www.og150.com

SSH MITM (Downgrade) Attack, Capturing Username Password Credentials

TABLE OF CONTENTS

www.og150.com

Follow @theog150

Introduction To SSH.

Legend has it that a researcher in Helsinki developed the first version of SSH after a password-sniffing attack at his university network in 1995. Since then, use of SSH has become very popular and is a more secure protocol than the plain text alternative known as Telnet. The first version of SSH was updated and SSH 1.5 became widespread and was typically referred to as SSH-1. Unfortunately, SSH-1 has inherent flaws which make it vulnerable. SSH-2 was later produced to fix the security holes and is much more secure than its SSH-1 predecessor. **Put simply, users should only use SSH-2**.

Interestingly, RFC 4253 specified that an SSH Server which supports both SSH-1 and SSH-2 should identify its protocol version as 1.99. How does this work? When an SSH Client connects to an SSH Server, the SSH Server will indicate which version(s) of SSH it supports. The SSH Server will present one of the following protocol versions:

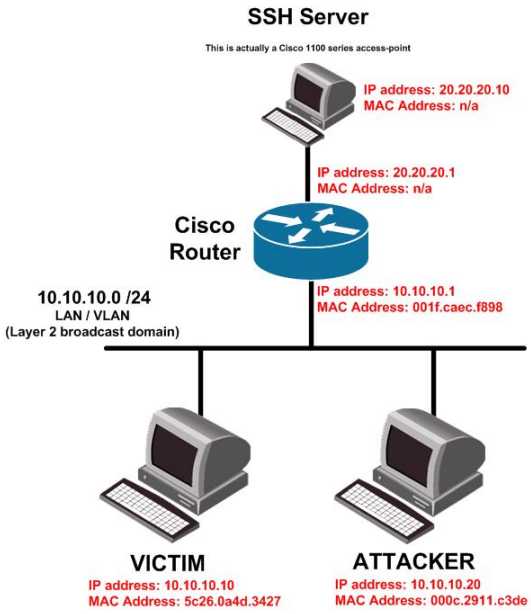
- ssh-2.xx The SSH Server supports only SSH-2
- **ssh-1.99 The SSH Server supports SSH-1 and SSH-2**
- ssh-1.51 The SSH Server supports only SSH-1

When the SSH Client receives this information, it can choose which SSH version he wants to use. In situations where the SSH Server supports ONLY SSH-1 or ONLY SSH-2, the user obviously has no choice. However, if protocol version ssh-1.99 is sent by the SSH Server, it is the SSH Client that chooses which to use. By default, a client that supports both SSH-1 and SSH-2 will always use the stronger version SSH-2.

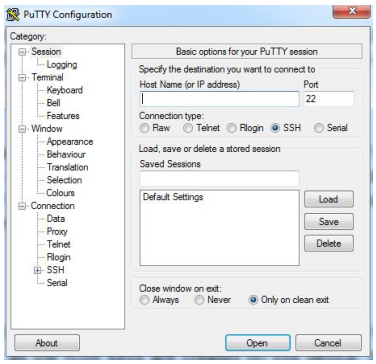
In instances whereby the SSH Server and SSH Client both support SSH-1 and SSH-2, if we can somehow 'force' the SSH Client and SSH Server to use SSH-1, we can capture the username/passwords credentials (we already know that SSH-1 is weak and can be compromised). OK, let's start.....

The demonstration topology is shown in Screenshot 1. Essentially, there is a *VICTIM* (using the SSH Client PuTTY) who needs to access the *SSH Server* using the SSH protocol. Importantly, the *SSH Server* supports SSH-1 AND SSH-2 (therefore its protocol version is ssh-1.99). The PuTTY SSH Client is freeware and can be seen in Screenshot 2.

Screenshot 1 – Demonstration topology



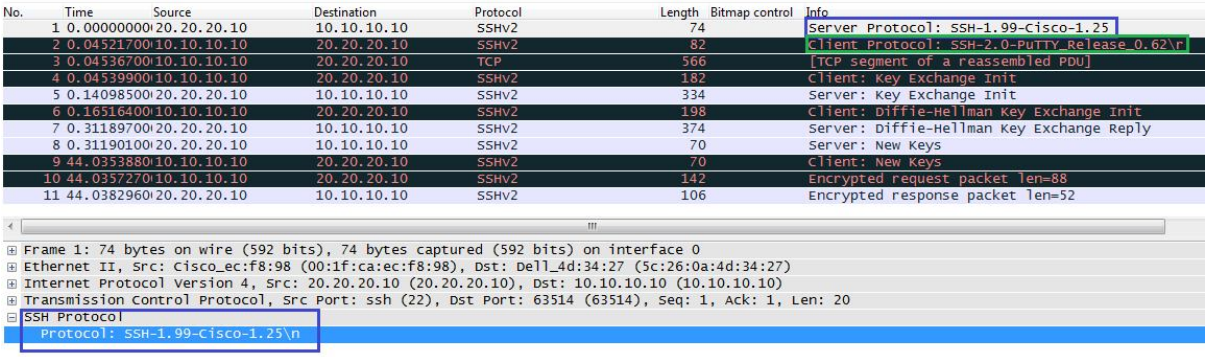
Screenshot 2 – PuTTY SSH Client



Please note: The *SSH Server* in this demonstration is an old Cisco 1120 series autonomous access-point. As soon as the RSA keys are created on this device, both SSH-1 and SSH-2 are **enabled by default**!

