

SECURE CODING ESSENTIALS

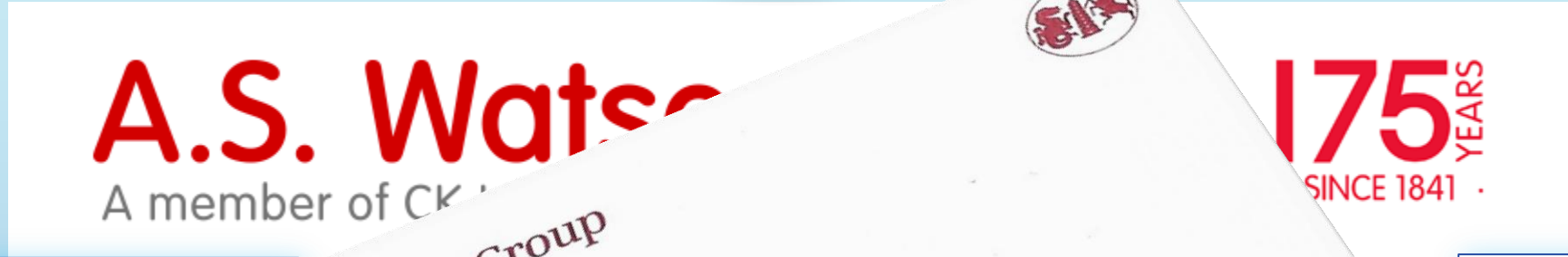
*DEFENDING YOUR WEB
APPLICATION AGAINST
CYBER ATTACKS*

*ROB AUGUSTINUS
30 MARCH 2017*



AGENDA

- *Intro - A.S. Watson and Me*
- *Why this Presentation ?*
- *Security Architecture*
- *Secure Code Design Principles*
- *The Process - Security in the SDLC*
- *Mission Accomplished ?*



Ports and related services

Structure

Energy and Telecom



270,000 people in over 50 countries across the world



A.S. WATSON RETAIL BRANDS EUROPE



NOT ALL OUR WEBSITE VISITORS ARE FRIENDLY...



AGENDA

- *Intro – A.S. Watson and Me*
- ***Why this Presentation?***
- *Security Architecture*
- *Secure Code Design Principles*
- *The Process - Security in the SDLC*
- *Mission Accomplished ?*

SHOULD DEVELOPERS BOTHER ABOUT SECURITY ?

Write clean, testable code against the infrastructure components of your choice and accomplish any task – without re-inventing the wheel.

**Build a better website
in less than an hour.**

Choose a template. Upload pictures. Edit text. Done!

Secure

CakePHP comes with built-in tools for input validation, CSRF protection, Form tampering protection, SQL injection prevention, and XSS prevention, helping you keep your application **safe & secure..**

No design skills needed

Choose from hundreds of gorgeous website templates or build your own design using HTML and CSS.

YES, BECAUSE...

Over 750,000 websites were breached

Arranged By: Severity Descending

🔒 126 Security Issues (296 variants) for 'My Application'

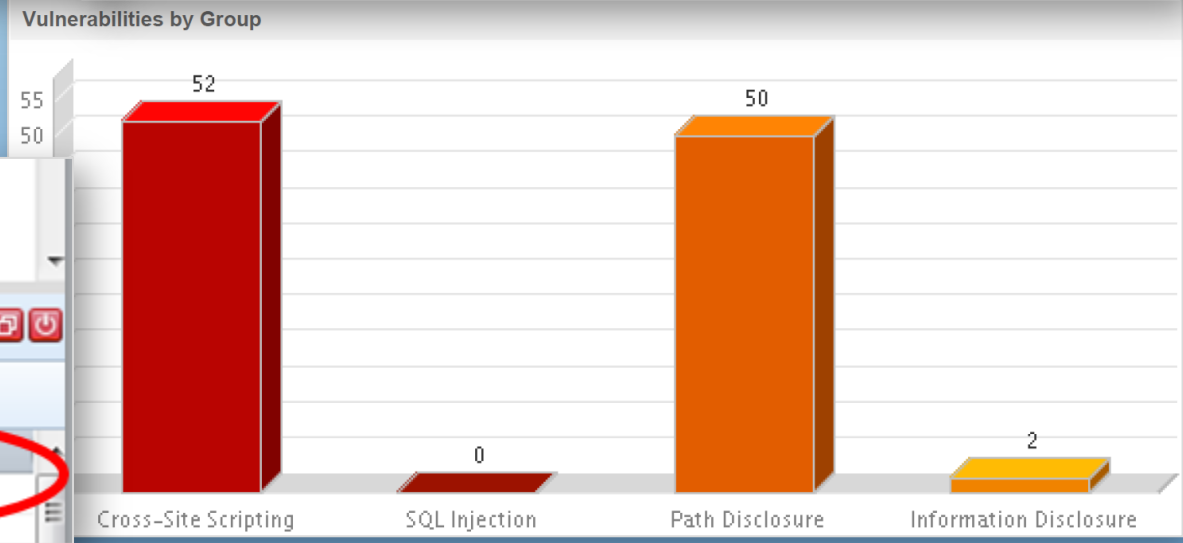
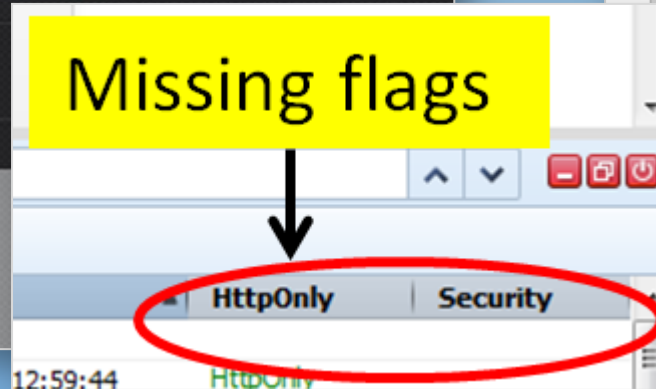
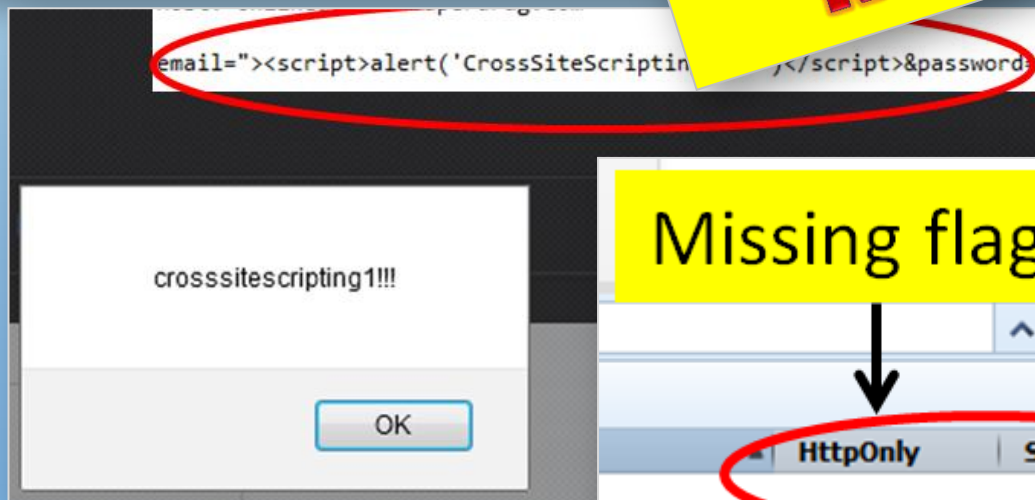
- ▶ Cross-Site Scripting (4)
- ▶ Link Injection (facilitates Cross-Sit
- ▶ Missing Secure Attribute in E
- ▶ Phishing Through Frames (1)

PENTEST RESULTS

Issue Types 10

Issue Type

- Blind SQL Injection
- Cross-Site Scripting
- Missing Secure Attribute in Encrypted Session (SSL) Cookie



