

1 PROBLEM STATEMENT

To identify the vulnerabilities of the website for the purpose of improving the security features and creating a blockchain based website. Website is made for the registration of the passport which contains the personal details of the individual. Attack is performed on the website which is created of our own using block chain and denied its service.

2 PROBLEM DESCRIPTION

The blockchain based website is created for the applicants to apply for the purpose of getting passport. The important thing is the blockchain websites are highly secured ones because of its distributed nature and decentralization. Hacking on such websites is really impossible as it is having less vulnerability. But it is possible to down the site . The more traffic on such sites made the purpose of attack the sites and finally the site is unavailable to the applicants. The security measures developed will be the need of the day.

3 WORK FLOW

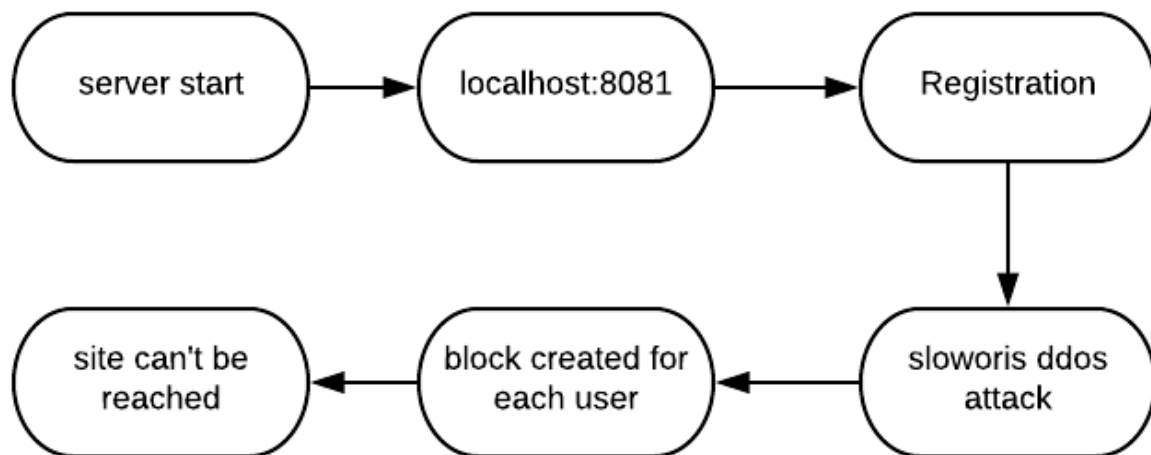


Fig 3.1 General outline of work flow.

4 INTRODUCTION

Hacking is usually legal as long as it is being done to find weaknesses in a computer or network system for testing purpose. This sort of hacking is what we call **Ethical**

4.1 Hacking

A computer expert who does the act of hacking is called a "Hacker". Hackers are those who seek knowledge, to understand how systems operate, how they are designed, and then attempt to play with these systems.

4.1.1 Advantages of Hacking

Hacking is quite useful in the following scenarios –

- To recover lost information, especially in case you lost your password.
- To perform penetration testing to strengthen computer and network security.
- To put adequate preventative measures in place to prevent security breaches.
- To have a computer system that prevents malicious hackers from gaining access.

4.1.2 Disadvantages of Hacking

Hacking is quite dangerous if it is done with harmful intent. It can cause –

- Massive security breach.
- Unauthorized system access on private information.
- Privacy violation.
- Hampering system operation.
- Denial of service attacks.
- Malicious attack on the system.

4.2 Types of attack

- SQL injection
- Phishing attack
- Malware
- DoS and DDoS attack
- Man in the middle attack
- Credential reuse
- Cross Site Scripting

4.3 Attacks

There are a number of ways in which a hacker can illegally gain access to an email account, and the majority of them rely on the behavior of the account's user.

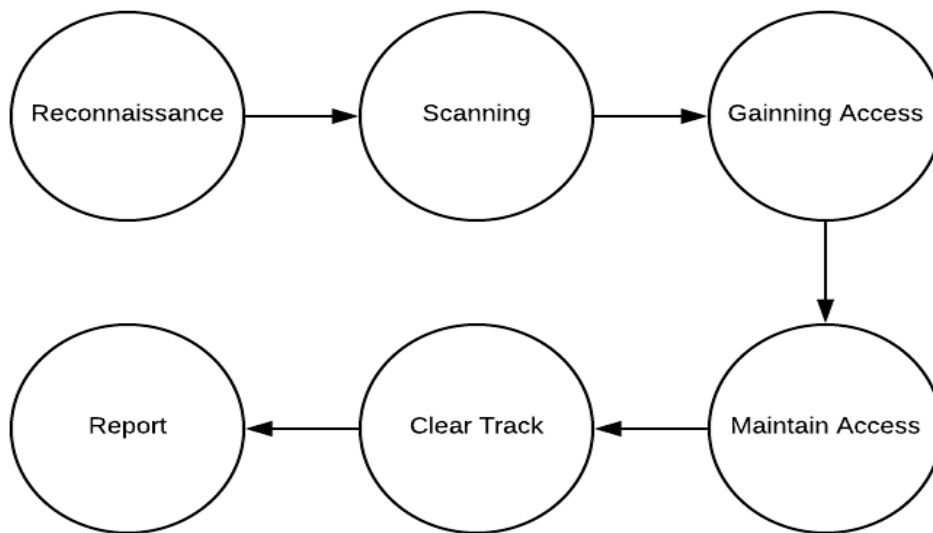


Fig 4.1 Steps in attack

5 METHODOLOGY

5.1 Block chain

A digital ledger in which transactions made in bitcoin or another cryptocurrency are recorded chronologically and publicly. Cryptography secures the records in a **blockchain** transaction, and each transaction is tied (in the chain) to previous transactions or records. Moreover, **blockchain** transactions are validated by algorithms on the nodes (computers in the network of participants in the distributed ledger).

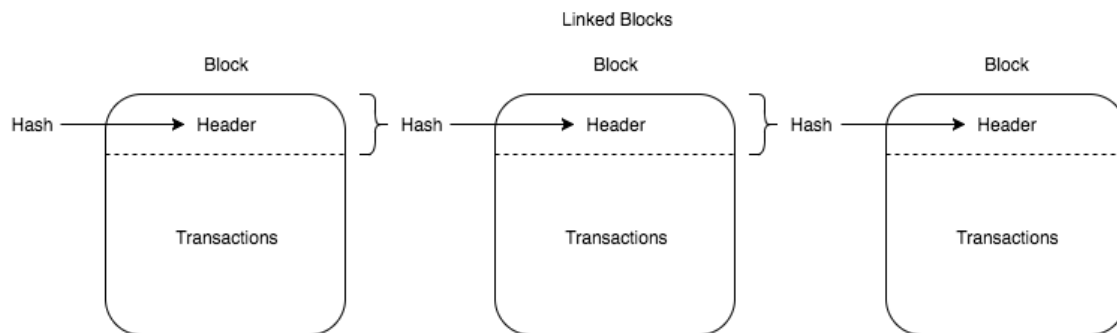


Fig 5.1 Structure of blockchain

5.2 DDOS ATTACK (Distributed Denial of Service)

In a DoS attack, an attacker with malicious intent prevents users from accessing a service. He does so by either targeting your computer and its network connection, or the computers and network of the website that you are trying to use. He can thus prevent you from accessing your email or online accounts.

