1. **Summary**

This Penetrtion test was performed on 10/05/2016.The detailed report about each task and our findings are described below.

The purpose of this test is to determine vulnerablities in the web application named Badstore. BadStore is an insecure application used for demonstration, security training, and testing purposes.The test are carried out assumeing the identity of an attacker or a user with malcious intent.

**3 Objective**

The objective of this test was to determine security vulnerabilities in the web server configuration and

website running on the server. The tests were carried out assuming the identity of an attacker or with

malicious intent. At the same time due care was taken not to harm the web server.

**2 Scope**

The scope of this penetration test was limited to the below mentioned IP addresses.

IP-http://192.168.0.113/

Name-badstore.net

3 Key Findings

**3.1 XSS**

Description : Cross-Site Scripting (XSS) attacks are a type of injection, in which malicious scripts are injected into otherwise benign and trusted web sites. XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user.

Affected Parameters-name, email, comments.

Affected Url-

1.http://192.168.2.22/cgi-bin/badstore.cgi?searchquery=%3Cscript%3E&action=search&x=18&y=16

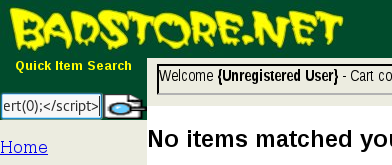
2.http://192.168.2.22/cgi-bin/badstore.cgi?action=doguestbook

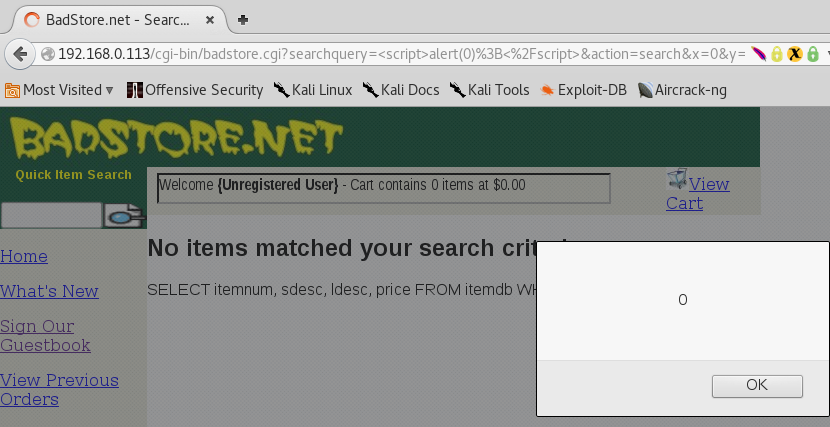
Affected method-GET, POST

Attack vector- <script>alert(0)</script>

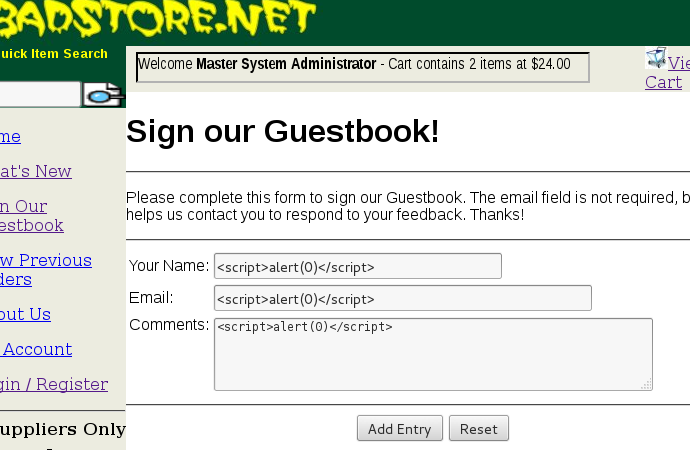
Analysis :

