Homework 4: The Final Homework

Due: Monday April 29 @ 11:59 pm ET

Overview and instructions

This homework has 3 problems:

- Problems 1–3 are required for all students
- There is no extra CS1620/CS2660 part for this assignment. Have fun with Dropbox!

Note on collaboration

You are welcome (and encouraged!) to collaborate with your peers, but the solutions you write down must be **your own work** (ie, written by you). You are responsible for independently understanding all work that you submit—after discussing a problem as a group, you should ensure that you are able to produce your own answers independently to ensure that you understand the problem. For more information, please see the course Collaboration Policy.

In your submission, we ask that you include a brief *collaboration statement* describing how you collaborated with others on each problem—see the next section for details.

How to submit

You will submit your work in PDF form on Gradesope. Your PDF should conform to the following requirements:

- **Do not** include any identifying information (name, CS username, Banner ID, etc.) in your PDF, since all homeworks are graded anonymously
- Each problem (where "problem" is one of the Problems 1–3) should start on a separate page. When you submit on Gradescope, you will be asked to mark which pages correspond to which problem
- At the start of each problem, write a brief *collaboration statement* that lists the names and CS usernames of anyone you collaborated with and what ideas you discussed together
- If you consulted any outside resources while answering any question, you should cite them with your answer

There is only one Gradescope submission for this assignment. All students should submit Problems 1–3 to the assignment labeled **"Homework 4"** on Gradescope.

Problem 1: TLS vs. Blue University

Relevant lectures: Lecture 21

Blue University worked out a really good deal with a the CA AwesomeTrust to get a TLS certificate for its main website, `blue.university`. However, based on what you know about Blue University and the reputation of their business partners, you know something is going to go very wrong here...

Consider the following scenarios, which examine what happens when different parts of a public key infrastructure are affected if a certain key is compromised. For each part, write a brief explanation (1-2 sentences) of what capabilities you (as an attacker) can do in each scenario. Specifically, for each question, consider:

- Who could you impersonate?
- Who would believe you? (ie, For what set of users would you be able to impersonate X? All users in the world, or just a specific group?)

Note: For considering who you can impersonate, you don't need to worry about *how* you'd actually perform the attack at the network layer—instead, just **focus on how the TLS authentication process would be affected if you control the key in question**.

Question a) You compromise a University webserver and obtain the private key for `blue.university`. If you have this private key, what capabilities do you have? Give your answer by considering the two bullet points above and briefly explain your reasoning. You can impersonate `blue.university` for anyone who connects to the website. (2pts, +2 for identifying who to impersonate, +2 for identifying who would believe you; half credit if intuition is close but something significant is missing)

Question b) It turns out that the CA AwesomeTrust is pretty shady (big surprise...) and you're able to obtain AwesomeTrust's CA private key. If you have AwesomeTrust's CA private key, what capabilities do you have? Give your answer by considering the two bullet points above and briefly explain your reasoning. You can now sign your own certificates, so you can now impersonate any website for any user (at least until AwesomeTrust gets revoked or removed from browsers).

(4pts, +2 for identifying who to impersonate, +2 for identifying who would believe you; half credit if intuition is close but something significant is missing. Not required to state that AwesomeTrust would get revoked eventually.)

Question c) After you compromised AwesomeTrust, Blue University gives up on the idea of delegating trust to a third-party: it decides to *make its own CA* and issue its own certificate for `blue.university`. To make this work, the University's IT policy requires all users to install the Blue University CA certificate on their systems.

(i) Why do Blue University users need to install the CA certificate? What happens if a browser **doesn't** have the CA certificate installed and connects to `blue.university` anyway? If the CA certificate isn't installed, browsers will flag it as not trusted because they can't verify `blue.university`'s public key. If the CA isn't installed, this will display a warning when the user tries to connect.

(3pts, +2 for recognizing that certificate is not trusted, +1 for noting that browser will display warning; half-credit as applicable)

(ii) If you're able to obtain the Blue University CA private key, what capabilities do you have? Give your answer by considering the two bullet points above and briefly explain your reasoning. You can now sign your own certificates and thus impersonate any website, and any user who has the CA certificate installed (ie, all Blue University students/employees) will believe you.

(4pts, +2 for identifying who to impersonate, +2 for identifying who would believe you; half credit if intuition is close but something significant is missing.)