In a DoS attack, the attacker sends out a flood of superfluous requests to the main server of the website in question, which basically overloads it and blocks out any further requests before the capacity is retained back. Other ways of attacking may involve preventing a particular person from accessing a certain website, obstructing the connection between two machines at the server end, therefore, disrupting the service etc.

5.2.1 HTTP Flood

HTTP flood is a type of DDoS attack in which the attacker exploits seemingly-legitimate HTTP GET or POST requests to attack a web server or application. HTTP flood attacks are volumetric attacks, often using a botnet “zombie army” a group of Internet-connected computers, each of which has been maliciously taken over, usually with the assistance of malware like Trojan Horses.

5.2.2 Ways to protect DDoS attack

1. Don't assume that only large-scale, volumetric attacks are the problem.
2. Don't rely on traffic monitoring or thresholds.
3. Make your architecture as resilient as possible.
4. Improve the security of your Internet of Things (IoT) devices.
5. Monitor traffic levels
6. Use a Content Delivery Network (CDN).
7. Use of extra bandwidth

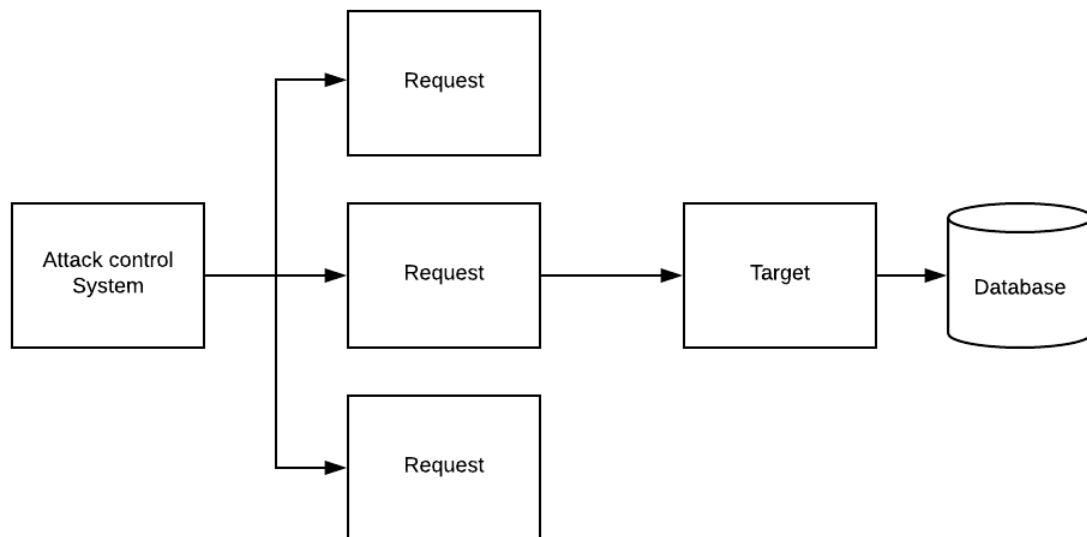


Fig 5.2 DDOS ATTACK

5.3 Burp suite

Burp or Burp Suite is a graphical tool for testing Web application security. The tool is written in Java and developed by Port Swigger Security.

Table 5.1 burpsuite tools

SNO:	TOOLS	USAGE
1.	Target	This tool contains detailed information about your target applications, and lets you drive the process of testing for vulnerabilities.
2.	Proxy	This is an intercepting web proxy that operates as a man-in-the-middle between the end browser and the target web application. it lets you intercept, inspect and modify the raw traffic passing in both directions.
3.	Spider	This is an intelligent application-aware web spider that can crawl an application to locate its content and functionality.
4.	Intruder	This is a powerful tool for carrying out automated customized attacks against web applications. it is highly configurable and can be used to perform a wide range of tasks to make your testing faster and more effective.
5.	Repeater	This is a simple tool for manually manipulating and reissuing individual http requests, and analyzing the application's responses.

6.	Sequencer	This is a sophisticated tool for analyzing the quality of randomness in an application's session tokens or other important data items that are intended to be unpredictable.
7.	Comparator	This is a handy utility for performing a visual "diff" between any two items of data, such as pairs of similar http messages.

5.4 Phishing attack

It actually means the cloning of original website into our local host to steal the information of the users of the original website. Generally try to steal the passwords, credit and debit card details etc., It is very simple in backtracking os but it may cause severe damage.

5.4.1 The 5 phases of an attack

While most organisations know what phishing is, few realise the phases of an attack and the extreme lengths to which a cyber criminal will go to initiate a phishing attack.

Step 1: Cyber thieves research using sites where information is open, such as LinkedIn, for gathering information to impersonate a business's information and find which employees to target.

Step 2: The criminals then register a domain name that appears very similar to the actual company domain.

Step 3: The chosen recipients receive a fake email that appears authentic, but has dangerous embedded links. These direct you to hoax sites where logins and personal details will be requested or a virus will be installed.

Step 4: The recipient believes the email is authentic, clicking on a dodgy link and engaging with the criminals. The phishing net has now been cast.

