Task 3.1

a)

1. Biometric authentication[1]

The biometric authentication is based on the human unique biological characteristics. The most popular might be fingerprint scanner, voice recognition and iris scanner; still there are many more unique characteristics. E.g. a fingerprint scanner is found nearly on every device nowadays, like smartphones or notebooks. It is widely known that the fingerprint of every human is unique. So the fingerprint scanner creates a digital likeness of the fingerprint and authenticates the human by mapping it to the registered fingerprint image.

+ Fast authentication, only have to use one finger

+ No need to remember a password. Biometric characteristics are always accessible

- Biometric characteristics can be lost for ever (loss of hand…)

- New attack target (from information based up to personnel based)

Popular product: Apple IPhone (5s and newer)

2. Authentication through possession in addition with biometric authentication[2]

The Nymi Band also uses biometrical characteristics like the unique heartbeat rhythm of a person. It is a wearable wristband that has to be worn to use its authentication features. For the authentication the wristband measures the chosen biometrical characteristic e.g. the heartbeat rhythm of its wearer and matches it with the preregistered rhythm to authenticate the person wearing the wristband.

+ Fast and easy authentication (no input necessary)

+ Pretty safe, only the preregistered wearer can authenticate

- Wristband must always be worn and can get lost

Popular service: Nymi Band

3. TAN authentication

TAN is the transaction authentication number and is usually used in online banking. Therefore e.g. the user registered a phone number to the online bank and at the time of payment, the service sends a message to the user's phone with the TAN. This TAN needs to be entered to authorize the payment. Mostly it is a 6 character long number.

+ No possession in form of a password is needed and has to be remembered

[1] Global source for this segment: "http://searchsecurity.techtarget.com/definition/biometric-authentication"
[2] Global source for this segment: "https://nymi.com/product_overview"

- The transmission path can be intercepted

Popular service: Kreissparkasse Köln

b)[3]

Googles offers an account confirmation in 2 steps for their google accounts to users. The first step is the normal password authentication. The only restriction on the password is that it has to be at least 8 characters long. The second step provides an authentication mostly over the user's telephone. Therefore a 6 long digit code is send to the telephone via SMS, phone call or a specially designated app. A login is only possible, if both steps are performed. Alternatively the second step can also performed offline, by downloading a list of backup-codes which can be used in case the telephone is not accessible. Another option is to use a Security Key with the open standard "FIDO Universal 2nd Factor (U2F)". Usually this security key comes within an USB HID and can be added to a user's google account. To authenticate it is only necessary to pluck in the security key.

Advantages:

- Even if the password of step 1 is thieved, your account can be still save through the second authentication step
- The security of your account doesn't depends as much on the password strength, thus weakly chosen passwords don't have a high severity
- Attackers have to gain access to a physical object (telephone, security key) to pierce through the second authentication phase

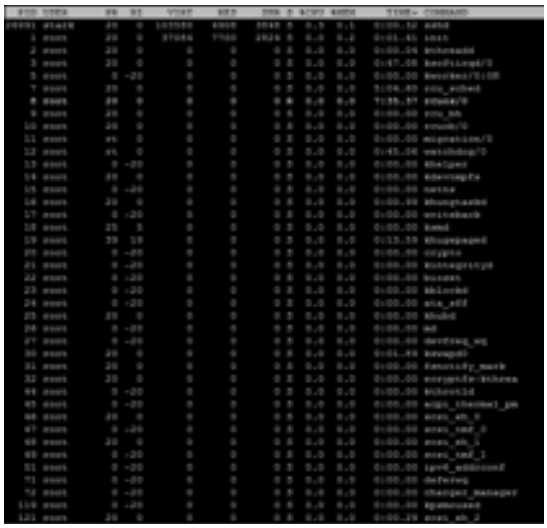
Disadvantages:

- For the second authentication phase, only a 6 long digit code is send to your device. It wouldn't take long to crack this code through a brute force attack
- Attackers can also gain access to physical objects like the registered security key, or the telephone. Therefore the attack isn't based any longer on a technical, thus a physical level
- The account authentication takes more time
- More circumstances when losing the telephone

