Week 54 - Securing Network Data

20th October 2020 at 11:51am

Contents

We are learning today:

- how computers exchange data
- how the data is secured and checked

By the end of the lesson you should be able to:

- state hardware that devices use to communicate with networks
- describe how data is sent to the right device
- explain how data is protected and checked

Device addresses

To send anything anywhere you need to know

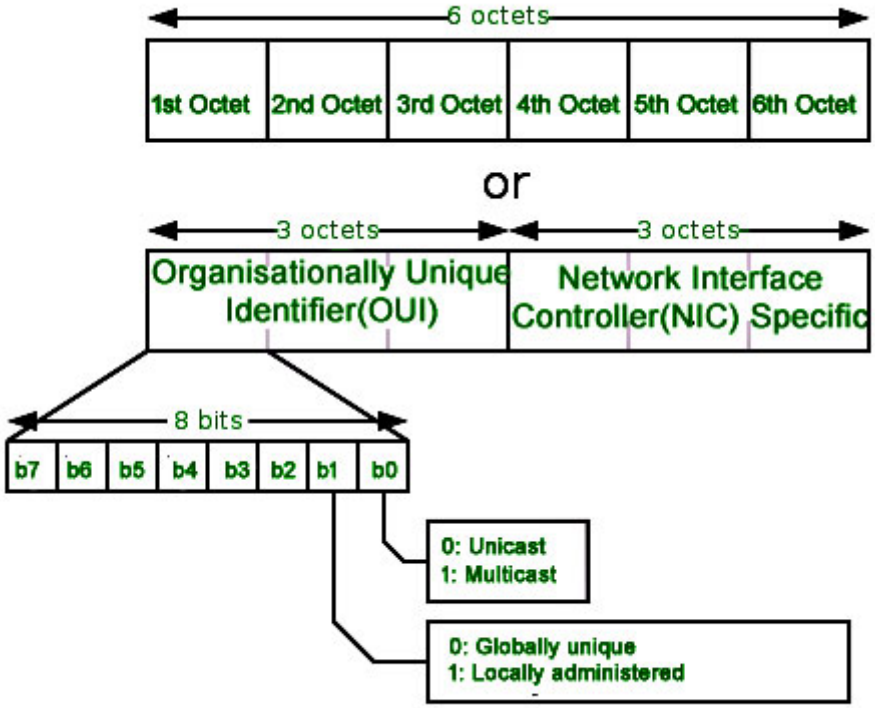
- what you are sending
- where it is going

Devices that exchange data have addresses on networks so they can be found and so data can be sent to them.

Data is sent from one computer to another in 'data packets'.

Data packets include the addresses of the devices they are going to and coming from. Computers need a network interface card to connect to a network. All devices on a network have a MAC address.