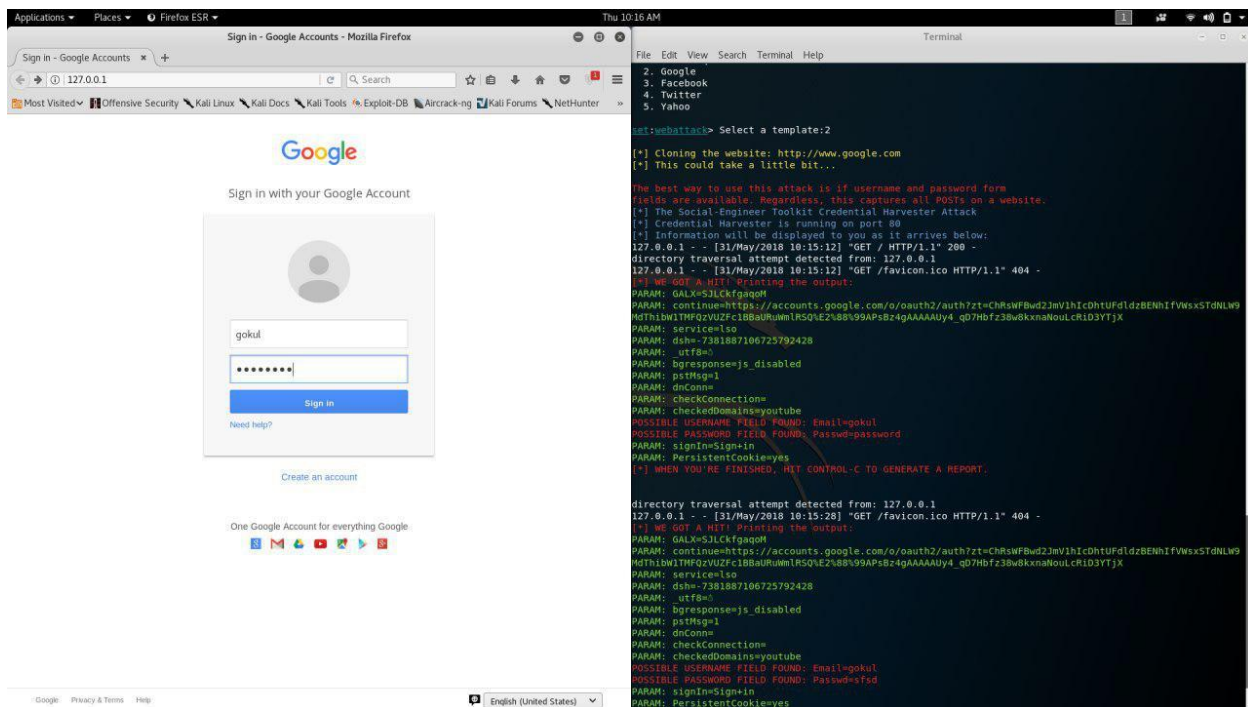


Fig 5.3 phishing attack

BackTrack has a tool to assist and automate social engineering attacks called **SET**, or the **Social Engineering Toolkit**. SET was developed by David Kennedy and simplifies a number of social engineering attacks such as phishing, spear-phishing, malicious USBs, etc. Furthermore, it has been integrated with Metasploit so that we can use Metasploit exploits and payloads in our social engineering attacks.

The current version of the Social Engineering Toolkit includes the following types of attacks.

- Spearphishing
- Websites
- Malicious USBs

6 IMPLEMENTATION

Table 6.1 tools

Operating System	Software
Kali linux	Burp suite
	Xampp server
	SET tool

6.1 Kali linux

Kali linux is a Debian-derived Linux distribution designed for digital forensics and penetration testing. Kali Linux is based on Debian Testing. Most packages Kali uses are imported from the Debian repositories. Kali Linux has over 600 preinstalled penetration-testing programs.

6.2 Burp suite:

An intercepting Proxy, which lets you inspect and modify traffic between your browser and the target application.

- An application-aware Spider, for crawling content and functionality.
- An advanced web application Scanner, for automating the detection of numerous types of vulnerability.
- An Intruder tool, for performing powerful customized attacks to find and exploit unusual vulnerabilities.
- A Repeater tool, for manipulating and resending individual requests.
- A Sequencer tool, for testing the randomness of session tokens.

