Akshay Srivatsan, Ayelet Drazen, & Jonathan Kula

Assignment #7—The (Public) Key to My Heart

Due: March 8, 2023 at 11:59pm

This assignment consists of two different components:

1. Spoof an email
2. Set up SSH keys to sign a message

We expect this assignment to take 1-3 hours depending on your proficiency level with the tools. If you find yourself unproductively stuck or unproductively struggling, ask on Ed and/or go to office hours!

> ***Warning:*** *This assignment teaches you a technique for email spoofing that could be used to compromise another's security.* You should not use this technique outside of this assignment. Neither CS45 course staff nor Stanford will take liability for any legal repercussions resulting from using this technique outside the context and parameters of this assignment.

Part I: Spoof an Email (1.5 points)

For this part of the assignment, you will get to spoof an email. Please note that we are showing you this technique as part of an assignment, with the purpose of teaching you about security techniques. Read the disclaimer above before starting this part of the assignment.

Thank you to Keith Winstein for inspiration for this part of the assignment.

Exercise 1: Send an Email to Yourself

1. To get started, you will want to log into our CS45 server. To log in, open a Terminal and type the following two lines, replacing `<SUNet>` with your SUNet:

```
Unset
export SUNET=<SUNet>
ssh s-cs45-assign7-$SUNET@192.9.152.85
```

   Once you are logged in, you will automatically be directed to run the Simple Mail Transfer Protocol (`smtp`). (We do this to ensure that you can't "explore" around our server and "accidentally" take it down... Security 😁).

   If all goes well, you should get immediately get some output that looks like the following immediately after logging in:

```
Unset
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
220 honeypot.vcn.oraclevcn.com ESMTP Postfix (Ubuntu)
```

2. Next, you'll want to specify who the email is from (which in this case, is yourself). To do so, you will want to type in `MAIL FROM:`, a space, your email address, and then hit the Enter key. This should look as follows:

```
Unset
MAIL FROM: <SUNET>@stanford.edu ↵
```

   If all goes well, you will see `250 2.1.0 Sender ok`.

   If you are running into an issue where the sender address is not being accepted (i.e. you are getting an error), make sure you don't have extra characters anywhere. You can also try encapsulating the address with `<` and `>` as in:

   `MAIL FROM: <adrazen@stanford.edu>`

3. Now, you will want to specify who the email is to (which in this case, is yourself). To do so, you will want to type in `RCPT TO:`, a space, your email address, and then hit the Enter key. This should look as follows:

```
Unset
RCPT TO: <SUNET>@stanford.edu ↵
```

   If all goes well, you will see `250 2.1.5 Recipient ok`.

   If you are running into an issue where the recipient address is not being accepted (i.e. you are getting an error), make sure you don't have extra characters anywhere. You can also try encapsulating the address with `<` and `>` as in:

   `RCPT TO: <adrazen@stanford.edu>`

4. Now it's time to construct the message itself. Type `DATA` and then hit the Enter key.

```
Unset
DATA ↵
```

If all goes well, you will see `354 End data with <CR><LF.<CR><LF>`.

First, we will add the headers. This may feel redundant to the `MAIL FROM:` and `RCPT TO:` that you entered above. However, the `MAIL FROM:` and `RCPT TO:` from above are part of the envelope, and are used by the SMTP protocol to specify specific delivery instructions for where the email needs to go.

Meanwhile, the headers that we will add below are used by email clients (i.e. the sender and receiver), but are *not* used for actually routing and delivering the email. You should add a `From`, `To` and `Subject` header. Make sure to add a blank line at the end of the headers. This should look as follows:

```
Unset
From: <SUNET>@stanford.edu ↵
To: <SUNET>@stanford.edu ↵
Subject: Hello from CS45! ↵
↵
```

5. Now you can construct the email body. You can write anything you want! When you are done constructing the email body, press the Enter key. Then add a line with just a `.` (period) on it and then press Enter. This should look something like this:

```
Unset
354 End data with <CR><LF.<CR><LF>
Hi this is my email! ↵
. ↵
```

If all goes well, you should see a confirmation message: `250 2.0.0 33h24dpdsr-1 Message accepted for delivery.`

6. When you are done, type `QUIT` and hit Enter to exit out of the email server.
7. If you go to your Stanford email inbox, you should see a new email from yourself! (If you don't see it after a few minutes, check your spam folder.)

