What is Insecure Deserialization?

```
$user→name = "carlos";
$user→isLoggedIn = true;

O:4:"User":2:{s:4:"name":s:6:"carlos"; s:10:"isLoggedIn":b:1;}
```

Insecure Deserialization is a web app vulnerability where the web app uses serialized objects in its function, but does not do so in a safe manner

What are Serialized Objects?

```
$user→name = "carlos";
$user→isLoggedIn = true;

O:4:"User":2:{s:4:"name":s:6:"carlos"; s:10:"isLoggedIn":b:1;}
```

Serialization is the process of converting programming objects into formats better suited to transmit between applications or across networks

What are Serialized Objects?

```
unserialize('O:4:"User":2:{s:4:"name":s:6:"carlos";
                s:10:"isLoggedIn":b:1;}')
```

```
$user->name = "carlos";
$user->isLoggedIn = true;
```

When the app needs to use a serialized object, it will use the **unserialize** function to convert it back into a PHP object

Why Can Deserialization be Insecure?

```
O:4:"User":2:{s:4:"name":s:6:"carlos"; s:7:"isAdmin":b:1;}
```

If users can get access to serialized objects and modify them, then an app could be vulnerable to an Insecure Deserialization attack

Why Can Deserialization be Insecure?

```
O:4:"User":2:{s:4:"name":s:6:"carlos"; s:7:"isAdmin":b:1;}
```

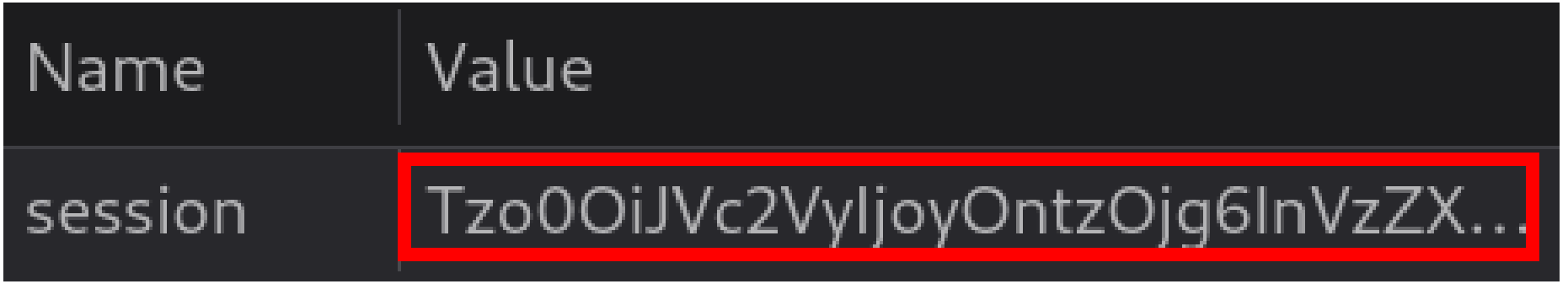
And such an attack could lead to unauthorized access, denial of service, remote code execution, and more

Lab: Modifying serialized objects

Let's learn about identifying and modifying serialized objects with a lab from Portswigger:

https://portswigger.net/web-security/deserialization/exploiting/lab-deserialization-modifying-serialized-objects

Serialized Objects in Cookies



One possible location users can get access to an app's serialized objects is if they are passed to the web browser as cookies

Serialized Objects in Cookies

```
O:4:"User":2:{s:8:"username";s:6:"wiener";s:5:"admin";b:0;}
```

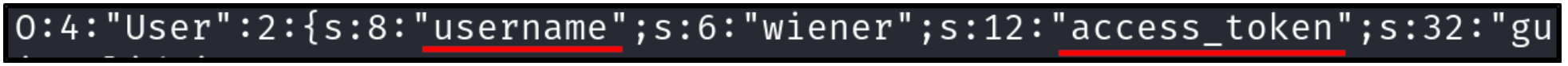
If we decode this cookie from base64, we see that it is a serialized object

Lab: Modifying serialized data types

Let's learn about modifying serialized data types with a lab from Portswigger:

https://portswigger.net/web-security/deserialization/exploiting/lab-deserialization-modifying-serialized-data-types

Modifying Serialized Objects



In this lab, the serialized object needs to be modified, and we need to modify the **username** and **access_token** value as well, since the admin user wouldn't have the same token as wiener

Lab: Modifying serialized data types

Let's learn about using deserialization with app functions using a lab from Portswigger:

https://portswigger.net/web-security/deserialization/exploiting/lab-deserialization-using-application-functionality-to-exploit-insecure-deserialization

Modifying App Settings with Serialized Objects

```
O:4:"User":3:{s:8:"username";s:6:"wien
tar_link";s:19:"users/wiener/avatar";}
```

In this lab, the app deletes all files associated with a user when the delete account function is used, and we can also specify a filepath through the serialized cookie we use

Lab: Arbitrary object injection in PHP

Let's learn about using deserialization and object injection with a lab from Portswigger:

https://portswigger.net/web-security/deserialization/exploiting/lab-deserialization-arbitrary-object-injection-in-php

Injecting Objects Through Deserialization

```
class CustomTemplate {
    private $template_file_path;
    private $lock_file_path;
```

```
function __destruct() {
    // Carlos thought this would be a good idea
    if (file_exists($this->lock_file_path)) {
        unlink($this->lock_file_path);
```

In this lab, through the deserialized cookie, we can inject a new object to achieve our goals

Injecting Objects Through Deserialization

```
class CustomTemplate {
    private $template_file_path;
    private $lock_file_path;
```

```
function __destruct() {
    // Carlos thought this would be a good idea
    if (file_exists($this->lock_file_path)) {
        unlink($this->lock_file_path);
```

The unlink() function in PHP deletes files