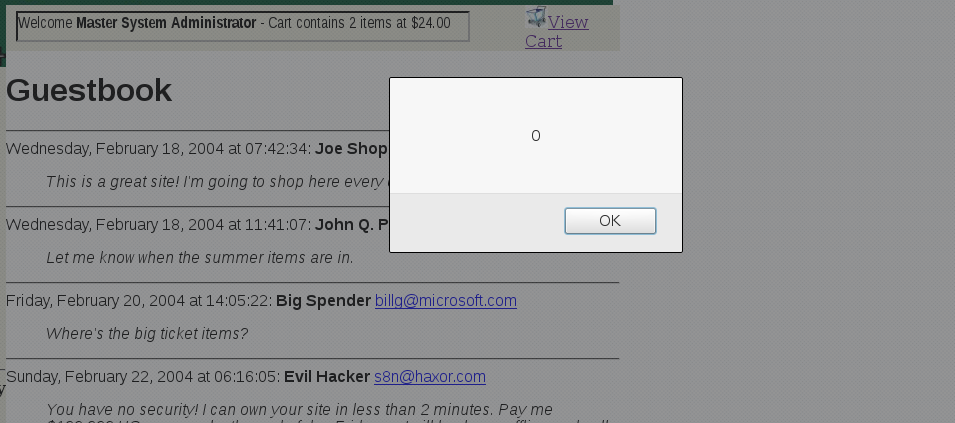
1. During analysis it was found that ‘searchquery’ parameter was vulnerable to XSS.





2. It was also found that parameters ‘name’, ‘email’, ‘comments’ were also vulnerable to XSS





Impact-

Cross-Site Scripting typically involves executing commands in a user's browser to display unintended content, or with the intent of stealing the user's login credentials or other personal information. This information can then be used by the attacker to access web sites and services for which the compromised credentials are valid (e.g., identity theft). In some cases, the attacker might be able to use this information to hijack or further compromise the user's HTTP sessions.

Recommendation:

The following general recommendations can help mitigate the risk associated with Cross-Site Scripting vulnerabilities.

1. Ensure that your web application validates all forms, headers, cookie fields, hidden fields, and parameters, and converts scripts and script tags to a non-executable form.  
2.Ensure that any executables on your server do not return scripts in executable form when passed scripts as malformed command parameters.  
3.Consider converting JavaScript and HTML tags into alternate HTML encodings (such as "<" to "&lt;>.  
4.If your site runs online forums or message boards, disallow the use of HTML tags and Scripting in these areas.

**3.2 SQL Injection**

**Description:** SQL injection is a [code injection](https://en.wikipedia.org/wiki/Code_injection) technique, used to [attack](https://en.wikipedia.org/wiki/Attack_(computing)) data-driven applications, in which malicious [SQL](https://en.wikipedia.org/wiki/SQL) statements are inserted into an entry field for execution.It is mostly known as an attack [vector](https://en.wikipedia.org/wiki/Vector_(malware)) for websites but can be used to attack any type of SQL database.

Affected Parameters-email, passwd.

Affected Url

1. http://192.168.2.22/cgi-bin/badstore.cgi?action=supplierportal

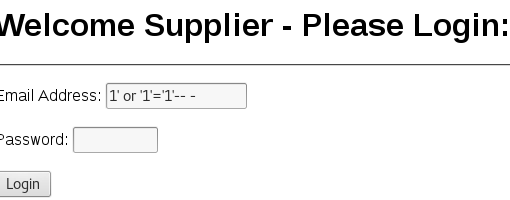
2.http://192.168.2.66/cgi-bin/badstore.cgi?action=login

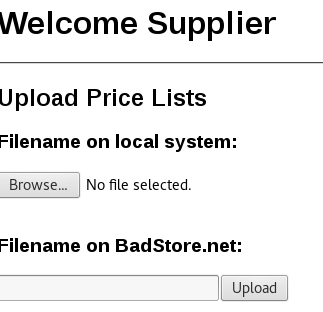
Affected method- POST

Attack vector- 1’ or ‘1’=’1-- -, admin’ or ‘1’=’1’