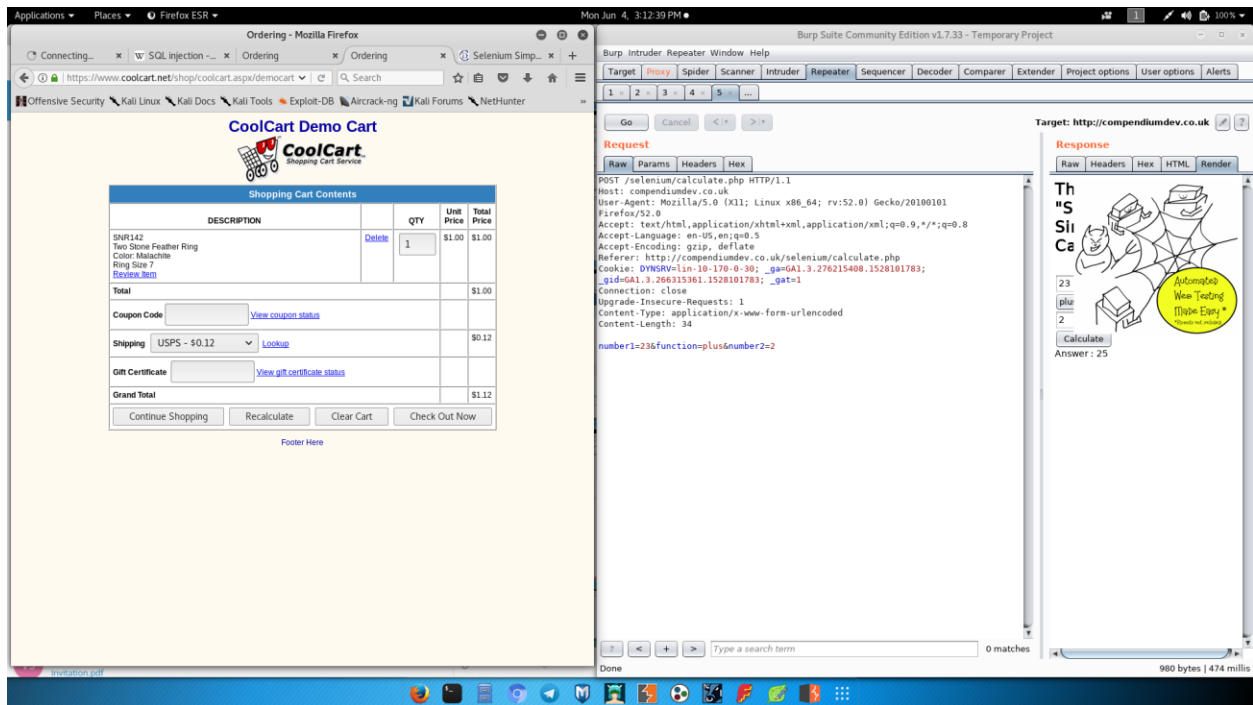


Fig 6.1 change the order value using proxy forward

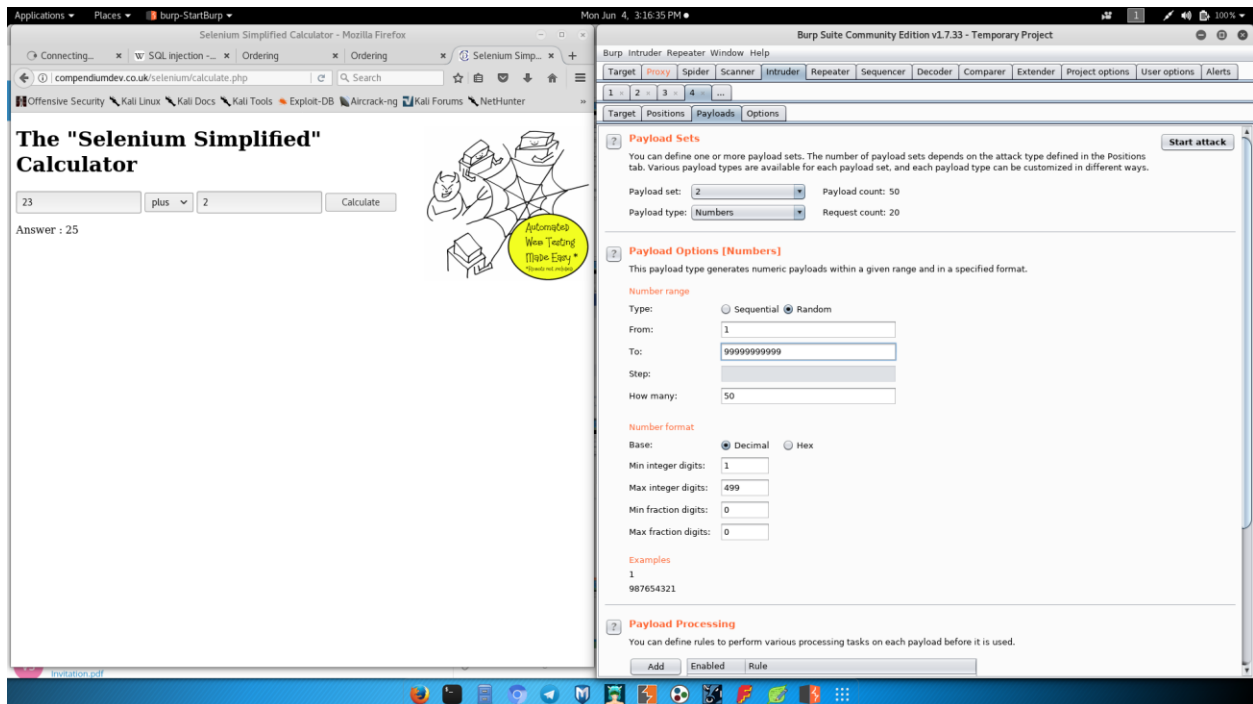


Fig 6.2 burp suite repeater

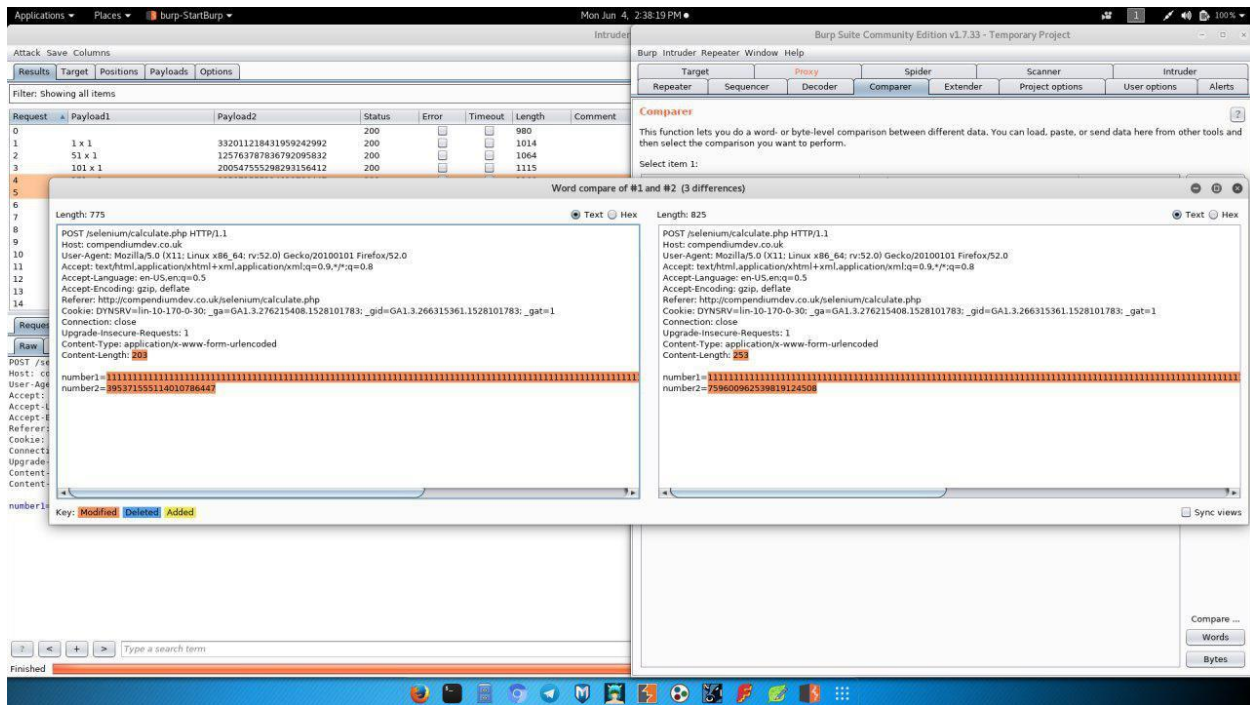


Fig 6.3 Comparator burp suite

6.3 Xampp server

XAMPP stands for x-os, apache, mysql, php , perl. (x-os means it can be used for any operating system. Xampp is a free and open source cross-platform webserver solution stack package developed by Apache Friends, consisting mainly of the Apache HTTP Server, MariaDB database, and interpreters for scripts written in the PHP and Perl programming languages.

To start the xampp server with apache and mysql

```
root@localhost: sudo /opt/lampp/lampp start
```

```
Starting XAMPP for Linux 7.2.5-0...
XAMPP: Starting Apache...ok.
XAMPP: Starting MySQL...ok.
XAMPP: Starting ProFTPD...ok.
```

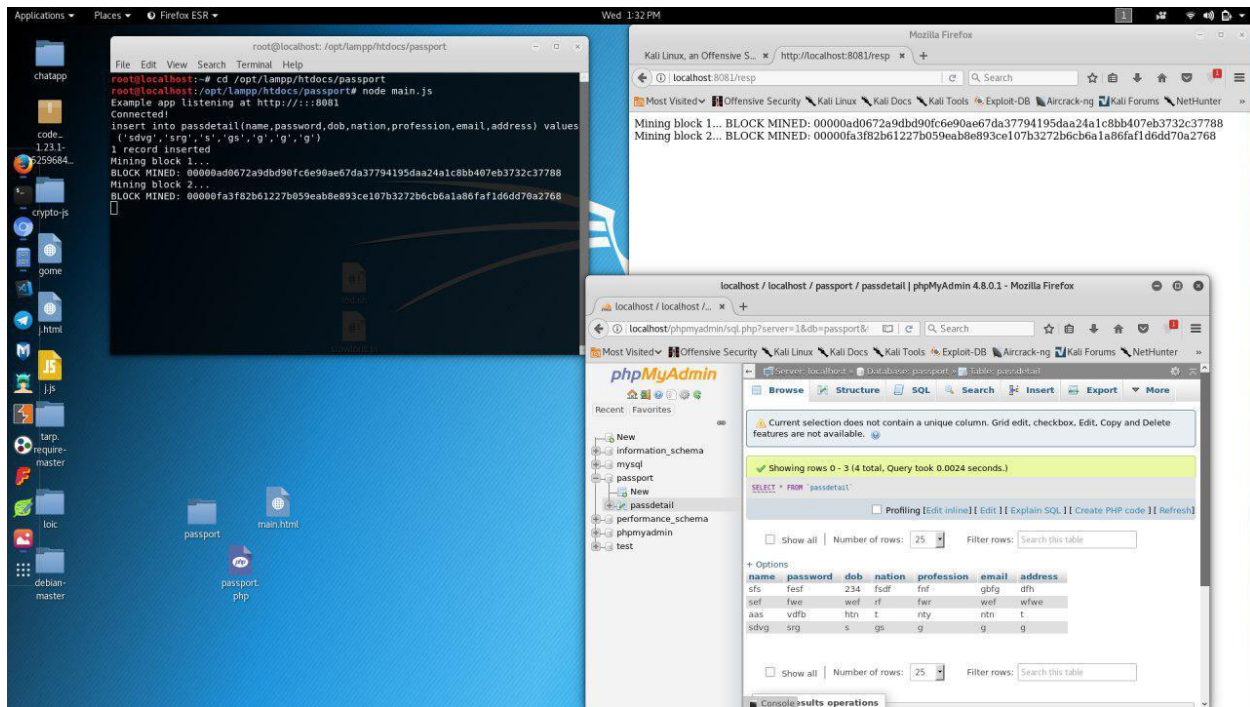


Fig 6.4 xampp server

6.4 Social Engineering Tool

The Social-Engineer Toolkit is an open-source penetration testing framework designed for Social-Engineering. SET has a number of custom attack vectors that allow you to make a believable attack in a fraction of the time.

