Integrated MSc Course on Informatics Engineering, DI/FCT/UNL
Computer Networks and Systems Security / Semester 1, 2019-2020
WORK-ASSIGNMENT #2 REPORT for Evaluation

Secure Blockchain-Enabled Auction Management System

REPORT

Authors:
Eduardo Brás Silva (emf.silva@campus.fct.unl.pt),
Rúben André Barreiro (r.barreiro@campus.fct.unl.pt)

Summary

With the popularization and emerging growth of ***Blockchain Systems***, such as ***Bitcoin***, ***Ethereum*** and many others, it is becoming increasingly common for *online services and platforms to adopt such technologies to conduct financial or banking transactions*. The use of this technology guarantees the *validation of all data present in a system since the very beginning of its runtime*, making use of the principles of *consensus* and *majorities* to decide if *given data is valid or not*. Making use of these properties, it is possible to create a system that handles **Auctions** and **Bids** in a way makes it possible to *verify each step the system took* to decide who has the winning **Bid** so that *no kind of foul play may happen*. Complementing this system with ways to guarantee that compromising information doesn't leak and that **Bids** *actually come from the people that really make them*, as well as the response from the system, is also of great importance and may be done using ***T.L.S. Communication Channels*** and **Asymmetric Cryptography**. Thus, with such tools in hand, the creation of a **Secure Blockchain-Enabled Auction Management System** is possible.

1. Introduction

A **Secure Blockchain-Enabled Auction Management System** *is a system that handles* ***Auctions*** *and* ***Bids*****,** as also *guarantees the success of the whole process of an* ***Auction***. For achieving a secure system, the *exchanged messages* between remote ***Clients*** and ***Server*** *must be protected to ensure that no malicious third party influences the interactions of the system, gets access to confidential information and other problems that may arise from* ***third party attacks***.

Two different systems are used to protect this service.

The use of **T.L.S. communication channels** enables the creation of ***<u>secure communication channels</u>*** that allows for ***<u>endpoint authentication</u>***, ***<u>data confidentiality</u>*** and ***<u>reliable connections</u>***. With both ***Clients*** and ***Server*** having ***Keystores***, ***Truststores*** and ***X.509 Certificates***, the endpoints are able to authenticate themselves to prove that they are who they say they are. And with the support of various **T.L.S. (Transport Layer Security)** parameters' configurations, such as *<u>supported protocols and ciphersuites</u>*, there's the ability to set the security as strong or lax as we deem required.

For the implementation of the **T.L.S. communication channels**, we will make use of **SSLSocket** and **SSLServerSocket** from the ***javax.net.ssl*** package.

The usage of **blockchains** enables us to create a ***Chain of Records*** of what happened in the system since its origin and enables us to verify how everything went in the system. In the case of the **Secure Blockchain-Enabled Auction Management System**, the idea is to maintain a **Chain of Blocks** containing **Bids,** so that ***Clients*** that do participate in ***Auctions*** by making **Bids** to the system can have a guarantee that there was no cheating involved in the process of a regular **Auction**. Implementation-wise, each **Auction** is to have its own **blockchain** and each **Client** will do **Proofs of Work** to be validated by the server, broadcast to the **Clients** the work done and add the new created blocks from the **Proof of Work** to the **blockchain**. The exchanged message, carrying these proofs and bids will be protected by using the **T.L.S. communication channel** described above and also by using **Asymmetric Cryptography** to *<u>cipher the data on "envelopes" and to sign them</u>*, to provide **Confidentiality, Integrity and Authentication**, in order to establish **Secure Sessions,** to the *<u>endpoints/parties</u>* involved, be able to use **Symmetric Cryptography** with **Secret keys**, from then.

2. System model and architecture

2.1. System model

The system model will be represented by 3 main ***principals/components***:

- ***Auction Server***:
  - The component responsible to attend ***Clients*** and provide services for them, as well as, communicating with the ***Auction Server Repository***, to keep all the data related to the system;

- ***Auction Repository Server:***
  - The component responsible to keep all the data related to the system;

- ***Client:***
  - The component which represents the final users, who use the system, requesting and performing services operations;

Each ***Client*** can perform the following operations:

- Create Auctions, from several types:
  1. ***Normal Auction*** with *no restrictions* for the ***Bids***, which will be performed;
  2. ***Auction*** with a *minimum initial value for the first **Bid** made*;
  3. ***Auction*** with a *minimum value for the next **Bid** made*, i.e., *a minimum higher value/threshold for the next **Bid**, related to the last one made*;

