

**Segurança de Redes e Sistemas de Computadores 2017/2018**  
**Trabalho Prático nº 1 (v1.1, 25/Mar/2018)**

**Protocolo SGCM e subprotocolos SGCP-TLP e SGCP-SAP**

**Especificações de Referência para Desenvolvimento**

**A - Especificação para o subprotocolo STGC-TLP (Fase 1)**

Este subprotocolo implementa o suporte de comunicação segura UDP/IPMC, com base num formato de encapsulamento seguro das mensagens UDP do nível aplicação em *datagramas* (como criptogramas UDP) seguros. Um *datagrama* seguro em STGC-TLP não é mais do que um formato de encapsulamento seguro de mensagens UDP com garantias de proteção como contra-medidas face à tipologia de ataques e modelo de adversário anteriormente definidos. Cada mensagem UDP do nível aplicação (que designamos por M), será protegida como M' da forma seguinte:

**M' = HEADER || PAYLOAD**

Em que o HEADER e o PAYLOAD são especificados do seguinte modo:

**HEADER** = VERSION-RELEASE || 0x00 || PAYLOAD\_TYPE || 0x00 || PAYLOAD\_SIZE

Version-Release: 1 byte (4 bits VERSION + 4 bits RELEASE)

Ex: 1.1 : 00010001

0x00 byte usado como separador

PAYLOAD\_TYPE: 1 byte, ex:

M: indica que payload contem mensagem de dados de aplicação

S: indica que payload contem uma mensagem STGC-SL

PAYLOAD\_SIZE: 2 bytes (ou short integer) contendo o tamanho do Payload transportado

**PAYLOAD** representa a carga da mensagem STGC que depende do PAYLOAD\_TYPE

PAYLOAD\_TYPE = M: Payload representa dados (bytes) de uma aplicação (ou seja, codificando as mensagens de um protocolo do nível aplicação)

PAYLOAD\_TYPE = S: Payload representa dados (bytes) para processamento segundo o subprotocolo STGC-SL.

Em todo o caso, o PAYLOAD é codificado com suporte criptográfico da seguinte forma:

$$\text{PAYLOAD} = E(K_S, [M_p || \text{MAC}_{K_M}(M_p)]) || \text{MAC}_{K_A}(C)$$

Sendo:

$$M_p = [\text{id} || \text{nonce} || M]$$

$$C = E(K_S, [M_p || \text{MAC}_{K_M}(M_p)])$$

Ks: chave de sessão (sessão de grupo multicast segura STGC)

KM: Chave de autenticidade e integridade na função MAC

KA: Chave de controlo para mitigação de ataques contra a disponibilidade

Nesta fase será implementado o protocolo STGC-TLP, sendo o mesmo integrado para proteger a aplicação escolhida pelos grupos de entre as duas disponibilizadas. A aplicação será assim usada como demonstrador da implementação. Para o efeito serão usados ficheiros de configuração manuais que serão usados em cada componente da aplicação, nomeadamente:

- Aplicações *multichat* (cliente) no caso da aplicação *multicast*
- Servidor e *proxy* (cliente) no caso da aplicação de *streaming multicast*

Os ficheiros de configuração serão os seguintes:

- Ficheiro *ciphersuite.conf*. Este ficheiro contem a *ciphersuite* e todos os parâmetros necessários a usar em cada grupo multicast em que o cliente vai participar

- Ficheiro **keystore.jceks**: Trata-se de uma *keystore* JAVA do tipo JCEKS (protegida por passwords) que contem todas as chaves necessárias, nomeadamente as chaves  $K_S$  e  $K_M$  que devem obviamente ser compatíveis com a configuração da *ciphersuite* que estiver no ficheiro *ciphersuite.conf*

Por exemplo, para participar na sala 224.10.10.10, o ficheiro *ciphersuite.conf* poderia ter a seguinte configuração, onde podem estar definidas outras associações de segurança para outros grupos multicast.

```
# Definição da ciphersuite, chaves e parâmetros para parametrizações de segurança de grupos
# multicast protegidos pelo protocolo STGC

<224.10.10.10>
CIPHERSUITE: <alg/mode/padding> # ex., AES/CBC/PKCS#5
KEYSIZE:      <valor>           # n. de bits da chave, ex: 256
KEYVALUE:     <valor>           # chave, representada em hexadecimal
MACKM:        <mac-alg>         # algoritmo MAC (HMAC ou CMAC), por ex: RC6-GMAC
MACKMEYSIZE:  <valor>           # n. de bits da chave do MAC, ex: 256
MACKMEYVALUE: <valor>           # chave MAC, (KM), ex: representada em hexadecimal
MACKA:        <mac-alg>         # algoritmo MAC (HMAC ou CMAC), por ex: RC6-GMAC
MACKAKEYSIZE: <valor>           # n. de bits da chave do MAC, ex: 256
MACKAMEYVALUE: <valor>         # chave MAC, (KM), ex: representada em hexadecimal
</224.10.10.10>
<239.7.22.122>
...
</239.7.22.122>
```

Para a Fase 1 considera-se que estes ficheiros se configuram e instalam manualmente e que a *keystore* com as chaves foi obtida de forma segura para ser instalada em cada computador que estará envolvido na aplicação em causa. O facto de possuírem o ficheiro (no pressuposto que o obtiveram licitamente) e que possuem a password com que está protegida a *keystore*, garante aos utilizadores poderem participar nos grupos multicast seguros pelo protocolo SGCM-TLP.