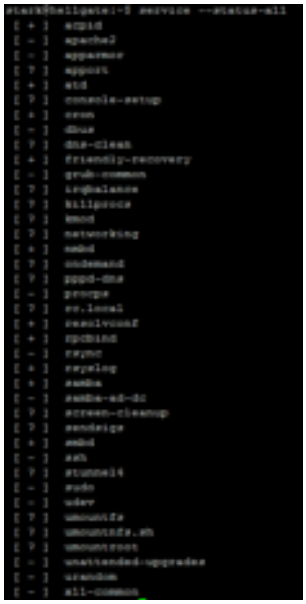
[3] Global source for this segment: "https://www.google.com/landing/2step/#tab=why-you-need-it"

Task 3.2

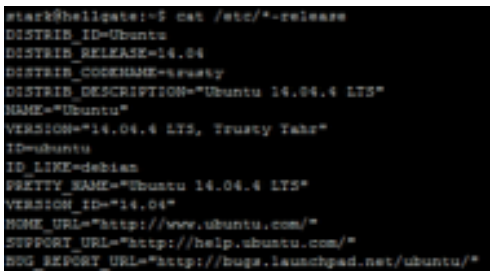
1) Show the process activity in real time with “top” or “ps –A” for all processes



2) Show all running or stopped services with “service --status-all”



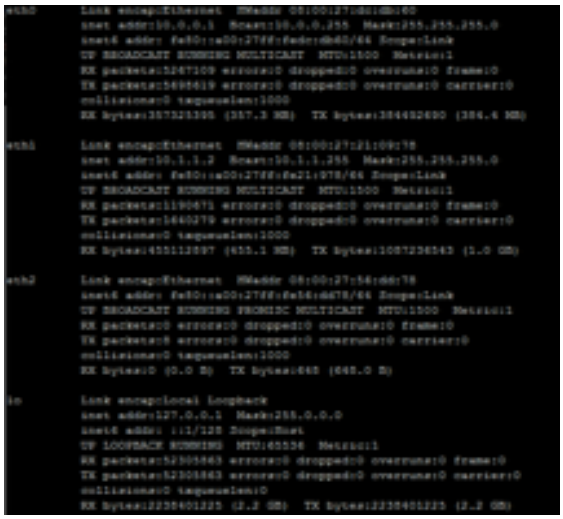
3) Finding out the operating system with “cat /etc/*-release



4) Finding out the openSSL version with “dpkg -l openssl”

```
stark@hellgate:~$ dpkg -l openssl
Desired=Unknown/Install/Remove/Purge/Hold
| Status=Not/Inst/Conf-files/Unpacked/half-conf/Half-inst/trig-aWait/Trig-pend
|/ Err?=(none)/Reinst-required (Status,Err: uppercase=bad)
||/ Name           Version       Architecture  Description
+++-==============-=============-============-=================================
ii  openssl        1.0.1f-1ubunt amd64         Secure Sockets Layer toolkit - crypto
```

5) Finding out the ip addresses with “/sbin/ifconfig”



6) Show all TCP ports with nmap using “nmap -sT 10.0.0.1”

```
stark@hellgate:~$ nmap -sT 10.0.0.1

Starting Nmap 7.12 ( https://nmap.org ) at 2016-05-24 15:24 CEST
Nmap scan report for 10.0.0.1
Host is up (0.000077s latency).
Not shown: 996 closed ports
PORT    STATE SERVICE
22/tcp  open  ssh
111/tcp open  rpcbind
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds

Nmap done: 1 IP address (1 host up) scanned in 16.57 seconds
```

Task 3.3

Important DNS Header fields:

ID : The identifier is necessary to math a reply to a specific request. Therefore this field is important to fake an answer for a specific request.

QR: This is a one bit field which makes a message a request (0) or a response(1). To spoof a response this field is important to be set at 1.

RCODE: This field needs to be set to 0 to communicate that there was no error at the name server while handling the request.