The Social-Engineer Toolkit (SET) is specifically designed to perform advanced attacks against the human element. SET was designed to be released with the <https://www.social-engineer.org> launch and has quickly become a standard tool in a penetration testers arsenal. SET was written by David Kennedy (ReL1K) and with a lot of help from the community it has incorporated attacks never before seen in an exploitation toolset. The attacks built into the toolkit are designed to be targeted and focused attacks against a person or organization used during a penetration test.

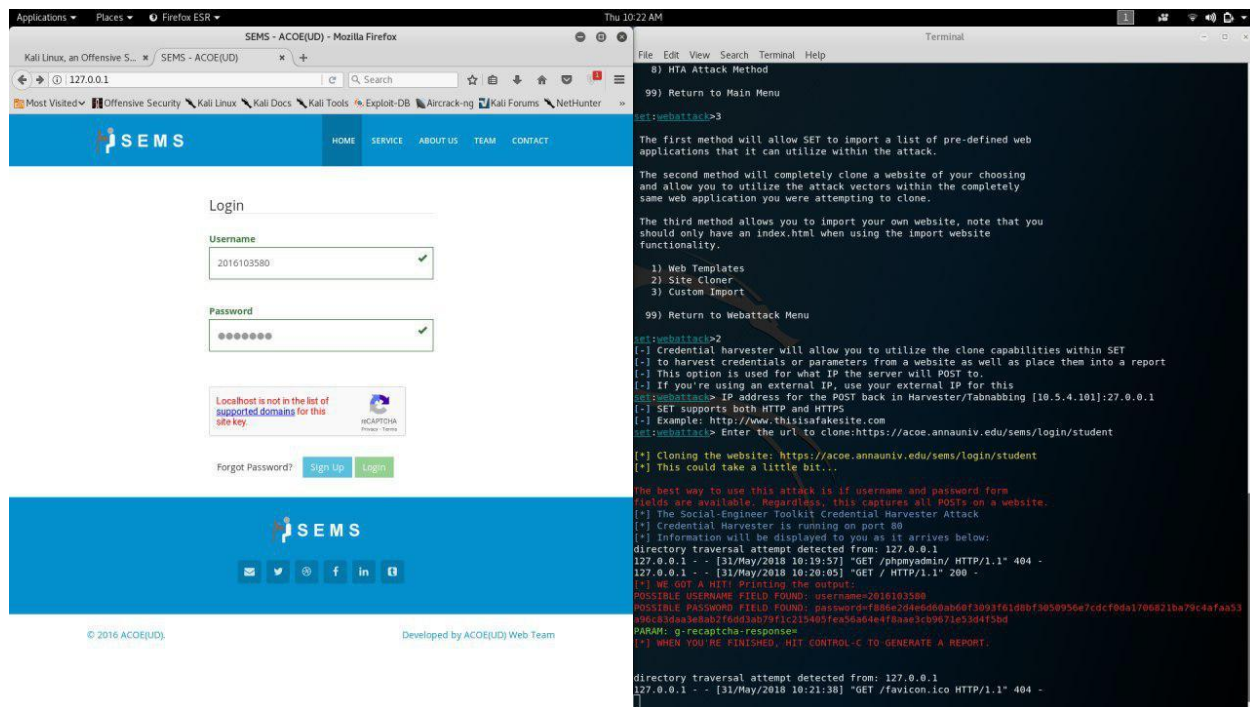


Fig 6.5 Cloning the original website

6.5 Illustration of block chain

A block chain based dynamic website is created using node js. The website consists of passport details of the user. The information includes details such as name, address, age, gender, etc. These details are stored in the database using local host port 8081.

Move to the directory which contains the node js file

```
root@localhost:cd /opt/lampp/htdocs/passport
root@localhost:node main.js
```

```
Example app listening at http://:::8081
Connected!
insert into passdetail(name,password,dob,nation,profession,email,address)
values
('Gowthamraj','1176','19','2eewq','hsjdjsj','gowthamraj06@gmail.com','ti
rupur')
1 record inserted
Mining block 1...
BLOCK MINED:
0000ad0672a9dbd90fc6e90ae67da37794195daa24a1c8bb407eb3732c37788
Mining block 2...
```

Once the user registered the details the details are stored in the data base and also its hash value and time of creation is calculated as a log. Vulnerability is identified in above created website to perform various attack mainly DDOS attack is performed and security features are added.

The javascript file is executed in the server side thus backend is triggered and also database is activated. It also makes the localhost to connect to the database otherwise xampp server-apache is not capable of connecting to the database.

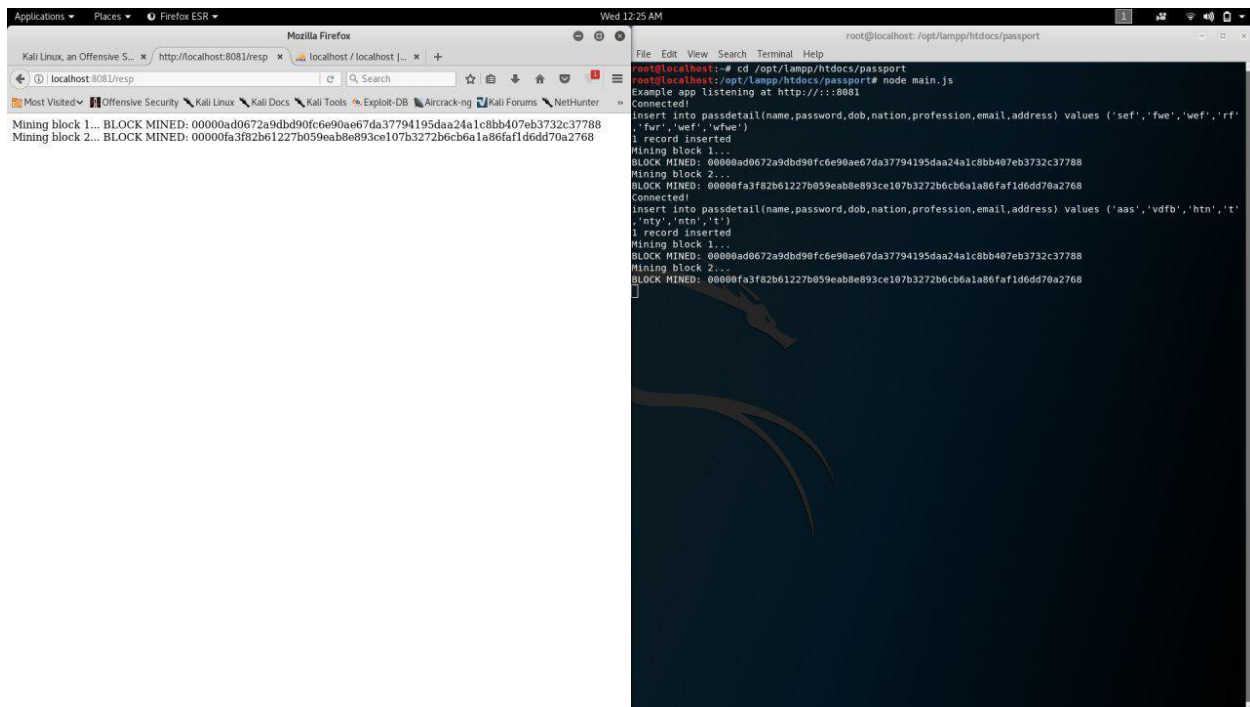


Fig 6.6 blockchain website mining using node js

7 CONCLUSION

Thus the high secured blockchain website is created and hosted in localhost 8081 port. The database is created for the use of storing the information of the user. For the successful login the web app create the hash code for the created blockchain. The hash code is displayed in the front for the referral purpose.