UNEVEN BATTLE

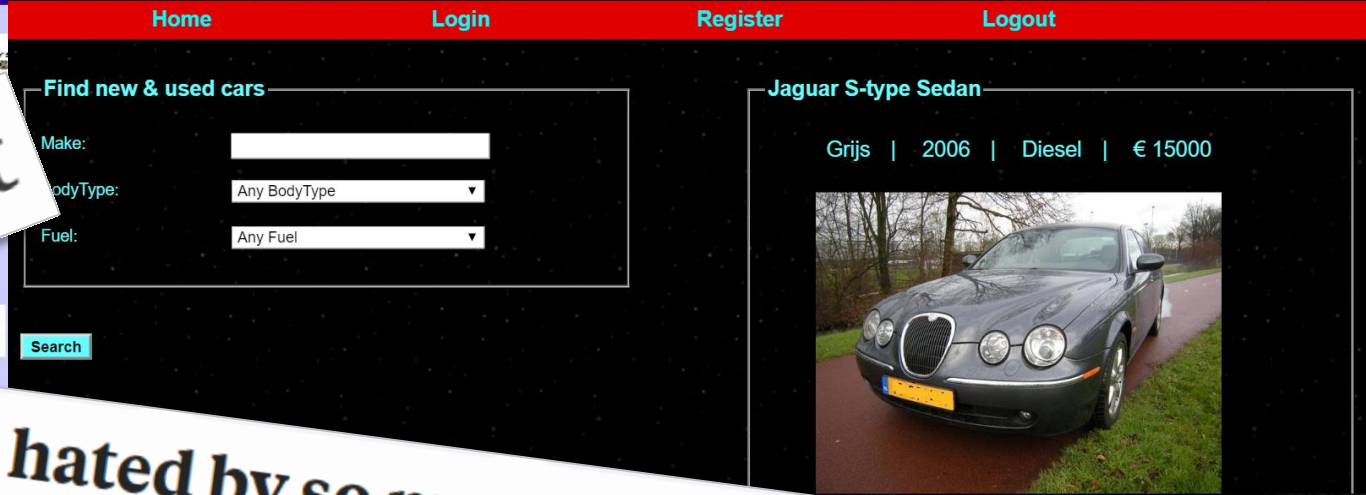


FOCUSED ON USE CASES
TIGHT DEADLINES
MANY SECURITY RISKS
TO THINK OF



FOCUSED ON ABUSE CASES
PLENTY OF TIME
FINDING ONE FLAW IS ENOUGH
TO HIT THE JACKPOT

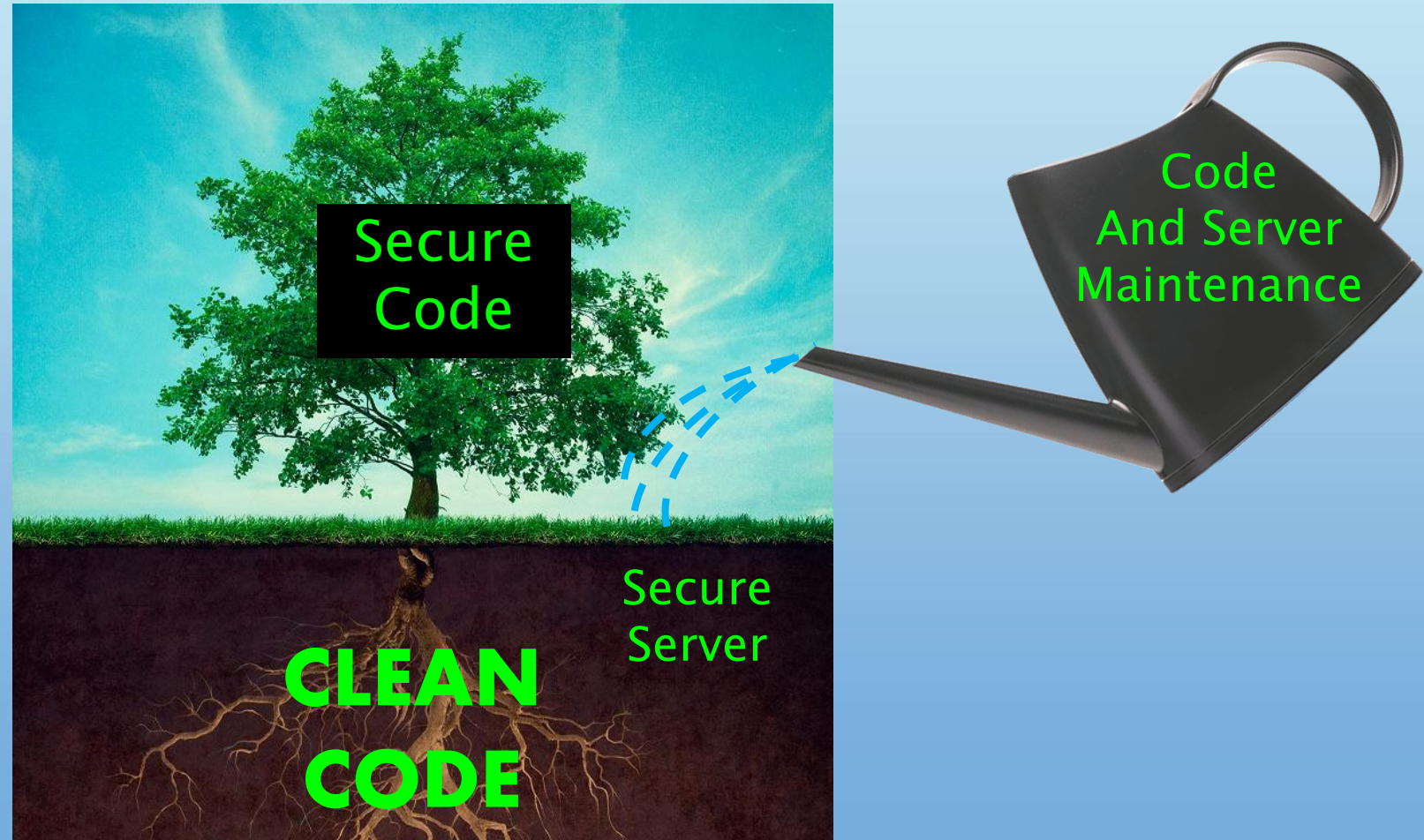
HOW TO WRITE IN/SECURE CODE



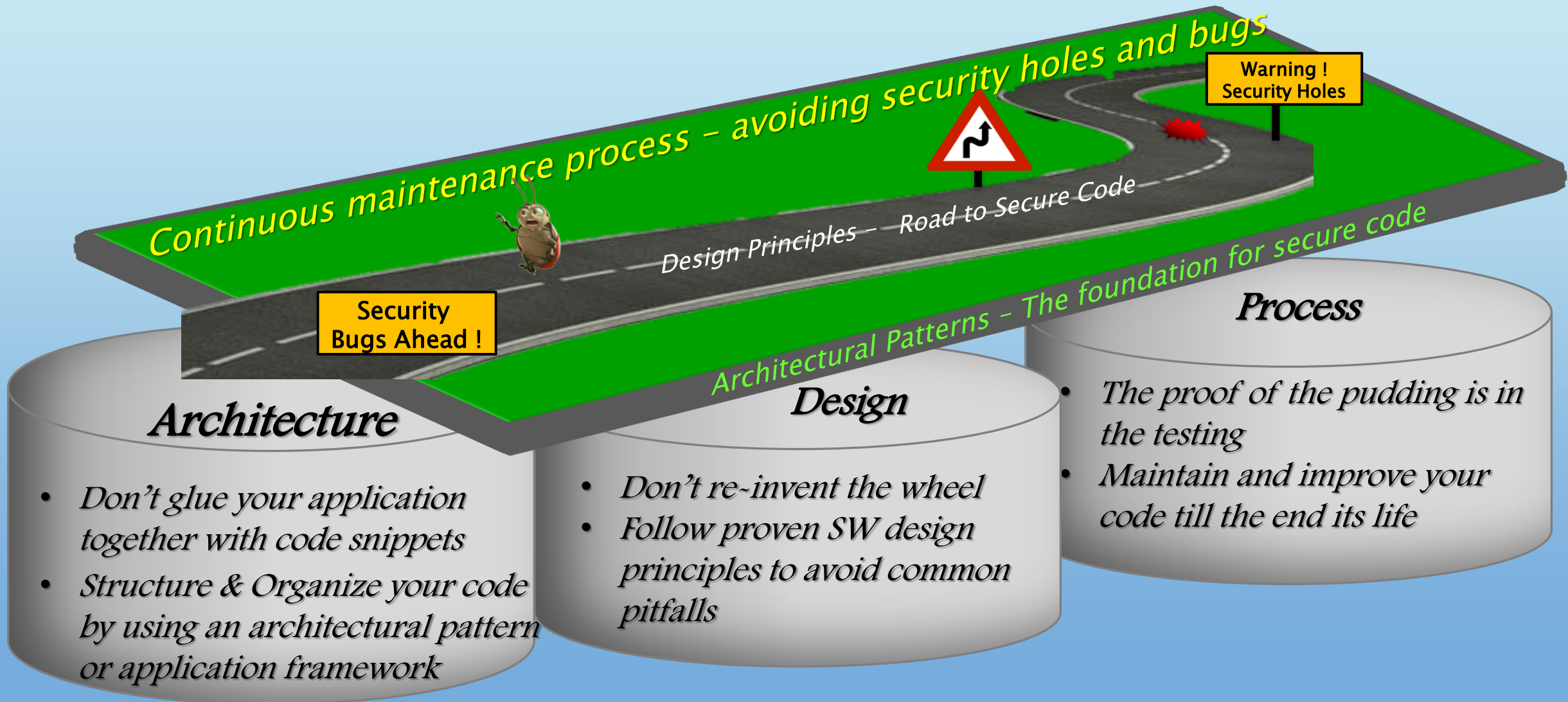
PHP is insecure by default

Why is PHP hated by so many developers?

SECURE CODE NEEDS A GOOD FOUNDATION



THE 3 PILLARS OF CLEAN CODE

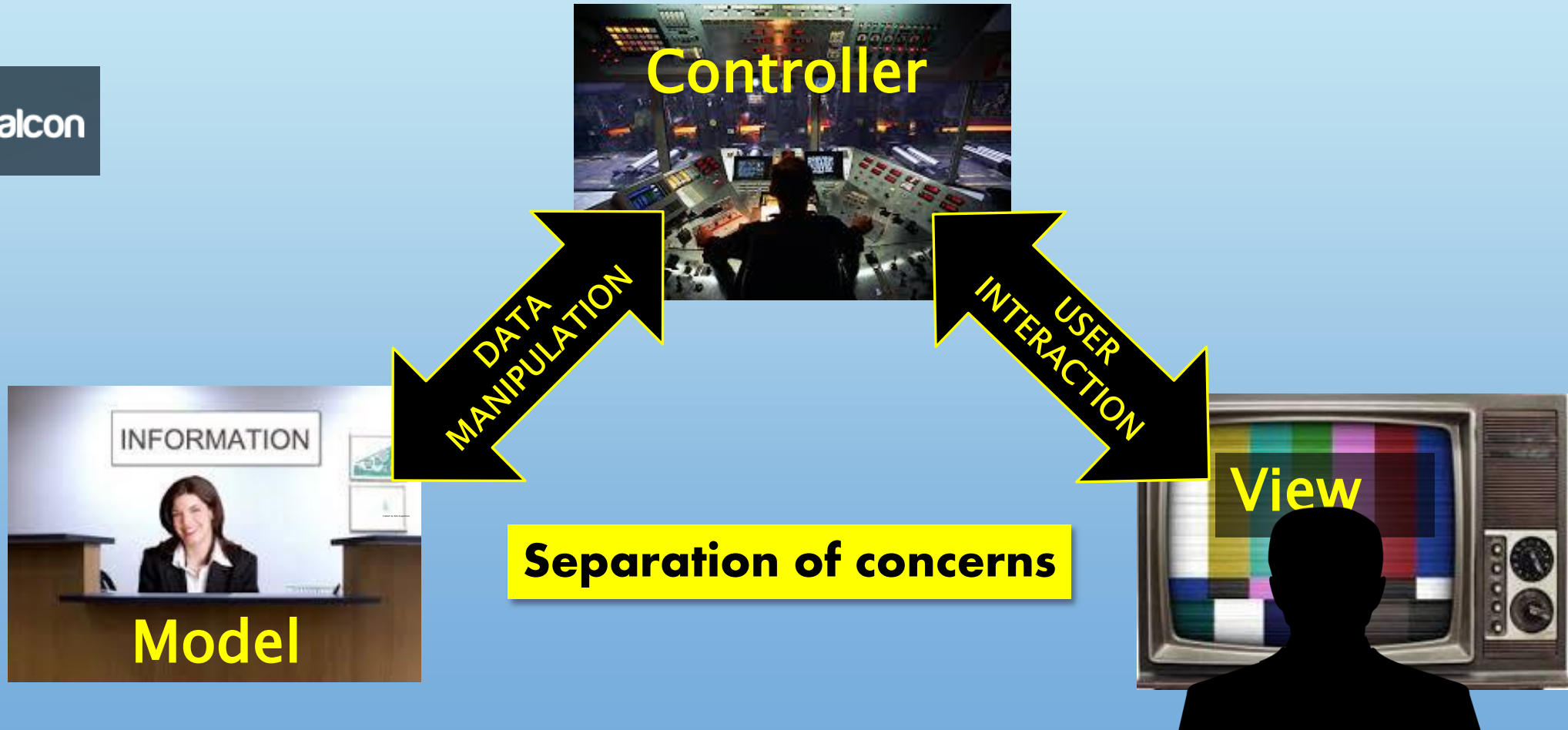


AGENDA

- *Intro – A.S. Watson and Me*
- *Why this Presentation ?*
- ***Security Architecture***
- *Secure Code Design Principles*
- *The Process - Security in the SDLC*
- *Mission Accomplished ?*

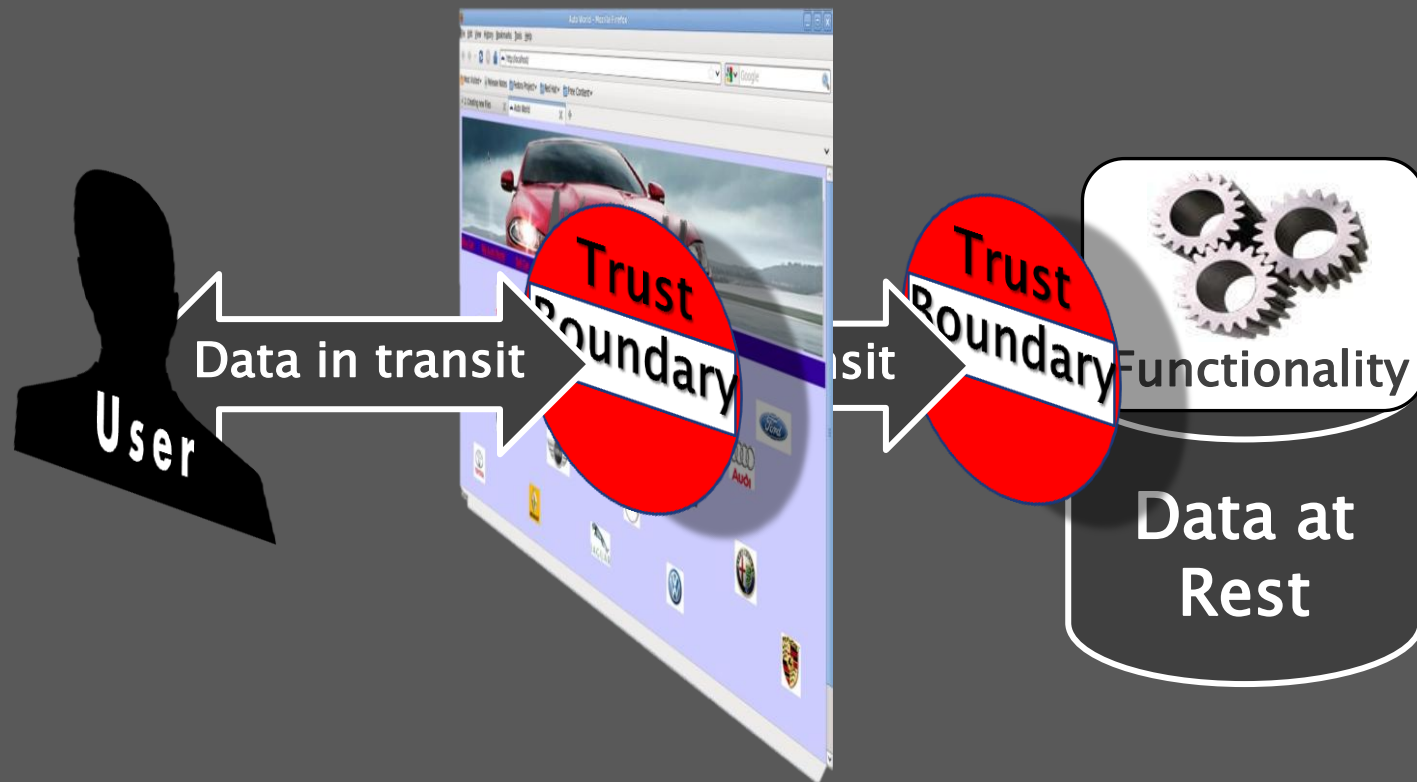


MVC ARCHITECTURE PATTERN



SECURITY ARCHITECTURE THREAT RISK MODELING

Application Modeling:



