ECE458/750: Computer Security
Assignment 2

Prepared by Tanmayi Jandhyala and Kami Vaniea

July 14, 2025

Due Date: 28th July, 2025 at 11:30 pm.

Lab setup requirements: The content of Question 1 of this assignment is closely related with, but is independent of, the SEED Lab linked below. You are **NOT** expected to do/submit all the tasks in this lab for this assignment. Please also note that this lab requires you to have a VM and Docker setup (within the VM). Question 1 in the assignment assumes that you do have this setup.
Sniffing and Spoofing Lab: https://seedsecuritylabs.org/Labs_20.04/Networking/Sniffing_Spoofing/

Office Hours: Mondays 4:00 - 5:30 PM at E7 4324. Piazza live Q&As will happen on Mondays from 4:00 PM - 6:00 PM. That basically means that a TA will be online on Piazza for those two hours to answer any questions.

Questions: Use Piazza for questions and concerns. Please come to office hours for any questions that you might have about the SEED Lab setup.

Submission: Use Crowdmark to submit, even for the code. Please make sure to add some helpful comments to your code to improve readability.

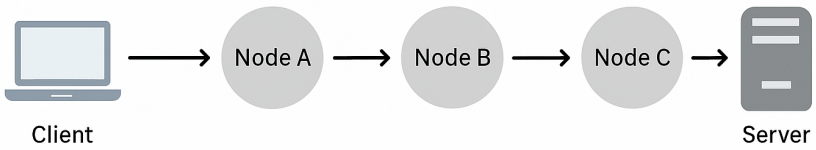

Figure 1: Onion routing network chosen by the Client.

1. We highly recommend that you work through Task 1 of the Sniffing and Spoofing SEED Lab before proceeding to answer the below question. The SEED lab linked above shows you how to get three containers up and running, one of which, is an "attacker" container which can see the packets from the other containers. We will begin by using **Scapy**, a python library that can parse packet capture information sniff, manipulate, and decode data, amongst other things. To familiarize yourself, once you set up your containers and before you can work on your SEED Lab task, you can try the commands listed in the 'first steps' section of this documentation: `https://scapy.readthedocs.io/en/latest/usage.html#first-steps`. You can also refer to the rest of the documentation while answering the below questions, if necessary.

   (a) **(5 marks)** You are an eavesdropper. Write a simple python script using Scapy to listen in on the network that you (the attacker) are connected to. Verify your setup by sending an ICMP ping to the network from one of the hosts, and receive an answer on the sniffer you wrote. Present your code and output. We will review the code, and only attempt to run it if its effects are unclear.

   (b) **(5 marks)** Utilising the Docker setup in the linked SEED Lab above, write a Python script using Scapy that performs the following:

   - Define the target subnet (this would be the one that you configured in your Docker container).
   - Iterate through all possible IP addresses within the subnet range.
   - For each IP address, craft an ICMP echo request packet (ping) using Scapy.
   - Send the ping packet and capture the response (if any).
   - Analyze the response: If a response (ICMP echo reply) is received, consider the device at that IP active. Print the discovered active IP addresses.

   Present your code. It is obvious that for the current setup, the output of the program does not matter. You will be evaluated for your program structure, we will only attempt to run code segments if the effects are unclear.

2. **(10 marks)** *Firewall Rules Game.*
   Below is a game created by one of Prof. Vaniea's past students. The game is fun, interactive, lets you test firewall rules for various topologies and helps you think both like a defender and an attacker, depending on the difficulty level. Feel free to explore and move up through the levels of the game.

   Blue Team : A firewall setup game

   Turn In: Provide screenshots (2 in total) that show the 'Good Job' page when you click on 'submit solution' for *two* levels that are *of level* 5 *and above*.

3. **(10 marks)** *Onion Routing.*

   Network security researchers in the 1990s developed the onion routing protocol (eventually leading to the TOR Project). Onion routing protects anonymity by ensuring that a user's traffic cannot be linked to the originating IP Address.

