

# Spam Pot

## Team Members

Siddharth Sharma

Saurabh Kulshreshtha

## Abstract:

Project Spam Pot is a project based on luring attackers to a do something humans wouldn't and using this information to detect and deflect them.

It can be used to detect and prevent D-DOS attacks.

Many pages take considerable server resources – which can be saved by filtering out spammers.

It can be used to filter out spammers posting garbage and unwanted content on the website

## Achievements:

A website that is able to filter out spammers, using a list of previous logged HostIDs and session cookies.

We plan to Implement a system that detects bots/spammers to make our system more secure and accessible to legitimate users.

Securing the system entails:

Every connection to the server is checked against a list of HostIDs addresses that are known to spam - if it matches to one we can redirect to 404 page

Every connection to server is also checked whether the cookies they own is associated with an attacker if so they are redirected to a 404-error page.

### **Specifications:**

Spambots usually fill in all the fields of a form. In this project we use this to protect the site from being spammed

A special field is created for the spambot to fill something in with a trap. The human is warned through the field placeholder to not fill anything in the field.

A spambot would fill something in this special field and would be detected as a spammer and his HostID is logged and they are sent a cookie for our server to later check whether we are dealing with a spammer and they are redirected to a 404 Page with warning notifying them that they have been now blacklisted as a spammer.

If the field is left blank the user is directed to the appropriate destination.

Also, if the spammer tries to brute force the submission page by leaving fields blank - they do not succeed because of the cookie and the HostID that is checked by our server to redirect them back to 404 warning page.

### **Layout:**

Spambots usually fill in all the fields of a form. In this project we use this to protect the site from being spammed. A special field is created for the spambot to fill something in with. The human is warned through the field placeholder to not fill anything in the field.

A spambot would fill something in this special field and would be detected as a spammer and his HostID is logged and they are sent a

cookie for our server to later check whether we are dealing with a spammer and they are redirected to a 404 Page with warning notifying them that they have been now blacklisted as a spammer.

If the field is left blank the user is directed to the appropriate destination. Also, if the spammer tries to brute force the submission page by leaving fields blank - they do not succeed because of the cookie and the HostID that is checked by our server to redirect them back to 404 warning page.

### **System Framework and Programming Platform:**

Apache: It is the server for running PHP

PHP: Language used is PHP to build our simple Spam Pot application to misguide the attacker so that we can protect our targeted application

### **Member Contributions:**

#### **Saurabh:**

Came up with and implementation cookie delivery and checking protocol.

Implemented bootstrap to the main page.

Testing of all use cases and fixing the issues.

#### **Siddhartha:**

Came up with the idea for the project.

Implemented a basic version with text field.

Implemented HostID storage and querying.

## Implementation:

Index.php: <Main page which consists of the form for the user to fill>

```
<?php
```

```
?>
```

```
<html>
```

```
  <head>
```

```
    <title>SPAM POT DEMO</title></head>
```

```
    <link href="https://stackpath.bootstrapcdn.com/font-  
awesome/4.7.0/css/font-awesome.min.css" rel="stylesheet"/>
```

```
    <link  
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.0/css/boot  
strap.min.css" rel="stylesheet"/>
```

```
  <body>
```

```
    <div class="container">
```

```
      <div class="wrapper">
```

```
        <div class="row" style="padding: 20px;">
```

```
          <div class="col-md-4">
```

```
            <p>Hello, we are proud to present our work:</p>
```

```
            <h2>Project SpamPot <i class="fa fa-  
forumbee"></i></h2>
```

`<p align="justify">`Spambots usually fill in all the fields of a form. In this project we use this to protect the site from being spammed.

`<br>`

A special field is created for the spambot to fill something in with. The human is warned through the field placeholder to not fill anything in the field.

`<br>`

A spambot would fill something in this special field and would be detected as a spammer and his HostID is logged and they are sent a cookie for our server to later check whether we are dealing with a spammer and they are redirected to a 404 Page with warning notifying them that they have been now blacklisted as a spammer.

`<br>`

If the field is left blank the user is directed to the appropriate destination.

`<br>`

Also if the spammer tries to bruteforce the submission page by leaving fields blank - they do not succeed because of the cookie and the HostID that is checked by our server to redirect them back to 404 warning page.

`</p>`

`</div>`

`<div class="col-md-4">`

`<div class="card bg-light">`

`<div class="card card-header">`

```
        <div class="card-title"><i class="fa fa-users"></i>
User Form</div>
    </div>
    <div class="card card-body">
        <form action="inter.php" method="POST">
            <div class="form-group">
                <label >UserID</label>
                <input name="username" class="form-control"/>
            </div>
            <div class="form-group">
                <label >Email</label>
                <input name="email" type="email" class="form-
control"/>
            </div>
            <div class="form-group">
                <label >Verify</label>
                <input name="honeypot" autocomplete="off"
placeholder="Don't touch if human." class="form-control"/>
                <input name="humantog" type="checkbox"
required/>
            </div>
            <input type="submit" value="Submit"/>
        </form>
```

</div>

</div>

</div>

<div class="col-md-4">

<ul class="list-group"><p><i class="fa fa-code"></i>

Developed by:</p>

<li class="list-group-item"><i class="fa fa-user"></i>

Siddharth Sharma</li>

<li class="list-group-item"><i class="fa fa-user"></i>

Saurabh Kulshretha</li>

</ul>

<div><p></p></div>

</div>

</div>

</div>

</div>

<script

src="http://code.jquery.com/jquery-3.3.1.slim.js"

integrity="sha256-

fNXJFilca05BIO2Y5zh1xrShK3ME+/lYZ0j+ChxX2DA="

crossorigin="anonymous"></script>

<script

src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.0/js/bootstrap.min.js"></script>