Identify Threats:

Data Integrity

- *Is your data clean and can it be trusted ?*

Data Access

- *Can users access only the data they should be able to access ?*

Data Protection

- *Is sensitive data well protected against unauthorized access ?*

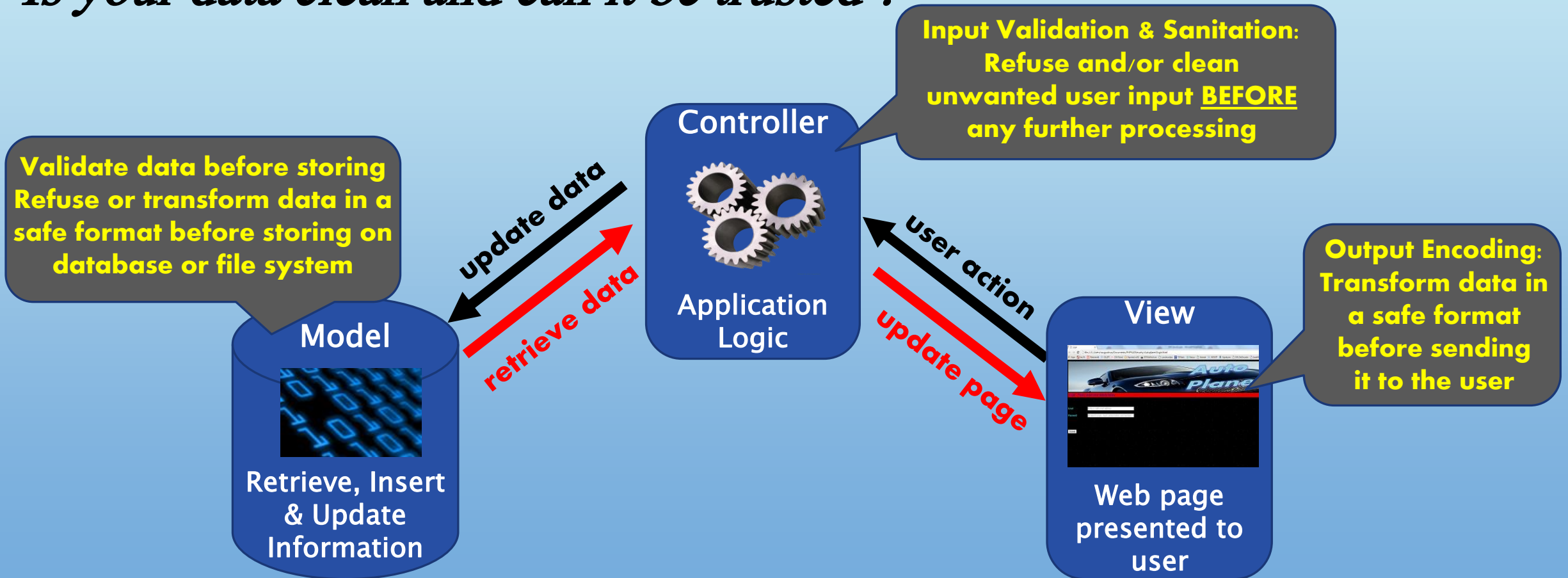
Functionality

- *Are features and application logic being used as expected ?*

SECURITY ARCHITECTURE

DATA INTEGRITY

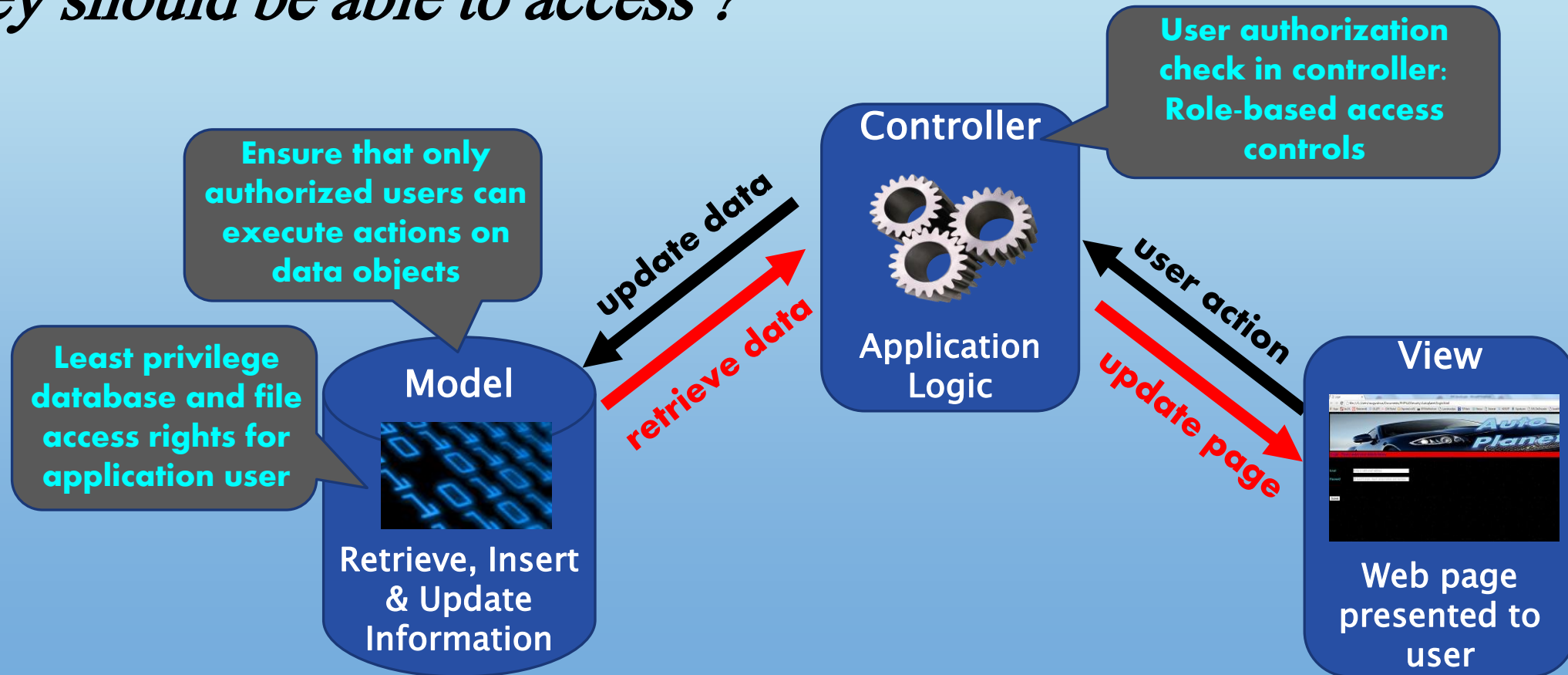
Is your data clean and can it be trusted ?



SECURITY ARCHITECTURE

DATA ACCESS

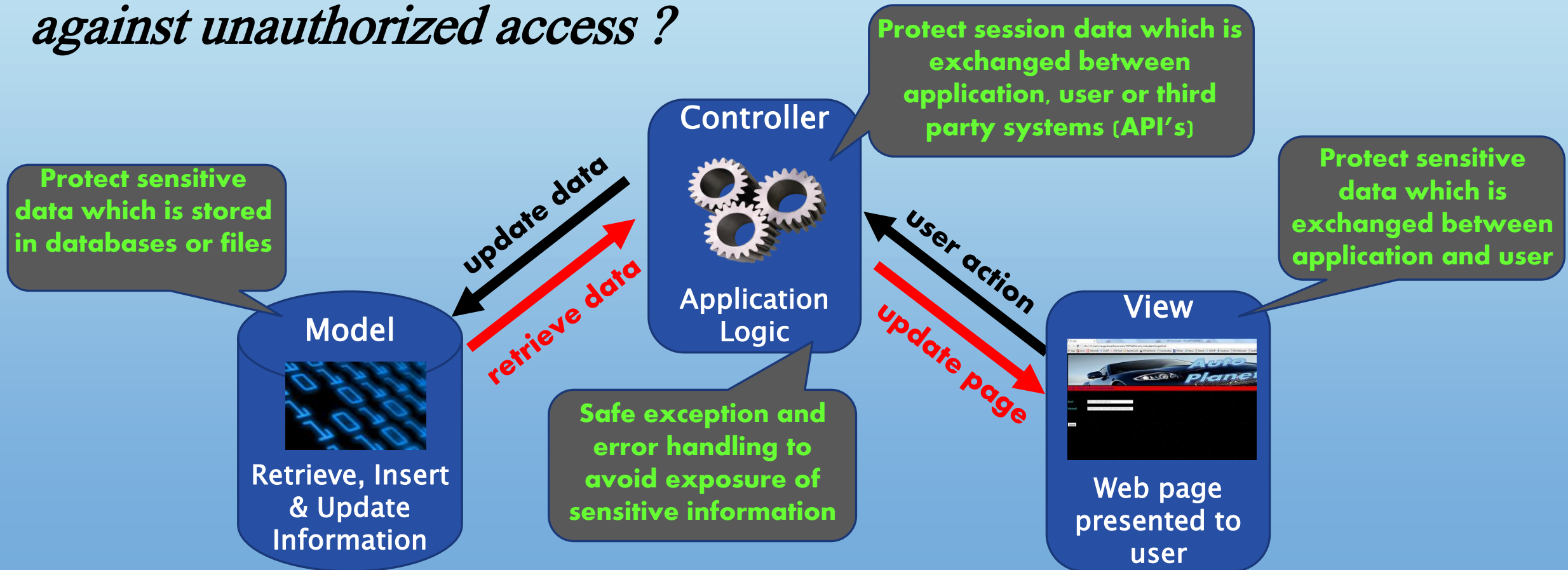
Can users access only the data they should be able to access ?



