Computer and Information Security

(ECE560, Fall 2023, Duke Univ., Prof. Tyler Bletsch)

Pointer to Homework 2

Question 0: Accessing the Homework (0 points, but necessary)

Homework 2 is encrypted with three stages of encryption. You'll need to use both your Windows VM and a new Kali Linux VM. The stages are:

1. The **inner layer** is a VeraCrypt encrypted disk image to be opened in Windows that contains a link to Homework 2; I explain how to find the key for this later in this document. You get to the inner layer by decrypting the outer layer.
2. The **outer layer** is encrypted with AES and is available for each student on the course site; the secret key is randomly generated per each student and is distributed by the **Encrypted Thing Giver** web app.
3. The **Encrypted Thing Giver** accepts an RSA public key, encrypts the random secret key using this public key. As owner of the corresponding private key, you'll be able to decrypt it in order to obtain the AES key for the outer layer.

The steps below will walk you through this crypto journey. **You will need to show your work later, so keep notes!**

Make an SSH key pair

You will need an SSH key pair. If you already created one, you can use it as-is (provided it is RSA-based, but most are).

If you don't have an SSH key pair, research how to create one with ssh-keygen. You may wish to do this on your local system, as you can set it up to let you SSH into Duke machines without a password (optional -- see Appendix A at the end of this document).

Note: the private key is `id_rsa`, the corresponding public key is `id_rsa.pub`. **Keep your SSH keys secure. Never divulge the private key to anyone, even me! Later assignments may rely on this keypair, so don't lose it.**

Accessing your Kali Linux installation

If you haven't already, reserve a Kali Linux virtual machine in VCM. This is labeled "ECE 560 F22: Kali" in VCM; we will call this your **Kali VM**. This is a Linux distribution focused on providing a wide variety of security tools built-in (both attack and defense).

Encrypted Thing Giver

For the outer layer of encryption, we will use the OpenSSL cipher suite. This tool is widely available, but you'll be using the one installed on your personal Kali Linux VM. Versions of OpenSSL differ significantly, and a version mismatch can make this step fail, so be sure to use the Kali Linux VM for this part, even if you have another environment with OpenSSL installed.

The SSH keypair you generated is based on the RSA algorithm, and can therefore also function as a traditional RSA key pair for asymmetric encryption/decryption in addition to SSH authentication. However, recall that RSA is much slower than symmetric cryptography, so in this step, a manually constructed *digital envelope* is employed using both your RSA keypair and a random secret key.

A 256-bit key was generated randomly for each student, and will be provided in a form that is RSA-encrypted using the student's submitted public key. This encrypted key is available as **`secret-<NETID>.key.enc`**. You can obtain this by submitting your public key to the Encrypted Thing Giver, a web tool developed for this purpose:

> http://target.colab.duke.edu:8000/

Convert your SSH private key's format

The tool you used to make an RSA key pair earlier was an SSH tool, `ssh-keygen`. Next, we'll use another tool, OpenSSL, to use this key to do RSA decryption. However, while the RSA *algorithm* is the same between both tools, they may use a differing format to store the key (depending on version).

Therefore, let's convert our SSH private key to the more generic "PEM" format. First, make a copy of you private key (replacing the filenames below with yours as needed):

```
cp id_rsa id_rsa.privpem
```

The following command does the actual conversion. This conversion is in-place, meaning the key file is modified (that's why we're doing it to a copy instead of the original key).

```
ssh-keygen -p -N "" -m pem -f id_rsa.privpem
```

In the subsequent RSA decryption step with OpenSSL, use this converted key.

NOTE: This converted file *also* represents your private key, and should be kept equally secure.

Outer layer of encryption

Using the **`openssl rsautl`** command on your Kali VM, decrypt the secret key from the Encrypted Thing Giver using your converted RSA private key. The result will be a text file showing the hex representation of a 256-bit AES key.

Now we can decrypt the payload, a 1MB data file available as **`data-<NETID>.dat`**, found in the directory linked from the course site. Use the **`openssl enc`** command to decrypt this ciphertext. Use the AES algorithm with a 256-bit key (provided with the -K option); the mode of operation should be the one that chains one block of ciphertext to the next block of plaintext using xor. You'll need to provide an **initialization vector**; simply use `dd1c46682156e180e85786dd821e9e59`.

To check the result of the decryption process, note that the SHA-256 hash of the output should be: `932f5cf0b48d86606c3629dd4f226d44c8f575f916e9dfc6dbea1c42402de905`. You can check this with **`sha256sum`**.

Inner layer of encryption

Once you decrypt the file, copy it to your Windows VM. The file is a disk image encrypted with 256-bit AES using VeraCrypt. VeraCrypt is a tool that allows one to store a read/write filesystem inside an encrypted container. Mount the volume and use the secret key to decrypt the volume.