4. ***Auction*** with a *maximum value for the next **Bid** made*, i.e., *a maximum higher value/threshold for the next **Bid**, related to the last one made*;
5. ***Auction*** with a *minimum and maximum values for the next **Bid** made*, i.e., *a minimum and maximum higher value/threshold for the next **Bid**, related to the last one made, forming a range of possible values for the next **Bids***;
6. ***Auction*** with a *limited set of **Client Bidders***, i.e., *only the **Client Bidders** present in that set, are allowed to perform **Bids***;

7. ***Auction*** with a *limited number of **Bids** for each **Client*** present in some set of ***Client Bidders*** predefined previously, i.e., *only the **Clients** in that set, are allowed to perform **Bids***, since *the number of **Bids** performed by that **Client** doesn't surpass the number predefined to it*;
8. ***Auction*** with a *limited number of **Bids***, i.e., *the **Client Bidders** are only allowed to perform **Bids**, since the total number of **Bids** performed until that moment, don't had surpassed that limit number of **Bids** allowed*;
9. ***Auction*** with a *limited time to perform **Bids***, i.e., *the **Client Bidders** are only allowed to perform **Bids**, since the time counted since the **Auction** was created don't surpassed that limit amount allowed for that **Auction***;

- List the ***Auctions***, from the several types:
    1. All the ***Auctions***, present in the ***Auction System***, i.e., both ***Opened*** and ***Closed Auctions***;
    2. ***Opened Auctions***, present in the ***Auction System***, i.e., which are *currently occurring* and allowing the ***Clients***, to perform ***Bids***;
    3. ***Closed Auctions***, present in the ***Auction System***, i.e., which are *no more occurring*, neither, allowing the ***Client Bidders***, to perform ***Bids***;

4. All the ***Auctions*** created by a certain ***Client Owner***, given its <u>*identification,*</u> i.e., both ***Opened*** and ***Closed Auctions***, <u>*which were created by that **Client Owner***</u>;
5. ***Opened Auctions*** created by a certain ***Client Owner***, given its <u>*identification*</u>, i.e., <u>*which were created by that **Client Owner***</u>;
6. ***Closed Auctions*** created by a certain ***Client Owner***, given its <u>*identification,*</u> i.e., <u>*which were created by that **Client Owner***</u>;

7. All the ***Auctions*** with a certain ***ID***, i.e., both ***Opened*** and ***Closed Auctions***, <u>*which were created with a certain **identification***</u>;
8. ***Opened Auctions*** with a certain ***ID***, i.e., <u>*which were created with a certain **identification***</u>;
9. ***Closed Auctions*** with a certain ***ID***, i.e., <u>*which were created with a certain **identification***</u>;

- Close an ***Auction***, given its <u>*identification*</u>, since it's present in the ***Auction System*** and it's <u>*currently in an **Opened Status***</u>;

- Create ***Bids***, given the <u>*identification*</u> of an ***Auction*** and the pretended <u>*value amount*</u> for the respective ***Bid***;

- List the ***Bids***, performed previously, in the ***Auction System***, through several manners:
    1. ***Bids*** of ***All Auctions***, given its *identification*, since a ***Bid*** with that *identification* was performed previously, and it's contained in ***All Auctions*** in the ***Auction System***, i.e., both ***Opened*** or ***Closed Auctions***;
    2. ***Bids*** of ***Opened Auctions***, given its *identification*, since a ***Bid*** with that *identification* was performed previously, and it's contained in ***Opened Auctions***;
    3. ***Bids*** of ***Opened Auctions***, given its *identification*, since a ***Bid*** with that *identification* was performed previously, and it's contained in ***Closed Auctions***;

    4. The set of ***Bids*** made in an ***Auction***, given the *identification* of the ***Auction***, present in ***All Auctions***, since an ***Auction*** with that *identification* was created previously, and it's contained in ***All Auctions*** in the ***Auction System***, i.e., both ***Opened*** or ***Closed Auctions***;
    5. The set of ***Bids*** made in an ***Auction***, given the *identification* of the ***Auction***, present in ***Opened Auctions***, since an ***Auction*** with that *identification* was created previously, and it's contained in ***Opened Auctions***;
    6. The set of ***Bids*** made in an ***Auction***, given the *identification* of the ***Auction***, present in ***Closed Auctions***, since an ***Auction*** with that *identification* was created previously, and it's contained in ***Closed Auctions***;

