

NAME: PRANATHEE H

ROLL NO.:241901078

EXPERIMENT 1

CAPTURE FLAGS-ENCRYPTION CRYPTO 101

AIM:

To capture the various flags in Encryption Crypto 101 in TryHackMe platform.

ALGORITHM:

1. Access the Passive reconnaissance lab in TryHackMe platform using the link below-

<https://tryhackme.com/r/room/encryptioncrypto101>

2. Click Start AttackBox to run the instance of Kali Linux distribution.

3. Solve the crypto math used in RSA.

4. Find out who issued the HTTPS Certificate to tryhackme.com

5. Perform SSH Authentication by generating public and private key pair using ssh-keygen

6. Perform decryption of the jpg encrypted file and find out the secret word.

OUTPUT:

The screenshot shows a dark-themed interface for a room titled "Encryption - Crypto 101". At the top, there's a navigation bar with "Learn > Encryption - Crypto 101", a lock icon, and the title "Encryption - Crypto 101". Below the title, it says "An introduction to encryption, as part of a series on crypto" and shows a progress bar at 45 min and 132,589 participants. A banner at the top indicates "Room completed (100%)".

A message box at the top states: "Our Asia Pacific VM region is now fully available and should offer you better performance." with "More Info" and "X" buttons.

The main content area lists 12 tasks, each with a green checkmark and a small circular icon:

- Task 1: What will this room cover?
- Task 2: Key terms
- Task 3: Why is Encryption important?
- Task 4: Crucial Crypto Maths
- Task 5: Types of Encryption
- Task 6: RSA - Rivest Shamir Adleman
- Task 7: Establishing Keys Using Asymmetric Cryptography
- Task 8: Digital signatures and Certificates
- Task 9: SSH Authentication
- Task 10: Explaining Diffie Hellman Key Exchange
- Task 11: PGP, GPG and AES
- Task 12: The Future - Quantum Computers and Encryption

Task 1 is expanded to show its details:

This room will cover:

- Why cryptography matters for security and CTFs
- The two main classes of cryptography and their uses
- RSA, and some of the uses of RSA
- 2 methods of Key Exchange
- Notes about the future of encryption with the rise of Quantum Computing

Note: This room expects some familiarity with tools, and some research into how to use them yourself!

Answer the questions below

I'm ready to learn about encryption

No answer needed ✓ Correct Answer

Task 1 ✓ What will this room cover?



Task 2 ✓ Key terms



Many of these key terms are shared with <https://tryhackme.com/room/hashingcrypto101>, so you might be able to skip over some if you're already familiar.

Ciphertext - The result of encrypting a plaintext, encrypted data

Cipher - A method of encrypting or decrypting data. Modern ciphers are cryptographic, but there are many non cryptographic ciphers like Caesar.

Plaintext - Data before encryption, often text but not always. Could be a photograph or other file

Encryption - Transforming data into ciphertext, using a cipher.

Encoding - NOT a form of encryption, just a form of data representation like base64. Immediately reversible.

Key - Some information that is needed to correctly decrypt the ciphertext and obtain the plaintext.

Passphrase - Separate to the key, a passphrase is similar to a password and used to protect a key.

Asymmetric encryption - Uses different keys to encrypt and decrypt.

Symmetric encryption - Uses the same key to encrypt and decrypt

Brute force - Attacking cryptography by trying every different password or every different key

Cryptanalysis - Attacking cryptography by finding a weakness in the underlying maths

Alice and Bob - Used to represent 2 people who generally want to communicate. They're named Alice and Bob because this gives them the initials A and B.

https://en.wikipedia.org/wiki/Alice_and_Bob for more information, as these extend through the alphabet to represent many different people involved in communication.

WARNING: This room is very theory heavy. Cryptography is a big topic, and this room is designed to just scratch the surface.

Answer the questions below

I agree not to complain too much about how theory heavy this room is.