The secret key is the *real* hostname of `target.colab.duke.edu`. DNS allows for aliases called CNAMEs, allowing one domain name to refer to another. Using the DNS tools of your choice, find what `target.colab.duke.edu` is an alias for (HINT: it will end in "`vm.duke.edu`", omit any trailing dot if your DNS tool includes one).

Enclosed in this volume is an HTML file that will link you to the homework.

Do not share the link with others: finding the link is part of the assignment. You may of course discuss conceptually how to perform the necessary steps with your colleagues, but please do not post or share information that short-circuits the above procedures (e.g. the inner-level encrypted file) or reduces them to commands to just blindly run.

Be sure to unmount your encrypted volume when you are done.

Appendix A: Using your SSH keypair for login

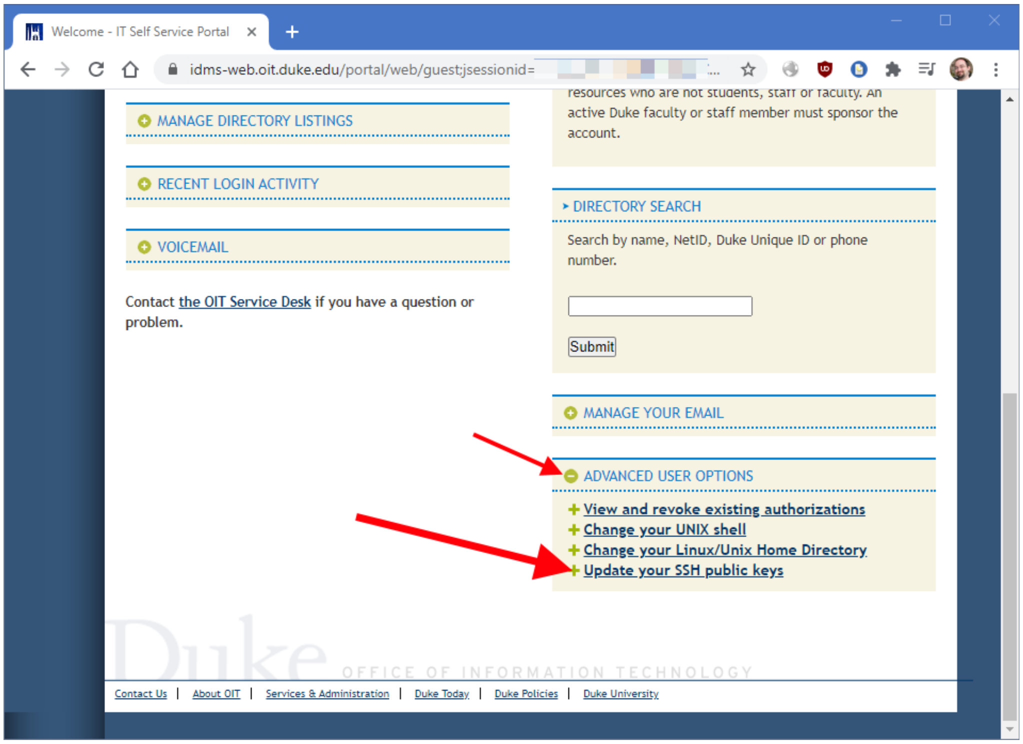
This is optional and is not required to access Homework 2, but it may improve your life.

The most common use for an SSH keypair is to allow passwordless login via SSH. On an individual system, this is normally configured by adding the key to your ~/.ssh/authorized_keys file. At Duke, however, we have a network-wide SSH key facility, allowing SSH keypair login to many Duke systems, including your VCM VMs!

To set your Duke authorized SSH public key, visit the Account Self Service site:

https://idms-web.oit.duke.edu/

Once there, access the “Update your SSH public keys” option under “Advanced User Options”:



Welcome - IT Self Service Portal

idms-web.oit.duke.edu/portal/web/guest;jsessionid=[redacted]...

resources who are not students, staff or faculty. An active Duke faculty or staff member must sponsor the account.

Contact the OIT Service Desk if you have a question or problem.

Search by name, NetID, Duke Unique ID or phone number.

MANAGE DIRECTORY LISTINGS
RECENT LOGIN ACTIVITY
VOICEMAIL
DIRECTORY SEARCH
Submit
MANAGE YOUR EMAIL
ADVANCED USER OPTIONS
View and revoke existing authorizations
Change your UNIX shell
Change your Linux/Unix Home Directory
Update your SSH public keys
Duke
OFFICE OF INFORMATION TECHNOLOGY
Contact Us | About OIT | Services & Administration | Duke Today | Duke Policies | Duke University

There you can paste as many SSH public keys as you want, one per line. After doing this, you should be able to SSH into VCM VMs with your NetID and private key.