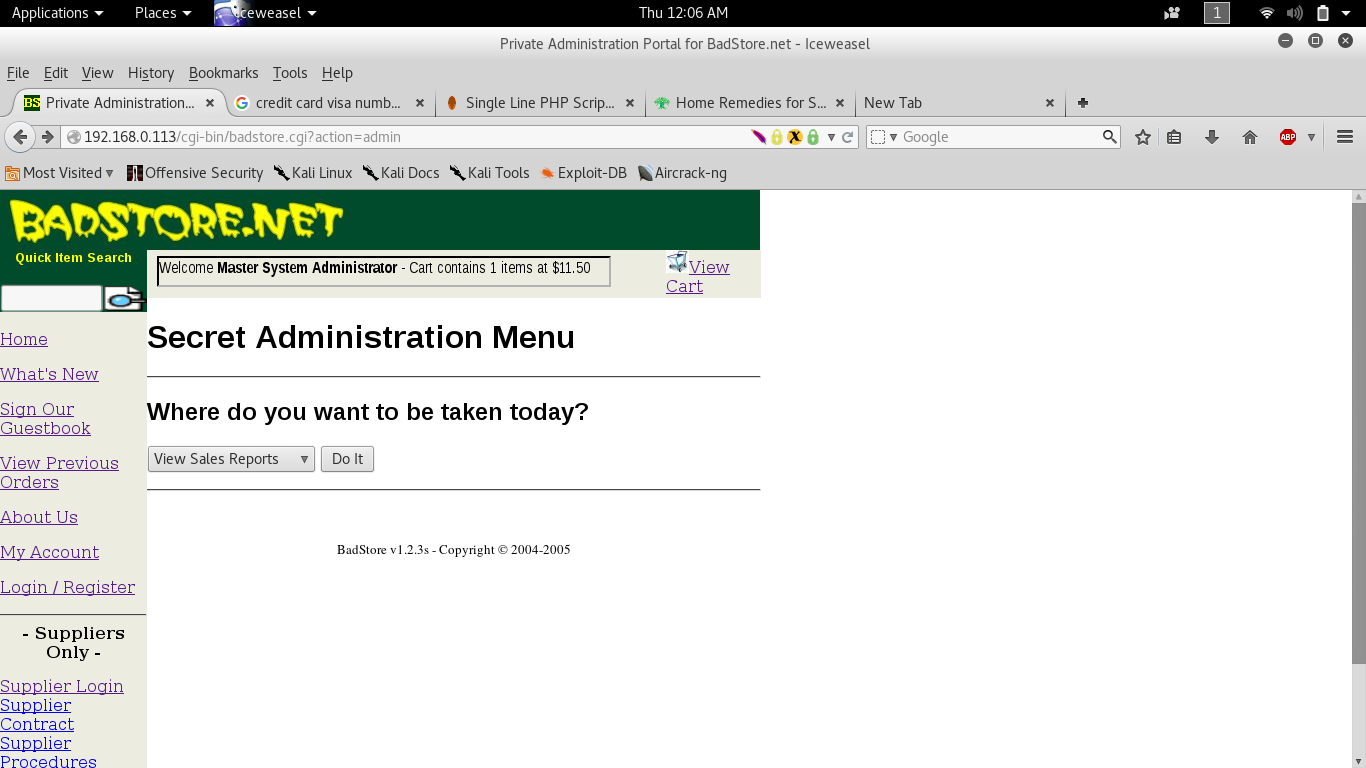
Analysis :

1. During analysis it was found that parameters ‘email’ and ‘passwd’ were vulnerable to sql injection.





**2. It was found that on doing sql injection we can access secret admin portal.**



**Impact-An attacker can use SQL injection to bypass authentication or even impersonate specific users.**

* One of SQL’s primary functions is to select data based on a query and output the result of that query. An SQL injection vulnerability could allow the complete disclosure of data residing on a database server.
* Since web applications use SQL to alter data within a database, an attacker could use SQL injection to alter data stored in a database. Altering data affects data integrity and could cause repudiation issues, for instance, issues such as voiding transactions, altering balances and other records.
* SQL is used to delete records from a database. An attacker could use an SQL injection vulnerability to delete data from a database. Even if an appropriate backup strategy is employed, deletion of data could affect an application’s availability until the database is restored.

Recommendation:

1. Use of Prepared Statements (Parameterized Queries)  
2. Use of Stored Procedures   
3. Escaping all User Supplied Input   
4. Also Enforce: Least Privilege  
5. Also Perform: White List Input Validation

# 3.3 Sensitive Data Exposure

Description: A data breach that is intentional or unintentional release of [secure information](https://en.wikipedia.org/wiki/Secure_information) to an untrusted environment.

