Network Security

Assignment 6, Friday, October 10, 2014, version 1.2

Handing in your answers: Submission via Blackboard (http://blackboard.ru.nl)

Deadline: Monday, October 20, 23:59:59 (midnight)

This final assignment is larger than the previous ones. Note the unusual deadline.
In this assignment you will be using the following tools:

- aircrack-ng: http://www.aircrack-ng.org/
- arpspoof: http://www.monkey.org/~dugsong/dsniff/
- nmap: http://nmap.org/
- sslstrip: http://www.thoughtcrime.org/software/sslstrip/, https://pypi.python.org/pypi/sslstrip/0.9.2
- openvpn: https://openvpn.net/index.php/open-source/documentation.html
- wireshark, tshark or tcpdump for packet capturing: https://www.wireshark.org/
- optionally virtualbox (or some other virtualization software): https://www.virtualbox.org/

Again, do not compile these programs from source, but install them using your distribution's package manager.

The assignment consists of a theoretical question and two practical exercises. Please turn in all your work in plain text files (program source files are also plain text). If you prefer a document with formatting for whatever reason, like including images, use the PDF format to turn in your work (most editors allow you to export to PDF). Note that it's okay to include images separately and then refer to them from within the text files.

Commands that need to be run with root rights are denoted by a prefix `#`. When a command should be run without root rights, it will be prefixed with `$`. Do not include the prefix when typing the command.

1. Create a folder called `exercise1`.

   This exercise is intended to teach you the basic use of OpenVPN. There is an abundance of documentation on the internet. A lot of good documentation is on the project's website, https://openvpn.net/index.php/open-source/documentation.html.

   (a) Create a subfolder called `exercise1a` to hold your answers and configuration files.

   Using the OpenVPN documentation you must set up an OpenVPN network between two machines. These can be physical machines, e.g. yours and your lab partner's laptops, but you can also use virtual machines. For the latter we recommend virtualbox. Setting up a virtual machine with e.g. Ubuntu is covered in tutorials so we will not cover that here.

   The minimum setup you should get working is a VPN with a static, pre-distributed key. You should use layer 3 tunneling (tun devices), not layer 2 (tap devices). Document the commands you use in a text file `commands`. Also include configuration files for both hosts, if applicable.

   Also perform a set of short packet captures on both ends of the connection, while doing a ping from the VPN server to the VPN client and vice versa. The packet captures on each end should be done on two interfaces: one capture on the tun-interface created for the VPN, and another capture on the network interface that is actually carrying the VPN-tunneled traffic (either your normal network interface, or a virtual interface created by e.g. virtualbox). So there should be four captures in total. Include these captures, and name them along the lines of `client-tun.cap` and `server-wlan0.cap`.

(b) Create a subfolder called `exercise1b` to hold your answers and configuration files.

Using the OpenVPN documentation and the previous exercise's answers, try to set up the VPN so that the VPN client uses the VPN for all its network traffic. This is a fairly common usage scenario, so it is fairly well covered in the basic documentation.

Document the commands in a text file `commands`. Include configuration files for both hosts. Perform the same kind of packet capture as in exercise 1a, however, this time the VPN client should ping some host on the internet (e.g. www.google.com) instead of the VPN server.

If you have network issues during this exercise, one thing to do would be to look at your routing table (`ip r show`; `route -n`) and see if you can figure out why traffic is or is not going through the VPN. Of course, send an e-mail or drop by my office if you're stuck.

2. Create a folder called `exercise2`.

This exercise is a multi-stage attack. Somewhere, there's a website containing your grades for this exercise. Everybody starts out with an O. It is up to you to give yourself the grade you want.

You are *not* allowed to sniff the general network traffic in order to eavesdrop on other groups performing the attack.

Please do not change other people's grades while performing this exercise, and don't do anything else on the target website.

(a) Although WPA2 is more secure than WEP, just like any other good cryptographic system it is only as strong as the key material in use. To demonstrate this, you will use aircrack-ng to crack the passphrase of the wireless network where the course administrator is working. We have already taken care of capturing the WPA2 connection handshake, you can download it at http://www.cs.ru.nl/~paubel/assignment6-handshake.cap.