No answer needed

✓ Correct Answer

Are SSH keys protected with a passphrase or a password?

passphrase

✓ Correct Answer

💡 Hint

Task 3 ✓ Why is Encryption important?



Cryptography is used to protect confidentiality, ensure integrity, ensure authenticity. You use cryptography every day most likely, and you're almost certainly reading this now over an encrypted connection.

When logging into TryHackMe, your credentials were sent to the server. These were encrypted, otherwise someone would be able to capture them by snooping on your connection.

When you connect to SSH, your client and the server establish an encrypted tunnel so that no one can snoop on your session.

When you connect to your bank, there's a certificate that uses cryptography to prove that it is actually your bank rather than a hacker.

When you download a file, how do you check if it downloaded right? You can use cryptography here to verify a checksum of the data.

You rarely have to interact directly with cryptography, but it silently protects almost everything you do digitally.

Whenever sensitive user data needs to be stored, it should be encrypted. Standards like PCI-DSS state that the data should be encrypted both at rest (in storage) AND while being transmitted. If you're handling payment card details, you need to comply with these PCI regulations. Medical data has similar standards. With legislation like GDPR and California's data protection, data breaches are extremely costly and dangerous to you as either a consumer or a business.

DO NOT encrypt passwords unless you're doing something like a password manager. Passwords should not be stored in plaintext, and you should use hashing to manage them safely.

Answer the questions below

What does SSH stand for?

Secure Shell

✓ Correct Answer

How do web servers prove their identity?

certificates

✓ Correct Answer

💡 Hint

What is the main set of standards you need to comply with if you store or process payment card details?

PCI-DSS

✓ Correct Answer

Task 4 Crucial Crypto Maths

There's a little bit of math(s) that comes up relatively often in cryptography. The Modulo operator. Pretty much every programming language implements this operator, or has it available through a library. When you need to work with large numbers, use a programming language. Python is good for this as integers are unlimited in size, and you can easily get an interpreter.

When learning division for the first time, you were probably taught to use remainders in your answer. $X \% Y$ is the remainder when X is divided by Y .

Examples

$25 \% 5 = 0$ ($5 * 5 = 25$ so it divides exactly with no remainder)

$23 \% 6 = 5$ (23 does not divide evenly by 6, there would be a remainder of 5)

An important thing to remember about modulo is that it's not reversible. If I gave you an equation: $x \% 5 = 4$, there are infinite values of x that will be valid.

Answer the questions below

What's $30 \% 5$?

✓ Correct Answer

What's $25 \% 7$?

✓ Correct Answer

What's $118613842 \% 9091$?

✓ Correct Answer

💡 Hint

Task 5 Types of Encryption

The two main categories of Encryption are symmetric and asymmetric.

Symmetric encryption uses the same key to encrypt and decrypt the data. Examples of Symmetric encryption are DES (Broken) and AES. These algorithms tend to be faster than asymmetric cryptography, and use smaller keys (128 or 256 bit keys are common for AES, DES keys are 56 bits long).

Asymmetric encryption uses a pair of keys, one to encrypt and the other in the pair to decrypt. Examples are RSA and Elliptic Curve Cryptography. Normally these keys are referred to as a public key and a private key. Data encrypted with the private key can be decrypted with the public key, and vice versa. Your private key needs to be kept private, hence the name. Asymmetric encryption tends to be slower and uses larger keys, for example RSA typically uses 2048 to 4096 bit keys.

RSA and Elliptic Curve cryptography are based around different mathematically difficult (intractable) problems, which give them their strength. More about RSA later.

Answer the questions below

Should you trust DES? Yea/Nay

✓ Correct Answer

💡 Hint

What was the result of the attempt to make DES more secure so that it could be used for longer?