Affected Parameters -

1. http://192.168.2.22/supplier/accounts

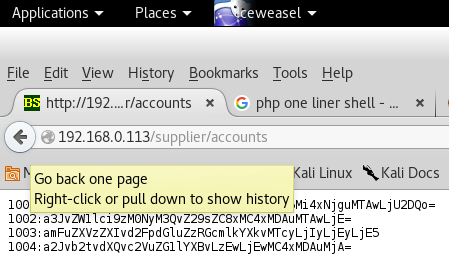
Affected Url- N/A

Affected method- N/A

Attack vector- N/A

Analysis:

1. It was found that the information kept about supplier accounts are simply Base64 encoded.



Impact-

1. Information leakage.  
2.Identity theft.

Recommendation:

1.Pages containing critical account details should not be kept on the web server.  
2. Critical information should be kept in either encrypted or hashed form, depending upon the usage.

3.4 Server Misconfiguration

Description: Server Misconfiguration attacks exploit configuration weaknesses found in web servers and application servers. Many servers come with unnecessary default and sample files, including applications, configuration files, scripts, and web pages. They may also have unnecessary services enabled, such as content management and remote administration functionality.

Affected Parameters- N/A

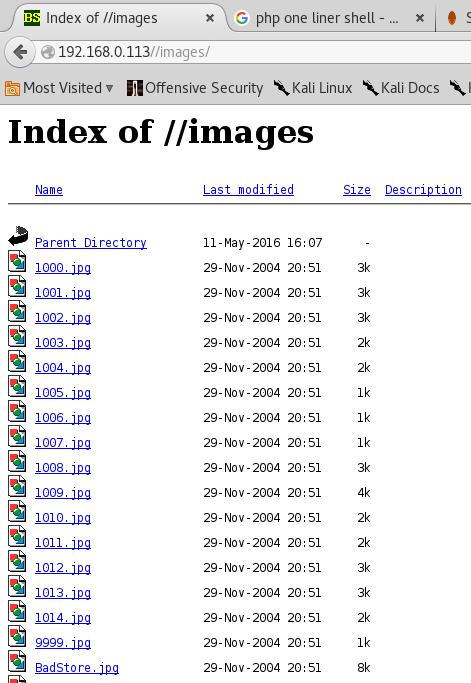
Affected Url-http://192.168.0.113/images/

Affected method- N/A

Attack vector- dirbuster

Analysis:

1. Directory listing was found in the ‘image’ folder.



Impact-

Exposing the contents of a directory can lead to an attacker gaining access to source code or providing useful information for the attacker to devise exploits, such as creation times of files or any information that may be encoded in file names. The directory listing may also compromise private or confidential data.

Recommendation-

Recommendations include restricting access to important directories or files by adopting a need to know requirement for both the document and server root, and turning off features such as Automatic Directory Listings that could expose private files and provide information that could be utilized by an attacker when formulating or conducting an attack.

**3.6 Back button browsing**

Description: Browsers have the ability to maintain a recent record of pages that were visited by a user. The Back and Forward buttons on browsers use this functionality to display the pages recently browsed. In addition, browsers also keep track of variables like username, password, credit card details, etc. that were POSTed to the server while fetching the page. If a user logs in to the website, performs some actions and then logs out, and an adversary has access to the same machine as the user, he can see the logout page that is displayed on the browser window. He can then click the Back button until he reaches the page shown after a successful login. Here, the attacker can click the Refresh button, and the browser automatically resubmits the request with all the information.

