LEI: Segurança Informática e nas Organizações 2019-2020

Projeto 2: Comunicações Seguras

Entrega: 17 de Novembro, 23:59

Objetivos

- Planeamento da segurança das comunicações
- Negociação de chaves
- Confidencialidade
- Integridade

1 Descrição

Este trabalho visa explorar os conceitos relacionados com o estabelecimento de uma sessão segura entre dois interlocutores. Explora conceitos relacionados com a troca de chaves, cifras simétricas e controlo de integridade.

2 Preparação

Deve-se considerar o código fornecido para o projeto como base. Este código implementa um cliente e um servidor usando um protocolo próprio baseado em mensagens JSON, sobre sockets TCP/IP. Por questões de simplicidade e de capacidade de análise das mensagens usando aplicações como a Wireshark, todas as mensagens transmitidas serão codificadas para texto (ex. usando base64).

Para execução do código, recomenda-se a criação de uma pasta com os ficheiros fornecidos. Depois, nesta pasta, deverão ser executados os seguintes comandos:

```
$ virtualenv -p python3 venv
$ source ./venv/bin/activate
$ pip3 install -r requirements.txt
```

O cliente aceita como argumento um ficheiro e um endereço de destino. O cliente irá:

- Abrir uma sessão TCP para o servidor.
- Enviar uma mensagem `OPEN` que especifica o nome do ficheiro. O servidor responde com uma mensagem `OK`.
- Enviar múltiplas mensagens`DATA` com partes consecutivas do ficheiro.

Caso exista um qualquer erro, o servidor responde com uma mensagem `ERROR`.

O servidor aceita como argumento um porto TCP e irá receber ficheiros transferidos por um cliente, guardando a informação recebida no sistema de ficheiros. O nome do ficheiro é filtrado, sendo que só se permitem letras, números e pontos.

A implementação de ambos os componentes é assíncrona, pelo que não será adequado utilizar código síncrono bloqueante (ex, `send->recv->send...`). O controlo do estado é realizado através de uma variável `self.state`, que indica o estado atual da ligação.

As mensagens consideradas são as seguintes:

Mensagem: OPEN

```
{
        "type": "OPEN",
        "file_name": "name of the file to transfer"
}
```

Mensagem: DATA

```
{
        "type": "DATA",
        "data": "base64 encoded data chunk"
}
```

Mensagem: CLOSE

```
{
        "type": "CLOSE"
}
```

Mensagem: OK

```
{
        "type": "OK"
}
```

Mensagem: ERROR

```
{
        "type": "ERROR",
        "message": "error message"
}
```

Os alunos podem alterar e ou adicionar as mensagens que considerem necessário.

3 Trabalho a realizar

O projeto consiste no desenho e implementação de um protocolo que permita a comunicação segura (confidencial e íntegra) entre dois pontos. Pretende-se que seja possível trocar um ficheiro entre o cliente e o servidor usando este protocolo.

O trabalho a realizar considera o planeamento, desenho, implementação e validação do protocolo, utilizando o código fornecido como base para o trabalho. A avaliação irá focar-se em cada um dos pontos a seguir descritos.

1. Desenho de um protocolo (planeamento e descrição) para o estabelecimento de uma sessão segura entre o cliente e o servidor, com: a) negociação dos algoritmos usados, b) confidencialidade, c) controlo de integridade, d) rotação de chaves. Deve-se considerar que tanto o cliente como o servidor suportem pelo menos duas cifras simétricas (ex, AES e Salsa20), dois modos de cifra (ex, CBC e GCM) e dois algoritmos de síntese (ex. SHA-256 e SHA-512). Por exemplo, considera-se que a combinação `DH_AES_128_CBC_SHA512` irá despoletar uma troca de chaves com o algoritmo Diffie-Hellman, depois utilizar uma cifra AES com chaves de 128 bits (negociada com o DH), aplicada no modo CBC e depois um controlo de integridade com o algoritmo SHA512. A sessão deverá iniciar-se com o acordo do conjunto de cifras a suportar, seguida da troca de chaves e finalmente troca do ficheiro. A rotação de chaves pode ser feita através de uma mensagem adicional enviada pelo cliente ou servidor.
2. Implementar a negociação de algoritmos de cifra entre cliente e servidor.
3. Implementar o suporte para confidencialidade, resultando em mensagens cifradas.
4. Implementar o suporte para integridade, resultando na adição de códigos de integridade às mensagens.
5. Implementar um mecanismo para rotação da chave utilizada após um volume de dados ou tempo decorrido.

Sugere-se que sejam implementadas funções genéricas de cifra/decifra/cálculo de um MAC/verificação de um MAC de textos. Estas funções podem aceitar o texto, algoritmo e outros argumentos, realizando uma ação específica.

Sugere-se ainda que sejam criadas mensagens novas (ex, `type=SECURE_X`) e que não alteradas as existentes. Estas novas mensagens correspondem ao estabelecimento da sessão e cifra. As mensagens atualmente existentes poderão estar incluídas (num formato cifrado e com integridade) como a parte do conteúdo destas novas a criar.

O esquema que se segue demonstra esta aproximação:

```
{
        'type': 'SECURE_X',
        'payload': <mensagem OPEN/DATA/CLOSE cifrada>
        .... // outros campos
}
```

A entrega deverá consistir nas chaves necessárias à operação (se algumas), o código desenvolvido e um relatório. O relatório deverá descrever o protocolo e demonstrar o funcionamento de cada funcionalidade (ex. execução mais capturas de ecrã).

4 Notas

Considera-se que os trabalhos são realizados por 2 alunos e que o documento final submetido é de sua autoria. A utilização de recursos existentes na Internet ou partilhado com outros colegas leva à anulação imediata do trabalho.

Podem e devem ser utilizadas bibliotecas criptográficas como a `Cryptography.io`. Podem também ser utilizadas outras bibliotecas, desde que forneçam apenas suporte à implementação dos mecanismos propostos.