Segurança Informática e nas Organizações

Blockchain-based Auction Management

Security Mechanisms Design

Group P1G9

Group P1G9
Tibério Baptista – 81258
Nuno Silva – 79958

28/01/2019

Contents

System Description

The project in question is an auction system that can manage and store auctions requested by clients. There are 2 sections of the system, the multiple clients and the servers, of which there are 2.

The Auction Client**:** Can request the auction creations to the Auction Manager, and perform bids to the Auction Repository. Each client can freely request a view of auction/bid details (within security limits that are discussed in the next section) but must provide is Citizen Card's digital info to be able to do all of these things.

The Auction Manager**:** Is able to request the creation of auctions and cease them, upon a time-out. The auctions created are either of the type "English Auction" or "Blind Auction", each with slightly different rules. It can also perform bid validations to hide private data..

The Auction Repository**:** Generates the auctions and handles bids requested by users. Said bids are passed to the Manager for validation. All auctions have a name, serial number (unique), a time limit and a description, and also a sorted linked list of bids, which may or may not be ordered by the bid value (depends on the type of bid). The Repository can also emit receipts for users, which can be checked with the repository.

Processos:

- Create/terminate an auction;

O utilizador pode criar uma auction definindo o seu nome, descrição, tipo (English ou Blind), e o tempo limite (em minutos)

- List open/closed auctions;

No menu apresentado ao utilizador é lhe dada a opção de listar as auction(as que estão a decorrer/ativas e as que já se encontram terminadas).

- Display all bids of an auction;

O cliente indica o id da “auction” para a qual pretende observar as bids e é recebido uma lista de todos os autores e as suas respetivas bids.

Nota: Se se tratar de uma bid que seja proveniente de uma “auction” do tipo “Open ascending price auction”, o nome irá ser apresentado cifrado. Os values são cifrados se for “Blind”

- Display all bids sent by a client:

  Nota: Não conseguimos realizar este processo.

- Check the outcome of an auction where the client participated;

  Nota: Não conseguimos realizar este processo.

- Validate a receipt.

Pode-se consultar e validar recibos de bids que o cliente executa. Será melhor abordado e explicado na seção “Segurança”.

Segurança:

• **Protection (encryption, authentication, etc.) of the messages exchanged;**

Ligação Inicial:

O estabelecimento de ligação inicial consiste na troca de desafios entre um e outro para provar que as suas chaves assimétricas são legítimas, seguido por guardar as chaves opostas e estabelecerem uma 'session key'. O AuctionClient assina o challenge dos servidores com o seu cartão de cidadão, em particular com a chave privada do certificado de autenticação, e transmite este ciphertext juntamente com a certidão de chave pública desta chave (No ponto seguinte explicamos como se valida o CC do cliente). Os servers também fazem um processo similar com o cliente, respondem a um challenge do cliente assinando-o com uma chave privada e enviando a certidão da pública, que pertence a uma chave assimétrica previamente criada e bem conhecida. Com isto concluído, é emitido um shared secret a ser partilhado entre 2 entidades para fins de comunicação.

É de notar que como as chaves públicas dos servidores são bem conhecidas, não é necessário trocas de challenges entre estes. Podemos assumir que já foram partilhadas e apenas partilhamos um secret.

Validação do Utilizador:

As entidades estabelecem uma troca de mensagens iniciais e pedidos/respostas de 'challenges' de cada um.

No lado do AuctionManager, antes de usar a certidão emitida pelo cliente, deve confirmar se esta é legítima. Portanto deve proceder à validação da certidão. A validação da certidão consiste em verificar o CA "issuer" da certidão e dos issuers subsequentes, e tentar detetar as suas existências nas CRLs , verificando se esta se encontra revogado(recusando-o se esta for detectada lá) e no ficheiro "PTeID.pem", adquirido a partir do website do professor Barraca.

A cadeia termina com um CA raiz. Se não se encontra nos CRLs mas está no PTeID,este é um certificado seguro. Para além disto, é também verificada a validade de cada CA e a verificação se os seus propósitos são válidos (se têm autorização para emitir certidões).
Este processo continua para cada "issuer" de cada CA intermédio detectado, até chegarmos ao CA raiz: O certificado de CA raiz é sempre assinado pela própria autoridade de certificação, portanto podemos concluir a pesquisa aqui e garantir que, baseado na nossa informação, a certidão emitida é segura.

Com a validação concluída, o server verifica a assinatura, comparando-o ao challenge que mandou anteriormente. Verificando que a assinatura é válida, o server emite um segredo para o cliente e este guarda-o localmente. O cliente recebe este segredo e guarda-o também.