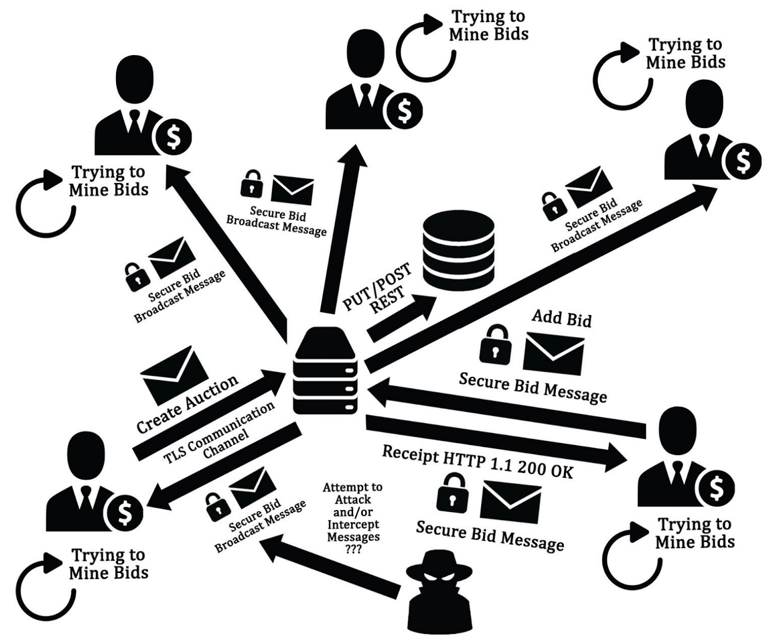
7. The set of ***Bids*** made in an ***Auction*** by a ***Client Bidder***, given the *identification* of both ***Auction*** and ***Client Bidder***, present in ***All Auctions***, since an ***Auction*** with that *identification* was created previously, and it's contained in ***All Auctions*** in the ***Auction System***, i.e., both ***Opened*** or ***Closed Auctions***, as also, that ***Auction*** *contains at least one* ***Bid*** from the ***Client Bidder*** with the given ***ID***;
8. The set of ***Bids*** made in an ***Auction*** by a ***Client Bidder***, given the *identification* of both ***Auction*** and ***Client Bidder***, present in ***Opened Auctions***, since an ***Auction*** with that *identification* was created previously, and it's contained in ***Opened Auctions***, as also, that ***Opened Auction*** *contains at least one* ***Bid*** from the ***Client Bidder*** with the given ***ID***;
9. The set of ***Bids*** made in an ***Auction*** by a **Client Bidder**, given the *identification* of both ***Auction*** and ***Client Bidder***, present in ***Closed Auctions***, since an ***Auction*** with that *identification* was created previously, and it's contained in ***Closed Auctions***, as also, that ***Closed Auction*** *contains at least one* ***Bid*** from the ***Client Bidder*** with the given ***ID***;

10. The set of ***All Bids*** made by a certain ***Client Bidder***, given its *identification*, i.e., both *mined and not mined* ***Bids***, made by the ***Client Bidder*** with the given **ID**;
11. The set of the ***Opened Bids*** made by a certain ***Client Bidder***, given its *identification*, i.e., the *not mined* ***Bids, made by the Client Bidder*** with the given ***ID***;
12. The set of the ***Closed Bids*** made by a certain ***Client Bidder***, given its *identification*, i.e., the *mined* ***Bids, made by the Client Bidder*** with the given ***ID***;

