Hello everyone, we are going to talk about our project related with the The Cork scenario.

So first we will show our infrastructure. As you can see in this picture, it is composed by two external machines being them the Client and the Bank. Then we have a Firewall that is responsible for filtering and forwarding the traffic and talking about the internal network we have the The Cork webserver and finally a database.

Based on this infrastructure, we developed two secure channels that are the Bank – WebServer channel and the Client – WebServer channel. We also assumed that the internal network is fully trusted.

Related with the keys and their distribution, we can tell that the WebServer has a private/public key pair being the public key known by the bank.

From the three possible security challenges proposed we chose the following one: TheCork’s users keep their credit card data information stored in the app in order to facilitate reservations at high-end restaurants that need you to leave a deposit. This data needs to be safe and confidential at the device level and on-the-wire when it gets sent to TheCork’s servers.

Talking now about our security protocol, we chose to protect the Client – WebServer channel by implementing the HTTPS protocol once it is one of the most known security protocols to confer privacy and security. and to the Credit Card company.

We also generated a self-signed certificate which was validated by a fictious CA.

The connection between Bank and WebServer is not secure so we designed a protocol inspired in TLS to protect it. Our protocol follows the steps on the next slides.

So, first the Bank generates a pre master secret that will be send to the WebServer in a packet when establishing connection with it. The packet information is a Json Object encrypted with WebServer's public key using RSA algorithm. After encrypted this packet, it will be sent to WebServer by a socket.

Once the WebServer receives the packet, it will decrypt it with its private key and RSA Algorithm to has access the pre master secret. Then both Bank and WebServer will generate a session key by first hashing the pre master secret with the function SHA3-256 and then using the AES algorithm.

From now on, the information sent between Bank and WebServer will be encrypted with the session key. This ensures confidentiality.

To ensure freshness, when the connection is established, the WebServer generates a token that is a random integer using the function Math.random(). This token will be present in the packets changed between Bank and WebServer to guarantee the freshness of the message.

When a message is received, the receiver checks if the token is incremented by one in relation to the last token sent by the receiver.

From now on the payload from WebServer to Bank contains the following information: Token, Credit Card number encrypted with the session key, 3digitCode encrypted with the session key and the validity date also encrypted with the session key.

To ensure integrity, we hash the payload with the SHA3-256 function to obtain the hmac that will be encrypted with the session key and sent alongside the payload. When the message arrives to the other side of the communication, the receiver must decrypt the hmac and hash the payload.

If the two values match, the integrity is ensured in this message.

Now we are going to talk about the communication between the WebServer and Database. In this communication we prevent SQL injection using he function PreparedStatement().

Also, when sending the information from the WebServer to the Database we encrypt the sensible information from our security challenge with the session key and AES algorithm in order to it to be protected in the Database.

Finally, we will talk about some vulnerabilities in our protocol. The site is vulnerable to interception attacks due to our protocol implementation between Bank and WebServer. Vulnerable to impersonation attacks once the information sent by the bank is not encrypted with bank's private key and finally vulnerable to DoS attacks due to the slowness of the Virtual Machines.

Now we will say some things we can improve in our protocol!

In the communication between WebServer to Bank, we encrypt the sensible information with the session key. In an ideal scenario, we would encrypt this information with a secret key and alongside the information we needed to send this secret key encrypted with the bank's public key.

This secret key would change each time every time we send a packet.

Now we are going to do our live demo. We are trying to do it on Hugo’s computer however the Virtual Machines are really slow so we have already recorded a video by cloning the VMs in case we need it.