MAC address



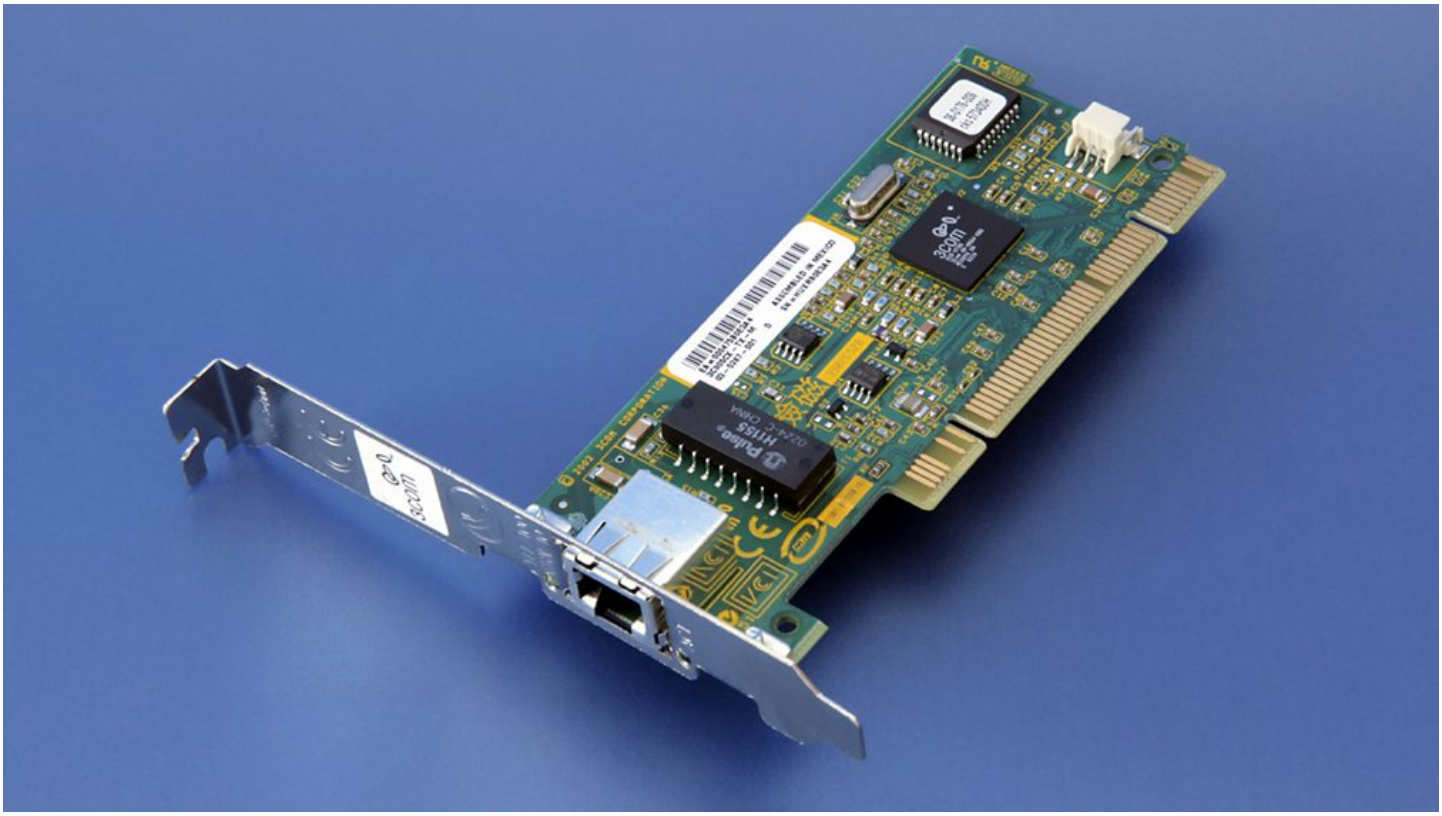
Every piece of hardware on a network has a unique MAC address. This is embedded in the hardware when the product is made in the factory, and the user cannot change it. On a computer, the MAC address is a unique code built into a NIC. No two computers have the same MAC address. A MAC address is made up of 48 bits of data, usually written as 12 hexadecimal characters.

The first half is assigned to the manufacturer and the second part is unique to the device. Some examples of manufacturer codes are:

```
CC:46:D6 - Cisco
3C:5A:B4 - Google, Inc.
3C:D9:2B - Hewlett Packard
00:9A:CD - HUAWEI TECHNOLOGIES CO.,LTD
```

> **Find** the manufacturer OUI MAC addresses for Apple, Samsung, Lenovo and Dell.
>
> 24 bits of data are used to identify the device. 2 bits would identify four devices, 3 bits would identify eight.
>
> **Calculate** how many devices can be identified with 24 bits.

Network interface card (NIC)



NICs enable desktop and laptop computers to connect to a network. NICs are small circuit boards that connect to the motherboard. Smartphones also use a GSM chip to connect to the telephone network. Games consoles contain a NIC card so users can access the internet, download games and play online.

> Go online and **find** the prices of NIC's for sale as separate items

Encryption

Encryption is the process of disguising a message so that it cannot be understood by anyone but its intended recipient. Encryption requires the use of a key. The key is secret as to how the message has been disguised.

> ***Watch*** the video https://www.bbc.co.uk/bitesize/guides/zr3yb82/revision/4

Imagine a classmate wants to send a secret message to a friend.

To make sure the people in between can't read the message, the sender could shift the the letters in the alphabet.

```
A>d, B>e, C>f, etc.
```

So long as the receiver knows the shift, they can read the message. This is a simple Caesar cipher and is quite easy to break.