Important DNS Answer fields:

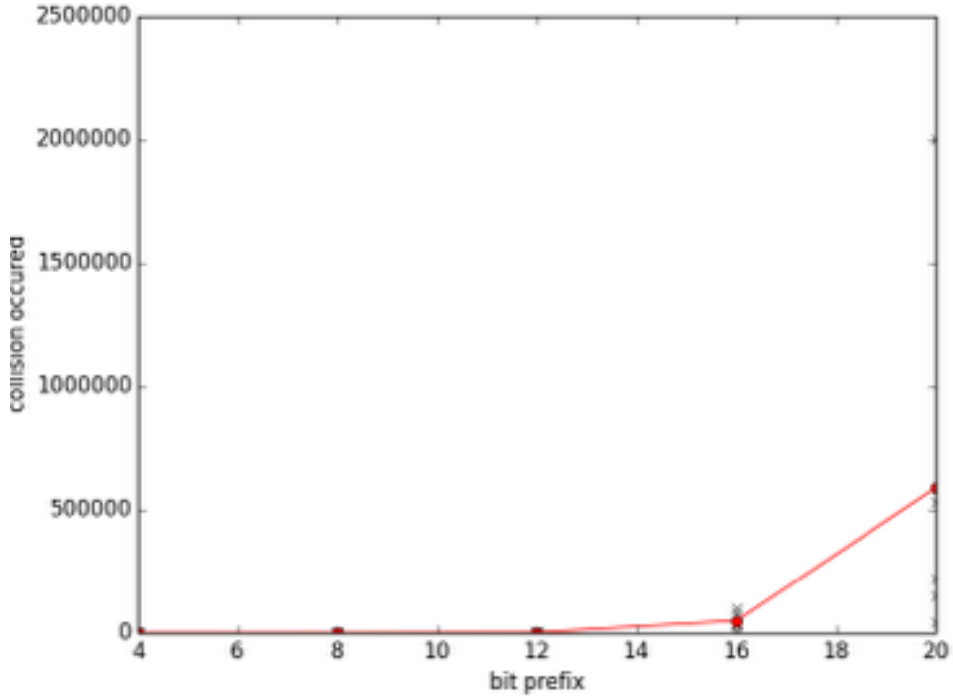
In my opinion, all fields of the DNS Answer are important to spoof a answer as this should be as realistic as possible and should be sure to be recognized as a real answer.

A special focus should be on the NAME field, which should be identically to the NAME of the query, and the RDATA which is the actual response.

The source code is located in the packetSniffer director.
The recorded output was piped to the snifferOutput.txt in the Task3_3 directory. Furthermore we recorded a pcap file to differ between Queries and Replies as well as see the actual Requests and Responses.

Task 3.4

The source code is located in the Task3_4 directory as well as the generated data in form of a txt file.
The plotting script (which is a part of the colliders source code) had the following output as a result:



Task 3.5

Part a)

Is this method secure?

Yes it is secure. At any time of delivery the box is locked by one or two padlocks. Since Alice and Bob are the only owner of the keys to unlock the box, it can't be open by the transmitter.

Does it also work with cryptographic means? Which problems could arise?

In my opinion it could be problematic, that the message is encrypted two times. The problem could arise when Bob tries to remove his encryption, after he got the message back from Alice. Since Alice didn't decrypt the message and only encrypts the already encrypted text, it could be hard for bob to remove his encryption.

Part b)

Does it work?

Yes, relating to the principles of the XOR operator, the mentioned encryption works. Since XOR is associative, the order of performing the XOR doesn't matter.

$$(key1 \oplus Text) \oplus key2 = key1 \oplus (Text \oplus key2)$$

Additionally XOR of the same element is always zero ($A \oplus A = 0$) and XOR with zero has no impact ($A \oplus 0 = A$). So Bob can easily remove his key, after obtaining the message back, from the encrypted text:

$$key1 \oplus \big(key1 \oplus (Text \oplus key2)\big) = 0 \oplus (Text \oplus key2) = Text \oplus key2$$

Can confidentiality and integrity be assured?

Confidentiality can be assured, because on every transmission the text is encrypted with either one or more key. No unauthorized access to the plain text is possible. (Of course only if the keys are not known for the attacker)

Integrity cannot be assured. Since the attacker can always intercept the message and encrypt it additionally by using XOR with his own key. It would be possible to obtain the third encryption key by using the already addressed XOR operations, but for this the two original keys plus the plain text have to be available to one member of the communication. But this is not the case at any time of the transmission.

Would choosing different random keys for each message have an impact?

No, it is regardless whether the keys are random on every new message, as long as the same key is used for en-and decryption.