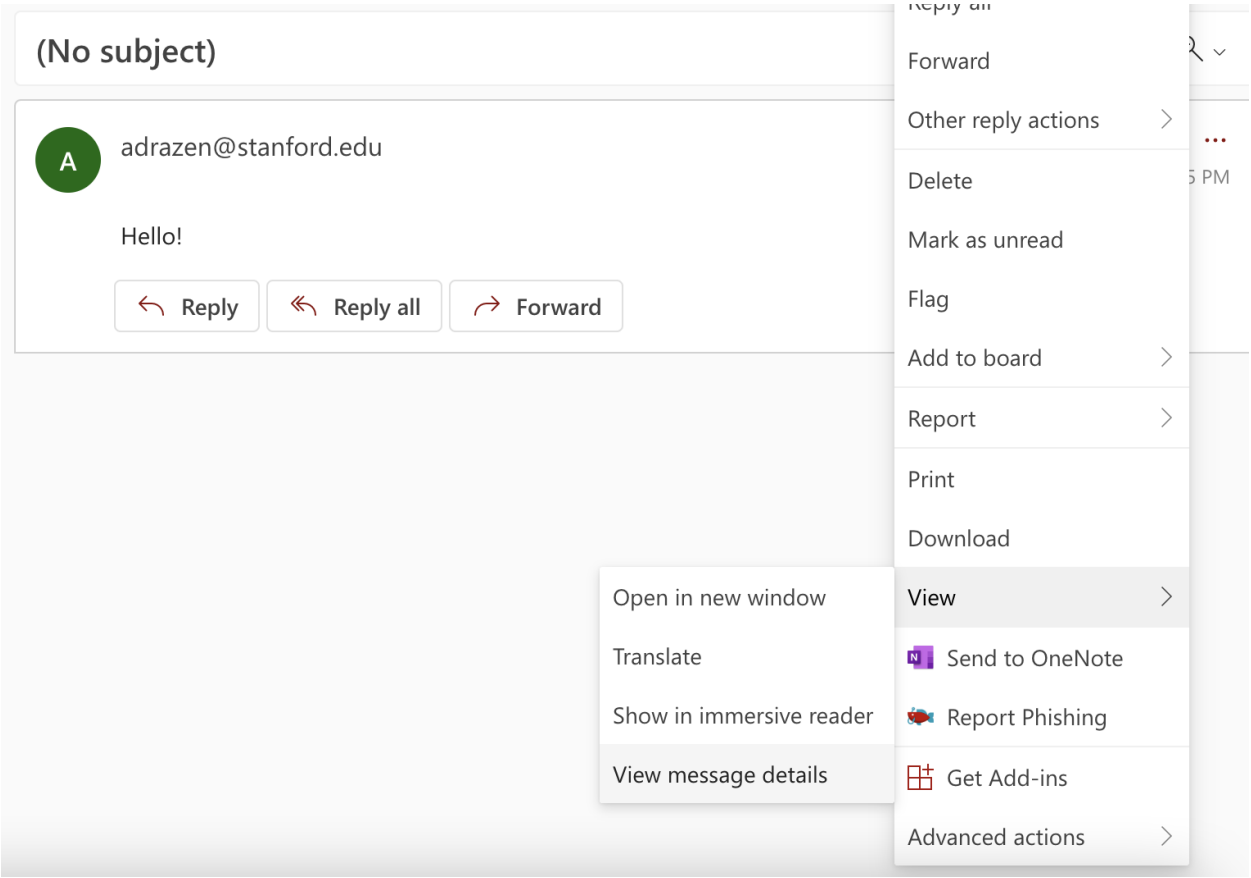
Exercise 2: Send a Spoofed Email to Yourself

To send a spoofed email to yourself, you will want to repeat the process from Exercise 1.

However, this time you should change the `MAIL FROM:` address in step 2 to be `cs45-admin@stanford.edu`. (This is a fake email address that we are allowing you to use as part of this assignment.) You will also want to change the `From:` header in Step 4 to be `cs45-admin@stanford.edu`.

What happens if you change one and not the other? Try changing just the `MAIL FROM:` address in step 2 and not the `From:` header in Step 4. Now try changing just the `From:` header in Step 4 and not the `MAIL FROM:` address in step 2. Can you see a difference?

To explore what the difference is, try opening the spoofed emails in the Outlook mail server. Open the "More Actions" menu (three dots) and then select "View > View Message Details". Can you see anything suspicious about the email here?



Exercise 3: Send a Spoofed Email to Us

Now you'll want to send us a spoofed email from `cs45-admin@stanford.edu`. You'll want to repeat the steps in Exercise 2 and send the email to `cs45-win2223-staff@mailman.stanford.edu`.