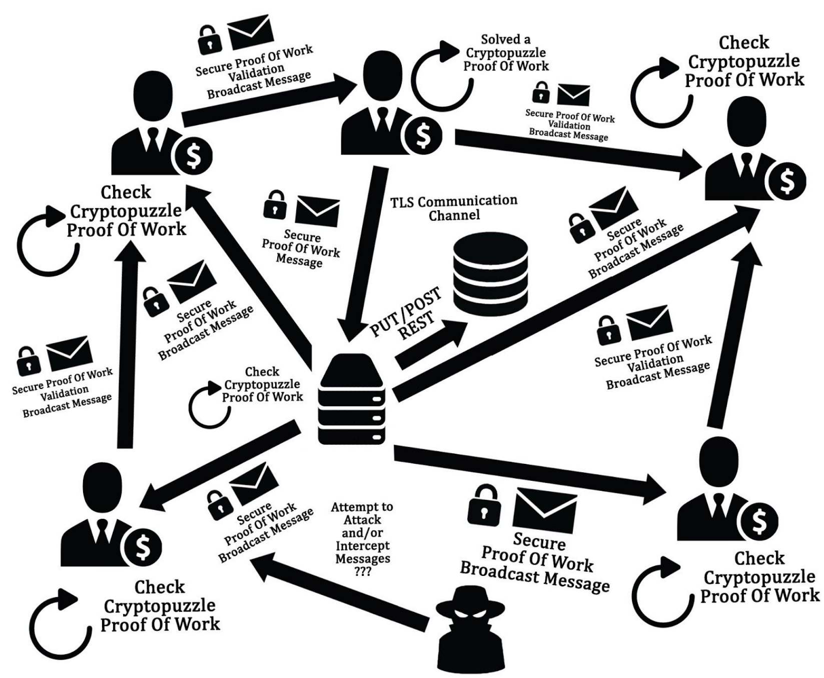
- Check the outcome of the ***Bids*** performed, on an ***Auction***, where a ***Client Bidder*** participated:
    13. Check the outcome of ***All Auctions*** where a certain ***Client Bidder***, made ***Bids*** and participated;
    14. Check the outcome of ***Opened Auctions*** where a certain ***Client Bidder***, made ***Bids*** and participated;
    15. Check the outcome of ***Closed Auctions*** where a certain ***Client Bidder***, made ***Bids*** and participated;

2.2 Architecture

All the ***Security Configurations*** were designed on ***Secure and Authenticated Communication Channels*** provided by ***T.L.S. (Transport Layer Security) Protocol Configurations***. The system supports ***Server-Side-Authentication*** and ***Mutual-Authentication*** modes making it strictly necessary that the ***Server*** *authenticate itself to* ***Clients*** or, both the ***Server*** and ***Client*** *authenticate themselves to each other* if they want to interact with one another, i.e., the ***T.L.S. Authentication*** was confirmed, by the exchanged ***Certificates in X.509*** format, during the initial setup of ***Secure, Encrypted and Authenticated Communication Channels*** (taking advantage of ***Tunneling*** techniques, for example), in a process called ***Handshake***, *confirmed and audited*, by using the ***Wireshark*** and ***testssl.sh*** frameworks/tools, to inspect the ***Datagrams/Messages*** exchanged.



Since, it's considered that an attacker it's allowed to perform attacks covered by the ***Security Threats/Mechanisms of the X.800 Framework***, major part of the messages exchanged must be protected, in order to guarantee the integrity of the same, as also, their both origin ***authenticity*** and ***peer-authenticity***.



For that reason, <u>*it was established some messages exchanged between the several principals of the system*</u>, which, unfortanley, some of them weren't implemented, due to given time to the realization of the project:

- ***Secure Bid Message:***
    - This message is sent by a ***Client*** to the ***Auction Server***, when the ***Client Bidder*** makes a ***Bid*** to a currently occurring ***Auction***, with the necessary security parameterizations, in order to, guarantees:
        - ***Confidentiality*** of the "envelopes" containing a ***Pair of Session Keys (K_{SYM}, K_{MAC})***, ciphered with the ***Public Key obtained from the Certificate X.509 of the Auction Server***, in a way that only the ***Auction Server***, which have its ***Private Key***, should be the only one to open that "envelope" to guarantee the ***DoS Mitigation***, ***Confidentiality***, ***Integrity*** and ***Origin-Data Authentication of the Payload Data*** of this message;
        - The message also contains a ***Digital Signature,*** signed by with the ***Private Key*** of the ***Client Bidder***, in order to guarantee the ***Origin-Peer Authentication*** and, in a way that can be verified by everyone who can obtain its ***Public Key from its X.509 Certificate***;
    - This message it was <u>***implemented***</u> and <u>***integrated***</u>;