Affected Parameters -N/A

Affected Url-N/A

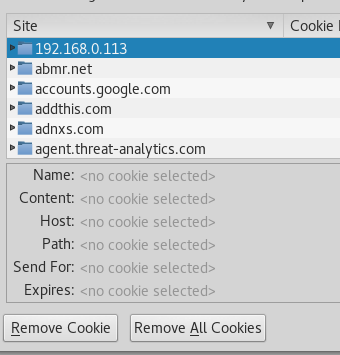
Affected method – N/A

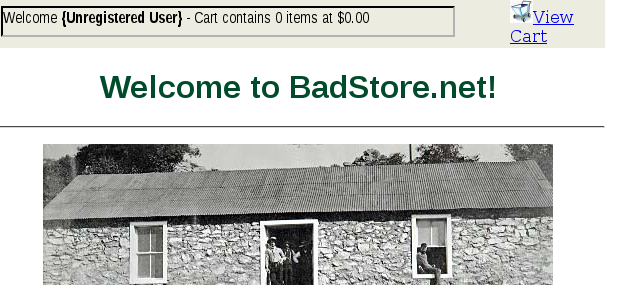
Attack vector-N/A

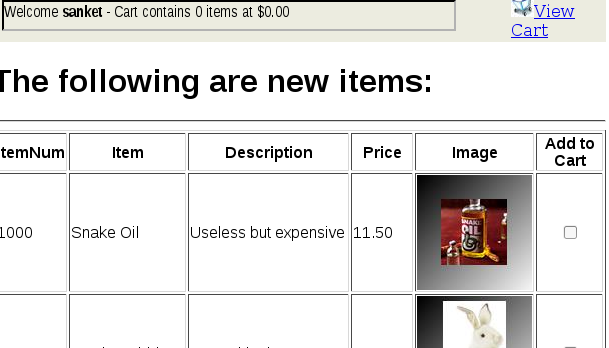
Analysis :

1. It is found that after logging in, we delete the cookie to log out. But on pressing the back button of browser we are able to see the cart of that user.









Impact-

1. Information leakage.

Recommendation:

This problem can be mitigated by setting proper cache control attributes in the response header.  
Mainly there are two types of cache attributes:  
1. Cache-control: no-cache  
The no-cache attribute indicates that the browser should not use the information that is cached for that particular request–response pair. The browser stores the cache, but instead of showing the content from the cache, it sends the request to the server each time. But again, the cache will be only be in the browser and can be easily accessed by an attacker or malicious user.  
2. Cache-control: no-store  
The no-store attribute indicates that the request–response pair should not be cached and stored in the browser. This applies to the entire page.

**3.7 Improper Handling of file upload**

**Description:** Uploaded files represent a significant risk to applications. The first step in many attacks is to get some code to the system to be attacked. Then the attack only needs to find a way to get the code executed. Using a file upload helps the attacker accomplish the first step.

Affected Parameters- N/A

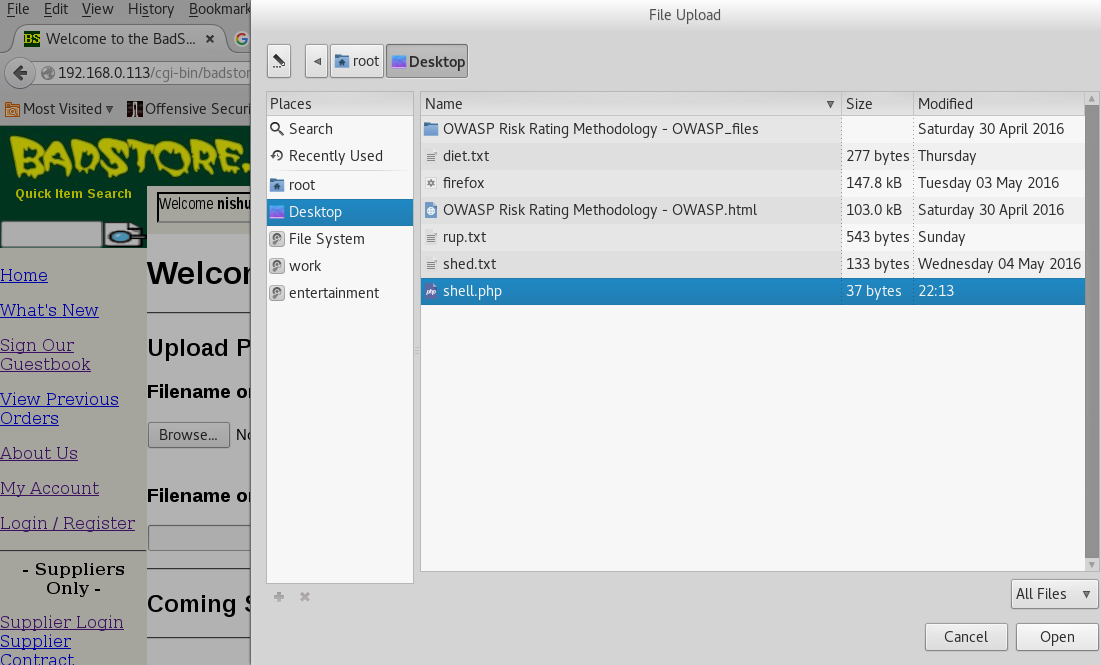
Affected Url- http://192.168.2.22/cgi-bin/badstore.cgi?action=supplierportal