SECURITY ARCHITECTURE

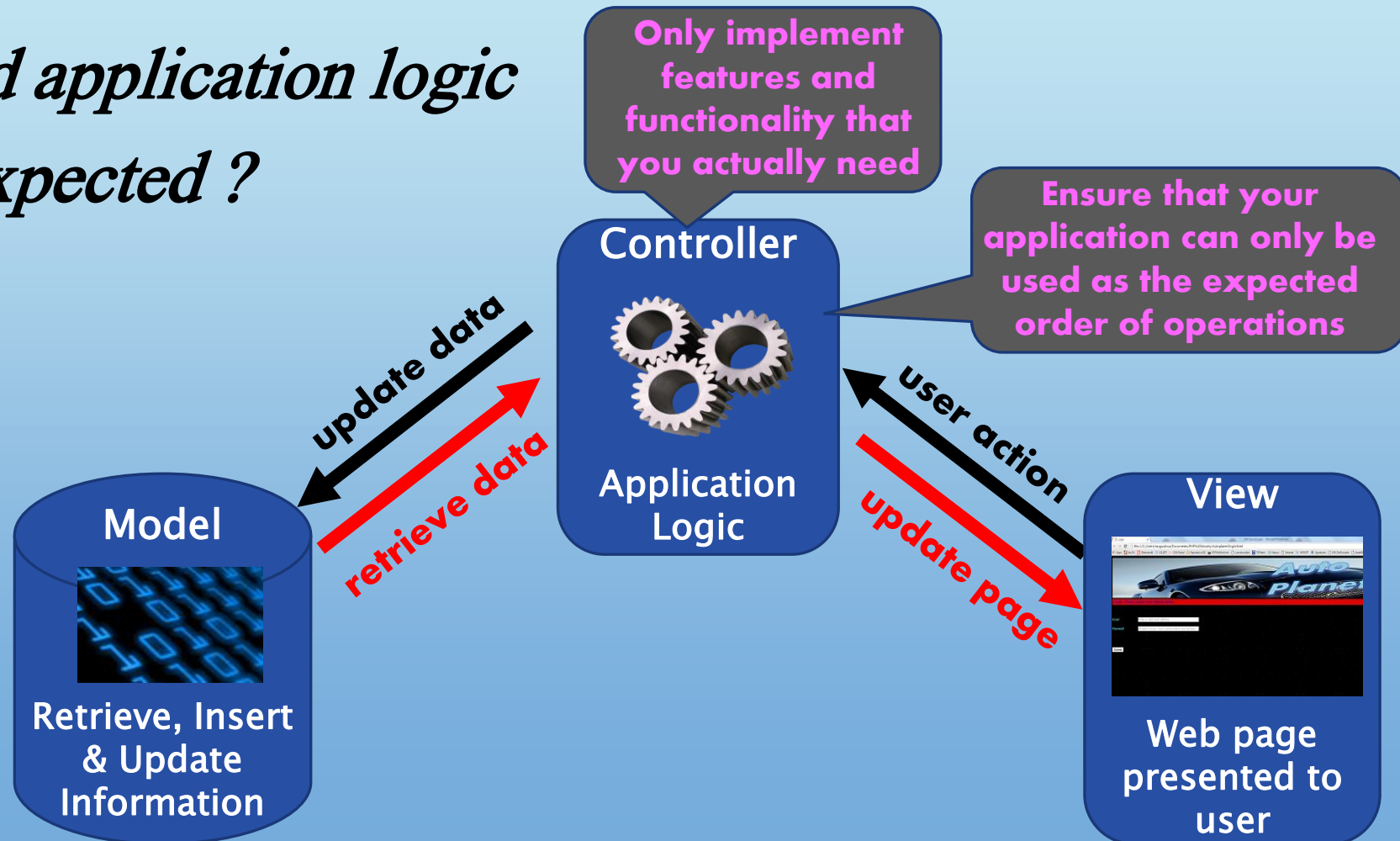
DATA PROTECTION

*Is sensitive data well protected
against unauthorized access ?*



SECURITY ARCHITECTURE FUNCTIONALITY

*Are features and application logic
being used as expected ?*



AGENDA

- *Intro – A.S. Watson and Me*
- *Why this Presentation ?*
- *Security Architecture*
- ***Secure Code Design Principles***
- *The Process - Security in the SDLC*
- *Mission Accomplished ?*

SOFTWARE DESIGN PRINCIPLES

**KEEP
IT
SIMPLE
STUPID**



SECURE CODING DESIGN PRINCIPLES

Data Integrity

- *Never, ever, trust the user*

Data Access

- *Control access to protected resources*
- *User rights: Less is more !*

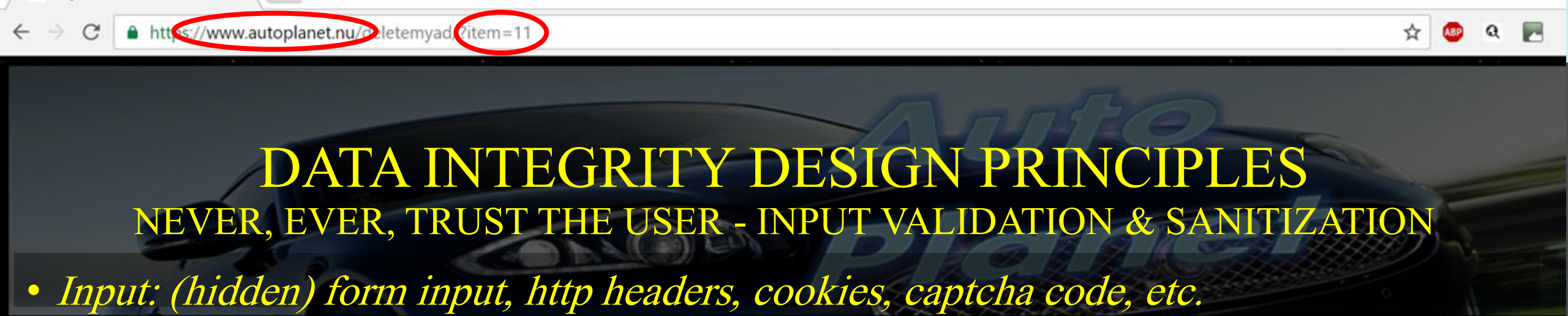
Data Protection

- *Keep sensitive information private*
- *Don't expose yourself*

Functionality

- *Expect the unexpected*
- *Minimize attack surface*





Sanitize unwanted input, but be careful...it can make your data unusable

```
// Call function san_chars to sanitize input from search form input  
list($Make, $BodyType, $Fuel)=san_chars(@$_GET['Make'], $_GET['BodyType'], @$_GET['Fuel']);
```

Validate and reject unwanted input ...always at the server side

```
// Validate e-mail address  
} elseif (!filter_var($Gebruikersnaam, FILTER_VALIDATE_EMAIL)) {  
echo "Please fill in a valid email address !";
```

Please enter your details below to contact the seller

Name:

Email:

Include a message:
Min: 200 characters

4 5 9 8 9

Copy the digits from above image into the box:

DATA INTEGRITY DESIGN PRINCIPLES

NEVER, EVER, TRUST THE USER – SQL INJECTION

Home

Sell car

My autoplanet

Logout

Please enter your details below to contact the seller

Name:

Rob' OR '1'='1

Enquiry For: Volvo 240 Station

- Prevent that user data can alter your SQL queries

Blauw | 1992 | Petrol | € 2450

Query with user input

```
//select entered username from users table
```

```
SELECT * FROM `users` WHERE `users`.`name` = ".$name."
```

1) Construct query string

```
SELECT * FROM `users` WHERE `users`.`name` = 'Rob' OR '1'='1'
```

2) Execute query

Database Driver

Database

This will return all rows from the table users, since WHERE 1=1 is always true

Query with prepared statements

```
//prepare select statement with parameters
```

```
$sth = $conn->prepare("SELECT * FROM users WHERE name = :name")
```

```
//bind the statement parameters to the input variables
```

```
$sth->bindParam(':name', $name, PDO::PARAM_STR, 30)
```

1) Prepare (compile) query

2) Add variable parameters

3) Execute query

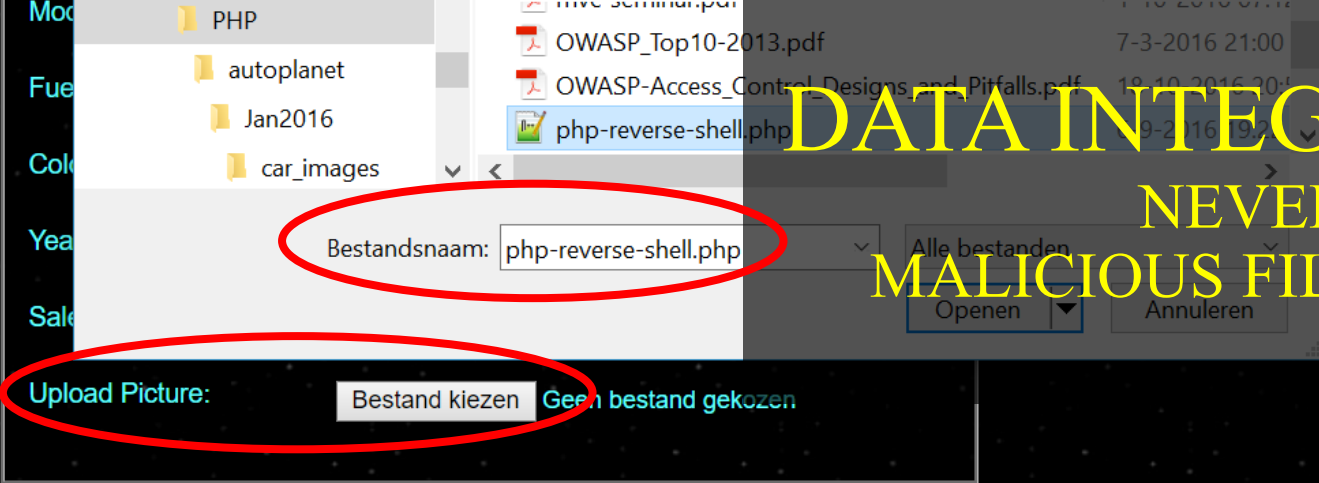
Abstraction Layer
(PDO)

Database Driver

Database

DATA INTEGRITY DESIGN PRINCIPLES

NEVER, EVER, TRUST THE USER - MALICIOUS FILE UPLOADS AND CODE INJECTIONS



Don't allow users to upload whatever they want

```
// Check if the uploaded picture isn't larger than 500kb
} elseif ($_FILES['carpic']['size'] > 500000) {
$warning='!!! Sorry, your picture file is too large !!!';
//
// Check if the uploaded picture is an image
} elseif ((getimagesize($_FILES['carpic']['tmp_name'])) == false) {
$warning='!!! File is not an image !!!';
}
```

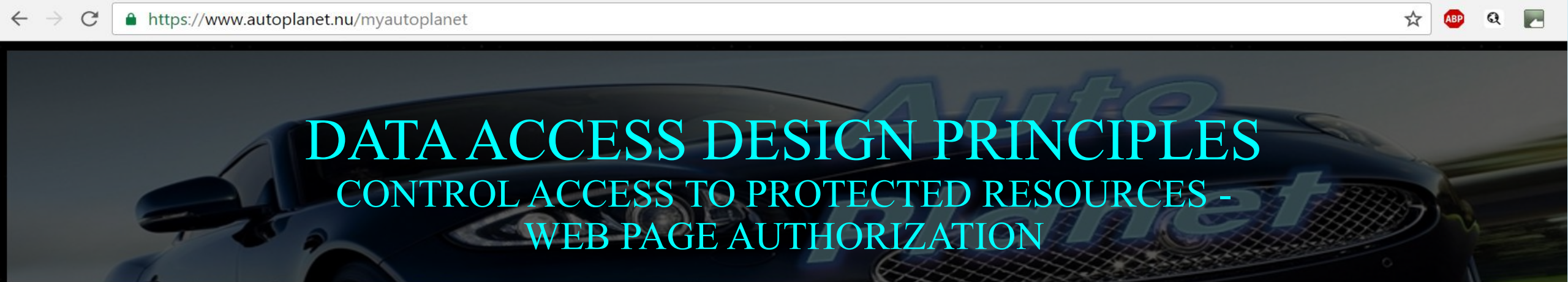
Prevent code injection vulnerabilities

```
; Whether to allow the treatment of URLs (like http:// or ftp://) as files.
; http://php.net/allow-url-fopen
allow_url_fopen = Off

; Whether to allow include/require to open URLs (like http:// or ftp://) as files.
; http://php.net/allow-url-include
allow_url_include = Off
```

Disable script execution in upload directories

```
[raugust@kangoos carimages]$ more .htaccess
php_flag engine off
```



Home

Login

Register

Logout

You need to login to access this page !



Check user authorization for each web page

```
// Check if user is authorised to access this page (depending on his role)
checkifuserauthorised("autoplanetuser");
```

Use centralized Authorization Routines

```
function checkifuserauthorised($role) {
// Run authorisation check for autoplanet users
if ($role=="anonymous") {
// Do nothing - No authorization check needed
}
if ($role=="autoplanetuser") {
// Check if user is authorised (logged in) to access this page
if(empty( $_SESSION['user_id'])) {
// If user isn't logged in show message that he needs to login to access this page
show_message("You need to login to access this page !");
exit();
}
```


DATA ACCESS DESIGN PRINCIPLES

CONTROL ACCESS TO PROTECTED RESOURCES - DATA OBJECT AUTHORIZATION

Home Sell car My autoplanet Logout

Selected car advert has been deleted
Returning to myautoplanet within 5 seconds...