   Figure 1 shows a small Onion routing network that has been selected by the Client. Assume that the Client has already setup the connections resulting in symmetric keys: Key A, Key B, and Key C corresponding with each of the nodes.

(a) Complete the table below (marked i. - xi.). Answer in terms of when the node has finished its internal processing of the message and is preparing to send it to the next node. The first row has been answered for your reference.

- Number of Encryption Layers - Onion routing works by concentric layers of encryption that must be added and removed in the correct order. How many layers does the message have when the node sends it to the next node. Answer only in terms of onion routing, other encryption such as HTTPS can be ignored for this question.
- Address(es) Known - Which IP addresses of other Nodes on this Onion Routing path does the node know?
- Key(s) Known - What symmetric keys does the node know?

Node	Number of Encryption Layers	Address(es) Known	Key(s) Known
Client	3	Node A, Node B, Node C	Key A, Key B, Key C
Node A	i.	ii.	iii.
Node B	iv.	v.	vi.
Node C	vii.	viii.	ix.
Server	x.	Node C (exit node's IP address)	xi.

(b) From your answers above, mention **two** vulnerabilities with onion routing. That is, two situations where a user could be connected with their traffic even if they use the onion routing correctly.

(c) Onion routing depends on there being nodes other than the client and server. Is it adequate for there to be only one such node? Justify your answer.

4. (**10 marks**) *Border Gateway Protocol (BGP)*

The Border Gateway Protocol (BGP) specifies how network paths across the Internet are formed. It divides the Internet into Autonomous Systems (ASes), each made up of multiple routers. Every AS announces the IP prefixes it owns to its neighboring ASes. These announcements are passed from AS to AS, and each router builds a routing table that records the sequence of ASes needed to reach each destination. Please refer to Lecture 10 for more details.

Figure 2 is a sample oversimplified map of part of the Internet. Use it to answer the questions below. You may assume that all the connections are bidirectional. Note that there is an important piece of information used in BGP routing that is not visible on the Figure.

(a) Under the BGP protocol are *AS200* and *AS140* both guaranteed to send traffic through *AS190* to reach IP addresses owned by *AS150*? Explain why or why not.

(b) ASes are managed by entities world-wide including companies, and countries. How might a country-owned AS, such as *AS180* in Figure 2 temporarily re-direct world-wide traffic to their own network?

(c) If *AS180* were to succeed at the attack in Question 4b, would they be able to Man-in-the-Middle the traffic? In other words, would it be possible to route the traffic through *AS180* while still ensuring it reaches its originally intended destination IP address? Explain why or why not.

(d) *Prefix Filtering:* is a security best practice where, as part of BGP routing, an AS only accepts IP-address ranges (prefixes) from its customers that it knows they are legitimately in control of. *Customers* here refers to smaller ASes such as *AS300* on the Figure that rely on larger better connected AS to send their traffic. What attack is prefix filtering defending against?

