Segurança

Segurança

- Os sistemas informáticos possuem muitas vulnerabilidades que são exploradas pelos mais diversos ataques
- Estas vulnerabilidades resultam de falhas técnicas e/ou do comportamento dos utilizadores
- Área muito vasta
- Muitas técnicas e mecanismos de defesa
- Existe uma área de especialização do MEIC dedicada a esta temática

Comunicação segura

Breve introdução a alguns mecanismos fundamentais

Comunicação segura

- Tipicamente, as entidades que pretendem estabelecer um canal seguro são designados por “Alice” e “Bob”
- Uma entidade que queira escutar ou perturbar a comunicação é frequentemente designado por “Trudy” (de “intruder”)
  - Um atacante que se limite a ver o fluxo de mensagens sem o alterar é geralmente designado por Eve (de “evesdropping”)
  - Um atacante malicioso pode também ser designado por “Mallory”
- Nestes slides uso simplesmente A, B e T.

Mecanismos básicos

- Cifra através do recurso a chaves simétricas
- Cifra através do recurso a chaves assimétricas
- Função de hash criptográfica
- *Nonce*

Cifra simétrica

- Existe uma chave secreta **Ks** que é **conhecida** por A e por B
- Existe um texto em claro (*plaintext*) **m**
- Existe uma função de cifra que produz um texto cifrado (*ciphertext*) a partir de **m** e de **Ks**, designado por **Ks(m)**
  - ***Ks(m)*** *representa o resultado de aplicar um algoritmo de cifra que usa* ***Ks*** *e* ***m*** *como input*
- Existe uma função para decifrar que usa a mesma chave **m=Ks(Ks(m))**
- Isto permite criar um canal seguro entre A e B desde que A e B conheçam previamente o segredo **Ks**
  - Por exemplo, A encontra B em pessoa e entrega-lhe a chave numa “pen USB”.
  - Ou a chave **Ks** é derivada de uma palavra passe previamente combinada entre A e B.

Algoritmos de cifra simétrica

- Exemplos:
  - **DES**: Data Encryption Standard
    - Chave simétrica de 56 bits
    - Inseguro: um ataque de força bruta descobre a chave em menos de 1 dia
  - **3DES**:
    - Usa 3 chaves distintas
  - **AES**: Advanced Encryption Standard
    - Usa chaves de 128, 192, ou 256 bits

- A segurança depende não só do algoritmo de cifra mas também do tamanho da chave e, implicitamente, do poder computacional do adversário.

Cifra assimétrica

- Cada entidade possui um par de chaves **(K+, K-)** designadas por chave pública e chave privada.
  - A chave pública **K+** é revelada às restantes entidades
  - A chave privada **K-** é mantida secreta
- Não é possível obter **K+** a partir de **K-**, nem **K-** a partir de **K+**.
- Estas chaves pode ser usadas para cifrar ou decifrar texto, tendo as seguintes propriedades:
  - **K+(K-(m))=m**
  - **K-(K+(m)=m**

Cifra assimétrica: dois usos

- Se a entidade A pretender enviar a B um texto **m** ao qual apenas B pode ter acesso, cifra **m** com a chave pública de B, **K_b+**
  - Apenas B pode decifrar esta mensagem com a sua chave privada
- Se a entidade A pretender provar a B que o texto **m** foi produzido por A, cifra **m** com a sua chave privada, **K_a-**.
  - B pode decifrar a mensagem recebida com a chave pública da A, **K_a+**. Se tiver sucesso, a mensagem foi produzida por A, pois apenas A conhece **K_a-**.

Cifras assimétricas

- Exemplos
  - Diffie-Helman
  - Rivest-Shamir-Adleman (**RSA**)
  - Outros...

Cifra mista

- A criptografia assimétrica é muito mais lenta que a criptografia simétrica
- Muitas vezes usamos cifra mista:
  - Cifra assimétrica usada para negociar uma chave simétrica, durante a fase criação do canal seguro.
  - A chave simétrica é posteriormente usada para cifrar a informação trocada no canal.
- Por exemplo, A cria uma chave simétrica **Ks**, cifra **Ks** com a chave pública de B (**Kb+**), e envia a chave cifrada **Kb+(Ks)** para B. Apenas B pode aceder a **Ks** através da sua chave privada **Kb-**.