- *Always ensure the data owner is referenced in queries*
- *Prevent tampering with query parameters*

Is the user who requests this delete action the owner of this record ?

Vulnerable SQL query:

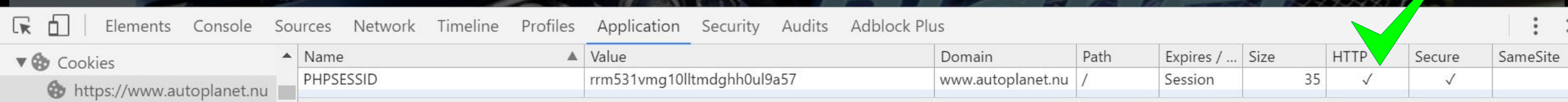
```
// Delete car advert from database  
$sth = $conn->prepare("DELETE FROM autoplanet WHERE ID = :ID");
```

Integrate access control in your SQL queries

```
// Delete car advert from database  
$sth = $conn->prepare("DELETE FROM autoplanet WHERE ID = :ID AND Name = :Name AND Email = :Email");
```

DATA ACCESS DESIGN PRINCIPLES

CONTROL ACCESS TO PROTECTED RESOURCES - SESSION DATA



Name	Value	Domain	Path	Expires / ...	Size	HTTP	Secure	SameSite
PHPSESSID	rrm531vmg10lltmdghh0ul9a57	www.autoplanet.nu	/	Session	35	✓	✓	

Prevent theft of (session) cookies by XSS attacks

Tell browsers that client scripts aren't allowed to access cookies

Prevent that attackers can obtain a valid session identifier (session hijacking)

Change session ID's at any transition in authentication state

Prevent that attackers can force users to use known session ID (session fixation)

Enforce that always a new session ID is created if a session is requested with a ID that does not exist

DATA ACCESS DESIGN PRINCIPLES

USER RIGHTS: LESS IS MORE ! - DATABASE PERMISSIONS

Registration - Create here an account to sell your car

Full Name:

Username (Your email):

Password (min 8 chars):

Don't use a single shared database account for your application

<input type="checkbox"/>	User name	Host name	Password	Global privileges	Grant	Action
<input type="checkbox"/>	autoplanetDBuser	localhost	Yes	USAGE	No	Edit privileges Export
<input type="checkbox"/>	autoplanetDBuseranonymous	localhost	Yes	USAGE	No	Edit privileges Export



Don't give application users complete access to your database

Edit privileges: User account '*autoplanetDBuser*'@'*localhost*'

Table-specific privileges

Table	Privileges	Grant	Column-specific privileges	Action
autoplanet	SELECT, INSERT, UPDATE, DELETE, REFERENCES, INDEX	No	No	Edit privileges Revoke
users	SELECT, INSERT, UPDATE, DELETE, REFERENCES, INDEX	No	No	Edit privileges Revoke

Edit privileges: User account '*autoplanetDBuseranonymous*'

Table-specific privileges

Table	Privileges	Grant	Column-specific privileges	Action
autoplanet	SELECT	No	No	Edit privileges Revoke

DATA PROTECTION DESIGN PRINCIPLES

Home

Login

Register

Logout

KEEP SENSITIVE INFORMATION PRIVATE - WEB FORMS

Welcome back, enter your details below

Email:

Password:

Name	Value	Domain	Path	Expires / ...	Size	HTTP	Secure	SameSite
PHPSESSID	2p8so0i1jd5vvc91733dqghjg4	www.autoplanet.nu	/	Session	35	✓	✓	

Use HTTPS and POST for sending sensitive information via web forms

```
<form name='form' action='/login' method='post'>
```

Prevent that usernames and passwords on shared computers get exposed to other users

```
name='Username' size='40' autocomplete='off' class='textbox-menu' required>
```


Home Login Register Logout

DATA PROTECTION DESIGN PRINCIPLES

KEEP SENSITIVE INFORMATION PRIVATE - SESSION DATA

Welcome back, enter your details below

Email:

Password:

Name	Value	Domain	Path	Expires / ...	Size	HTTP	Secure	SameSite
PHPSESSID	2p8so0i1jd5vvc91733dqghjg4	www.autoplanet.nu	/	Session	35	✓	✓	

Prevent that session data is visible in the URL (enforce that only cookies are used)

Encrypt session data between user and application (set session.cookie_secure flag)

Prevent that session data is being transmitted in plain text (enforce HTTPS for whole site)

```
RewriteEngine On
RewriteCond %{SERVER_PORT} 80
RewriteRule ^(.*)$ https://www.autoplanet.nu/$1 [R,L]
```

DATA PROTECTION DESIGN PRINCIPLES

KEEP SENSITIVE INFORMATION PRIVATE - CREDENTIALS

Welcome back, enter your details below

Email:

pieter@post.nl

Password:

pieter@post.nl
with 0 chars, letters and numbers

Use a recent salted password hashing algorithm (argon2, bcrypt, scrypt)

```
$HashPW = password_hash($Second, PASSWORD_DEFAULT); (PASSWORD_DEFAULT for salted bcrypt hash)
```

Name	Username	HashPW
Pieter Post	pieter@post.nl	\$2y\$10\$Lt5SYWdcGjvXhIrhNE5p0u3pXCARWPW

Keep database connection strings out of your source code (keep them in a trusted place)

```
-rwx----- 1 root root 69 Jan 31 2016 stuff
[root@kangeos secret]# more stuff
SetEnv DB_USER # Load config files in the "/etc/httpd/conf.d" directory, if any
SetEnv DB_PASS include /var/secret/stuff
```

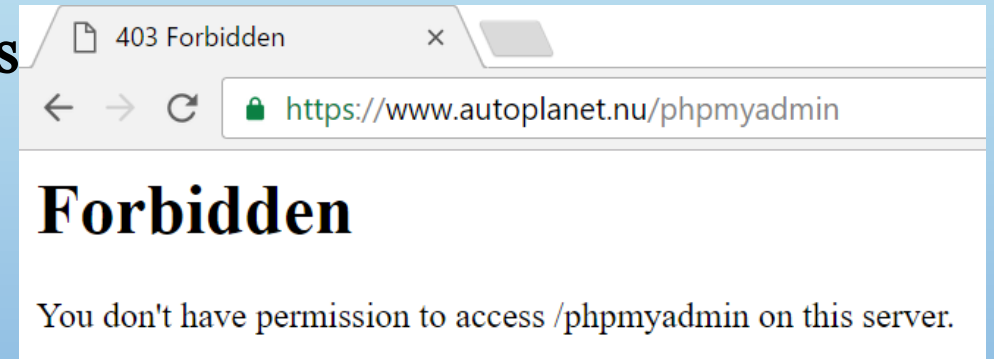
```
function search_user($X) {
// Fetch database credentials
$username = $_SERVER['DB_USER'];
$password = $_SERVER['DB_PASS'];
//
// Connect to database with PDO
$conn = new PDO("mysql:host=$hostname;dbname=autodb", $username, $password);
```


DATA PROTECTION DESIGN PRINCIPLES

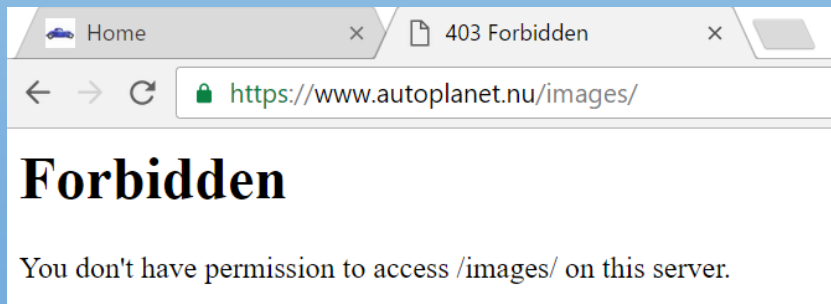
DON'T EXPOSE YOURSELF - SENSITIVE CONTENT

Use URL rewriting and routing to prevent access to physical file paths and files

Prevent attackers to discover admin consoles or pages



Don't allow attackers to view sensitive content in your web directories



```
# Deny listings of the directories below
<Directory "/var/www/html/autoplanet/public/images">
Options -Indexes
</Directory>
#
<Directory "/var/www/html/autoplanet/public/carimages">
Options -Indexes
</Directory>
```



DATA PROTECTION DESIGN PRINCIPLES

DON'T EXPOSE YOURSELF - WEB APPLICATION FINGERPRINTING

Don't expose valuable information to attackers such as PHP and Web server version

expose_php = Off in php.ini

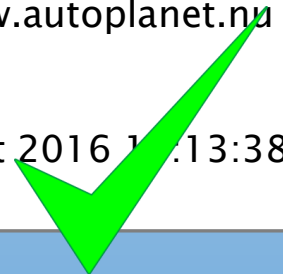
Apache httpd.conf:

ServerTokens Product

ServerSignature Off

```
GET https://www.autoplanet.nu
-- response --
200 OK
Date: Fri, 28 Oct 2016 15:54:56 GMT
Server: Apache/2.4.23 (Fedora) OpenSSL/1.0.2j-fips PHP/5.6.27
X-Powered-By: PHP/5.6.27
```

```
GET https://www.autoplanet.nu
-- response --
200 OK
Date: Fri, 28 Oct 2016 15:13:38 GMT
Server: Apache
```



Don't put information in HTML or JavaScript comments

```
<link rel="wivmanitest" type="application/wivmanitest+xml" />
<meta name="generator" content="WordPress 4.6.1" />
<meta name="generator" content="WooCommerce 2.4.10" />
```

DON'T EXPOSE YOURSELF

INFORMATION EXPOSURE THROUGH ERROR MESSAGES

Sorry, something went wrong... Please try again later.

```
exception 'PDOException' with message 'SQLSTATE[HY000] [2002] No
such file or directory' in
/var/www/html/autoplanet/application/models/searchcar.php:12
Stack trace: #0
/var/www/html/autoplanet/application/models/searchcar.php(12):
PDO->__construct('mysql:host=loca...', 'autoplanetDBuse...',
'KraftwerkAut0h...' ) #1
/var/www/html/autoplanet/application/controllers/searchcar.php(38):
```

Fail safe: Error messages reveal sensitive information such as passwords, file paths, etc.

```
// Display error message in case of database error, close connection and exit script
}
catch(PDOException $e) {
show_message("Sorry, something went wrong... Please try again later.");
// Uncomment line below if you need detailed error messages for debugging purposes
// show_message($e);
$conn = null;
exit();
```

```
display_errors = Off
; Default Value: On
; Development Value: On
; Production Value: Off
```

FUNCTIONALITY DESIGN PRINCIPLES

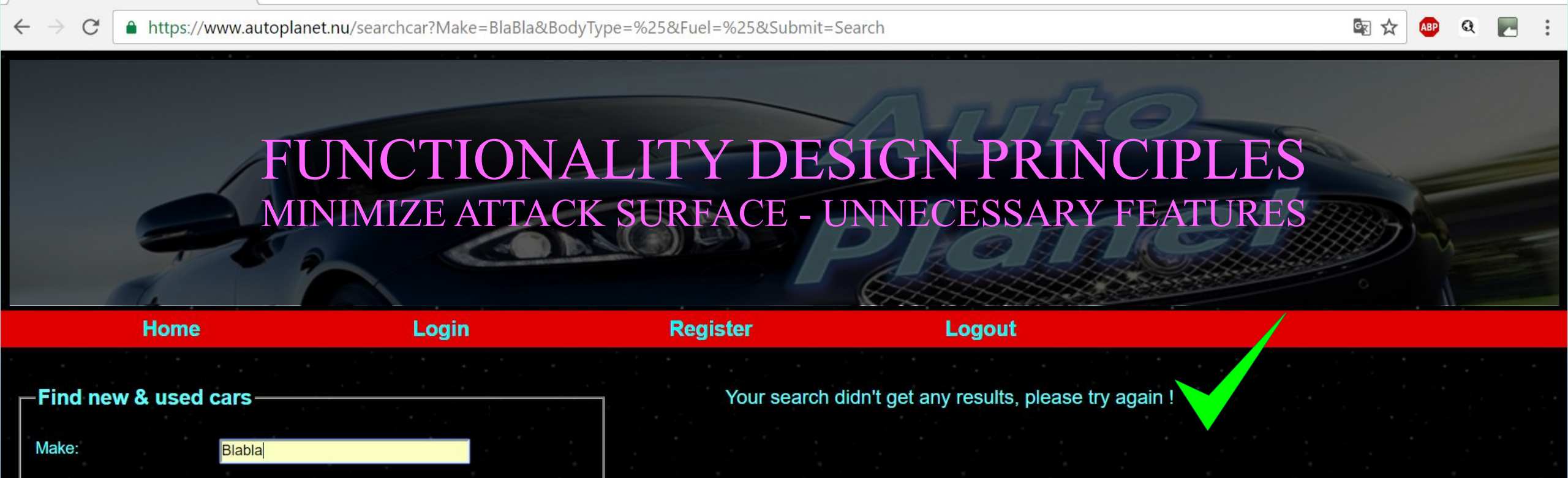
EXPECT THE UNEXPECTED - APPLICATION LOGIC ATTACKS