**Nota importante:** se suportar chaves protegidas na *keystore* **keystore.jceks**, todos os campos relativos a chaves (incluindo as usadas em construções MAC) poderão ficar com o valor \* (asterisco) já que estes valores decorrerão naturalmente do conteúdo da *keystore*, sob a forma de diferentes chaves criptográficas simétricas, geradas externamente (keytool ou KeyStore Explorer, por exemplo).

## B - Especificação de referência de partida para o subprotocolo STGC-SAP (Fase 2)

O protocolo STGC-SAP implementa um protocolo de autenticação de utilizadores, para suportar autenticação e controlo de acesso de utilizadores e para estabelecimento de parâmetros de associações de segurança que são depois usados para participação em grupos seguros de comunicação *multicast*, suportados pelo protocolo STGC-TLP. No caso de utilizadores devidamente autenticados e com autorização para se juntarem a tais grupos *multicast*, o protocolo STGC-SAP permitirá que se possam obter dinamicamente e de forma segura as configurações e parametrizações criptográficas específicas (*ciphersuites* e *chaves criptográficas necessárias*) que depois são adoptados no subprotocolo STGC-TLP.

Note-se que do ponto de vista do suporte de comunicação, o subprotocolo STGC-SAP deverá ele próprio ser suportado nos formatos definidos pelo subprotocolo STGC-TLP.

O protocolo STGC-SAP implementará o suporte de autenticação e controlo de acesso a um grupo multicast com base em passwords, usando construções criptográficas HMAC ou CMAC bem como construções para encriptação baseada em passwords (ou PBE - *Password-Based Encryption*). Para o efeito, será preciso implementar-se o subprotocolo STGC-SAP e um servidor de autenticação, que será usado como um servidor SSO (*Single Sign On*) para autenticação centralizada.

### Requisitos de especificação inicial para o protocolo STGC-SAP

O protocolo STGC-SAP deverá ser concebido na sua completude e depois implementado a partir dos seguintes requisitos primários, sendo a correção da completude da concepção e implementação considerados fatores qualitativos de avaliação.

O protocolo deve atuar em duas rondas de mensagens trocadas com o servidor AS:

#### Ronda 1:

**Cliente > AS:    Cliente ID || NonceC || IPMC || AutenticadorC**

// IPMC é um endereço Multicast a que o cliente se quer juntar

// NonceC é um desafio – inteiro, fabricado a partir de geração segura, usado como um nonce

```
// por parte do cliente
// O AutenticadorC do cliente será fabricado pelo cliente do seguinte modo:
  E [ K, ( Nonce C || IPMC || SHA-512(pwd) || MACK(X) ) ]
// usando um esquema do tipo Password-Based Encryption , em que se usou SHA-512 (pwd)) como
// chave K e MD5 (NonceC || SHA-512(pwd) ) como chave da privacidade MAC.
// X corresponde ao plaintext Nonce C || IPMC || SHA-512(pwd)
```

Depois do servidor AS verificar que o cliente em causa é autêntico e está autorizado a juntar-se ao endereço IPMC pretendido envia a mensagem na ronda 2.

## Ronda 2:

**AS > ClienteID:**  $E[K_{PBE}, ( \text{NonceC}+1 || \text{NonceS} || \text{TicketAS} ) || \text{MAC}_K(X) ]$

```
// NonceC+1 é a resposta ao desafio NonceC da ronda 1, por parte do servidor.
// NonceS é um nonce gerado pelo servidor
// A mensagem vai cifrada usando uma cifra do tipo password-based encryption, em que
// A password-seed usada corresponde a: Hpwd || NonceC+1
// TicketAS será uma estrutura de dados que (deverá ser especificada e implementada por
// cada grupo) e que conterá todas as informações que deverão ser enviadas de forma /
// segura e que permitem ao cliente ter todas as parametrizações criptográficas que
// Permitirão entrar numa sessão segura multicast, nomeadamente, os dados
// sobre a configuração da ciphersuite (no ficheiro ciphersuite.conf)
// bem como das chaves KS e KM que se destinam a ser usadas na sessão (keystore.jacks)
// A chave para o MAC pode ser a mesma como derivada na ronda 1, podendo X ser o conteúdo da primeira parte
// da mensagem cifrada
```

Deste modo, com a informação do TicketAS já não será preciso do lado do cliente que hajam configurações estáticas e manuais (ficheiros de parametrização usados na fase1), passando estes ficheiros a existir apenas do lado do servidor AS.