Funções de hash criptográficas

- Função criptográfica que, dado um texto **m** cria uma síntese de tamanho fixo deste texto (designado por *digest*), **H(m)**
- Uma propriedade fundamental destas funções é que é computacionalmente inviável encontrar em tempo útil outro texto **m'** tal que **H(m') = H(m)**

Funções de hash criptográficas

- Exemplos:
  - **MD5**:
    - muito inseguro, muito fácil obter colisões
  - **SHA-1**:
    - inseguro, já foram encontradas colisões
  - **SHA-2 e SHA-3:**
    - considerados seguros atualmente
    - US standards
    - digests de 224 a 512 bits

Nonce

- Um ***nonce*** é uma palavra única que é usada para identificar uma troca de mensagens e que nunca é usada de novo em trocas posteriores.
- O objectivo da utilização de *nonces* é evitar o que designa por um "replay attack":
  - Um intruso captura todas as mensagens trocadas e, mesmo tendo apenas acesso ao conteúdo cifrado, re-executa posteriormente a mesma sequência.

Nonce

- Tipicamente baseados numa das seguintes técnicas (ou na sua combinação):
  - Estampilhas temporais: a segurança deste mecanismo está dependente da segurança do relógio da máquina.
  - Números de sequência monotonicamente crescentes: a segurança deste mecanismo depende da capacidade das entidades de memorizarem o último número usado.

Combinando os mecanismos básicos

- Combinando a criptografia simétrica, a criptografia assimétrica, as funções de hash criptográficas e os nonce é possível obter funcionalidades mais sofisticadas:
  - Assinaturas digitais
  - Infra-estruturas de chaves públicas e certificados
  - Troca de e-mail seguro
  - Canais seguros
  - Sistemas de autenticação com “single sign-on”

Assinaturas digitais

- Com chave simétrica
- Com chave assimétrica

Assinaturas digitais com chave simétrica

- A entidade A quer assinar um texto **m** de forma a que outra entidade B, com quem partilha um segredo **Ks**, possa confirmar que esse texto foi gerado por A e não foi alterado durante o percurso
- A entidade A cria uma versão estendida do texto **(m|Ks)**, adicionando ao texto **m** o segredo **Ks** partilhado com B e usa uma função de hash criptográfica para gerar o digest da versão extendida **Sm = H(m|Ks)**
- A envia a B o texto **m** mais a assinatura **Sm**
- B verifica se **Sm = H(m|Ks)**
- Este tipo de assinatura só pode ser usado entre duas entidades que partilham um segredo e é tipicamente designado por **Message Authentication Code (MAC)**

Assinaturas digitais com chaves assimétrica

- A entidade A quer assinar um texto **m** de forma a que qualquer outra entidade possa confirmar que esse texto foi gerado por A e não foi alterado
- A usa uma função de hash criptográfica para gerar o digest da mensagem **H(m)**
- A usa a sua chave privada **Ka-** para cifrar **H(m)**
- **Ka-(H(m))** serve como assinatura de m
- Qualquer entidade pode usar a chave pública **Ka+** de A para validar a assinatura