✓ Correct Answer

💡 Hint

Is it ok to share your public key? Yea/Nay

✓ Correct Answer

Task 6 RSA - Rivest Shamir Adleman

The math(s) side

RSA is based on the mathematically difficult problem of working out the factors of a large number. It's very quick to multiply two prime numbers together, say $17 \times 23 = 391$, but it's quite difficult to work out what two prime numbers multiply together to make 14351 (113x127 for reference).

The attacking side

The maths behind RSA seems to come up relatively often in CTFs, normally requiring you to calculate variables or break some encryption based on them. The wikipedia page for RSA seems complicated at first, but will give you almost all of the information you need in order to complete challenges.

There are some excellent tools for defeating RSA challenges in CTFs, and my personal favorite is <https://github.com/Ganapati/RsaCtfTool> which has worked very well for me. I've also had some success with <https://github.com/ius/rsatool>.

The key variables that you need to know about RSA in CTFs are p, q, m, n, e, d, and c.

"p" and "q" are large prime numbers, "n" is the product of p and q.

The public key is n and e, the private key is n and d.

"m" is used to represent the message (in plaintext) and "c" represents the ciphertext (encrypted text).

CTFs involving RSA

Crypto CTF challenges often present you with a set of these values, and you need to break the encryption and decrypt a message to retrieve the flag.

There's a lot more maths to RSA, and it gets quite complicated fairly quickly. If you want to learn the maths behind it, I recommend reading MuirlandOracle's blog post here: <https://muirlandoracle.co.uk/2020/01/29/rsa-encryption/>.

Answer the questions below

p = 4391, q = 6659. What is n?

29239669

✓ Correct Answer

✗ Hint

I understand enough about RSA to move on, and I know where to look to learn more if I want to.

No answer needed

✓ Correct Answer

Task 7 Establishing Keys Using Asymmetric Cryptography

A very common use of asymmetric cryptography is exchanging keys for symmetric encryption.

Asymmetric encryption tends to be slower, so for things like HTTPS symmetric encryption is better.

But the question is, how do you agree a key with the server without transmitting the key for people snooping to see?

Metaphor time

Imagine you have a secret code, and instructions for how to use the secret code. If you want to send your friend the instructions without anyone else being able to read it, what you could do is ask your friend for a lock.

Only they have the key for this lock, and we'll assume you have an indestructible box that you can lock with it.

If you send the instructions in a locked box to your friend, they can unlock it once it reaches them and read the instructions.

After that, you can communicate in the secret code without risk of people snooping.

In this metaphor, the secret code represents a symmetric encryption key, the lock represents the server's public key, and the key represents the server's private key.

You've only used asymmetric cryptography once, so it's fast, and you can now communicate privately with symmetric encryption.

The Real World

In reality, you need a little more cryptography to verify the person you're talking to is who they say they are, which is done using digital signatures and certificates. You can find a lot more detail on how HTTPS (one example where you need to exchange keys) really works from this excellent blog post. <https://robertheaton.com/2014/03/27/how-does-https-actually-work/>

Answer the questions below

I understand how keys can be established using Public Key (asymmetric) cryptography.

No answer needed

✓ Correct Answer

Task 8 Digital signatures and Certificates

What's a Digital Signature?

Digital signatures are a way to prove the authenticity of files, to prove who created or modified them. Using asymmetric cryptography, you produce a signature with your private key and it can be verified using your public key. As only you should have access to your private key, this proves you signed the file. Digital signatures and physical signatures have the same value in the UK, legally.

The simplest form of digital signature would be encrypting the document with your private key, and then if someone wanted to verify this signature they would decrypt it with your public key and check if the files match.

Certificates - Prove who you are!

Certificates are also a key use of public key cryptography, linked to digital signatures. A common place where they're used is for HTTPS. How does your web browser know that the server you're talking to is the real tryhackme.com?