Troca de Mensagens:

As entidades estabelecem uma troca de mensagens iniciais e pedidos/respostas de 'challenges' de cada um. É de notar que as mensagens em si são do formato JSON, serializados. Após a ligação, têm a informação necessária para a transmissão segura de mensagens. Para enviarmos uma mensagem entre 2 entidades, de 'A' para 'B'. Utilizamos uma chave híbrida:

- A mensagem JSON de 'A' é serializada e encriptada com um segredo + iv gerados aleatoriamente;
- O segredo prévio é cifrado com a session key que A e B partilham;
- O segredo, cifrado com a session key, é depois cifrado com a chave pública de B;
- Um 'tag' chamado "requestType", o ID de A ("Manager"/"Repository" para os servers, ou o serial id do cartão de cidadão para os clientes), a mensagem cifrada, o segredo, cifrado e o IV são enviados como um JSON serializado.

No lado de B, este deve de fazer o processo inverso para descobrir a sub-mensagem original. Este método de cifragem garante que os canáis de comunicação são seguros.

- **Protection of the bids until the end of their auction;**

O manager, na criação de auctions, define auctions "English" e auctions "Blind" consoante o que o utilizador envia. A identidade do autor da bid é cifrado fazendo uso da chave pública deste se esta for do tipo "English"bids e são ordenadas por ordem ascendente, enquanto que nas "Blind" auctions o valor da bid é cifrado usando a chave pública. A chave é do tipo pública, gerado previamente pelo cliente que criou a auction.

- **Identification of the bid author with a Citizen Card**

Para atingir este objetivo, o preço da bid e o número de série do utilizador são agrupadas por uma funão de 'xor' e, posteriormente, são assinados com a chave privada do cartão de cidadão, e esta assinatura será um dos atributos necessário aquando da criação de "bid".

- **Exposure of the necessary bids at the end of an auction;**

Não foi possível concluír os métodos de exposição de bids relevantes. Seria adquirir a chave privada que o cliente cria para uma auction e usá-la para decifrar campos relevantes. Nas English auctions seria apenas decifrar o nome do vencedor, e nas Blind seria decifrar todos os valores.

- **Validation of bids using dynamic code**

Bids do tipo “English” são recusadas se forem inferiores à bid mais alta, garantindo que a ordem das bids em auctions do tipo English é ascendente pelo bid value

• Modification of validated bids by dynamic code;

´É realizada a cifragem de certos campos de bids consoante que tipo de bid que esta é. Para as “bids” do tipo “English” o valor do autor é cifrado ,enquanto que nas “bids” do tipo “Blind” o campo que se cifra é o valor.

• Construction of a blockchain per auction

Cada auction possui uma linked list, contendo bids. Cada bid contém um campo “digest” que contém um hash value, este hash é gerado com os atributos “creation_date” e “autor” da bid, e também a hash da bid prévia. Para assegurar a integridade das bids de uma auction, temos o método “Verify_Integrity()” no AuctionRepository que compara o cálculo de uma hash calculado do início ao fim da linked list com o valor do digest da ùltima bid. Se houve qualquer alteração na linked list, a hash do cálculo ficará diferente do hash da bid mais recente, e a auction é vista como comprometida.

• Deployment of Cryptopuzzles

No momento antes da criação da “bid”, o repositório pede ao cliente para resolver um cryptopuzzle e enviar para este o resultado. O cryptopuzzle em questão é um gerador de valores binários que apenas pára quando tiver uma quantidade de “1”s menor que um certo valor. O repositório irá receber o resultado do cliente e verificar rapidamente se é um resultado válido. Sendo válido o processo de bidding pode continuar.

A função “Cryptopuzzle()” recebe dois argumentos, argumentos estes que servem para alterar/influenciar a dificuldade do “cryptopuzzle”, tal como pedido. Por cada bid adicionada, o número de bits que são gerados aumenta, +1 por cada bid para subir o tempo média de espera. O número de bits inicial é 32.

• Production and validation of bid receipts

Após o processo de adicionar a bid a uma auction, o repositório envia o recibo para o cliente e este guarda-o numa pasta denominada com o seu serial id do CC. O recibo contém o timestamp da inserção, o índice da auction, o valor da bid, uma assinatura do repositório.

• Validation of a closed auction (by a user client);

O “Verify_Integrity()” referido anteriormente no “Construction of a blockchain per auction” serve para isto. **Podemos manualmente pedir esta função depois.**

Deficiências:

- No fim de uma auction, o cliente não envia a chave privada para o repositório para que seja possível decifrar os dados relevantes de uma bid;
- O sistema não consegue verificar as assinaturas de bids, para confirmar autores;
- A validação de recibos não funciona com sucesso;
- Auctions são guardadas em binary files regulares, não estão encriptadas
- Assinatura autor está sempre vazia nos recibos.

Links Consultados Durante a Realização do Trabalho:

https://support.dnsimple.com/articles/what-is-ssl-certificate-chain/

https://docs.python.org/2/library/pickle.html

https://cryptography.io/en/latest/

https://en.wikipedia.org/wiki/Chain_of_trust

https://pki.cartaodecidadao.pt/publico/lrc/

https://joao.barraca.pt/teaching/sio/2018/p/PTEID.pem

https://knowledge.digicert.com/solution/SO16297.html