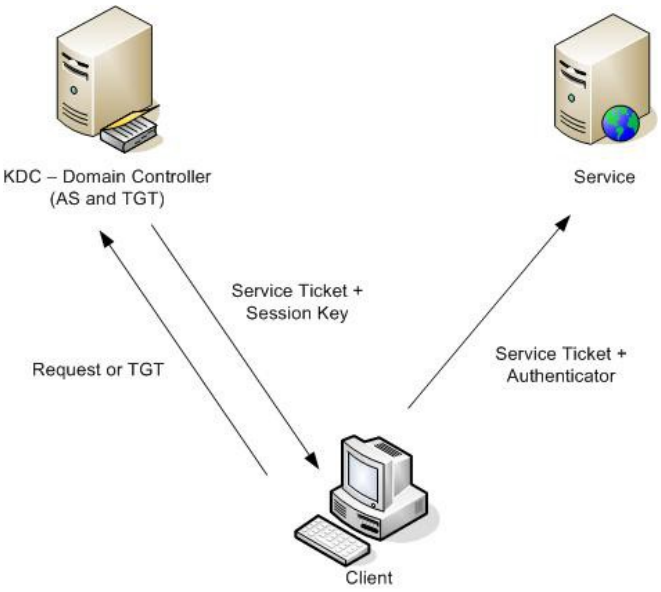
Relatório

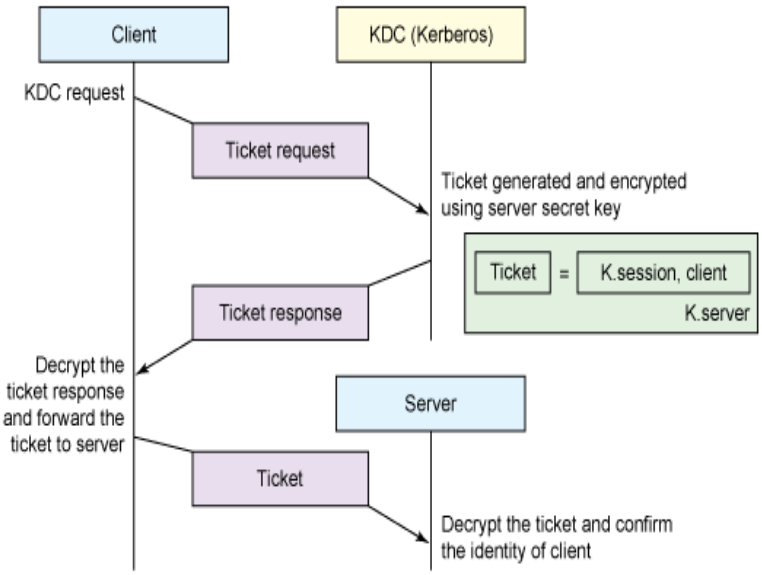


SD-ID.A

Neste relatório concluímos que a solução a considerar para a integração e implementação do *Kerberos* seria utilizar o Algoritmo de Encriptação **TripleDES**, pois mostrou-se ser o que tinha um critério segurança/simplicidade mais adequado à nossa implementação.

De uma forma a melhor compreender o problema temos as seguintes imagens do *Protocolo Kerberos*:





Resumidamente a ideia é, o **Cliente** ao pedir informação de acesso ao **Servidor Primário** terá que primeiramente fazer um pedido de autenticação ao **Servidor de Autenticação** (Saut) e após isto, o **Servidor de Autenticação** responde ao **Cliente** com um *ticket* e com uma chave da sessão. Por fim o **Cliente** autentica-se usando um **desafio** que o autentique juntamente com o *ticket* enviado do **Servidor de Autenticação**.

Não é necessariamente o mais seguro em termos matemáticos. Os ataques que se conhecem são numa vertente de ataques do canal lateral, por outras palavras, são reaproveitadas as falhas no canal, pelos atacantes, ou os mesmos conhecem a plataforma de encriptação.

A cifra **TripleDES** foi escolhida pois parece ser o algoritmo mais aperfeiçoado para, em uma variedade de diferentes sistemas, em todos os tamanhos de bits testados, ser também extremamente seguro. Se tivermos controlo sobre os nossos sistemas, que aqui assumimos tê-lo, é uma forma “barata” de substituir o **DES** pelo **TripleDES**, uma variante do DES cifrada iterativamente três vezes.

Outras cifras altamente seguras e que gostaríamos de debater com o Professor são:

- **Twofish**;
- **Serpent**;
- **RC6**;

No entanto não nos sentimos muito confortáveis em ao não compreendê-las na íntegra o que sugere alguma ajuda da parte dos Professores sobre esta matéria.

As cifras que se seguem, pelo o que lemos, estão desatualizadas, o que nos levou a escolher as anteriores, mais recentes:

- **DES** (muito baixo rendimento);
- **AES** (bastante bom, mas falamos a tentativa de implementação);
- **Blowfish** (pelo que percebi há especificidade);
- **MARS** (não compreendi esta também);

Em suma, existem variadíssimas escolhas e grande parte delas foram usadas e criadas no âmbito de casos particulares sendo que apenas algumas, como as que falei sendo altamente seguras, poderão merecer uma análise e retrospectiva de utilização possível.

Na nossa implementação, as chaves são geradas através da class *SymKey* que melhor facilita a sua integração tanto na camada servidor como cliente.

O programa cliente consegue obter correctamente a chave da sessão e o ticket no entanto o Protocolo *Kerberos* encontra-se numa versão simplificada e por concluir.

Falta também enviar a mensagem MAC encriptada, algo que não é feito na nossa solução por escassez de tempo.

Store.B

Para este requisito, a parte dos quóruns que implementam a solução de replicação para as operações *createDoc* e *listDoc* não ficaram implementadas na integra e deve ser algo a ter em conta a melhorar.

Não foram também adaptados para camadas assíncronas, não tendo desta forma uma melhor eficiência.