The answer is certificates. The web server has a certificate that says it is the real tryhackme.com. The certificates have a chain of trust, starting with a root CA (certificate authority). Root CAs are automatically trusted by your device, OS, or browser from install. Certs below that are trusted because the Root CAs say they trust that organisation. Certificates below that are trusted because the organisation is trusted by the Root CA and so on. There are long chains of trust. Again, this blog post explains this much better than I can.
<https://robertheaton.com/2014/03/27/how-does-https-actually-work/>

You can get your own HTTPS certificates for domains you own using Let's Encrypt for free. If you run a website, it's worth setting it up.

Answer the questions below

What can you use to verify that a file has not been modified and is the authentic file as the author intended?

Digital Signature

✓ Correct Answer

Task 9 SSH Authentication

Encryption and SSH authentication

Download Task Files

By default, SSH is authenticated using usernames and passwords in the same way that you would log in to the physical machine.

At some point, you're almost certain to hit a machine that has SSH configured with key authentication instead. This uses public and private keys to prove that the client is a valid and authorised user on the server. By default, SSH keys are RSA keys. You can choose which algorithm to generate, and/or add a passphrase to encrypt the SSH key. ssh-keygen is the program used to generate pairs of keys most of the time.

SSH Private Keys

You should treat your private SSH keys like passwords. Don't share them, they're called private keys for a reason. If someone has your private key, they can use it to log in to servers that will accept it unless the key is encrypted.

It's very important to mention that the passphrase to decrypt the key isn't used to identify you to the server at all, all it does is decrypt the SSH key. The passphrase is never transmitted, and never leaves your system.

Using tools like John the Ripper, you can attack an encrypted SSH key to attempt to find the passphrase, which highlights the importance of using a secure passphrase and keeping your private key private.

When generating an SSH key to log in to a remote machine, you should generate the keys on your machine and then copy the public key over as this means the private key never exists on the target machine. For temporary keys generated for access to CTF boxes, this doesn't matter as much.

How do I use these keys?

The ~/.ssh folder is the default place to store these keys for OpenSSH. The authorized_keys (note the US English spelling) file in this directory holds public keys that are allowed to access the server if key authentication is enabled. By default on many distros, key authentication is enabled as it is more secure than using a password to authenticate. Normally for the root user, only key authentication is enabled.

In order to use a private SSH key, the permissions must be set up correctly otherwise your SSH client will ignore the file with a warning. Only the owner should be able to read or write to the private key (600 or stricter). ssh -i keyNameGoesHere user@host is how you specify a key for the standard Linux OpenSSH client.

Using SSH keys to get a better shell

SSH keys are an excellent way to "upgrade" a reverse shell, assuming the user has login enabled (www-data normally does not, but regular users and root will). Leaving an SSH key in authorized_keys on a box can be a useful backdoor, and you don't need to deal with any of the issues of unstabilised reverse shells like Control-C or lack of tab completion.

Answer the questions below

I recommend giving this a go yourself. Deploy a VM, like Linux Fundamentals 2 and try to add an SSH key and log in with the private key.

No answer needed

✓ Correct Answer

Hint

Download the SSH Private Key attached to this room.

No answer needed

✓ Correct Answer

Hint

What algorithm does the key use?

RSA

✓ Correct Answer

Hint

Crack the password with John The Ripper and rockyou, what's the passphrase for the key?

delicious

✓ Correct Answer

Hint

Task 10 Explaining Diffie Hellman Key Exchange

What is Key Exchange?

Key exchange allows 2 people/parties to establish a set of common cryptographic keys without an observer being able to get these keys. Generally, to establish common symmetric keys.

How does Diffie Hellman Key Exchange work?

Alice and Bob want to talk securely. They want to establish a common key, so they can use symmetric cryptography, but they don't want to use key exchange with asymmetric cryptography. This is where DH Key Exchange comes in.

Alice and Bob both have secrets that they generate, let's call these A and B. They also have some common material that's public, let's call this C.

We need to make some assumptions. Firstly, whenever we combine secrets/material it's impossible or very very difficult to separate. Secondly, the order that they're combined in doesn't matter.