A cifra na mensagem da ronda 2 deverá ser feita com base num modelo do tipo *Password-Based Encryption*

## Servidor AS

Será implementado um servidor que implementa um servidor de autenticação. Este servidor é contactado pelos clientes (MCHAT Client no caso do *Multicast* ou o *Proxy* no caso da aplicação de *streaming*) para que os respetivos utilizadores sejam autenticados. Este servidor será designado por AUTHENTICATION SERVER (AS) e será o responsável pela execução do protocolo STGC-SAP.

O servidor AS tem uma tabela DAC – *Descriptive Access Control* que não é mais do que um ficheiro de controlo de acessos **dacl.conf** que tem o registo de autorizações de acesso a grupos *multicast* seguros

224.12.12.12: maria, jose, hj, ...
224.6.23.22: maria, manuel, filipe, ...

O servidor AS possui uma *keystore* que contém entradas de chaves necessárias para cada sessão de grupo *multicast* seguro.

Finalmente, o servidor AS gere uma tabela de autenticação de utilizadores, que mapeia utilizadores registados (sendo o registo prévio e manual). Só utilizadores registados podem autenticar-se e, no caso de serem autorizados, aceder à informação que será necessária para entrarem num grupo seguro *multicast*.

A tabela de utilizadores será a seguinte (ficheiro **users.conf**)

maria/maria@hotmail.com: H(password-da-maria)
jose/jose@gmail.com: H(password-do-jose)
jfaustino:/j.faustino@campus.fct.unl.pt: H(password-do-jfaustino)

Este ficheiro estará cifrado com *Password Based Encryption*, de modo que só o servidor AS conseguia obter o seu conteúdo. Por outro lado, as passwords presentes deverão estar representadas com base numa síntese de segurança (H(pwd)), utilizando-se uma função segura de síntese (hash), por exemplo SHA3 / SHA-512. No processo de autenticação de um utilizador com base no subprotocolo STGC-SAP nunca deverá ser necessário ao servidor AS manipular a password original de cada utilizador/cliente sem ser na sua forma resumida (*hashed*) durante o procedimento de autenticação.

Além disso, o servidor AS terá os ficheiros *ciphersuite.conf*. e *keystore.jecks* que na Fase 1 eram instalados manualmente nos clientes.

Cada vez que um cliente (cliente MCHAT no caso do Multicast Chat ou proxy e servidor de envio de filmes no caso da aplicação de *streaming*) se autenticar perante o AS com base no protocolo STGC-SAP, receberá toda a informação dos referidos ficheiros, de forma segura, e que será necessária para poder entrar no grupo IPMC.

### **Ciphersuites e parametrizações do subprotocolo STGC-SAP**

A *ciphersuite* dos algoritmos PBE usados para o protocolo STGC deverá ser preferencialmente especificada num ficheiro de configuração pré-existente, no AS e clientes (aplicação). A estrutura de referencia deste ficheiro deverá ser a seguinte:

Ficheiro: *stgcsap.auth*

PBE: algoritmo PBE Encryption em causa, podendo ser usadas por exemplo:

PBE: PBESWithSHAAnd3KeyTripleDES

Deste modo, podem usar-se por configuração quaisquer construções criptográficas deste tipo, por exemplo:

PBESWithSHAAnd3KeyTripleDES  
PBESWITHSHA256AND256BITAES-CBC-BC  
PBESWITHSHA-1AND256BITAES-CBC-BC  
PBESWithHmacSHA224AndAES\_256

E qualquer construção MAC, sejam construções HMAC ou CMAC, por exemplo:

HmacSHA1  
HMAC/SHA384  
HMAC-SHA3-224  
HMAC-SHA3-256  
HMAC-SHA512  
...etc  
SKIPJACKMAC  
AESGMAC  
RC6GMAC  
RC5MAC  
DES  
... etc

Por exemplo, o ficheiro *stgcsap.auth* pode ter uma qualquer parametrização (entre outras) como a seguir se exemplifica:

STGC-SAP: PBESWithSHAAnd3KeyTripleDES:HmacSHA1:
---

Claro que este ficheiro deve estar parametrizado de forma semelhante quer do lado do servidor AS quer do lado dos clientes que vão autenticar-se no AS. Na verdade bastará que exista do lado do AS, já que os clientes podem receber a informação na ronda 2 pelo AS a cada cliente que se queira autenticar para receber a informação na ronda 2.

Nesta caso a mensagem da ronda 2 enviada pelo AS pode ter por exemplo o seguinte formato:

**AS > ClienteID:**

**$E[K_{PBE}, (NonceC+1 || Nonces || TicketAS) || MAC_K(X)] || CS$**

**Sendo no caso CS = "PBESWithSHAAnd3KeyTripleDES:HmacSHA1"**