- *Some jokers don't respect your rules...*



Prevent that application logic steps can be bypassed

Avoid being helpful for attackers



Don't display search queries back to the user (XSS risks)

Don't use the URL as input for the navigation bar

Disable default application settings which you don't need

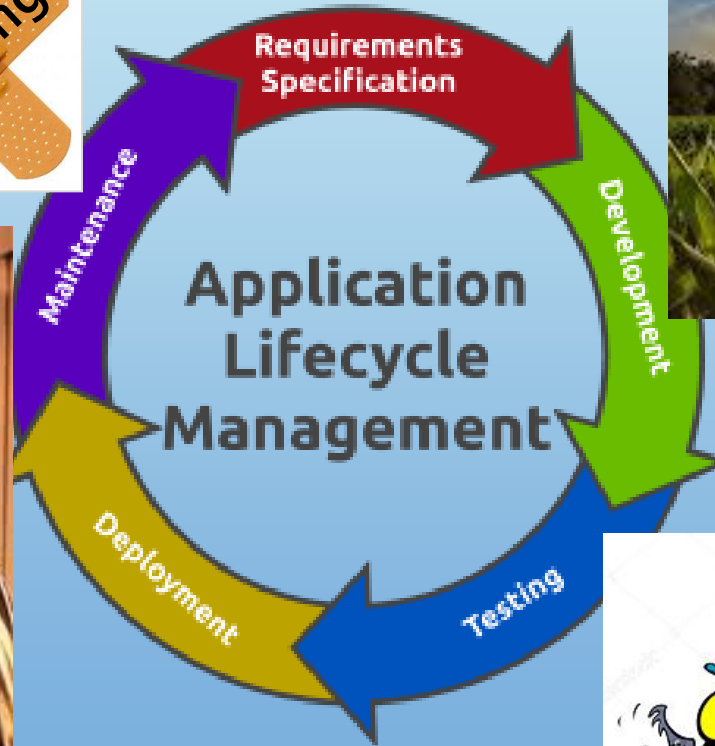
AGENDA

- *Intro – A.S. Watson and Me*
- *Why this Presentation ?*
- *Security Architecture*
- *Secure Code Design Principles*
- ***The Process - Security in the SDLC***
- *Mission Accomplished ?*

SECURITY IN THE SDLC



All bugs definitely fixed?



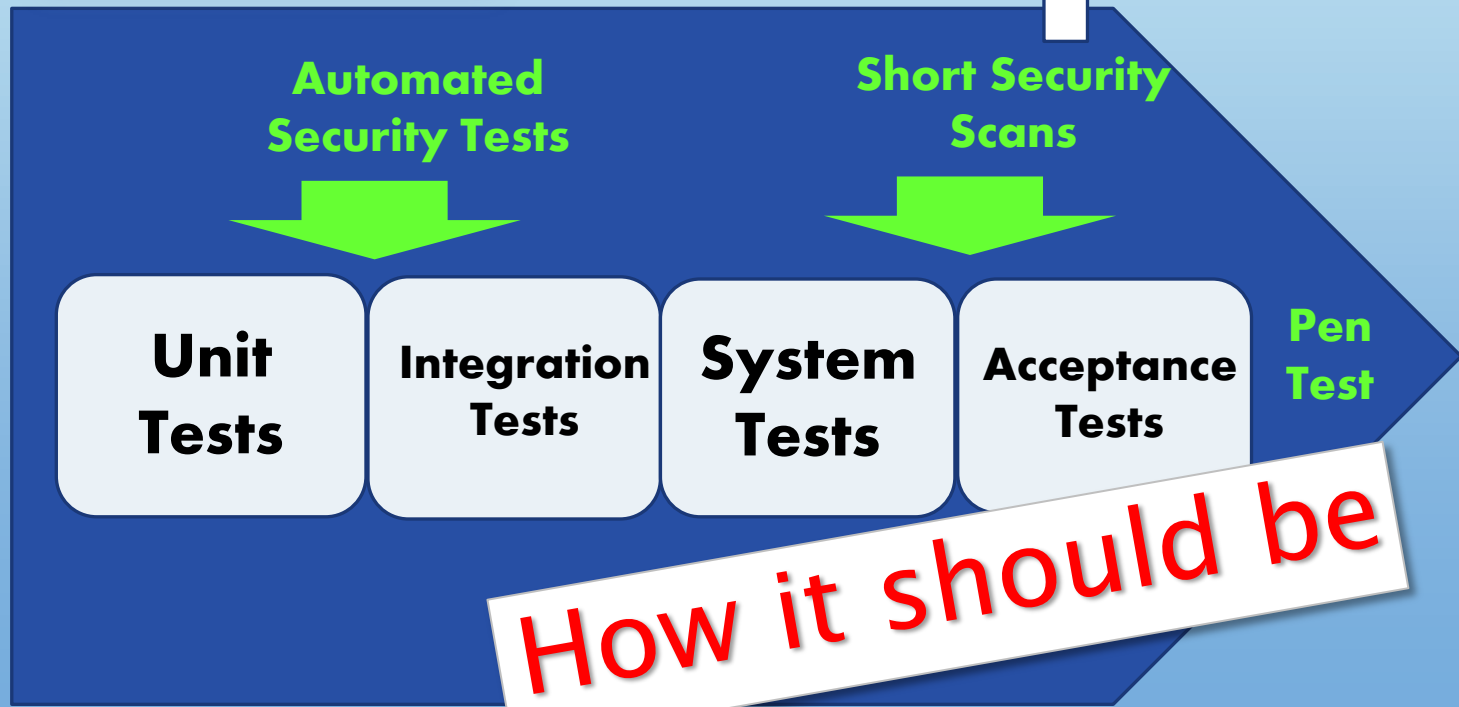
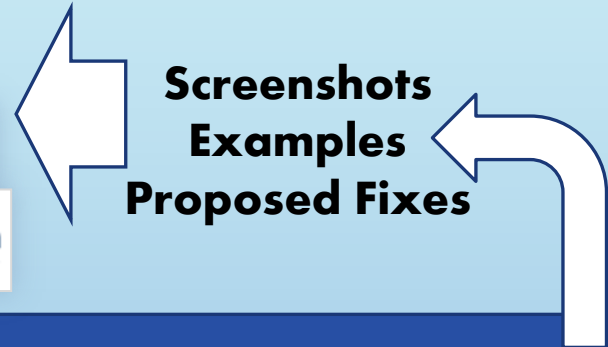
Debugging

SECURITY TESTING

One week before launch...



How it is



How it should be

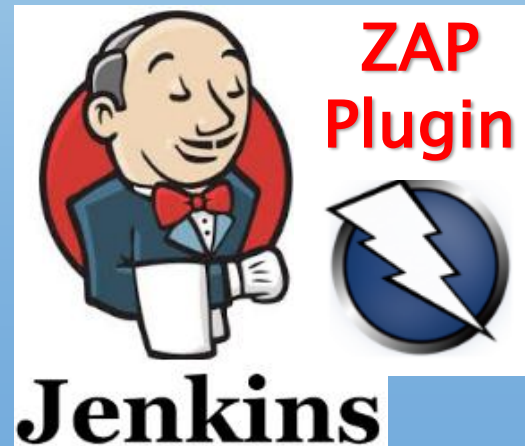
INTEGRATING SECURITY IN THE SDLC

Agile

Sec DevOps

**Automated
Security Testing**

Evil User Stories



AGENDA

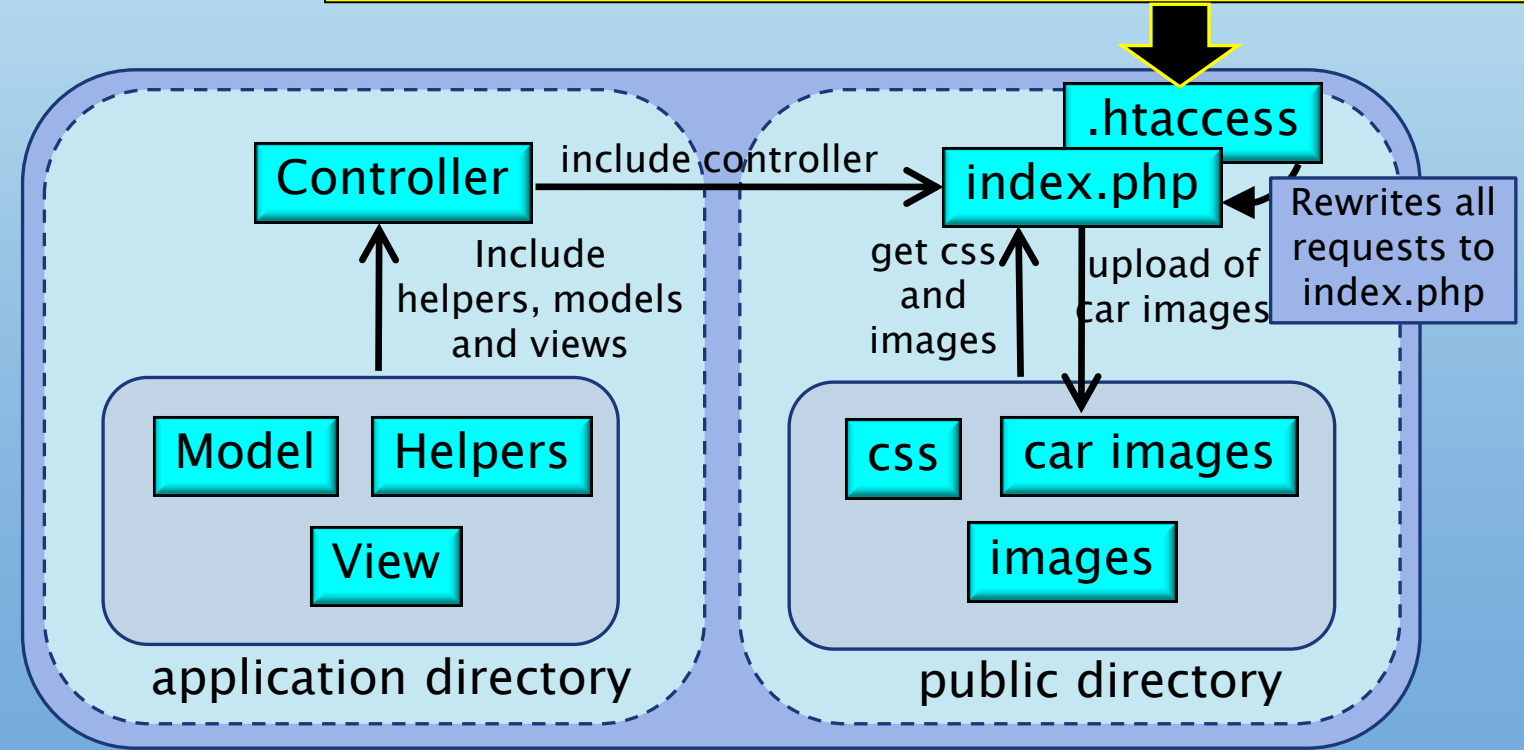
- *Intro – A.S. Watson and Me*
- *Why this Presentation*
- *Security Architecture*
- *Secure Code Design Principles*
- *The Process - Security in the SDLC*
- ***Mission Accomplished ?***

AUTOPLANET – A SECURE WEB APPLICATION ?