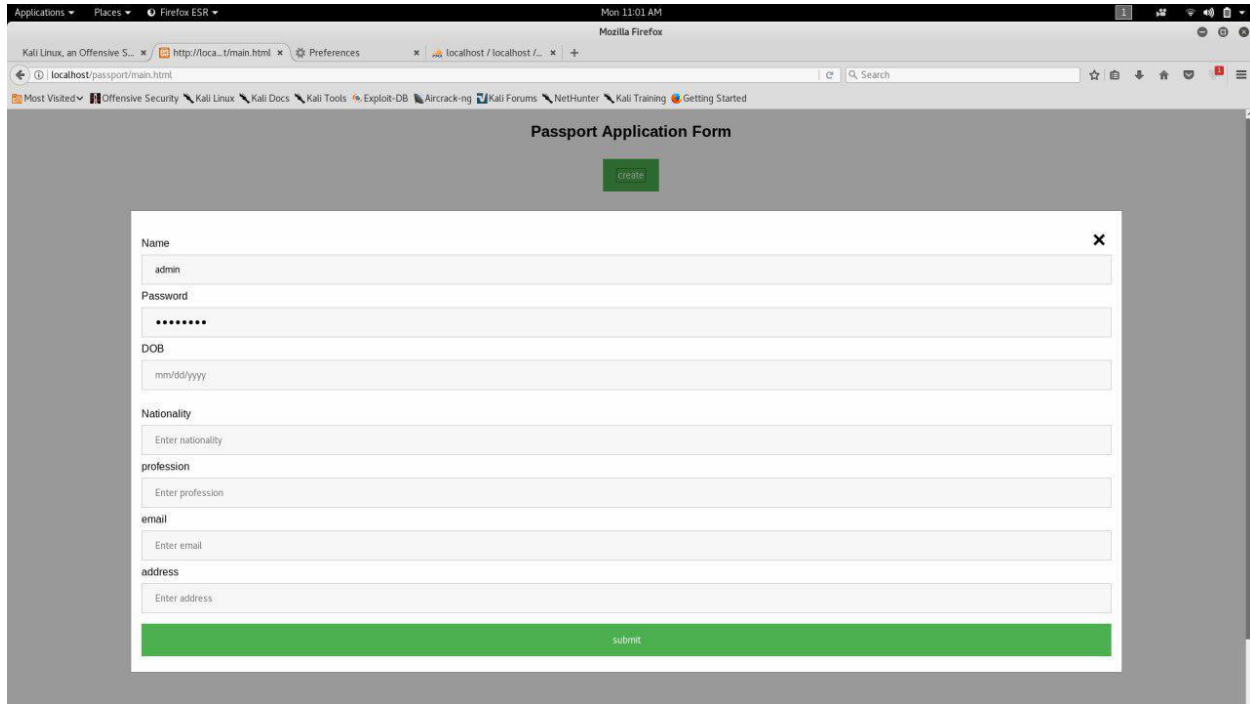
The denial of service attack is performed on it using slowloris tools. It create multiple request to the terminal and thus it slow down or affect the localhost. It prevents the the page from loading and unable the user to performany action in the web. Though it is less vulnerable there is a possibility of down the site and it is tested.

Moreover the vulnerability in the web application such as sql injection, credentials reuse, randomness in web, changing of tokens are found by performing various attack in the web by the use of tools in burp suite. Burp suite is connected to the proxy setup in the browser. Hence the request were forwarded through burp suite. Various tools in burp suite such as intruder, repeater, scanner, comparator helps to catch the packets that are transfered in the burp suite. These packets contain the cookies, tokens and param. These are

We also done the phishing attack (website cloning) using SET tools in kali linux. We cloned the original website and hosted in our localhost using social engineering tool in kali. This website looks like a normal website expect the address. When a user entre the credentials in the login field the forwarded to the original page and the credentials are stored in the attacker's terminal.

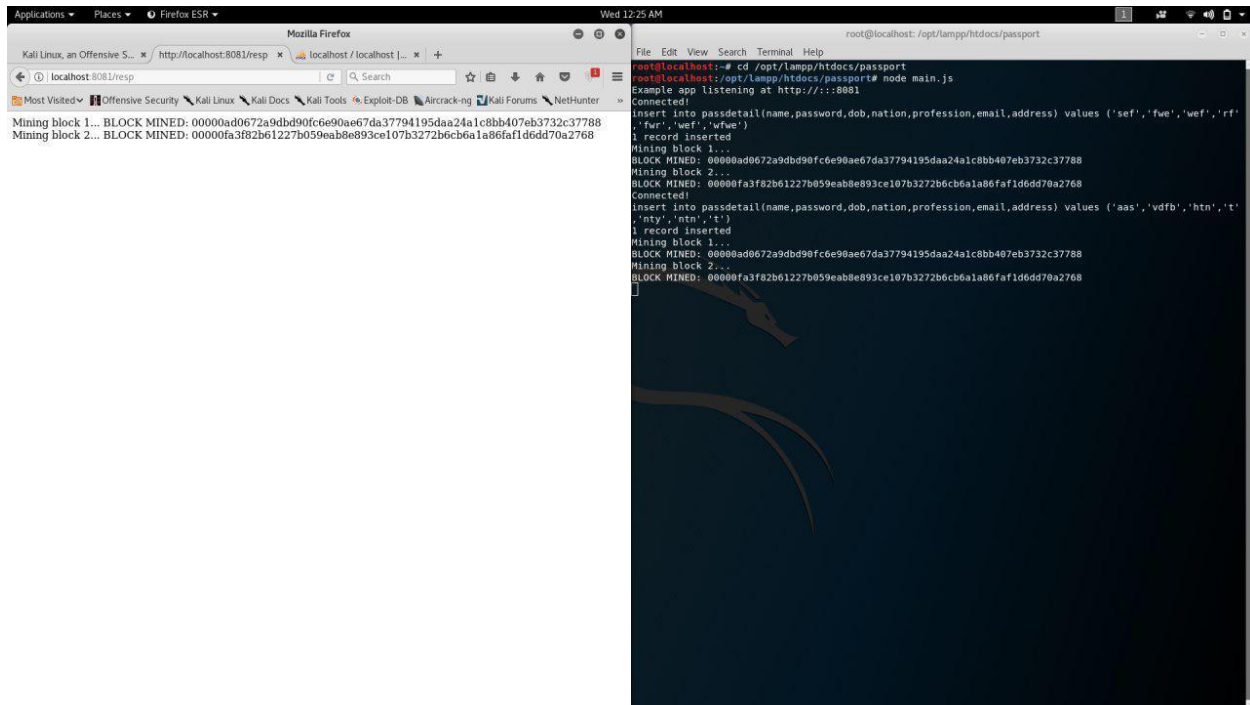
Thus we performed various activities in the ethical hacking by the use of various resources which are mention below. We came to know about the working of network, web app, and the security features involved.

