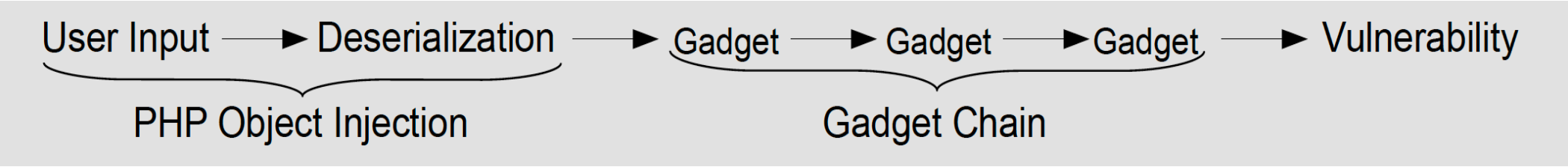
Defending against Code Reuse Attacks in PHP

Penghui Li

2020/08/09

Workflow of PHP code reuse attacks

- Use a PHP Object Injection (POI) vulnerability to trigger gadget chains
- Can lead to multiple types of vulnerabilities, e.g., SQL injection, arbitrary code execution, file inclusion, etc.



Examples (1)

- user.php?usr=O:9:"UserClass":2:{s:3:"age";i:18; s:4:"name";s:3:"Tom";}
  - Normally: print string “User Tom is 18 years old. ”
- user.php?usr=O:9:"FileClass":1:{s:8:"filename"; s:10:"config.php";}
  - Unexpectedly: dump the content in config.php

```
<?php
class FileClass {
 public $filename = "error.log";
 public function __toString() {
    return @file_get_contents($this->filename);
 }
}
class UserClass {
 public $age = 0;
 public $name = '';
 public function __toString() {
     return 'User '.$this->name." is ".$this->age.' years old. <br/>';
 }
}
$obj = unserialize($_GET['usr']);
echo $obj; // trigger __toString() method in the object
```

Examples (2)

- news.php?user=O:4:"User":2:{s:3:"age";i:20;s:4:"name";s:4:"John”;}
  - Normally
- news.php?user=O:8:"LogClass":1:{s:11:"logfilename";s:9:".htaccess";}
  - Unexpectedly: delete the .htaccess file

```
<?php
class LogClass {
 public $logfilename = "";
 public function logdata($text) {
  file_put_contents($this->logfilename,$text,FILE_APPEBD);
 }
 public function __destruct() {
   unlink(dirname(__FILE__).'/'.$this->logfilename);
 }
}
class User {
 public $age = 0;
 public $name = '';
 public function print_data() {
  echo "User".$this->name."is".$this->age."years old.<br/>";
 }
}
$usr = unserialize($_GET[“user”]); // trigger __destruct() method if it exists
```

Potential research direction

- Instead of detecting or confirming PHP code reuse attacks (or gadget chains), can we perform runtime defense?
- Code reuse attacks in other languages
  - Locate the addresses of code to be reused
    - ASLR, etc.
  - Use gadget chains
- General defense mechanisms for SQL injection
  - Input validation and sanitization
  - Benign usage profile (training process) and check whether the issued SQL query matches it or not at runtime (applying process)
    - Structural query match
    - Syntax modification match