In order to confirm that your email gets successfully delivered, we recommend BCCing yourself on the email. In order to do that, you will want to add an additional recipient using `RCPT TO:`. In other words, you should repeat Step 3 from above twice. The first time, you should add the recipient as `cs45-win2223-staff@mailman.stanford.edu` and the second time you should add the recipient as your own email address. You can set the `To:` header to just be `cs45-win2223-staff@mailman.stanford.edu`.

Make sure to include your SUNet somewhere in the message body as this is how you will get credit for this part of the assignment.

Part II: Generate SSH Key to Sign A File (1.5 points)

For this next part of the assignment, we'll get to use asymmetric cryptography! We'll set up SSH (Secure Shell) keys and then see how we can use these keys to sign a file.

Step 1: Setting up SSH Keys

To begin, we need to set up our SSH keys. You may have set up SSH keys before in the past. If you already have SSH keys, you can skip to Step 2.

SSH keys are *incredibly* useful.

To check whether you have existing SSH keys, you can run the following command:

```
Unset
ls -al ~/.ssh
```

The output of this command will list all of your SSH keys. If you have existing SSH keys, the output will contain two lines that look something like this:

```
Unset
-r--------   1 adrazen  staff   411 Sep 22  2021 id_ed25519
-rw-r--r--   1 adrazen  staff   102 Sep 22  2021 id_ed25519.pub
```

Notice the different file permissions between the first line (containing the private key) and the second line (containing the public key).

If you don't have SSH keys, you'll want to run the following command, replacing `your_email@example.com` with your Stanford email address:

```
Unset
ssh-keygen -t ed25519 -C "your_email@example.com"
```

Once you hit Enter, the command will tell you that it's generating your Public/Private key pair with the message `Generating public/private ed25519 key pair`.

It will then prompt you asking you where to save your SSH keys:

```
Unset
Enter a file in which to save the key
(/Users/your_username/.ssh/id_ed25519): [Press enter]
```

You should hit the Enter button in order to use the default location (`/Users/your_username/.ssh/id_ed25519`).

You will then be prompted to provide a secure passphrase to protect your SSH keys. Choose a passphrase that you won't forget.

Congratulations! You've successfully generated your SSH keys. You can now examine your newly-minted SSH keys by navigating to `~/.ssh`. Inside of this directory you should see two files: `id_ed25519` and `id_ed25519.pub`. These are your private and public keys respectively. Feel free to print them out using `cat` to examine their contents (in particular, you'll need the contents of the `id_ed25519.pub` file in the next part). Remember, it's okay to share your public key with others, but your private key should always be a secret. If someone has your private key, your key is "compromised" (they can now pretend to be you and take actions on your behalf).

Step 2: Signing A File

Let's sign a file using your SSH key! First you will need to create a file. Create a file called `file.txt` with a fun message or fact that you want to sign and send to us!

To sign a file using your SSH key, you will want to run the following command:

```
Unset
ssh-keygen -Y sign -n file -f ~/.ssh/id_ed25519 file.txt
```

If you're interested in learning about the flags we are using with `ssh-keygen`, you can run `man ssh-keygen` to learn more.

If all goes well, running this command should sign the provided file and generate a signature file ending in `.sig`.

Now, you'll want to provide us with all of the necessary information so we can verify that you indeed signed this file.

To do so, you'll want to create a file called `allowed_signer` that contains your email, the cryptography algorithm you used (`ed25519`) and your public key. The contents of `allowed_signer` file should look as follows:

```
Unset
<email_address> ssh-ed25519 <your-public-key>
```

Make sure to give us your **public** key (from the `.pub` file), not your private key. No one should ever see your private key except you!

Now it's time to submit! You will need to submit your signature file (ending in `.sig`), your `allowed_signer` file, and your original file. We'll verify your signature to make sure it's really your file.

Feedback Survey (0.5 points)

Once you have completed the assignment, you can earn an additional 0.5 points by completing our anonymous feedback survey. Given this is the first offering of the course, we want to collect as much feedback as possible to improve the course in the future. You can complete the survey here. Once you complete the survey, you will receive a completion code which you should place in a text file named `survey.txt`. The survey is anonymous so submitting the completion code is the only way to verify that you completed the survey. *Please do not share this code with anyone, as that would constitute a breach of the honor code.*

Submitting Your Assignment

Once you have finished this assignment, you will need to upload your files to Gradescope in addition to sending the spoofed email in Part 1. Make sure to upload all files to the Assignment 7 submission page. You should also upload `survey.txt` if you completed the survey.

This autograder for this assignment will grade Part II and your survey code.

```
Unset
zip -jv assign7_submission.zip ./survey.txt ./file.txt
./file.txt.sig ./allowed_signer
```

All files must have the same name as specified above.