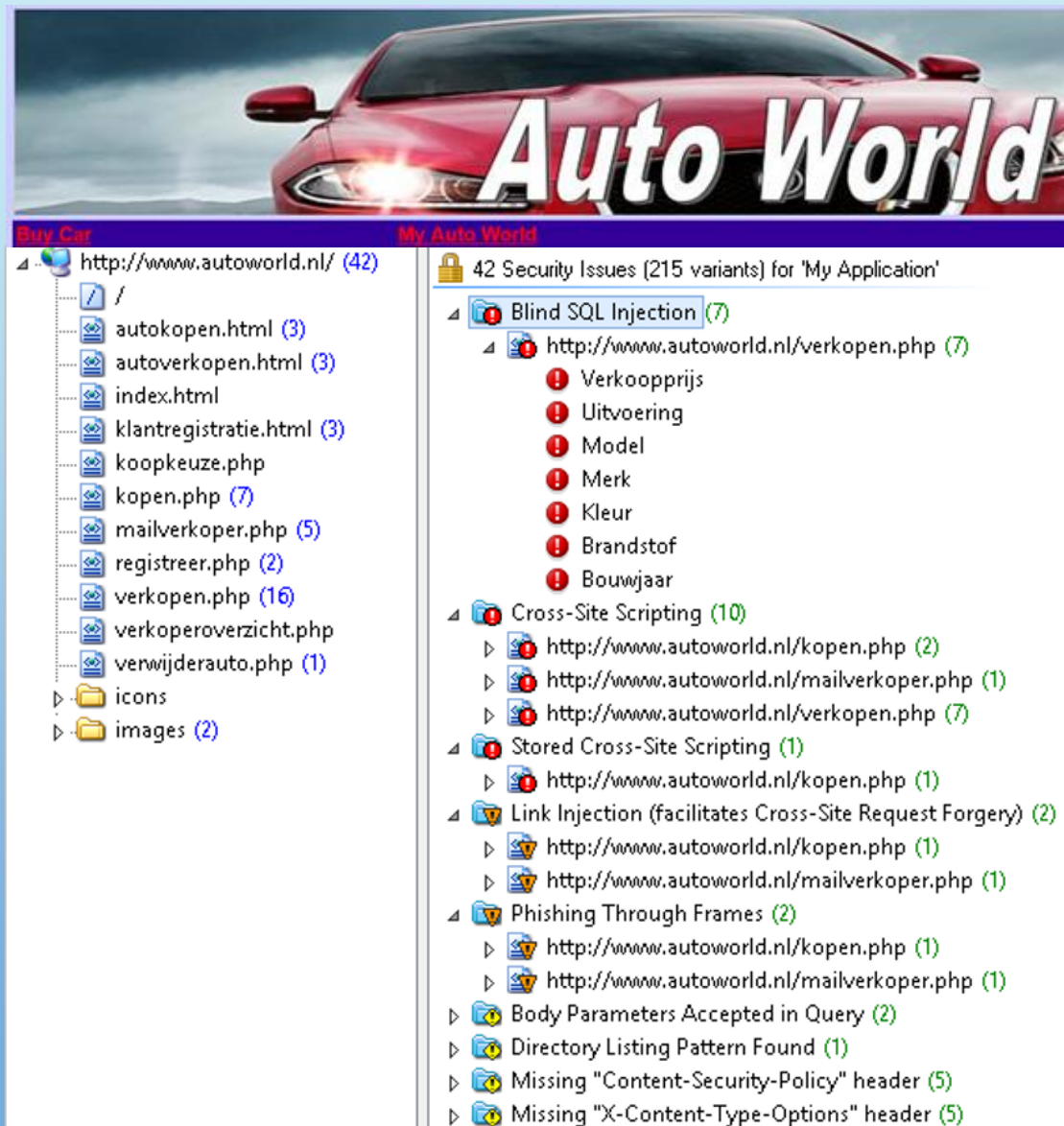
Home

Login

`https://home-url/controller?param1=value1¶m2=value2...`



APPSCAN TEST RESULTS



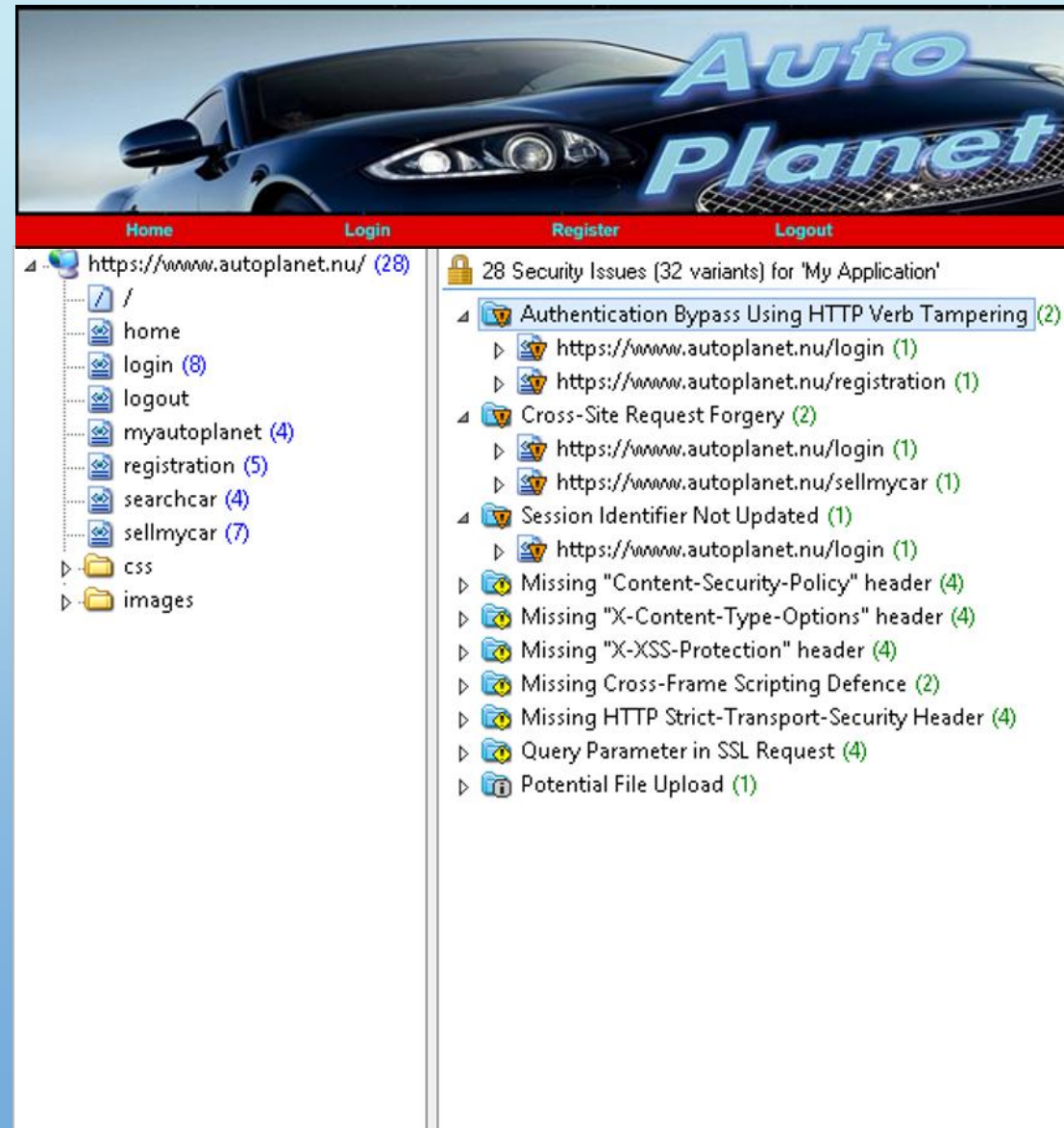
Buy Car **My Auto World**

42 Security Issues (215 variants) for 'My Application'

- Blind SQL Injection (7)
 - http://www.autoworld.nl/verkopen.php (7)
 - Verkoopprijs
 - Uitvoering
 - Model
 - Merk
 - Kleur
 - Brandstof
 - Bouwjaar
- Cross-Site Scripting (10)
 - http://www.autoworld.nl/kopen.php (2)
 - http://www.autoworld.nl/mailverkoper.php (1)
 - http://www.autoworld.nl/verkopen.php (7)
- Stored Cross-Site Scripting (1)
 - http://www.autoworld.nl/kopen.php (1)
- Link Injection (facilitates Cross-Site Request Forgery) (2)
 - http://www.autoworld.nl/kopen.php (1)
 - http://www.autoworld.nl/mailverkoper.php (1)
- Phishing Through Frames (2)
 - http://www.autoworld.nl/kopen.php (1)
 - http://www.autoworld.nl/mailverkoper.php (1)
- Body Parameters Accepted in Query (2)
- Directory Listing Pattern Found (1)
- Missing "Content-Security-Policy" header (5)
- Missing "X-Content-Type-Options" header (5)

File list:

- autokopen.html (3)
- autoverkopen.html (3)
- index.html
- klantregistratie.html (3)
- koopkeuze.php
- kopen.php (7)
- mailverkoper.php (5)
- registreer.php (2)
- verkopen.php (16)
- verkoperoverzicht.php
- verwijderauto.php (1)
- icons
- images (2)



Home **Login** **Register** **Logout**

28 Security Issues (32 variants) for 'My Application'

- Authentication Bypass Using HTTP Verb Tampering (2)
 - https://www.autoplanet.nu/login (1)
 - https://www.autoplanet.nu/registration (1)
- Cross-Site Request Forgery (2)
 - https://www.autoplanet.nu/login (1)
 - https://www.autoplanet.nu/sellmycar (1)
- Session Identifier Not Updated (1)
 - https://www.autoplanet.nu/login (1)
- Missing "Content-Security-Policy" header (4)
- Missing "X-Content-Type-Options" header (4)
- Missing "X-XSS-Protection" header (4)
- Missing Cross-Frame Scripting Defence (2)
- Missing HTTP Strict-Transport-Security Header (4)
- Query Parameter in SSL Request (4)
- Potential File Upload (1)

File list:

- home
- login (8)
- logout
- myautoplanet (4)
- registration (5)
- searchcar (4)
- sellmycar (7)
- css
- images