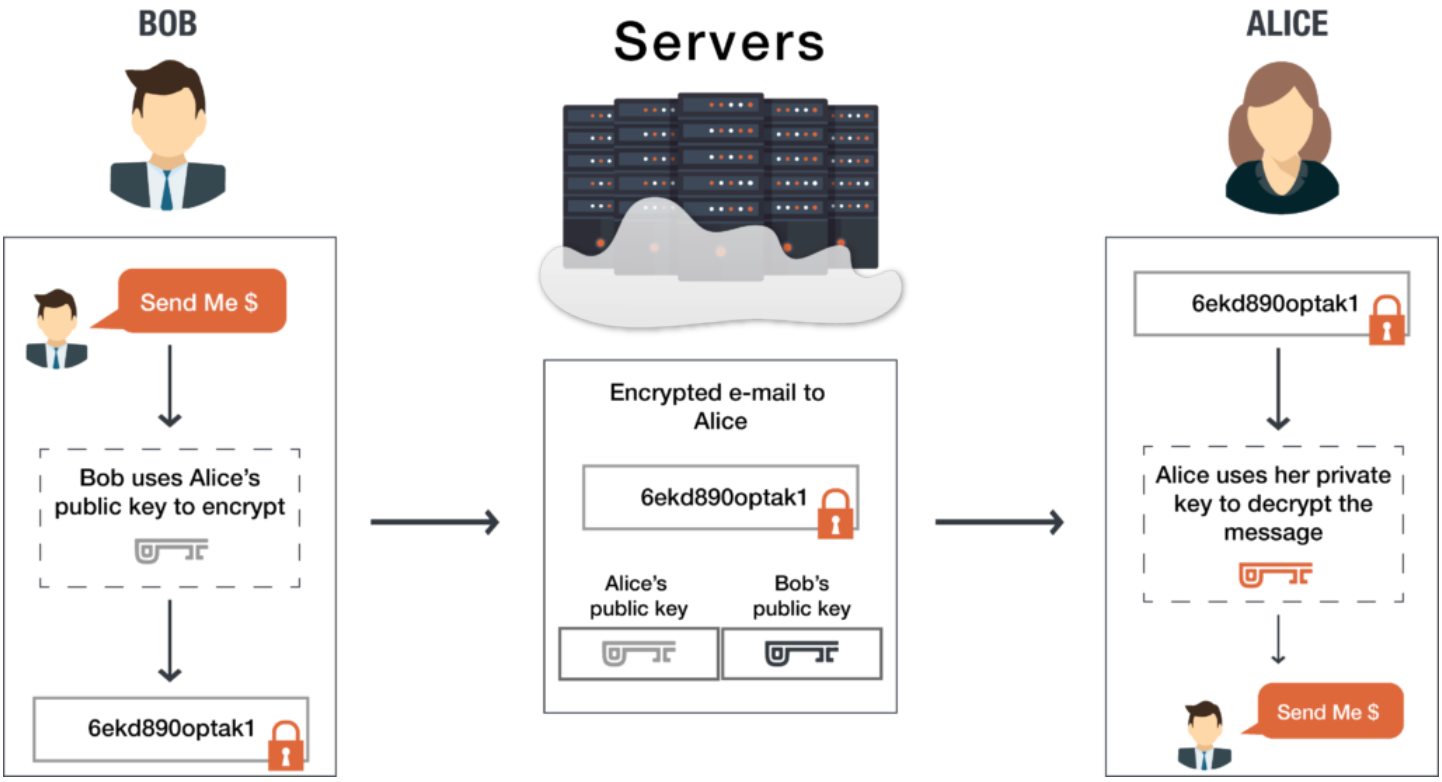
> You can try it here https://cryptii.com/pipes/caesar-cipher
>
> **Send** a message via email to a friend encrypted with a Caesar cipher and see how long it takes them to decode it.

This is an example of **symmetric encryption.** *One key changes the plain-text into cipher-text and visa versa*. It doesn't matter how sophisticated the key is, once someone has it, any message can be decoded.

Public Key Encryption

Public-key cryptography (also known asymmetric cryptography) has a solution for this.

Each person creates two keys—a public key and a private key. The two keys are connected. If you encode a message using a person's public key, they can decode it using their matching private key.



Both keys are needed to read the message.

> This is cipher-text
>
> ```
> aabbb aabaa ababa ababa abbab babaa abbab baaaa ababa aaabb
> ```
>
> It was encoded twice at https://cryptii.com/
>
> First, it was encoded with the Enigma machine cipher (default settings) Then, it was encoded with the Bacon cipher (default settings)
>
> Can you **decode** it?