Let's see what happens to an SSH connection between the *VICTIM* and the *SSH Server* in a normal situation (when there is no attack in progress). Upon completion of the TCP 3 way handshake, the SSH Server (20.20.20.10) informs the SSH Client that it supports protocol version ssh-1.99 (which means the SSH Client can use either SSH-1 or SSH-2). This is highlighted by the blue rectangles in Screenshot 3. The SSH Client (using the PuTTY application) chooses the strongest version – SSH-2. This is highlighted by the green rectangle in Screenshot 3. The SSH-2 connection is then created.

Screenshot 3 – Standard SSH connection packet trace



Just for reference, the ARP cache on the Cisco Router prior to the attack is shown in Screenshot 4.

Screenshot 4 – Cisco Router ARP cache 'before' attack

```
Router#show arp
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  10.10.10.1              -   001f.caec.f898  ARPA   FastEthernet0/0
Internet  10.10.10.10             0   5c26.0a4d.3427  ARPA   FastEthernet0/0
Internet  10.10.10.20             0   000c.2911.c3de  ARPA   FastEthernet0/0
Internet  20.20.20.1              -   001f.caec.f899  ARPA   FastEthernet0/1
Internet  20.20.20.10            59   0011.93f1.0c08  ARPA   FastEthernet0/1
Router#
```

SSH MITM (Downgrade) Attack.

The essence of this attack is this; the *SSH Server* supports SSH-1 and SSH-2. The SSH Client also supports SSH-1 and SSH-2. However, the problem is that the more secure version (SSH-2) will always be used. **If we can somehow 'change' the SSH protocol version sent by the *SSH Server* from ssh-1.99 to ssh-1.51, the client will 'think' that the *SSH Server* only supports SSH-1**. The SSH Client will then setup an SSH-1 connection which we can compromise ☺

The first step is to create a MITM scenario, for this I will use ARP spoofing (for further information on ARP spoofing please refer to the tutorial "ARP Spoofing MITM Attack, Capturing Telnet Data" at www.og150.com). With the MITM in place, all traffic to and from the *VICTIM* will traverse the *ATTACKER* machine (running Kali). The *ATTACKER* machine will now 'look' for any SSH sessions that specify protocol version ssh-1.99 and will **re-write the protocol version as ssh-1.51**, before sending it onto the *VICTIM*. All this is achieved using the awesome tool – Ettercap.

As shown in Screenshot 5, ensure that Ettercap is installed on the *ATTACKER* machine (it is installed in Kali by default).

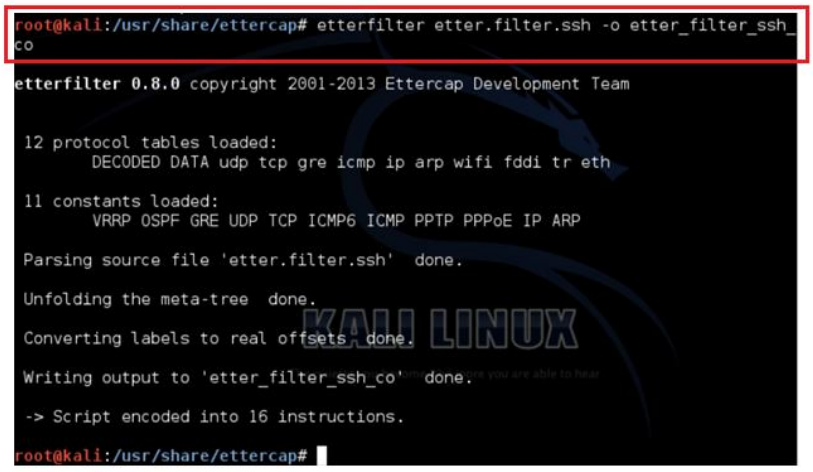
Screenshot 5 – Verify Ettercap is installed

We will be using an Ettercap SSH filter for this attack, which will downgrade (re-write) the SSH protocol version from ssh-1.99 to ssh-1.51. As shown in Screenshot 6, you will need to change to the Ettercap directory and confirm that the SSH filter exists (it is installed in Kali by default).

