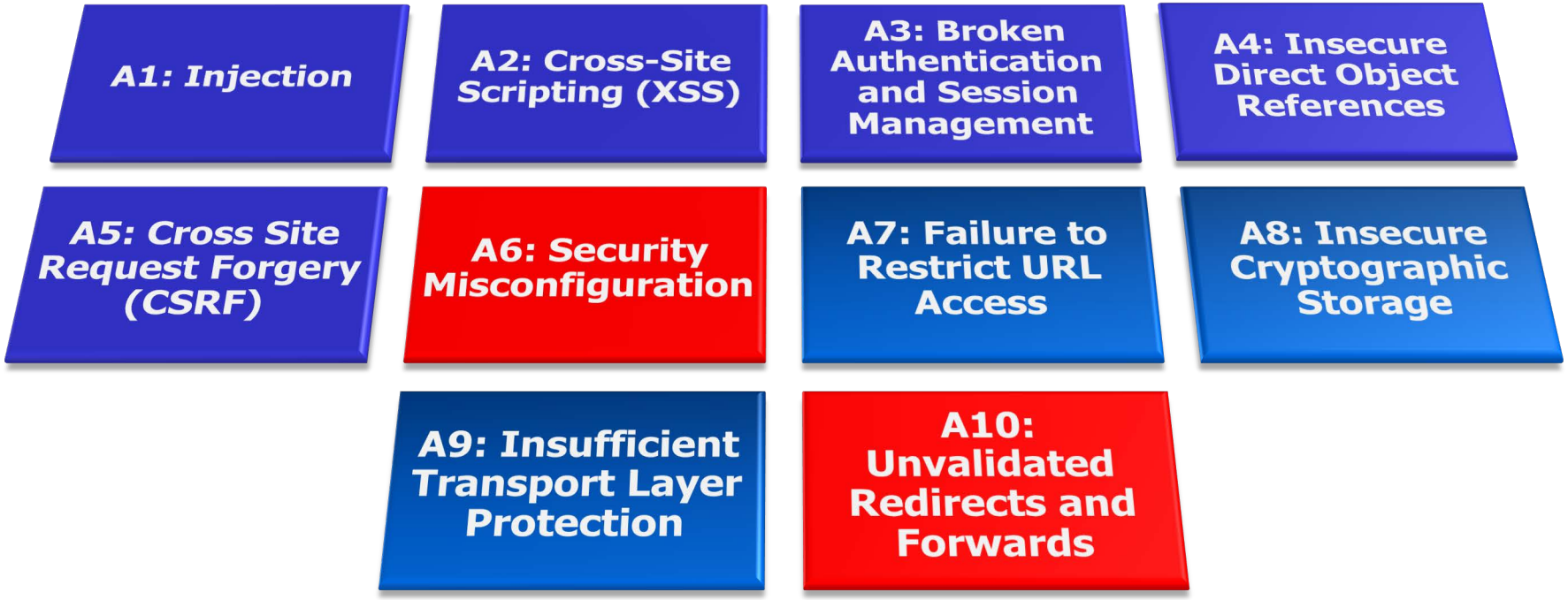
To avoid these problems:

- Open Web Application Security Project OWASP
- Guide 200+ pages **(READING ASSIGNMENT)**

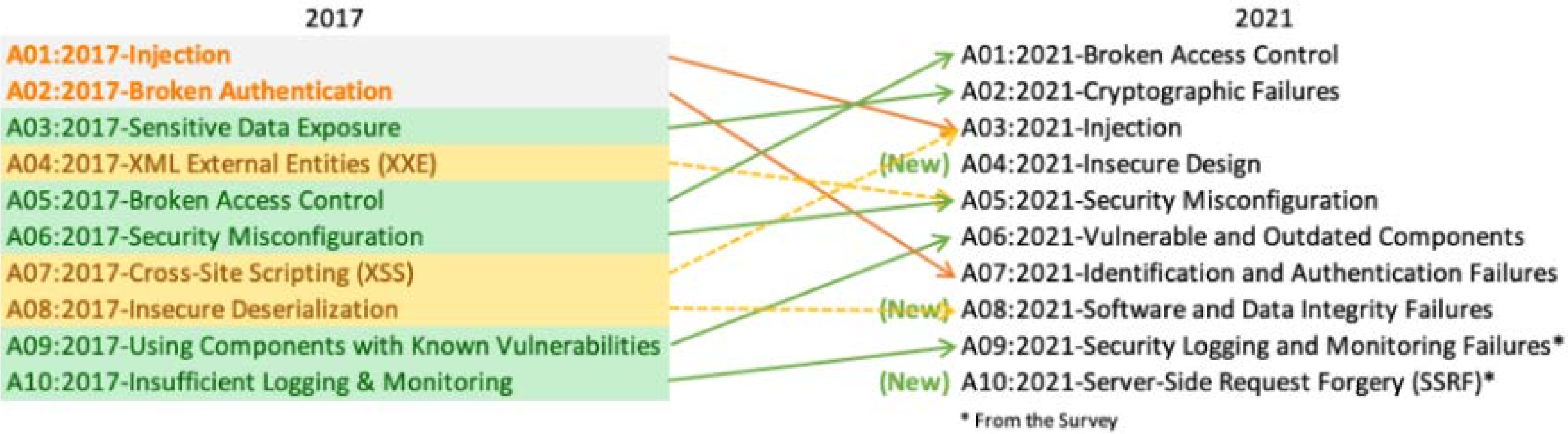
- Remember that "they are out to get you"

Web Application Risks

OWASP Top 10 (2010)



OWASP Top 10


2017
A01:2017-Injection
A02:2017-Broken Authentication
A03:2017-Sensitive Data Exposure
A04:2017-XML External Entities (XXE)
A05:2017-Broken Access Control
A06:2017-Security Misconfiguration
A07:2017-Cross-Site Scripting (XSS)
A08:2017-Insecure Deserialization
A09:2017-Using Components with Known Vulnerabilities
A10:2017-Insufficient Logging & Monitoring
2021
A01:2021-Broken Access Control
A02:2021-Cryptographic Failures
A03:2021-Injection
(New) A04:2021-Insecure Design
A05:2021-Security Misconfiguration
A06:2021-Vulnerable and Outdated Components
A07:2021-Identification and Authentication Failures
(New) A08:2021-Software and Data Integrity Failures
A09:2021-Security Logging and Monitoring Failures*
(New) A10:2021-Server-Side Request Forgery (SSRF)*
* From the Survey