To crack WPA2 passphrases, you need wordlists. A tutorial on how to crack WPA is on http://www.aircrack-ng.org/doku.php?id=cracking_wpa. Ignore the stuff about injecting packets, capturing the handshake etc. We've already taken care of that. The interesting part is section 4. Pointers on where to find wordlists are on http://www.aircrack-ng.org/doku.php?id=faq#how_can_i_crack_a_wpa-psk_network. Since it is not our intention to have you spend hours on WPA cracking, use the wordlist at http://gdataonline.com/downloads/GDict/GDict_v2.0.7z. Note that you have to unzip it first (`7z x GDict_v2.0.7z`).

The bssid of the network is 48:5B:39:89:8C:10. If you want to decrypt the capture to see whether you have the correct key, you also need the essid. This is "NetSec Homework Net (Pol)". The capture should contain a single DHCP packet. Beware of the Ubuntu decryption bug, however: if you see other stuff you may still have the correct key. The best way to check is to try to connect to the network.

Keep in mind that the network may not have a running DHCP server so if you fail to connect, try to set a static IP address in the 192.168.84.100–149 range, with netmask 255.255.255.0 and gateway 192.168.84.10.

Write the passphrase you found to a file called `exercise2a`.

(b) Connect to the network. There should be a DHCP server running. If not, use an IP address in the range of 192.168.84.100–149, with netmask 255.255.255.0 and gateway 192.168.84.10.

Use nmap to scan this network. Find the hosts in the range 192.168.84.1–99. Disable reverse DNS lookup to speed up things. There should be at least 2 hosts, apart from the gateways (192.168.84.10–15) and access point (192.168.84.1). Write which hosts you find to `exercise2b`.

(c) From this point onwards you will need to coordinate with other groups, since there is only a limited number of hosts to arpspoof. Do not get in each others way.

Pick one of the hosts that are not the gateways (192.168.84.10–15) or access point (192.168.84.1). Its gateway is matched on the second digit (so 192.168.84.32 and 192.168.84.42 would both have

gateway 192.168.84.12, whereas 192.168.84.33 would have gateway 192.168.84.13).[1]

Using arpspoofing and wireshark, figure out which websites this host is contacting. Save the network capture in `exercise2c.cap`. Write the URLs to `exercise2c`. Note that you may need to also arpspoof its gateway. Do *not* arpspoof the access point (192.168.84.1).

NOTE: There is some delay between requests in order to not abuse the target website. This delay is approximately 1 minute as of this writing.

(d) Now, use sslstrip (`http://www.thoughtcrime.org/software/sslstrip/`, `https://pypi.python.org/pypi/sslstrip/0.9.2`) to strip out SSL from its web traffic. Look at the traffic in wireshark and figure out the login credentials to use. Save the network capture in `exercise2d.cap` and write the login credentials you found to `exercise2d.creds`.

(e) Finally, log in to the website, find your grades and edit them to your desired result. After that, write your student numbers and the result you set to `exercise2e`.

3. During the lecture, Peter and Christiaan demonstrated ongoing research to figure out which websites somebody using TLS is visiting. The method demonstrated is based on counting the number of image requests performed. Assume that you are a passive MITM, so you cannot modify traffic, only observe it. Try to think of other methods that might work to perform the same kind of tracking, i.e. to figure out which Wikipedia page (or sequence of Wikipedia pages) somebody is visiting, without stripping away TLS. Explain these methods and their operation. Write your answer to `exercise3`.

4. Place the files and directories `exercise1`, `exercise2`, and `exercise3` and all their contents in a folder called `netsec-assignment6-SNR1-SNR2`. Replace `SNR1` and `SNR2` by your respective student numbers, and accomodate for extra / fewer student numbers. Make a `tar.gz` archive of the whole directory `netsec-assignment6-SNR1-SNR2` and submit this archive in Blackboard.

[1] This somewhat weird network configuration is required to enable you to arpspoof in parallel with other groups. Without going into too much detail, the problem is that if we only had one gateway IP address, we would need as many hardware devices as IP addresses for you to spoof. In most normal situations, a network only has one gateway which all clients will use.