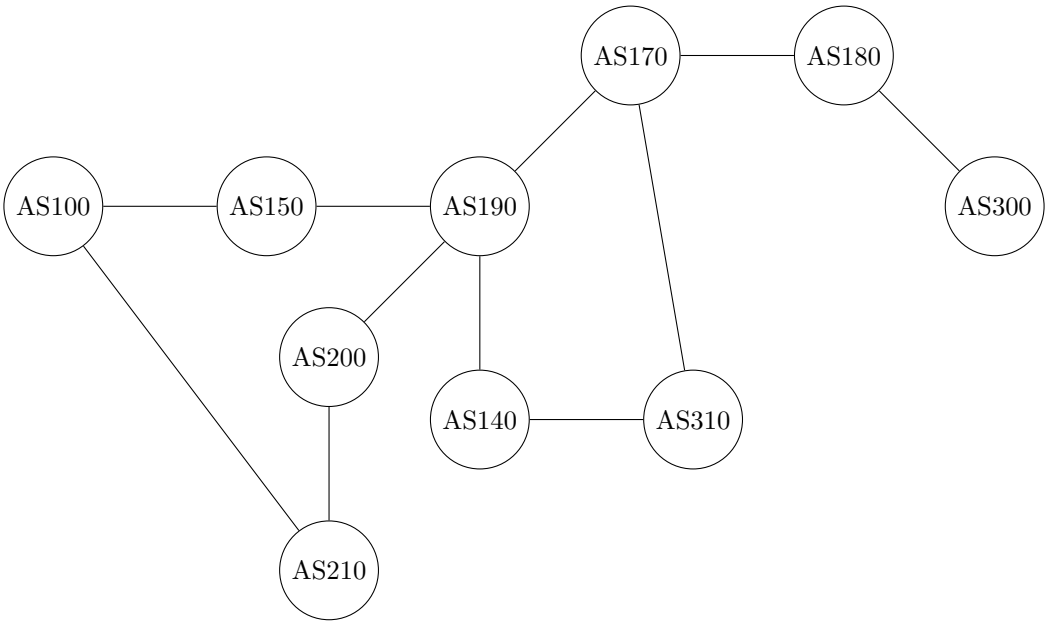

Figure 2: A sample oversimplified map of part of the Internet.

5. (**5 marks**) *Web Certificates*

   (a) (**3 marks**) Find the intermediate certification authority's certificate and the trusted root certification authority's certificate on your browser for `https://www.eff.org/`.

      i. Check for validity of the certificates.

      ii. How long are each of these certificates' public keys?

      iii. What algorithm is used to generate the signatures?

   (b) (**2 marks**) Modern browsers like Chrome, Firefox, Safari, and Edge all automatically verify certificates on behalf of users. Describe a situation where a user explicitly enters a correct URL (e.g. `https://eff.org`), the certificate is correctly validated by the browser, but the web page that is delivered is controlled by an attacker. It may help to think about what type of attacker would be able to accomplish such an attack, rather than a generic attacker.

Unmarked Question

The questions below will not be marked or impact your grade, but there will be space provided in Crowdmark for you to submit if you wish to. You also do NOT need a VM setup to do the below questions. We will provide some course-wide feedback and the correct answers for these questions, but we will not individually mark them.

TLS Handshake

Recall the handshake protocol for TLS, and think about how it can ensure a secure communication over the internet.

(a) Give two scenarios during the TLS handshake when the certificate verification can be mis-authenticated by the server.

(b) The Online Certificate Status Protocol (OCSP) enables clients to query a server (an OCSP responder) in real-time to determine the status of a certificate, receiving a response that is authenticated by the issuing CA. This protocol was developed to overcome the limitations associated with Certificate Revocation Lists (CRLs). Generally, an OCSP response is smaller than 1 kB. Can you think of a way in which an attacker can exploit this protocol to get verified by a server during authentication?

(c) Imagine that a client pre-establishes trust with a server. At a later time, the client begins to self-sign certificates and uses them to authenticate itself, and server validates it because of the pre-established trust. Is this mechanism fool-proof? If yes, why? If not, why?

(d) Session resumption and Pre-Shared Key (PSK): A pre-shared key (PSK) is a secret mutually shared by two parties through a secure channel before its utilization. It's possible to create a PSK during one TLS handshake and then employ it to initiate a new connection in another handshake, known as session resumption using a PSK. The server can transmit to the client a PSK identity corresponding to a unique key derived from the initial handshake. The client can subsequently employ this PSK identity in future handshakes to negotiate the use of the associated PSK. Does this circumvent the vulnerabilities with using a trusted third-party (CA) to help validate a server's authenticity? Share your thoughts briefly.

(e) The cipher suites used in TLS1.2 (the older version) used an optional compression mechanism in the Client Hello message to reduce the bandwidth. In cookie information, for example, the compression algorithm basically replaces repeated byte sequences with a pointer to the first instance of that sequence. This was a vulnerability once. Can you think of a way in which an attacker who knew that a handshake was using compression, could reconstruct the cookie of a victim's browser? Hint: It is similar to the underlying concept of an adaptive chosen ciphertext attack.