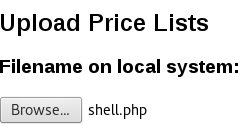
Affected method - GET

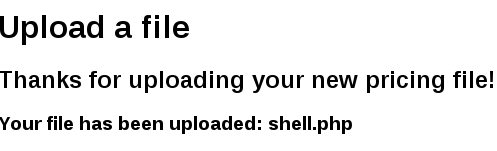
Attack vector- shell.php

Analysis -

1.It is found MIME types are not being checked while uploading a file. We can upload any file with file extension.







**Impact-**

**1.** The web server can be compromised by uploading and executing a web-shell which can run commands, browse system files, browse local resources, attack other servers, and exploit the local vulnerabilities, and so forth. This may also result in a defacement. **2.** An attacker might be able to put a phishing page into the website.

**Recommendation-**

**1.** Never accept a filename and its extension directly without having a white-list filter.2. It is necessary to have a list of only permitted extensions on the web application. And, file extension can be selected from the list. For instance, it can be a “select case” syntax (in case of having VBScript) to choose the file extension in regard to the real file extension.  
3. Limit the filename length. For instance, the maximum length of the name of a file plus its extension should be less than 255 characters (without any directory) in an NTFS partition.  
4.Uploaded directory should not have any “execute” permission.

**3.8 Improper Password Reset Functonality**

**Description:**

**It is common for an application to have a mechanism that provides a means for a user to gain access to their account in the event they forget their password. Very often the password recovery mechanism is weak, which has the effect of making it more likely that it would be possible for a person other than the legitimate system user to gain access to that user's account.**

Affected Parameters- email, pwdhint, DoMods=Reset+User+Password

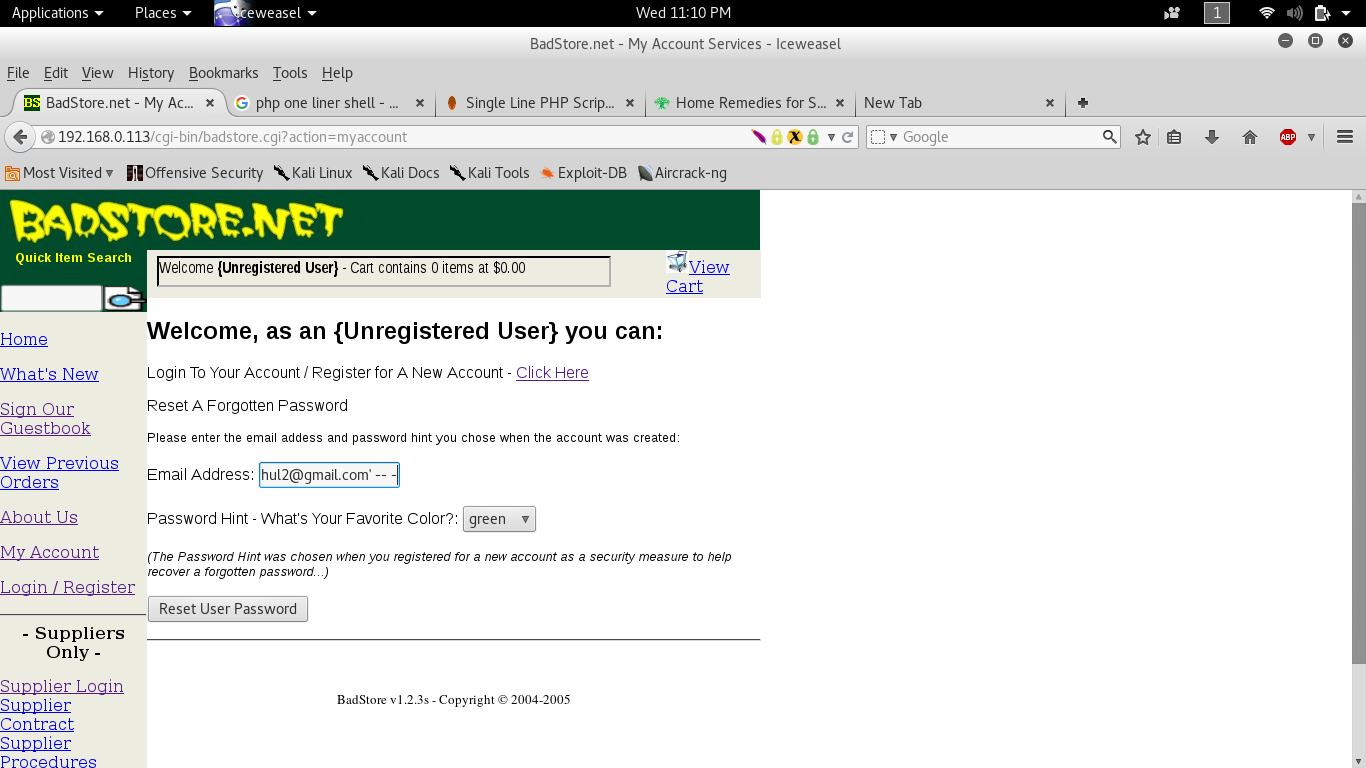
Affected Url- http://192.168.2.22/cgi-bin/badstore.cgi?action=myaccount