8 RESULT AND SCREENSHOTS



The screenshot shows a web browser window with the title "Passport Application Form". The browser's address bar shows "localhost/passport/main.html". The form itself is a white box with a green "create" button at the top. It contains several input fields: "Name" (with "admin" entered), "Password" (with "*****" entered), "DOB" (with "mm/dd/yyyy" entered), "Nationality" (with "Enter nationality" entered), "profession" (with "Enter profession" entered), "email" (with "Enter email" entered), and "address" (with "Enter address" entered). A green "submit" button is at the bottom of the form.

Fig 8.1 passport application



The screenshot shows a terminal window with the following output:

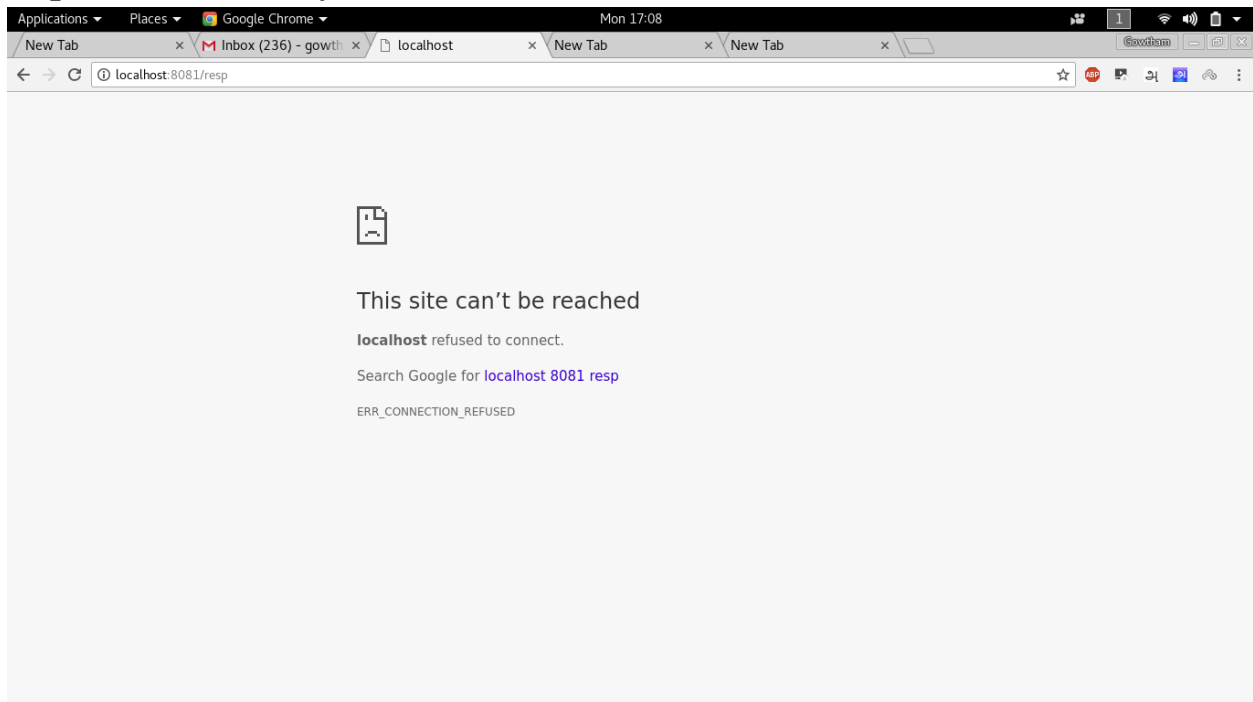
```
root@localhost: /opt/lampp/htdocs/passport
root@localhost:~# cd /opt/lampp/htdocs/passport
root@localhost:~# cd /opt/lampp/htdocs/passport
root@localhost:~# node main.js
Example app listening at http://:::8081
Connected!
Insert into passportdetail(name,password,dob,nation,profession,email,address) values ('sef','fwe','wef','rfr','fur','sef','ufwe')
1 record inserted
Mining block 1...
BLOCK MINED: 00000ad0672a9dbd90fc6e90ae67da37794195daa24a1c8bb407eb3732c37788
Mining block 2...
BLOCK MINED: 00000fa3f82b61227b059eab8e893ce107b3272b6cb6a1a86faf1d6dd70a2768
Connected!
Insert into passportdetail(name,password,dob,nation,profession,email,address) values ('aas','vdfb','htn','t','sty','nta','st')
1 record inserted
Mining block 1...
BLOCK MINED: 00000ad0672a9dbd90fc6e90ae67da37794195daa24a1c8bb407eb3732c37788
Mining block 2...
BLOCK MINED: 00000fa3f82b61227b059eab8e893ce107b3272b6cb6a1a86faf1d6dd70a2768
```

Fig 8.2 blocks created in blockchain and values stored in db

The following code provide DDOS attack in the blockchain website present in the localhost:

```
root@localhost:~# cd slowloris.pl
root@localhost:~/slowloris.pl# ./slowloris.pl -dns 10.5.4.205
Welcome to Slowloris - the low bandwidth, yet greedy and poisonous HTTP
client by Laera Loris
Defaulting to port 80.
Defaulting to a 5 second tcp connection timeout.
Defaulting to a 100 second re-try timeout.
Defaulting to 1000 connections.
Multithreading enabled.
Connecting to 10.5.4.205:80 every 100 seconds with 1000 sockets:
    Building sockets.
    Building sockets.
    Building sockets.
    Building sockets.
    Building sockets.
    Building sockets.
    Building sockets.
    Building sockets.
    Building sockets.
    Building sockets.
```

Finally the site cannot be reached because the overflow of the traffic of the request to the site by the attacker



9 REFERENCES

Online refernces

- a. For installation of burpsuite:
<https://support.portswigger.net/customer/portal/topics/718317-installing-and-configuring-burp/articles>
- b. For CA certificate:
https://portswigger.net/burp/help/proxy_options_installingcacert
- c. For dvwa setup:
<https://cloudup.com/iHWKaT9dxbj>
- d. For blockchain code:
<https://github.com/SavjeeTutorials/SavjeeCoin/tree/master/src>
- e. For SET tool:
<https://github.com/trustedsec/social-engineer-toolkit/>
- f. For change the order value:
<https://www.coolcart.net/jewelrystore.html>

Book references

- a. For basic details in ethical hacking
https://www.tutorialspoint.com/ethical_hacking/ethical_hacking_tutorial.pdf
- b. For blockchain reference
<http://scet.berkeley.edu/wp-content/uploads/BlockchainPaper.pdf>
- c. For burp suite
https://cdn.ttgtmedia.com/rms/pdf/SearchSecurity.in_Burp_%20Suite_tutorial_Part_01.pdf