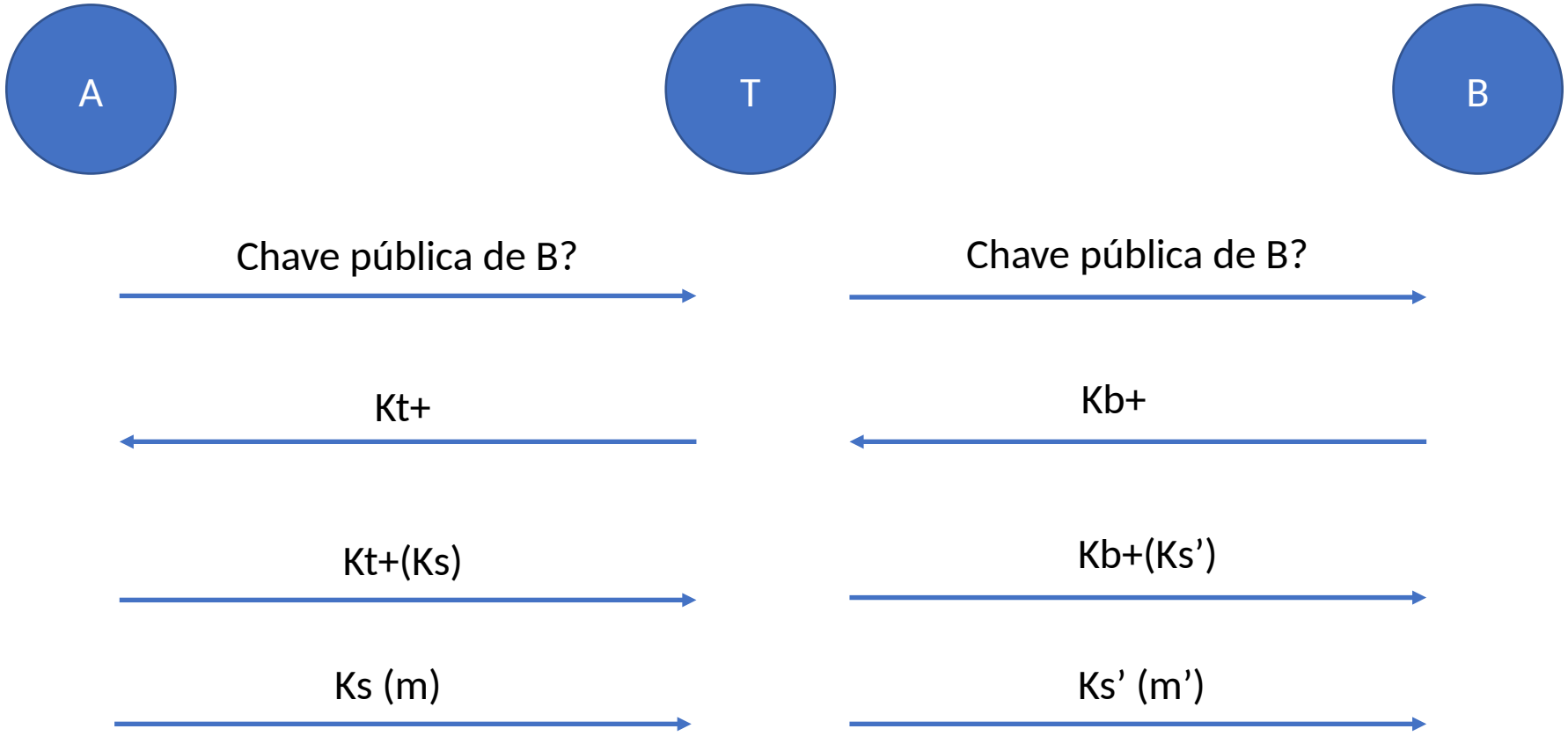
Motivação: o Bob precisa distribuir uma chave...


Alice,
se quiseres
enviar-me
mensagens
confidenciais,
cifra-as com AES
usando esta
chave secreta:
0xF43087AB
Bob
Alice,
se quiseres
enviar-me
mensagens
confidenciais,
cifra-as com RSA
usando a minha
chave pública:
0x12287AA0
Bob

Comecemos por um exemplo de movitação

- A pretende estabelecer uma ligação segura com B mas não conhece à partida a chave pública de B
- Um algoritmo (errado) poderia ser o seguinte:
  - A pede a B a sua chave pública
  - A cria uma chave simétrica e cifra a mesma com a chave pública de B
  - B obtém a chave simétrica e a partir desse momento A e B usam a chave simétrica para estabelecer o canal seguro

Man-in-the-middle attack



Man-in-the-middle attack

- Na execução anterior, o atacante T intercepta e modifica as mensagens enviadas entre A e B sem que A (ou B) se apercebam desse facto
- Isto acontece porque A não conhece previamente a chave de B nem tem meio de verificar se a chave pública que recebe é de facto de B

Man-in-the-middle attack: como prevenir?

