CyberSecurity: Principle and Practice

BSc Degree in Computer Science
2025-2026

Lesson 9: Language Vulnerabilities and Injection attack pt 1

Prof. Mauro Conti
Department of Mathematics
University of Padua
mauro.conti@unipd.it
http://www.math.unipd.it/~conti/

Teaching Assistants
Giulio Umbrella
giulio.umbrella@phd.unipd.it
Francesco De Giudici
francesco.degiudici@studenti.unipd.it

Università degli Studi di Padova

SPRITZ Security & Privacy Research Group

Dipartimento Matematica

Introduction

- How many of you think about security during a system deployment?

Introduction

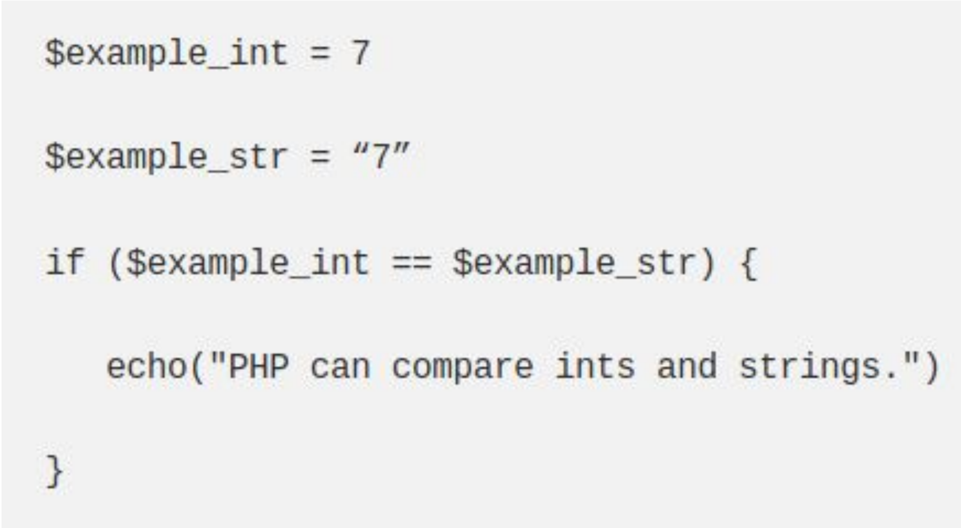
- How many of you think about security during a system deployment?
- Hope some of you …
- But what about the security derived from the program language that you are using?

Program Languages Vulnerabilities

- Program Languages are well known for several security threats that they provide
- Some functions might expose your application to threats
- It is a good practice to be aware of these risks
    - to prevent attacks

Type Juggling

- PhP, as JS and Python, is a dynamically typed programming language
- the variables types are checked at runtime
- this sometimes can be a problem …
- (“7 puppies” == 7) -> True
- see more at link1 and link2



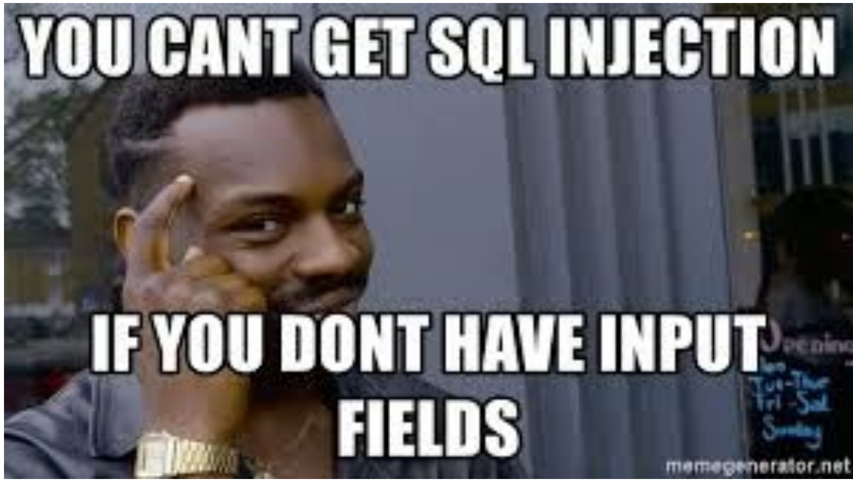
```
$example_int = 7

$example_str = “7”

if ($example_int == $example_str) {

   echo("PHP can compare ints and strings.")

}
```

How to do in practice

1. Identify the programming language used in the application
2. Identify the version
3. Identify possible libraries used
4. Check on Google for possible vulnerabilities

Injection attacks

- **Injection Attacks** are a class of attacks
- the attacker provides an untrusted input to our application
- the program processes the input and executes a function in an anomalous way
- it is considered the most dangerous class of attacks in web applications
- slides inspired by [link]



Case 1: Code Injection

- The attacker injects application code written in the application language
- potential impact: full system compromised
- the attacker might try to run OS command with program privileges
- e.g., in the following example we get the php version info
  - (using “phpinfo()” in code)

```
**
* Get the code from a GET input
* Example of Code Injection-
http://example.com/?code=phpinf
o();
*/
$code = $_GET['code'];

/**
* Unsafely evaluate the code
* Example - phpinfo();
*/

 eval("\$code;");
```

Case 6: OS command Injection

- Injection of OS commands that will run with application privileges
- For example, a PHP application might execute a ping to a given IP address

```
<?php
$address = $_GET["address"];
$output = shell_exec("ping -n 3 $address");
echo "<pre>$output</pre>";
?>
```

- The request is done via GET request
  - parameter name: address
- An attacker might request the following, displaying ping and list of files in the directory

  http://example.com/ping.php?**address**=8.8.8.8**%26**ls

  - Note that 26 -> &

Questions? Feedback? Suggestions?