Network Security

Assignment 5, Friday, September 30, 2015, version 1.0

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

Deadline: Monday, October 19, 23:59:59 (midnight)

This final assignment is larger than the previous ones. ***Note the unusual deadline.*** Please make sure to start on time. There is one more lecture and tutorial during this exercise, on October 14 and 15, resp. However, you should be able to do most of this assignment without the information from the lecture. If you're stuck, please send us an e-mail or ask questions during the tutorial.

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
- dig, drill, or similar DNS query tools: http://www.isc.org/downloads/bind/, https://www.nlnetlabs.nl/projects/ldns/
- 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 several practical exercises and theoretical questions. Please turn in all your work in plain text files (program source files are also plain text), unless specified otherwise. 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 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. Also, 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/assignment5-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 `exercise1a`.

   (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 `exercise1b`.

   (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 `exercise1c.cap`. Write the URLs to `exercise1c`. 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. sslstrip will be briefly touched upon in the lecture on October 14, but the documentation and explanation on the websites will probably be enough to get it working.
   Look at the traffic in wireshark and figure out the login credentials to use. Save the network capture in `exercise1d.cap` and write the login credentials you found to `exercise1d.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 `exercise1e`.

[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.

2. This exercise is about DDOS attacks using DNS amplification. Create a folder `exercise2` to contain the files with your answers.

   (a) Using any tool, script or program you want, figure out the DNS query that gives you the largest DNS amplification. E.g. a query that's 100 bytes and generates a response of 1000 bytes gives you an amplification factor of 10. You are not allowed to use DNS servers under your own control for this, but apart from that you are free to pick any server and any query you want. To make sure that we can verify your answer, make a packet capture of the outgoing query and the incoming response.

   Members of the group with the largest amplification will get a prize: a copy of "Ghost in the Wires: My Adventures as the World's Most Wanted Hacker" by Kevin Mitnick. In the case of a tie, the first submission in Blackboard wins. Don't spend all your time doing this, however. Find a reasonable query, then do the other exercises before coming back to improve on this answer.

   Write your answer, preferably as a `drill` query, to `exercise2a`. Also store the packet capture as `exercise2a.cap`. If you programmed something for this, include the source code.

   (b) Now imagine that you are in a LAN with a *non-NATing* gateway router. Explain how you would use this DNS query to take down a server which has been annoying you for a while, e.g. blackboard.ru.nl. Describe the packet you need to craft, and its relevant features, at DNS level, UDP level, IP level and ethernet level. Do not actually perform the attack. Write your answer to `exercise2b`.

   (c) Suppose you are the administrator of this network. You want to make sure that, from the LAN, nobody can use this kind of DNS amplification attack. The LAN network is 203.0.113.0/24, the gateway's internal IP address is 203.0.113.1, and its external IP address is 198.51.100.78.

   What firewall measures (iptables rules) would be effective in preventing this kind of attack *without* impeding normal operation of the network? Describe these measures in detail, and also try to come up with actual iptables rules for them. Write your answer to `exercise2c`.

3. Create a folder called `exercise3`.

   Assume you're an attacker who wants to trick a DNS cache into believing your server is actually hosting blackboard.ru.nl. You try to race a legitimate DNS server to provide the answer faster.

   (a) How would you ensure that you can predict the queries that the cache is going to produce, and how would you ensure that your answers will be accepted (i.e. pass the bailiwick check)? Describe the setup and/or process. Write your answer to `exercise3a`.

   (b) QID randomization and port randomization are (somewhat) effective countermeasures against cache poisoning. If you craft a single blind response, to a single DNS query, what are the odds that you guess right if the DNS cache is only using QID randomization in its queries?

   What are the odds if the cache is also using source port randomization?

   Write your answers to `exercise3b`.

   (c) Imagine that on top of that, these DNS servers also deploy 0x20 randomization (see slides, the random capital letters in the query). What are the odds now that you will guess right on a query for the blackboard.ru.nl host? Why? Write your answer to `exercise3c`.

   (d) How could you still try to get a good success rate, even though your odds of guessing correctly are low? Describe the general idea behind the attack, exact calculations of probability are not required. Write your answer to `exercise3d`.

   (e) Explain, in your own words, why all these randomization countermeasures do not work against a passive MitM attacker. Write your answer to `exercise3e`.

4. The firewall configuration you made in assignment 4, exercise 1a, should still allow DNS conversations. However, DNS usually runs over UDP and UDP is a connectionless protocol. Try to explain how the firewall still knows that it should allow this DNS traffic. Write your answer to `exercise4`.

5. Create a folder called `exercise5`.

   This exercise is intended to teach you the basic use of OpenVPN. The general concept of VPNs will be further explained in the next lecture on October 14, but most of this exercise is based on OpenVPN which will not be explained in depth during the lecture. However, 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. We expect you to be able to finish this exercise using this documentation and by asking us specific questions.

   (a) Create a subfolder called `exercise5a` 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. Note that you may not be able to reach each other's machines through eduroam, but a direct link using an ethernet cable or an ad-hoc WiFi network usually works. Setting up a virtual machine with e.g. Ubuntu is covered in tutorials so we will not cover that here. You *can* use the virtual machine we've provided but note that you may need to change the MAC address or even the type of its virtual interface if you're trying to connect two of them.

   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)[2]. 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 `exercise5b` 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. You can use `traceroute` or `mtr` to figure out whether traffic is actually going through the VPN or whether it's taking the normal route to the internet.

   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 Peter's office (M1.03.18) if you're stuck.

6. Place the files and directories `exercise1`, `exercise2`, `exercise3`, `exercise4`, and `exercise5` and all their contents in a folder called `netsec-assignment5-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-assignment5-SNR1-SNR2` and submit this archive in Blackboard.

[2] The reason for this is that it's harder to get layer 2 OpenVPN working. Layer 2 OpenVPN does have some additional tunneling overhead but is not inherently worse than layer 3, but it does require you to do most of the network setup (e.g. DHCP, routing) yourself.