- Pode ser evitado se A conhecer previamente a chave púbica de B
  - podendo, neste caso, enviar de imediato a chave simétrica a B, cifrada com a chave pública de B
  - Um adversário que intercepte a comunicação não consegue obter a chave

- Ou se existir um certificado digital que confirma que a chave que recebe pertence de facto a B
  - Neste caso, B pode enviar o certificado para A

Infra-estrutura de chave pública e certificados

Infra-estrutura de chave pública

- Conjunto de autoridades de certificação que emitem **certificados que associam uma chave pública a uma entidade**
- Um certificado é um documento que tem o identificador de uma entidade (por exemplo, um endereço web) e a sua chave pública, sendo assinado digitalmente pela autoridade de certificação
- Assume-se que **a chave pública da autoridade de certificação é conhecida**, de modo a que qualquer entidade do sistema possa validar um certificado emitido por esta

Certificados X509

Figure 11.12 X509 Certificate format

Subject	Distinguished Name, Public Key
Issuer	Distinguished Name, Signature
Period of validity	Not Before Date, Not After Date
Administrative information	Version, Serial Number
Extended information	

Infra-estrutura de chave pública

- Porque é que pode ser útil uma infra-estrutura com autoridades de certificação que certificam chaves públicas?
- Assuma que A pretende estabelecer uma ligação segura com B mas não conhece à partida a chave pública de B
- Um algoritmo (errado) poderia ser o seguinte:
  - A pede a B a sua chave pública
  - A cria uma chave simétrica e cifra a mesma com a chave pública de B
  - B obtém a chave simétrica e a partir desse momento A e B usam a chave simétrica para estabelecer o canal seguro

Troca de e-mail seguro

PGP

PGP: exemplo de troca de email seguro

- PGP: pretty good privacy
- Cada utilizador possui um par de chaves assimétricas
- O utilizador divulga a sua chave pública de diversas formas, de modo a que os outros utilizadores possam ter acesso à mesma e confiar que possuem a chave correcta
  - Pode entregar em mão a sua chave pública ao destinatário
  - Coloca na sua página web
  - Divulga por outros utilizadores de confiança que certificam a chave públicadesse utilizador usando as suas próprias chaves
  - Em versões mais recentes, também suporta infraestruturas de chave pública (hierarquias de CAs)

PGP

- Para assegurar que um email não é alterado:
  - Gera uma assinatura digital do conteúdo da mensagem
  - O destinatário pode assim verificar que a mensagem recebida

- Para assegurar que só o destinatário lê o email:
  - Cria uma chave simétrica para cifrar o conteúdo do email
  - Cifra a chave simétrica com a chave pública do destinatário
  - Envia o conteúdo cifrado conjuntamente com a chave cifrada
  - O destinatário é o único que consegue extrair a chave simétrica e decifrar o email

Canais seguros

SSL/TLS

TLS/SSL: Secure Sockets Layer

- Protocolo de segurança **implantado globalmente**
  - Suportado por quase todos os browsers e servidores web

- Oferece:
  - Confidencialidade
  - Integridade
  - Autenticação

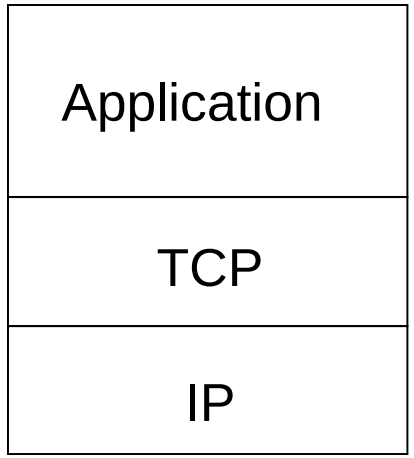
- Disponível para todas as aplicações TCP
  - Através da interface ***secure socket***