10 APPENDIX

Node js code

```
var express = require('express');
var app = express();
var swig = require('swig');
var mysql = require('mysql');

var bodyParser = require('body-parser');
const SHA256 = require("crypto-js/sha256");
class Block {
  constructor(index, timestamp, data, previousHash = "") {
    this.index = index;
    this.previousHash = previousHash;
    this.timestamp = timestamp;
    this.data = data;
    this.hash = this.calculateHash();
    this.nonce = 0;
  }
  calculateHash() {
    return SHA256(this.index + this.previousHash + this.timestamp + JSON.stringify(this.data) +
this.nonce).toString();
  }
  mineBlock(difficulty) {
    while (this.hash.substring(0, difficulty) !== Array(difficulty + 1).join("0")) {
      this.nonce++;
      this.hash = this.calculateHash();
    }
    console.log("BLOCK MINED: " + this.hash);
    return "BLOCK MINED: " + this.hash;
  }
}
class Blockchain{
  constructor() {
    this.chain = [this.createGenesisBlock()];
    this.difficulty = 5;
  }
  createGenesisBlock() {
    return new Block(0, "01/01/2017", "Genesis block", "0");
  }
  getLatestBlock() {
    return this.chain[this.chain.length - 1];
  }
}
```

```

    }

    addBlock(newBlock) {
        newBlock.previousHash = this.getLatestBlock().hash;
        var st = newBlock.mineBlock(this.difficulty);
        this.chain.push(newBlock);
                                return st + "\n";
    }

    isChainValid() {
        for (let i = 1; i < this.chain.length; i++){
            const currentBlock = this.chain[i];
            const previousBlock = this.chain[i - 1];

            if (currentBlock.hash !== currentBlock.calculateHash()) {
                return false;
            }

            if (currentBlock.previousHash !== previousBlock.hash) {
                return false;
            }
        }

        return true;
    }
}

function poost(nme,psw,dob,nationality,pro,email,address,res){
var con = mysql.createConnection({
    host: "localhost",
    user: "root",
    password: "",
    database:"passport"
});
var resp;
con.connect(function(err) {
    if (err) throw err;
    console.log("Connected!");
    var sql = "insert into passdetail(name,password,dob,nation,profession,email,address) values (" +
nme+', '+psw+', '+dob+', '+nationality+', '+pro+', '+email+', '+address + ')";
    console.log(sql);
    con.query(sql, function (err, result) {
        if (err) throw err;
        console.log("1 record inserted");
        res.send(oncl());
    });
});

```

```

    });
  });
}
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({
  extended: true
}));

function oncl(res){
  var resp = "";
  let savjeeCoin = new Blockchain();
  console.log('Mining block 1...');
  resp += 'Mining block 1...\n';
  resp += savjeeCoin.addBlock(new Block(1, "20/07/2017", { amount: 4 }));

  console.log('Mining block 2...');
  resp += 'Mining block 2...\n';

  resp += savjeeCoin.addBlock(new Block(2, "20/07/2017", { amount: 8 }));
  return resp;
}

app.post('/resp',function(req,res){
  var name="" + req.body.nme + "";
  var pass="" + req.body.psw + "";
  var dob="" + req.body.dob + "";
  var nation="" + req.body.nationality + "";
  var pro="" + req.body.pro + "";
  var email="" + req.body.email + "";
  var address="" + req.body.address + "";
  var resp = poost(name,pass,dob,nation,pro,email,address,res);
});

app.get('/',function(req,res){
  var s = swig.renderFile('/opt/lampp/htdocs/passport/main.html',{ });
  res.send(s);
});

app.get('/oncl',function(req,res){
  var response = oncl();
  res.send(response);
});

var server = app.listen(8081, function () {
  var host = server.address().address
  var port = server.address().port
  console.log("Example app listening at http://%s:%s", host, port)
})

```

HTML code:

```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1">
<style>
body { font-family: Arial, Helvetica, sans-serif;}
input[type=text], input[type=password] {
    width: 100%;
    padding: 12px 20px;
    margin: 8px 0;
    display: inline-block;
    border: 1px solid #ccc;
    box-sizing: border-box;
}
button {
    background-color: #4CAF50;
    color: white;
    padding: 14px 20px;
    margin: 8px 0;
    border: none;
    cursor: pointer;
    width: 100%;
}

button:hover {
    opacity: 0.8;
}
.cancelbtn {
    width: auto;
    padding: 10px 18px;
    background-color: #f44336;
}
.imgcontainer {
    text-align: center;
    margin: 24px 0 12px 0;
    position: relative;
}

img.avatar {
    width: 40%;
    border-radius: 50%;
}
```

```

.container {
  padding: 16px;
}

span.psw {
  float: right;
  padding-top: 16px;
}

.modal {
  display: none; /* Hidden by default */
  position: fixed; /* Stay in place */
  z-index: 1; /* Sit on top */
  left: 0;
  top: 0;
  width: 100%; /* Full width */
  height: 100%; /* Full height */
  overflow: auto; /* Enable scroll if needed */
  background-color: rgb(0,0,0); /* Fallback color */
  background-color: rgba(0,0,0,0.4); /* Black w/ opacity */
  padding-top: 60px;
}

.modal-content {
  background-color: #fefefe;
  margin: 5% auto 15% auto; border: 1px solid #888;
  width: 80%; /* Could be more or less, depending on screen size */
}

.close {
  position: absolute;
  right: 25px;
  top: 0;
  color: #000;
  font-size: 35px;
  font-weight: bold;
}

.close:hover,
.close:focus {
  color: red;
  cursor: pointer;
}

.animate {

```

```

    -webkit-animation: animatezoom 0.6s;
    animation: animatezoom 0.6s
}

@-webkit-keyframes animatezoom {
    from {-webkit-transform: scale(0)}
    to {-webkit-transform: scale(1)}
}

@keyframes animatezoom {
    from {transform: scale(0)}
    to {transform: scale(1)}
}

@media screen and (max-width: 300px) {
    span.psw {
        display: block;
        float: none;
    }
    .cancelbtn {
        width: 100%;
    }
}
</style>
</head>
<body>

<script type="text/javascript" src="/root/Desktop/tarp.require-master/require.min.js"></script>
<center><h2>Passport Application Form</h2>
<button onclick="document.getElementById('id01').style.display='block'"
style="width:auto;">create</button>
</center>

<div id="id01" class="modal">
    <form class="modal-content animate" method="POST" action="/resp">
        <div class="imgcontainer">
            <span onclick="document.getElementById('id01').style.display='none'" class="close" title="Close
Modal">&times;</span>
        </div>

        <div class="container">
            Name
            <input type="text" placeholder="Enter Name" id="nme" name="nme" required>
            Password

```



```

        <input type="password" placeholder="Enter Password" name="psw" required>
        DOB
        <input type="text" placeholder="mm/dd/yyyy" name="dob" required>
    <br/>
    <br/>
    Nationality
    <input type="text" placeholder="Enter nationality" name="nationality" required>
    profession
    <input type="text" placeholder="Enter profession" name="pro" required>

    email
    <input type="text" placeholder="Enter email" name="email" required>


    address
    <input type="text" placeholder="Enter address" name="address" required>
    <button type="submit" a href="/oncl">submit</button>
</div>

</form>
</div>

<script>
var modal = document.getElementById('id01');

window.onclick = function(event) {
    if (event.target == modal) {
        modal.style.display = "none";
    }
}
</script>

</body>
</html>

```