- ***Secure Receipt Message:***
    - This message is sent by the ***Auction Server*** to a ***Client Bidder***, after the same made a ***Bid***, in order to the ***Client Bidder*** have guarantees that the ***Auction Server*** validate and accepted the ***Bid*** made.
    - The message was defined with the necessary security parameterizations, in order to, guarantees:
        - ***Confidentiality*** of the "envelopes" containing a ***Pair of Session Keys (K_{SYM}, K_{MAC})***, ciphered with the ***Public Key obtained from the***

        Certificate X.509 of the Client, in a way that only the ***Client***, which have its ***Private Key***, should be the only one to open that "envelope" to guarantee the ***DoS Mitigation***, ***Confidentiality***, ***Integrity*** and ***Origin-Data Authentication of the Payload Data*** of this message;
        - The message also contains a ***Digital Signature,*** signed by with the ***Private Key*** of the ***Auction Server***, in order to guarantee the ***Origin-Peer Authentication*** and, in a way that can be verified by everyone who can obtain its ***Public Key from its X.509 Certificate***;

Secure Bid Broadcast Message:
  - This message is sent by the ***Auction Server*** to all the ***Clients***, in order to, all the other ***Clients*** have locally, all the ***Bids***, to be able to select some of them, to try to mine, and close ***Blocks of Bids***, through ***Proof Of Works***;
  - This message it was ***implemented*** and ***integrated***, but ***not built under Security assumptions***;

- ***Secure Proof Of Work Message:***
  - This message is sent by a ***Client,*** *who solves a challenge, by mining and closing a* ***Block of Bids****, through a* ***Proof Of Work****, and close a* ***Block of Bids***. This message is sent to the ***Auction Server***, in order for the ***Auction Server*** to *validate* it and then, *"broadcast" the same message to all the other* ***Clients*** *to validate the* ***Proof Of Work***, in order for the ***Proof Of Work*** to be considered *valid*, by at least, the *majority of the principals of the* ***Auction System***;
  - The message was defined with the necessary security parameterizations, in order to, guarantees:
    - ***Confidentiality*** of the "envelopes" containing a ***Pair of Session Keys*** (K_{SYM}, K_{MAC}), ciphered with the ***Public Key obtained from the Certificate X.509 of the Auction Server***, in a way that only the

Auction Server, which have its ***Private Key***, should be the only one to open that "envelope" to guarantee the ***DoS Mitigation***, ***Confidentiality***, ***Integrity*** and ***Origin-Data Authentication of the Payload Data*** of this message;
        - The message also contains a ***Digital Signature,*** signed by with the ***Private Key*** of the ***Client***, in order to guarantee the ***Origin-Peer Authentication*** and, in a way that can be verified by everyone who can obtain its ***Public Key from its X.509 Certificate***;
    - This message was ***implemented*** and ***integrated***;

- ***Secure Proof Of Work Broadcast Message:***
    - This message is sent by the ***Auction Server*** to all the other ***Clients***, in order for the ***Proof Of Work*** to be *verified and validated by, at least, the majority of the principals of the* ***Auction System***.
    - The message was defined with the necessary security parameterizations, in order to, guarantees:
        - ***Confidentiality*** of the "envelopes" containing a ***Pair of Session Keys*** ***(K_{SYM}, K_{MAC})***, ciphered with the ***Public Key obtained from the Certificate X.509 of all the other Clients (except, the one responsible for solving the challenge and Proof Of Work)***, in a way that only the same, individually, should be the only one to open that "envelope", through their ***Private Keys***, to guarantee the ***DoS Mitigation***, ***Confidentiality***, ***Integrity*** and ***Origin-Data Authentication of the Payload Data*** of this message;
        - The message also contains a ***Digital Signature,*** signed with the ***Private Key*** of the ***Auction Server***, in order to guarantee the ***Origin-Peer Authentication*** and, in a way that can be verified by everyone who can obtain its ***Public Key from its X.509 Certificate***;
    - This message was ***implemented*** and ***integrated***, despite being used the

same ***Pair of Session Keys (K_{SYM}, K_{MAC})*** of ***Proof Of Work Message***, due to the time available to perform the project and to increase the performance of the **Auction System**;

- ***Secure Proof Of Work Validation Broadcast Message:***
    - This message is sent by all the ***Clients*** and the ***Auction Server*** who received the ***Proof Of Work*** and validate it, to each other, by “broadcast” communication, in order to the ***Auction System***, guarantees that it’s reached a *majority and consensus validation of the* ***Proof Of Work***;
    - This message should have a computational complexity of $\boldsymbol{O((n-1)^2)}$, because *all the principals of the* ***Auction Server*** *must verify the* ***Proof Of Work****, and send a message, confirming the verification and validation of the same*;
    - This message was ***not implemented*** and ***integrated***, due to the time available to implementation of the project and its *scale complexity*;

.