</body>

</html>

**Inter.php:**

<?php

if(isset(\$\_COOKIE['backdoor']))

{

header('Location: backdoor.php');

}

else{

\$valid = TRUE;

\$myFile = 'honey\_log.txt';

require\_once("getIP.php");

if (!file\_exists(\$myFile)) {

}

else if(!\$fh = fopen(\$myFile, 'a')) {

}

else {

\$arr = file(\$myFile);

foreach(\$arr as \$i){



```

        if(strcmp($i, gethostbyaddr(getRealIpAddr()))){
            $valid = FALSE;
            setcookie('backdoor', "KKK", time() + (86400 * 30));
        }
    }
}

```

```

if(!(isset($_POST['humantog']) && $_POST['honeypot'] == '')){
    header('Location: backdoor.php');
}

```

```

else{
    //setcookie("backdoor", time() - 450);
    header('Location: hp-conf.php');
}
}
?>

```

**getIP.php:**      <Notes the IP of the spammer>

```
<?php
```

```
function getRealIpAddr()
```

```

{
    if (!empty($_SERVER['HTTP_CLIENT_IP'])) //check ip from share
internet
    {
        $ip=$_SERVER['HTTP_CLIENT_IP'];
    }

    elseif (!empty($_SERVER['HTTP_X_FORWARDED_FOR'])) //to
check ip is pass from proxy
    {
        $ip=$_SERVER['HTTP_X_FORWARDED_FOR'];
    }
else
{
    $ip=$_SERVER['REMOTE_ADDR'];
}
return $ip;
}

?>

```

**hp.conf.php:**    <The page to be displayed if the user is valid>

```

<!DOCTYPE html><html><head><meta charset="UTF-8">
<title>Simple Confirmation | { visibility: inherit; }</title>

```

```
</head>
<body>
<p>Thank you human! Your submission has been accepted!</p>
<p><a href="index.php">&laquo; Back To Spam Pot Demo</a></p>
</body>
</html>
```

**Backdoor.php:** <The page to be displayed to the spammer>

```
<?php
require_once("getIP.php");
$ip = getRealIpAddr();
header("HTTP/1.0 404 Not Found");
if(isset($_COOKIE['backdoor'])){
    echo "You have already been blacklisted once!";
    // setcookie("backdoor", "", time() - 3600);
}
else{
    echo "You have been added to blacklist! You have been warned!";
}
if(isset($ip))
{
    $myFile = 'honey_log.txt';
    if (!file_exists($myFile)) {
    }
```

```
else if(!$fh = fopen($myFile, 'a')) {  
}  
else {  
    $output = gethostbyaddr($ip);  
    fwrite($fh, $output."\n");  
    fclose($fh);  
}  
}  
?>
```

## Snapshots of the Project:

### Main Page:

Hello, we are proud to present our work:

#### Project SpamPot


Spambots usually fill in all the fields of a form. In this project we use this to protect the site from being spammed.

A special field is created for the spambot to fill something in with. The human is warned through the field placeholder to not fill anything in the field.

A spambot would fill something in this special field and would be detected as a spammer and his HostID is logged and they are sent a cookie for our server to later check whether we are dealing with a spammer and they are redirected to a 404 Page with warning notifying them that they have been now blacklisted as a spammer.

If the field is left blank the user is directed to the appropriate destination.

Also if the spammer tries to bruteforce the submission page by leaving fields blank - they do not succeed because of the cookie and the HostID that is checked by our server to redirect them back to 404 warning page.

 User Form

UserID


Email

Verify

☐

</> Developed by:

 Siddharth Sharma

 Saurabh Kulshretha

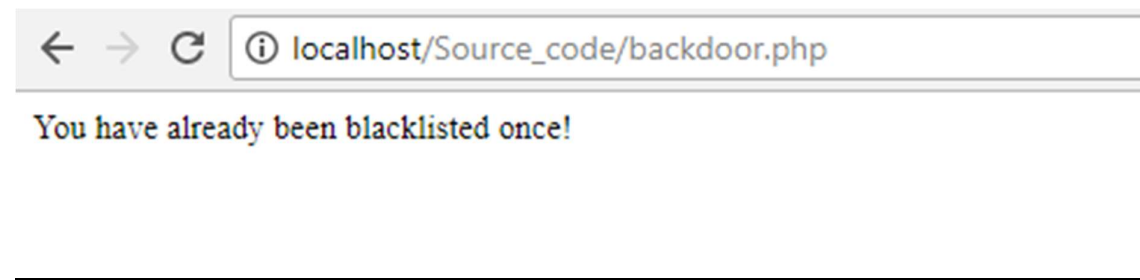
### Valid Users:



**Thank you human! Your submission has been accepted!**

[« Back To HoneyPot Demo](#)

**Spammer User:**



## **References:**

- Wang, J. (2008). Computer network security: Theory and practice. Beijing: Higher Education Press.
- Ramachandram Understanding the Network-Level Behaviour of Spammers  
[https://www.cc.gatech.edu/classes/AY2007/cs7260\\_spring/papers/p396-ramachandran.pdf](https://www.cc.gatech.edu/classes/AY2007/cs7260_spring/papers/p396-ramachandran.pdf)
- Spam Honey Pot Research  
[https://link.springer.com/chapter/10.1007%2F1-84628-352-3\\_3](https://link.springer.com/chapter/10.1007%2F1-84628-352-3_3)