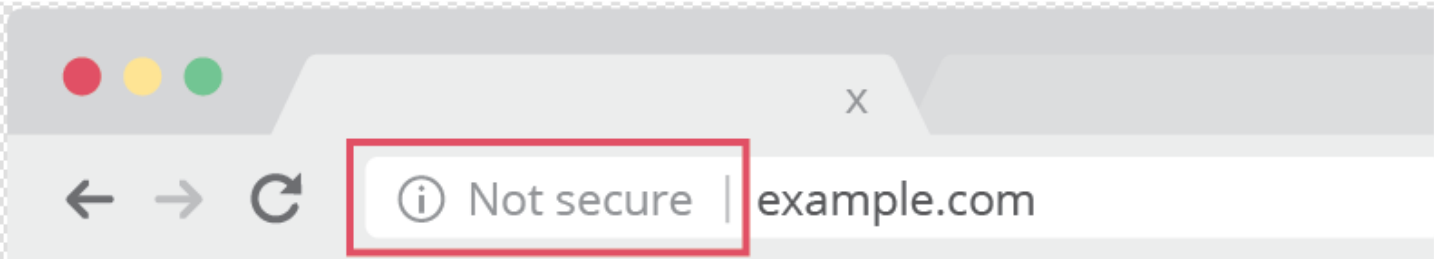
To decode this in the real world you would need to know both keys and the settings.

Internet encryption

Hypertext transfer protocol secure (HTTPS) is the secure version of HTTP, which is the primary protocol used to send data between a web browser and a website. HTTPS is encrypted in order to

increase security of data transfer. This is particularly important when users transmit sensitive data, such as by logging into a bank account, email service, or health insurance provider.

Any website, especially those that require login credentials, should use HTTPS. In modern web browsers such as Chrome, websites that do not use HTTPS are marked differently than those that are. Look for a green padlock in the URL bar to signify the webpage is secure. Web browsers take HTTPS seriously; Google Chrome and other browsers flag all non-HTTPS websites as not secure.



How does HTTPS work?

HTTPS uses an encryption protocol to encrypt communications. The protocol is called Transport Layer Security (TLS), although formerly it was known as Secure Sockets Layer (SSL). This protocol secures communications by using what's known as an asymmetric public key infrastructure. This type of security system uses two different keys to encrypt communications between two parties:

- The private key - this key is controlled by the owner of a website and it's kept, as the reader may have speculated, private. This key lives on a web server and is used to decrypt information encrypted by the public key.
- The public key - this key is available to everyone who wants to interact with the server in a way that's secure. Information that's encrypted by the public key can only be decrypted by the private key.

> **Explain** the steps involved in a *TLS handshake* over https

Caesar.py

> Copy the code into IDLE and get the code to work.
>
> Look up these functions and **explain** what they do
>
> > isalpha()
> >
> > isupper()
> >
> > split()
> >
> > chr()
> >
> > ord()
>
> Revise how WHILE loops work
> https://www.w3schools.com/python/python_while_loops.asp

```
MAX_KEY_SIZE = 26
```

```
def getMode():
    while True:
        print('Do you wish to encrypt or decrypt a message?')
        mode = input().lower()
        if mode in 'encrypt e decrypt d'.split():
            return mode
        else:
            print('Enter either "encrypt" or "e" or "decrypt" or "d".')


def getMessage():
    print('Enter your message:')
    return input()


def getKey():
    key = 0
    while True:
        print('Enter the key number (1-%s)' % (MAX_KEY_SIZE))
        key = int(input())
        if (key >= 1 and key <= MAX_KEY_SIZE):
            return key


def getTranslatedMessage(mode, message, key):
    if mode[0] == 'd':
        key = -key
    translated = ''
```

```
    for symbol in message:
        if symbol.isalpha():
            num = ord(symbol)
            num += key

            if symbol.isupper():
                if num > ord('Z'):
                    num -= 26
                elif num < ord('A'):
                    num += 26
            elif symbol.islower():
                if num > ord('z'):
                    num -= 26
                elif num < ord('a'):
                    num += 26

            translated += chr(num)
        else:
            translated += symbol
    return translated

mode = getMode()
message = getMessage()
key = getKey()

print('Your translated text is:')
print(getTranslatedMessage(mode, message, key))
```