3 Implementation details

It was implemented several _secure messages_, in order to be _securely exchanged between the involved principals_, using _security techniques_, such as:

- ***Symmetric Cryptography Ciphersuites***;
- ***HMACs/MACs (Hash Message Authentication Codes/Message Authentication Codes) Functions for Origin-Data Authentication and Integrity of the exchanged Messages***;
- ***SHAs (Secure Hash Algorithm) Functions for quick Integrity of exchanged Messages;***
- ***Asymmetric Cryptography for protocols of secure distribution of "envelopes" containing Session Keys (Symmetric and HMAC Keys), defined by ourselves, guaranteeing also, its Origin-Peer Authenticity, as also, Origin-Data Authenticity and Integrity, using techniques as Digital Signatures and SHAs (Secure Hash Algorithm) Functions***;
- ***X.509 Certificates, for Origin-Peer Authentication***;
- ***Digital Signatures for Origin-Peer Authentication, using Public Key Infrastructure/System Algorithms, such as, R.S.A. (Rivest-Shamir-Adleman) Algorithm, per example***;

- ***Keystores***, using ***Password-Based Encryption*** techniques for each entry:
    - ***C.A. Root (Certification Authority Root)***, containing:
        - The ***Pair of Keys*** (***Public and Private Keys***) of the ***C.A. Root (Certification Authority Root)***, _used to sign the **X.509 Certificates** of the principals involved in the **Auction System**_;

    - ***Auction Server***, containing:
        - The ***Pair of Keys*** (***Public and Private Keys***) of the ***Auction Server***;
        - The ***Certificate X.509*** of the ***C.A. Root (Certification Authority Root)*** and as its descendant, the ***X.509 Certificate*** of itself (***Auction Server)***, forming a ***Chain of X.509 Certificates;***
        - _Example_:
            - ***auction-server:***
              |--- *x509-certificate-ca-root*
              |--- *x509-certificate-auction-server*

    - ***Clients***:
        - Containing the ***Pair of Keys*** (***Public and Private Keys***) of each ***Client***;
        - The ***Certificate X.509*** of the ***C.A. Root (Certification Authority Root)*** and as its descendant, the ***X.509 Certificate*** of itself (***Client***), forming a ***Chain of X.509 Certificates***;
        - _Example_:
            - ***client:***
              |--- *x509-certificate-ca-root*
              |--- *x509-certificate-client*

- ***Truststores***:

    - ***Auction Server***:
        - Containing the ***X.509 Certificate*** of the ***C.A. Root (Certification Authority Root)*** and as its descendant, the ***X.509 Certificate*** of itself, forming a ***Chain of X.509 Certificates***;
        - <u>*Example*</u>:
            - ***auction-server:***
              |--- *x509-certificate-ca-root*

    - ***Clients***:
        - Containing the ***Certificate X.509*** of the ***C.A. Root (Certification Authority Root)*** and as its descendant, the ***X.509 Certificate*** of itself, forming a ***Chain of X.509 Certificates***;
        - <u>*Example*</u>:
            - ***client:***
              |--- *x509-certificate-ca-root*

- ***Cryptopuzzles/Proofs Of Work:***
    - It was implemented ***Cryptopuzzles/Proofs Of Work***, <u>*in order to mine and close **Blocks of Bids**, with difficulty sizes varying between 1 and 4, and with 3 strategies for the variance of the **nonce***</u> (***increasing, decreasing and random strategies***), <u>*in order to generate different **Hashes**, through **SHAs (Secure Hash Algorithms) Function**, to guess and solve the challenge*</u>;