Screenshot 6 – Verify Ettercap SSH filter is available

You now need to compile the SSH filter as shown in Screenshot 7.

Screenshot 7 – Compile Ettercap SSH filter



Finally, you can start the Ettercap attack using the command shown in Screenshot 8. This command will start the ARP MITM and will actively monitor SSH sessions and downgrade (re-write) the protocol version from ssh-1.99 to ssh-1.51.

Screenshot 8 – Start the Ettercap attack!

```
root@kali:/usr/share/ettercap# ettercap -T -q -i eth0 -M arp -F etter_filter_ssh
_co // //
```

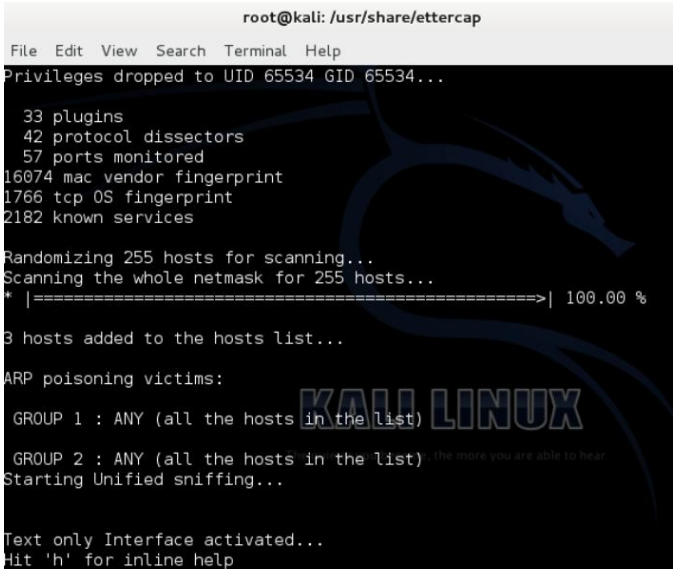
The ARP cache on the Cisco Router is shown in Screenshot 9 and confirms that traffic destined for the *VICTIM* (10.10.10.10) is being sent via the *ATTACKER* machines MAC address – the MITM is working. Note: Compare this output to Screenshot 4 to see what is different.

Screenshot 9 – Cisco Router ARP cache with attack in progress

```
Router#show arp
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  10.10.10.1               -   001f.caec.f898  ARPA   FastEthernet0/0
Internet  10.10.10.10              0   000c.2911.c3de  ARPA   FastEthernet0/0
Internet  10.10.10.20              0   000c.2911.c3de  ARPA   FastEthernet0/0
Internet  20.20.20.1               -   001f.caec.f899  ARPA   FastEthernet0/1
Internet  20.20.20.10             24   0011.93f1.0c08  ARPA   FastEthernet0/1
Router#
```

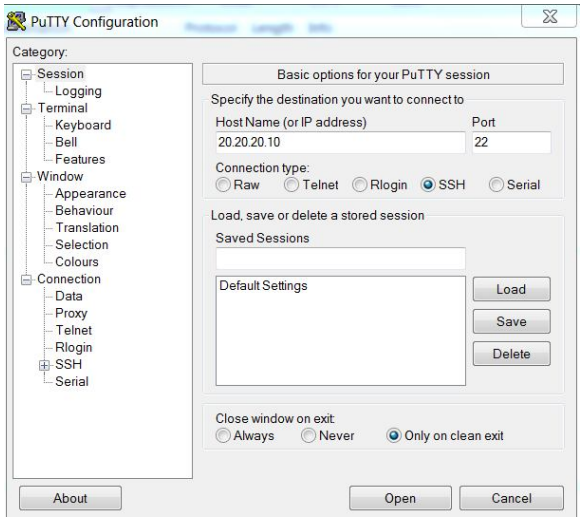
Screenshot 10 illustrates that the attack is in progress and we are patiently waiting for an SSH connection to compromise.....

Screenshot 10 – Ettercap attack is running and waiting for an SSH connection to compromise



On the *VICTIM*, open the PuTTY SSH Client and connect to the *SSH Server* as shown in Screenshot 11.