Alice and Bob will combine their secrets with the common material, and form AC and BC. They will then send these to each other, and combine that with their secrets to form two identical keys, both ABC. Now they can use this key to communicate.

Extra Resources

An excellent video if you want a visual explanation is available here. <https://www.youtube.com/watch?v=NmM9HA2MQGI>

DH Key Exchange is often used alongside RSA public key cryptography, to prove the identity of the person you're talking to with digital signing. This prevents someone from attacking the connection with a man-in-the-middle attack by pretending to be Bob.

Answer the questions below

I understand how Diffie Hellman Key Exchange works at a basic level

No answer needed

✓ Correct Answer

Task 11 PGP, GPG and AES

What is PGP?

Download Task Files

PGP stands for Pretty Good Privacy. It's a software that implements encryption for encrypting files, performing digital signing and more.

What is GPG?

GnuPG or GPG is an Open Source implementation of PGP from the GNU project. You may need to use GPG to decrypt files in CTFs. With PGP/GPG, private keys can be protected with passphrases in a similar way to SSH private keys. If the key is passphrase protected, you can attempt to crack this passphrase using John The Ripper and gpg2john. The key provided in this task is not protected with a passphrase.

The man page for GPG can be found online [here](#).

What about AES?

AES, sometimes called Rijndael after its creators, stands for Advanced Encryption Standard. It was a replacement for DES which had short keys and other cryptographic flaws.

AES and DES both operate on blocks of data (a block is a fixed size series of bits).

AES is complicated to explain, and doesn't seem to come up as often. If you'd like to learn how it works, here's an excellent video from Computerphile <https://www.youtube.com/watch?v=O4xNUsjtN6E>

Answer the questions below

Time to try some GPG. Download the archive attached and extract it somewhere sensible.

No answer needed

✓ Correct Answer

You have the private key, and a file encrypted with the public key. Decrypt the file. What's the secret word?

Pineapple

✓ Correct Answer

✗ Hint

```
File Edit View Search Terminal Help
root@ip-10-10-18-189:~# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa): myKey
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in myKey.
Your public key has been saved in myKey.pub.
The key fingerprint is:
SHA256:mYLMN1vmJnLZgFjuatvJ+maOmK9HcIARIE//j0dXt9s root@ip-10-10-18-189
The key's randomart image is:
+---[RSA 2048]---+
|==   . |
|o.. + . |
| ... o . |
| ..o.o + |
| .o+ = S . |
| ..o O o.. |
| .+ + =. . . |
| +.0+=. . . |
| ++*OX. ..E |
+---[SHA256]---+
root@ip-10-10-18-189:~# ls
burp.json    Downloads      myKey.pub    Rooms          Tools
CTFBuilder   Instructions  Pictures     Scripts        welcome.txt
Desktop      myKey         Postman     thinclient  drives  welcome.txt.gpg
```

```
root@ip-10-10-18-189:~# gpg --import tryhackme.key
gpg: /root/.gnupg/trustdb.gpg: trustdb created
gpg: key FFA4B5252BAEB2E6: public key "TryHackMe (Example Key)" imported
gpg: key FFA4B5252BAEB2E6: secret key imported
gpg: Total number processed: 1
gpg:           imported: 1
gpg:   secret keys read: 1
gpg:   secret keys imported: 1
```

```
root@ip-10-10-18-189:~# gpg message.gpg
```

```
gpg: WARNING: no command supplied. Trying to guess what you mean ...
gpg: encrypted with 1024-bit RSA key, ID 2A0A5FDC5081B1C5, created 2020-06-30
    "TryHackMe (Example Key)"
```

```
gpg: WARNING: no command supplied. Trying to guess what you mean ...
gpg: encrypted with 1024-bit RSA key, ID 2A0A5FDC5081B1C5, created 2020-06-30
    "TryHackMe (Example Key)"
```

RESULT:

Thus, the various flags have been captured in Encryption Crypto 101 in TryHackMe platform.