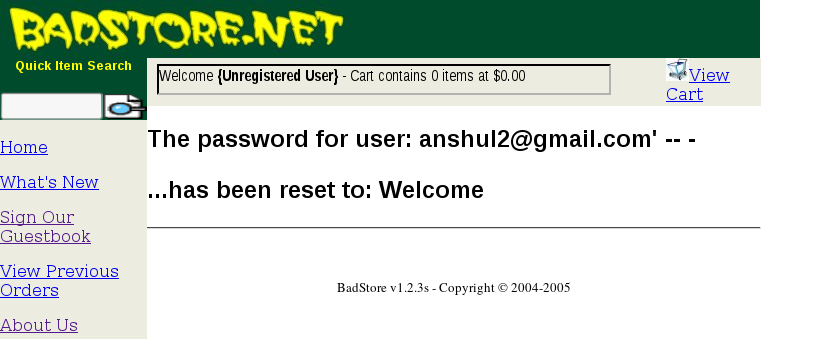
Affected method -POST

Attack vector- N/A

Analysis:

1.It was found that for any user ,the password was being changed to ‘welcome’.





Impact-

1. Bypass protection mechanism.  
2. Gain privileges.  
3. Identity theft.

Recommendation**-**

1. Do not use standard weak security questions and use several security questions.  
2. Make sure that there is throttling on the number of incorrect answers to a security question. Disable the password recovery functionality after a certain (small) number of incorrect guesses.  
3.Never allow the user to control what e-mail address the new password will be sent to in the password recovery mechanism.  
4. Assign a new temporary password rather than revealing the original password.

**3.9 Business logic flaw**

**Description:**

Most security problems are weaknesses in an application that result from a broken or missing security control (authentication, access control, input validation, etc...). By contrast, business logic vulnerabilities are ways of using the legitimate processing flow of an application in a way that results in a negative consequence to the organization. For example:

* Purchase orders are not processed before midnight
* Written authorization is not on file before web access is granted
* Transactions in excess of $2000 are not reviewed by a person

Affected Parameters- cartitem, Place+Order=Place+Order

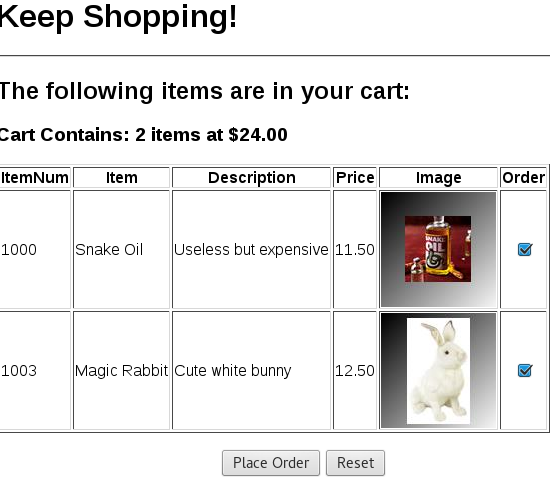
Affected Url- http://192.168.2.66/cgi-bin/badstore.cgi?action=submitpayment

Affected method - POST

Attack vector- N/A

Analysis-

1.It is found that on changing the value of parameter ‘cartid’ using a proxy, we can order different product with the cost of the previous order that we selected.