Screenshot 11 – *VICTIM* creates SSH connection to *SSH Server*

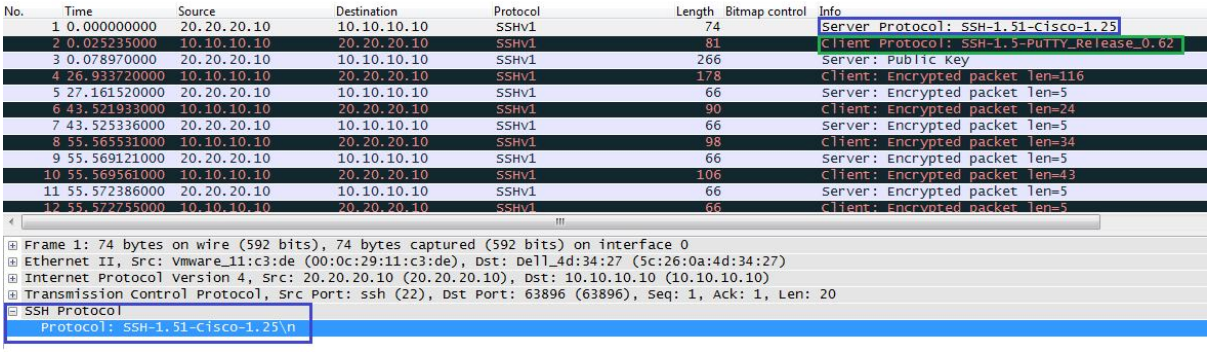


At this point, Ettercap on the *ATTACKER* machine will downgrade (re-write) the *SSH Server* response from ssh-1.99 to ssh-1.51. The Ettercap output shown in Screenshot 12 confirms the SSH downgrade took place!

Screenshot 12 – SSH connection was downgraded

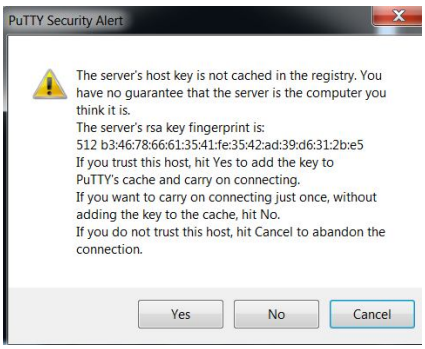
The SSH downgrade can be confirmed by a Wireshark packet capture taken on the *VICTIM* machine. The SSH Server (20.20.20.10) informs the SSH Client that it supports ssh-1.51 (which means the SSH Client can only use SSH-1) – this is because Ettercap re-wrote the *SSH Servers* protocol version from ssh-1.99 to ssh-1.51. This is highlighted by the blue rectangles in Screenshot 13. The SSH Client (using the PuTTY application) has no option but to use SSH-1 and therefore an SSH-1 connection is created. This is shown by the green rectangle in Screenshot 13. Please compare the output in Screenshot 3 with the output in Screenshot 13 to see how Ettercap manipulated the *SSH Servers* protocol version.

Screenshot 13 – Downgraded SSH connection packet trace



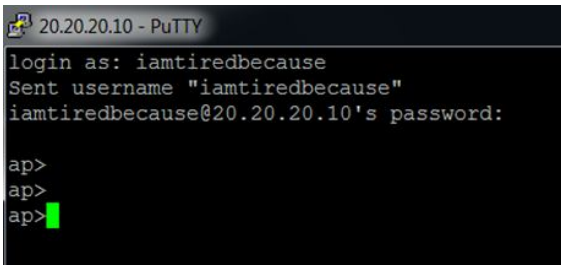
The *VICTIM* is then prompted to accept the *SSH Servers* SSH-1 RSA key as shown in Screenshot 14.

Screenshot 14 – PuTTY security prompt



Unless the *VICTIM* is security aware, at this point he will have no idea that he/she has connected using SSH-1. Therefore, the user logs in as normal (as shown in Screenshot 15).

Screenshot 15 – SSH login



The *VICTIM* now has an SSH session to the *SSH Server* (the Cisco access-point in this demonstration). As far as he/she is concerned, they are using a secure/encrypted SSH session. Take a look at the Ettercap output in Screenshot 16 though – we captured the username and password that the user logged into the SSH session with ☺

Screenshot 16 – WE GOT THE USERNAME PASSWORD! ☺

And that, my friends, is game over.

SSH Quick Facts.

Here are some quick facts about SSH:

- Recent versions of OpenSSH has only SSH version 2 enabled (by default) = secure.
- Cisco IPS software (Intrusion Prevention System) has only SSH-1 enabled! To enable SSH-2 you have to gain 'root' access to the software. Sadly, Cisco IPS is a security device that is not secure... ☹
- The PuTTY SSH Client, by default, will use either SSH-1 or SSH-2 (although SSH-2 will be preferred). This means this SSH Client is vulnerable to an SSH downgrade attack. If you use PuTTY, please change to use SSH-2 ONLY.
- The SecureCRT client (which I use), asks you what version you want to connect with before the connection. If you say you want to connect using SSH-2 then it will ONLY use SSH-2. In this respect, SecureCRT is more secure than PuTTY.
- To find out if your *SSH Server* is vulnerable to an SSH downgrade attack, you need to check what SSH version(s) it supports. One simple tool you can use for this is 'ncat' as shown in Screenshot 17. Notice that this server (running OpenSSH) uses ONLY SSH-2 and is therefore not vulnerable to an SSH downgrade attack.

Screenshot 17 – 'ncat' check for SSH protocol support