4. Work Evaluation and Validation

The ***Security Parameterizations*** of the ***Ciphersuites*** and ***Protocols*** provided by the ***Auction Server***, through the ***T.L.S. Protocol***, were *implemented and checked*, using the *audit frameworks and tools*, ***Wireshark*** and ***testssl.sh***;

```
 Testing protocols via sockets except NPN+ALPN 

 SSLv2      not offered (OK)
 SSLv3      not offered (OK)
 TLS 1      not offered
 TLS 1.1    not offered
 TLS 1.2    offered (OK)
 TLS 1.3    not offered and downgraded to a weaker protocol
 NPN/SPDY   not offered
 ALPN/HTTP2 not offered

 Testing cipher categories 

 NULL ciphers (no encryption)                  not offered (OK)
 Anonymous NULL Ciphers (no authentication)    not offered (OK)
 Export ciphers (w/o ADH+NULL)                 not offered (OK)
 LOW: 64 Bit + DES, RC[2,4] (w/o export)       not offered (OK)
 Triple DES Ciphers / IDEA                     not offered (OK)
 Obsolete: SEED + 128+256 Bit CBC cipher       offered
 Strong encryption (AEAD ciphers)              not offered
```

Image - Part of the result of the testssl.sh tool with the configuration delivered on Github

The exchange of the ***Certificates***, during the *handshake process* of ***T.L.S. Protocol***, were also *implemented and checked*, guaranteeing the ***Mutual Authentication***, of both ***Client*** and ***Auction Server***, to each other, or only ***Server-side Authentication*** using the previously mentioned *audit frameworks and tools*;

No.	Time	Source	Destination	Protocol	Length	Info
3	0.000023896	127.0.0.1	127.0.0.1	TCP	66	39304 → 8443 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=1790903456 TSecr=1790903456
4	0.007413544	127.0.0.1	127.0.0.1	TLSv1.2	195	Client Hello
5	0.007429304	127.0.0.1	127.0.0.1	TCP	66	8443 → 39304 [ACK] Seq=1 Ack=130 Win=65408 Len=0 TSval=1790903464 TSecr=1790903464
6	0.010753070	127.0.0.1	127.0.0.1	TLSv1.2	1773	Server Hello, Certificate, Certificate Request, Server Hello Done
7	0.010777505	127.0.0.1	127.0.0.1	TCP	66	39304 → 8443 [ACK] Seq=130 Ack=1708 Win=64128 Len=0 TSval=1790903467 TSecr=1790903467
8	0.050345482	127.0.0.1	127.0.0.1	TLSv1.2	1868	Certificate, Client Key Exchange
9	0.050365466	127.0.0.1	127.0.0.1	TCP	66	8443 → 39304 [ACK] Seq=1708 Ack=1932 Win=64000 Len=0 TSval=1790903507 TSecr=1790903507
10	0.077018901	127.0.0.1	127.0.0.1	TLSv1.2	335	Certificate Verify
11	0.077030704	127.0.0.1	127.0.0.1	TCP	66	8443 → 39304 [ACK] Seq=1708 Ack=2201 Win=64768 Len=0 TSval=1790903533 TSecr=1790903533
12	0.077073666	127.0.0.1	127.0.0.1	TLSv1.2	72	Change Cipher Spec
13	0.077078236	127.0.0.1	127.0.0.1	TCP	66	8443 → 39304 [ACK] Seq=1708 Ack=2207 Win=64768 Len=0 TSval=1790903533 TSecr=1790903533
14	0.091169546	127.0.0.1	127.0.0.1	TLSv1.2	151	Encrypted Handshake Message
15	0.091182527	127.0.0.1	127.0.0.1	TCP	66	8443 → 39304 [ACK] Seq=1708 Ack=2292 Win=64768 Len=0 TSval=1790903548 TSecr=1790903548
16	0.152132195	127.0.0.1	127.0.0.1	TLSv1.2	72	Change Cipher Spec
17	0.152148778	127.0.0.1	127.0.0.1	TCP	66	39304 → 8443 [ACK] Seq=2292 Ack=1714 Win=65536 Len=0 TSval=1790903609 TSecr=1790903609
18	0.152440510	127.0.0.1	127.0.0.1	TLSv1.2	151	Encrypted Handshake Message
19	0.152447008	127.0.0.1	127.0.0.1	TCP	66	39304 → 8443 [ACK] Seq=2292 Ack=1799 Win=65536 Len=0 TSval=1790903609 TSecr=1790903609
20	41.892708565	127.0.0.1	127.0.0.1	TLSv1.2	135	Application Data
21	41.892722180	127.0.0.1	127.0.0.1	TCP	66	8443 → 39304 [ACK] Seq=1799 Ack=2361 Win=65536 Len=0 TSval=1790945349 TSecr=1790945349
22	43.023174493	127.0.0.1	127.0.0.1	TLSv1.2	183	Application Data
23	43.023193641	127.0.0.1	127.0.0.1	TCP	66	8443 → 39304 [ACK] Seq=1799 Ack=2478 Win=65536 Len=0 TSval=1790946480 TSecr=1790946480