**Impact-**

1.May lead to huge financial losses.

Recommendation**-**

1. In forged and predictive parameter request testing, we verify that the application does not allow users to submit or alter data to any component of the system that they should not have access to, are accessing at that particular time or in that particular manner. This is important because without this safeguard attackers may be able to “fool/trick” the application into letting them into sections of thwe application of system that they should not be allowed in at that particular time, thus circumventing the applications business logic workflow.  
2. In integrity check and tamper evidence testing, we verify that the application does not allow users to destroy the integrity of any part of the system or its data. This is important because without these safe guards attackers may break the business logic workflow and change of compromise the application/system data or cover up actions by altering information including log files.

**4.0 Critical Data in Session ID**

**Description-**

In this Critical information such as username and password are sent in the Session ID.

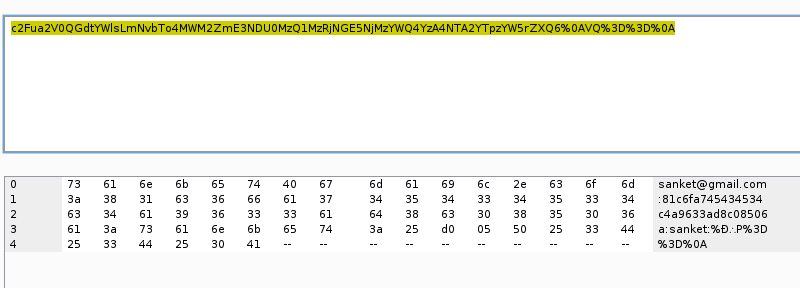
Affected Parameters :N/A

Affected Url:N/A

Affected method :N/A

Attack vector:N/A

Analysis:



Impact-

1.identity theft.  
2.Information leakage.

Recommendation:

1. session id should not contain any critical information such as credentials.  
2.session id should be atleast 128 characters long.

**4.1 Same Session ID for Multiple Logins**

**Description:**

In this Same session ID are alloted to a particular user each time he logs in.

Affected Parameters- N/A

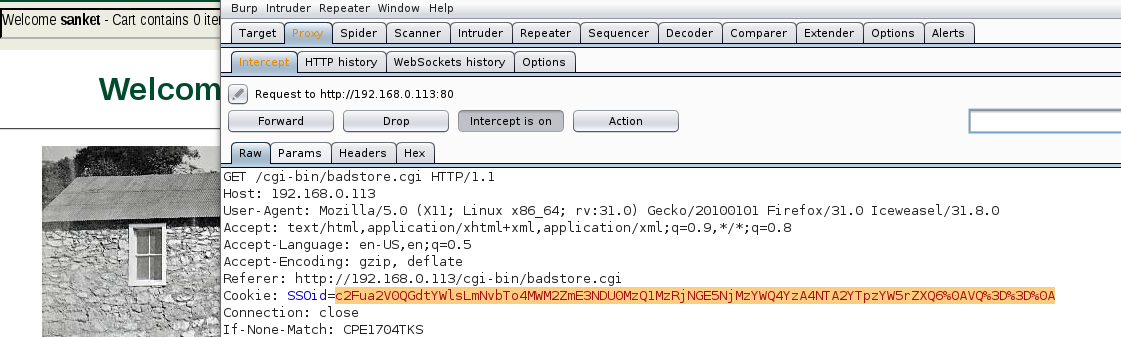
Affected Url- N/A

Affected method :N/A

Attack vector:N/A

Analysis:

1. It is found that a user is getting same session id every time he logs in.



Impact-

1. Session hijacking.  
2. Identity theft.

Recommendation-

1. Session-id should be new every time a user log in.  
2. Session-id should be random and unguessable.