Problem 2: Onion-Flavored Handins

Relevant lectures: Lecture 22

The CS666 course at Blue University has its own Tor network to allow students to submit their code "super-anonymously" to a custom testing server. Unlike Gradescope, the testing server doesn't require students log in and instead just sends back a test report so students can check their work against the autograder. This way, only students' final, best submissions need to be uploaded to Gradescope and recorded with their names.[1]

Figure 1 depicts the CS666 Tor network and a specific circuit established between you and the testing server. Lines depict the specific Tor circuit established by your client using the nodes highlighted in blue. In this network, each Tor node i has a public key PK_i, which is known to all other Tor nodes and the client[2].

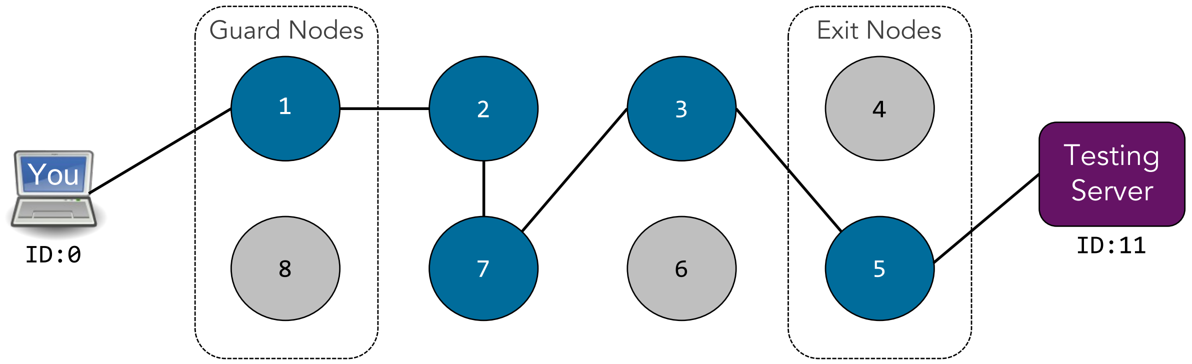

Figure 1: The CS666 Tor network. Note that the lines indicate connections in the Tor circuit, not individual network links.

Question a) To send a message M to the testing server using the circuit in Figure 1, what is the packet your Tor browser sends out?
You can express your answer in the format: `[dest-id, data]`
where `dest-id` is the ID of a node as labeled in Figure 1 and `data` is the packet content. For example, `[1, "Hello"]` sends the string "Hello" (in plaintext) to node 1. You can denote encryption of message M as $E(PK_i, M)$, where PK_i is the public key for a node with ID i. You can assume that each key is known to all Tor nodes and the client (this is established during circuit setup).

Question b) **True or False**: even if you connect to the testing server using plain HTTP (**not** HTTPS), no one can read M except the testing server—that is, TLS isn't required in order to have *confidentiality* in the presence of an eavesdropper in the network because the traffic is encrypted. **Explain your reasoning.**

Sample TA Solution

a) The packet is:

$$[1, E(PK_1, [2, E(PK_2, [7, E(PK_7, [3, E(PK_3, [5, E(PK_5, [11, M])])])])])]$$

Note that the final message from the exit node to the testing server is not encrypted.

5pts total. +2 points for correct ordering/onion structure (except last hop), +3 points for recognizing that the last hop is not encrypted. Partial credit as applicable.

[1] I promise you that we don't actually care about your non-final submissions and don't look unless we think there's an autograder bug. It makes a nice case study for Tor, though. ☺

[2] This is a simplification of the actual Tor protocol, which uses a combination of asymmetric and symmetric keys to set up circuits.

b) False. Based on our observation from part (a), we can see that confidentiality of M is not maintained as M is sent in plaintext (since HTTP is used, not HTTPS) between node 5 and the Gradescope server. Thus, a network eavesdropper between those two machines can read the contents of M.

5 pts total. For full credit, answer correctly reasons that HTTPS is required because data on the last hop is not encrypted using Tor. If answer for part (a) stated that the last hop was encrypted, full credit if answer states that HTTP is **not** required because last hop is encrypted (ie, no double jeopardy if last part is wrong). Partial credit as applicable if intuition is correct.

Problem 3: Lab: Wireshark

This problem is a short "lab" designed to give you practice using Wireshark.

To complete the lab, follow the instructions here: `https://hackmd.io/@cs1660/wireshark-lab`

At the end of the lab, you'll be asked find some items in a packet capture and include them in your submission. To receive credit for this problem, just include these items in your submission.

Graded on completion, +5 for including a screenshot that looks reasonable.