But Wait...
**THERE'S
MORE!**

Access-Control-Allow-Origin

Password recovery attacks

pollution

iFrame injection

X-Path injection

Email injection

Blind XSS

Default passwords

DOM based XSS attacks

Slow HTTP attacks

XML external entity attacks

Vulnerable Javascript library

Third party includes/Plugins

Flash parameter ScriptAccess

Cross Site Request Forgery

Secure HTTP response headers

TRACK method is enabled

Apache user Chroot

Security logging & auditing

Javascript eval() usage

HTTP Response Splitting

Oversized POST requests

Data encryption

Brute Force Attacks

Rename uploaded files

Session expiration

Clickjacking

SE Linux

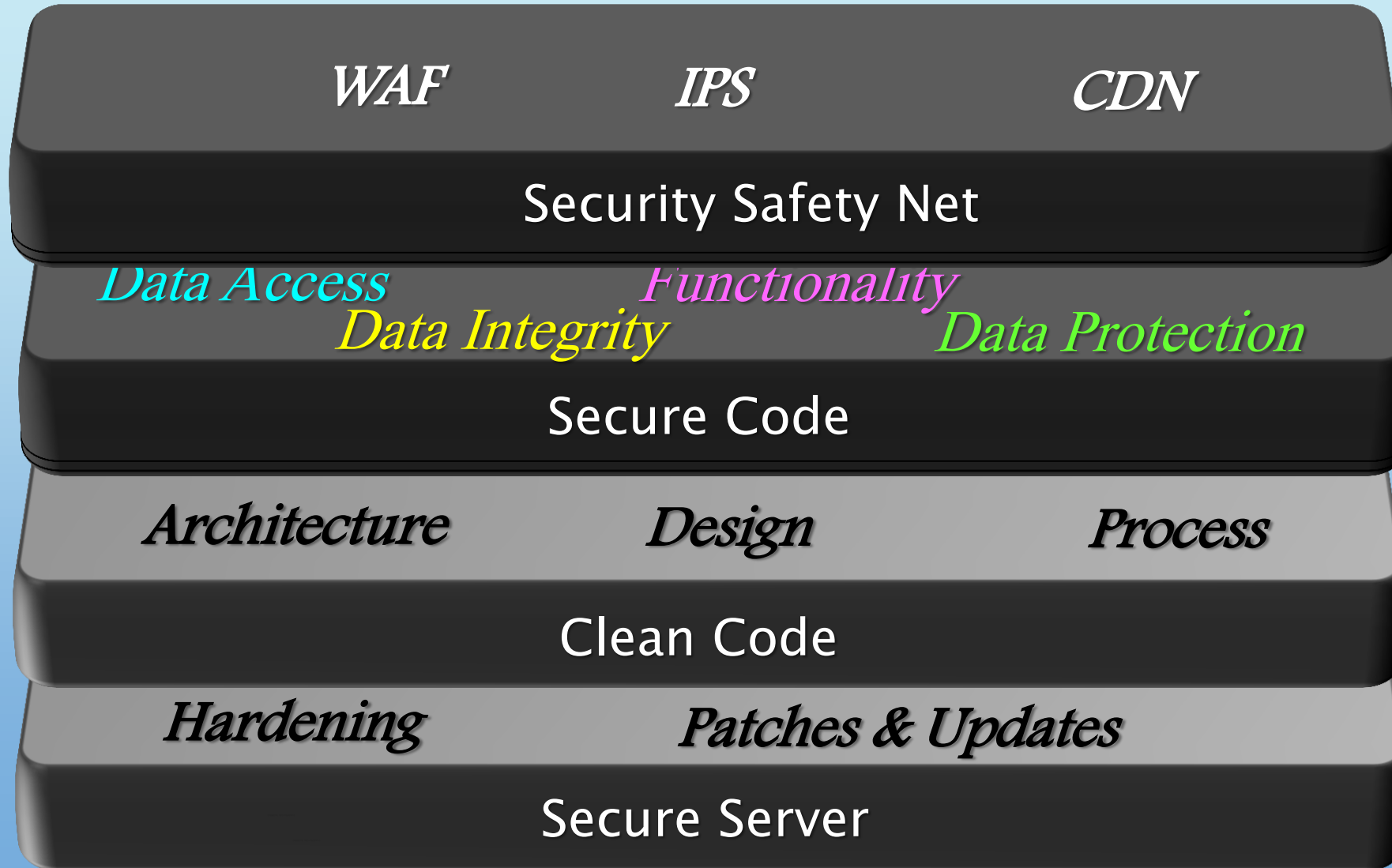
HTTP Parameter pollution

Insufficient Entropy

Http redirect security bypass

TRACE method is enabled

THE WHOLE PICTURE



MISSION ACCOMPLISHED OR..... **MISSION:** Impossible?

*Can we build with secure coding a
Trump wall to keep out the hackers ?*



*Can we do much better than the millions of
poorly secured websites on the Internet ?*





rob.augustinus@the-future-group.com

CONTROLLER STRUCTURE (SEARCH FORM)

```
<?php
// Home Page & Search Car Controller
//
// Begin this session
session_start();
// Include function to generate view for page header
require_once('/var/www/html/autoplanet/application/helpers/sanchars.php');
// Include function to generate view for showing a message
require_once('/var/www/html/autoplanet/application/views/headerview.php');
// Include function to generate view for showing search results
require_once('/var/www/html/autoplanet/application/views/searchcarview.php');
// Include function to check if user is authorised to access the web page
require_once('/var/www/html/autoplanet/application/helpers/checkuserauth.php');
//
// Generate the page header view depending on session status
page_header('Home', $_SESSION['user_id']);
//
check_authorisation("anonymous");
//
// Set value for search form status depending on if input is submitted
if (empty($_GET['Make'])) {
    $searchformstatus=0;
}
else {
    $searchformstatus=1;
}
//
// Call function san_chars to sanitize input from search form input
list($Make, $BodyType, $Fuel)=san_chars(@$_GET['Make'], $_GET['BodyType'], @$_GET['Fuel']);
//
// Call datamodel function to search for cars in the database
//
$foundcars=search_car($Make,$BodyType,$Fuel);
//
// Generate the view for the search page
search_page($foundcars,$searchformstatus);
//
```

Load includes (views, model, helpers)

Display menu (page header view)

Authorization check (page access)

Initialize form (based on submit status)

Validate & sanitize form input

Search car in database (data model)

Display search results (page view)

DATA INTEGRITY SECURITY TESTING

How does the application respond to unexpected input ?

- Test input such as special characters and SQL queries, try parameter tampering
- How does the input appear in the returned HTML ? (dangerous characters not removed / encoded)
- Can SQL queries as input cause (SQL) error messages or stack traces ?

The screenshot illustrates a security test on a web application. On the left, a browser window shows the URL `https://www.autoplanet.nu/buycar?iter=<script>alert(111)</script>` and a red navigation bar with 'Home' and 'Login' buttons. Below the browser, a registration form is visible with a red navigation bar and a 'Name:' field containing the payload `admin' or '1'='1'--`. A yellow error message states: "Something went wrong...Please try later again." On the right, a network tool (Fiddler) shows a POST request to `https://www.autoplanet.nu/login` with a body containing the payload `Username=pieter%40post.nl&Password=<script>alert(111)</script>`. Red circles highlight the malicious payloads in the browser address bar, the registration form, and the network tool body.

What can you upload and do with it ?

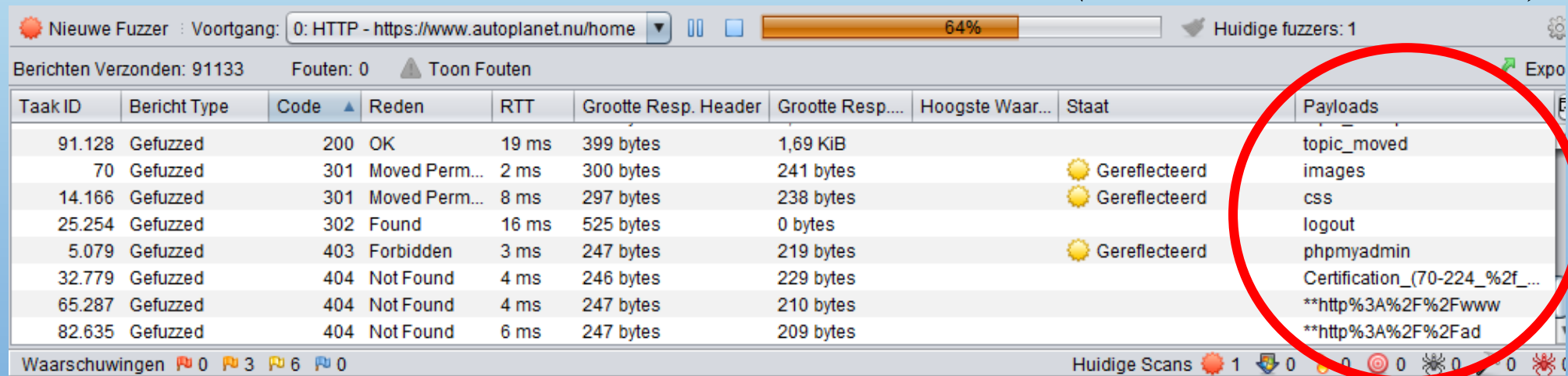
- Test uploading of files with unexpected extension, type, content, size, Can uploaded HTML or scripts be executed ?

The screenshot shows a file upload interface. The text "Upload Picture:" is followed by a button labeled "Bladeren...". A file named "test.php" is listed next to it, circled in red. Below the file list is a "submit" button.

DATA ACCESS SECURITY TESTING

Unauthorized access to protected resources

- *Try to access to webpages or data which you shouldn't be able to access*
- *Try if session cookies are protected (cookie HttpOnly flag), are they visible in URL or (hidden) parameters?*
- *Test for directories and/or files which shouldn't be visible (use fuzzer tool to discover)*



The screenshot shows a fuzzer tool interface with a table of test results and a list of discovered payloads. A red circle highlights the 'Payloads' column.

Taak ID	Bericht Type	Code	Reden	RTT	Grootte Resp. Header	Grootte Resp....	Hoogste Waar...	Staat	Payloads
91.128	Gefuzzed	200	OK	19 ms	399 bytes	1,69 KiB			topic_moved
70	Gefuzzed	301	Moved Perm...	2 ms	300 bytes	241 bytes		Gereflecteerd	images
14.166	Gefuzzed	301	Moved Perm...	8 ms	297 bytes	238 bytes		Gereflecteerd	css
25.254	Gefuzzed	302	Found	16 ms	525 bytes	0 bytes			logout
5.079	Gefuzzed	403	Forbidden	3 ms	247 bytes	219 bytes		Gereflecteerd	phpmyadmin
32.779	Gefuzzed	404	Not Found	4 ms	246 bytes	229 bytes			Certification_(70-224_%2f_...
65.287	Gefuzzed	404	Not Found	4 ms	247 bytes	210 bytes			**http%3A%2F%2Fwww
82.635	Gefuzzed	404	Not Found	6 ms	247 bytes	209 bytes			**http%3A%2F%2Fad

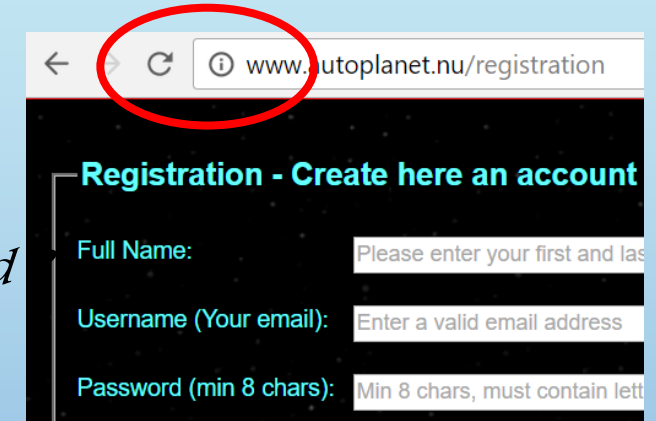
User rights

- *Review database permissions for application users*
- *Do they use the lowest possible level of permissions ?*

DATA PROTECTION SECURITY TESTING

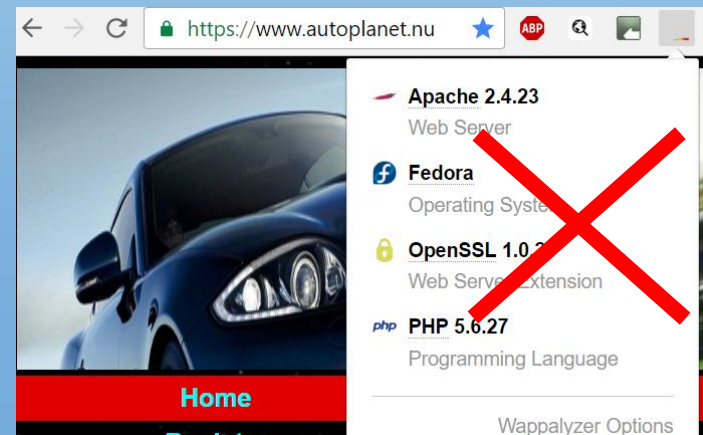
Is sensitive information well protected ?

- *Check if forms with sensitive data can't be sent via GET and/or HTTP*
- *Check for autocomplete on form fields with sensitive data*
- *Check session.cookie_secure flag and what happens if cookies are denied*



Information exposure

- *Check server response for webserver version and other information (Wappalyzer)*
- *Force stack trace errors (e.g. stop database) and check if sensitive information is exposed*
- *Check source code for hard coded credentials*



FUNCTIONALITY SECURITY TESTING

EVIL USER STORIES

Be creative: What data, steps, parameters, etc. can be tampered with ?

- *Try to resubmit email reply forms an unlimited number of times*
- *Intercept POST requests and remove or change parameters before submitting*
- *See what happens if you enter negative amounts or item numbers*
- *Change your token or account nr to another number to see what will happen*