Image - TLS handshake procedure of a mutual authentication handshake on Wireshark

It was used also the ***Keytool*** to generate the auto-signed the ***X.509 Certificates*** by a fictitious ***C.A. Root (Certification Authority Root)*** and also, the ***Keystore Explorer*** tool, to inspect its content;

It was also used the ***Keytool*** to generate the ***X.509 Certificates*** of both, ***Auction Server*** and ***Clients*** signed by the ***C.A. Root (Certification Authority Root)***, forming a ***Chain of X.509 Certificates***;

It was implemented the <u>*mining process*</u> of ***Block of Bids***, through ***Cryptopuzzles/Proofs Of Work***, with many variations and difficulty levels but not integrated with a ***Data Structure*** properly designed for that, such a ***Merkle Tree***, forming a ***Chain of Hashes*** in a ***Binary Tree***, where t<u>*he validation of the **Blocks** are reenforced by any **Block** added, because contains in it, all the previous **Hashes***</u>, which it's the <u>*true vision and function*</u> of a ***Blockchain*** system. The ***Merkle Tree*** <u>*it was also adapted/implemented, but **not** used*</u>, due to the given time to implement the project;

It wasn't implemented the <u>*behaviour of interrupting the mining process of a **Block***</u> by a ***Client***, in the case of, receive a ***Proof Of Work Message***, <u>*which it's valid and contains at least one **Bid** inside the **Block of Bids**, which the current **Client** it's trying to mine, at the moment*</u>. This can be implemented by a concurrent ***Thread***, checking this, every time, the

Client receives a ***Secure Proof Of Work Message***;

The following *exchanged messages are completely developed and integrated with the **Auction System** and built under **Security** assumptions*:

- **Secure Bid Message**;
- **Secure Receipt Message**;
- **Secure Proof Of Work Message**;
- **Secure Proof Of Work Broadcast Message** (but not implemented with stronger ***Security assumptions***);

The following *exchanged messages are developed and integrated with the **Auction System** but **not** built under **Security** assumptions*:

- **Secure Bid Broadcast Message**;

The following *exchanged messages aren't completely developed neither integrated with the **Auction System**, at all*:

- **Secure Proof Of Work Validation Broadcast Message**;

5. Conclusion

The emergence of ***Blockchain Systems*** are starting to become very popular, not only for banking and money transactions, but mostly because of the strong property of validation that provides.

But most part of this kind of systems use processes which can't be considered fair, such the ***Cryptopuzzles/Proof Of Works***, because *this process considers only the computational power of the machine, which it's performing it*. Because of that, are being discussed *other different mining processes*, such as, ***Proof-of-Stake, Proof of Authority, Proof of Space, Proof of Burn***, among others.

Solving the ***Cryptopuzzles/Proof Of Works*** _to reach a consensus of all the members of the System_, require a _high computational power and have a high computational complexity_, which can have the performance of this process improved, by taking advantage of ***Parallel Computing*** or ***High Performance Computing*** using ***GPUs***, per example, or even, taking advantage, in the near future, of the emergent ***Quantum Computing***, through the properties of ***Quantum Superposition*** and _simultaneously processing properties_, which it's viewed as a great threat to ***Blockchain Systems***.

The **Secure Blockchain-Enabled Auction Management System** successfully implemented secure **T.L.S. communication channels** and partially implemented a complete **blockchain** service.

The usage of **T.L.S. channels** allowed us to learn how to protect messages between principals, how to use audit tools to check the correct usage of said channels and how authentication works with the use of **asymmetric cryptography** and on protecting messages with "**envelopes**" and signatures for confidentiality and authentication. Furthermore, it gave us the opportunity to mix **symmetric cryptography** for more protections.

The **blockchain** portion of the system ended up not being completely implemented and integrated but still gave us an opportunity to see how to use and how to implement a **blockchain** service.

6. References

[1] Course on Computer Networks and Systems Security, MSc Program in Informatics Engineering, DI/FCT/UNL 2019/2020, Work-Assignment #2 Statement and Initial Specifications, November/2019.