SSL/TLS: criação de um canal seguro

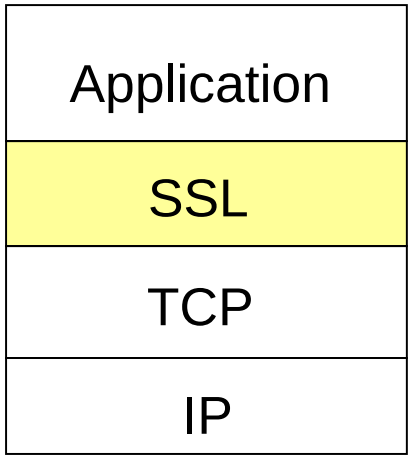
- SSL: Secure Sockets Layer
  - Entretanto substituído pelo TLS (Transport Layer Security)
- Seja por exemplo A um browser e B um servidor WWW. Este protocolo permite criar um canal seguro entre A e B.
- Nos slides seguintes, apresentamos uma versão simplificada do protocolo:
  - O protocolo real é mais complexo. Entre outras coisas, permite que A e B escolham o tipo de cifras que vão usar, dentro de um conjunto de alternativas suportado por ambos

TLS/SSL e TCP/IP

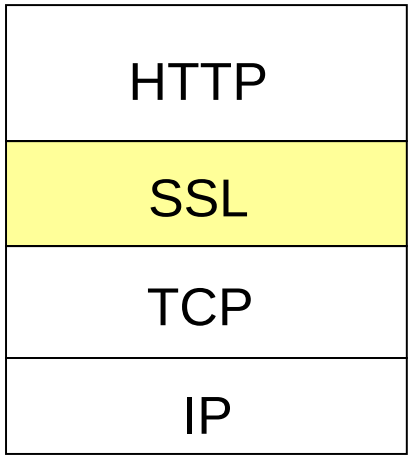
- O SSL providencia uma API para as aplicação
  - Há bibliotecas/classes C e Java disponíveis



Aplicação normal



Aplicação com SSL



HTTPS

TLS/SSL (versão simplificada)

- A envia a B um pedido, solicitando a chave pública de B
- A envia também a B um *nonce*
- B envia a A um certificado com a sua chave pública
- B envia também a A um *nonce*
- A verifica se o certificado é válido, abortando a ligação se tal não acontecer

TLS/SSL (versão simplificada), continuação

- A utiliza os *nonces* para gerar um segredo (designado por "**master secret**") que irá partilhar com B
  - Este segredo é único para esta ligação
- A envia a B o segredo, cifrando-o com a chave pública de B
- B fica a conhecer também o "master secret"
- A e B criam, de forma determinista, um conjunto de chaves simétricas que usam para concretizar o canal seguro
  - Estas chaves são geradas a partir do *master secret* e dos *nonce* anteriormente trocados

SSL

- São criadas 4 chaves simétricas:
  - Kc = chave usada para cifrar os dados enviados do cliente para o servidor
  - Mc = chave usada assinar, com um MAC, os dados enviados do cliente para o servidor
  - Ks = chave usada para cifrar os dados enviados do servidor para o cliente
  - Ms = chave usada para assinar, com um MAC, os dados enviados do servidor para o cliente

- A utilização de várias chaves torna a comunicação mais robusta

SSL

- Os dados enviados entre as duas entidades são agrupadas em blocos designados por “records”.
- Cada record é assinado pelo emissor com um MAC
  - Para gerar o MAC de um dado “record”, o emissor usa a sua chave M, o tipo do block, e um **número de sequência**
  - MAC = Hash(record|M|type|sequence_number)
  - Isto impede que um record seja alterado e que uma sequência de records seja re-ordenada sem que isso seja detectado.
- Para além disto, os record são cifrados antes de serem enviados, para assegurar a confidencialidade

Autenticação do cliente

Autenticação do Cliente

- Nos exemplo anteriores, quando A estabelece um canal com B, A garante que está a comunicar efectivamente com B
  - Mas a identidade de A é irrelevante para B

- No entanto, em muitos casos, A também precisa de se autenticar face a B
  - Caso B forneça serviço apenas a clientes autorizados para o efeito

Possível solução?

- Solução: A possuir também um par de chaves assimétricas devidamente certificadas
- Problema: pouco prático
  - A pessoa teria que manter o seu certificado, com a sua chave pública, em todos os dispositivos que usasse
  - Certificado teria que estar protegido para evitar a sua utilização indevida caso o dispositivo fosse roubado

Autenticação do Cliente

- Em muitos casos a autenticação é feita recorrendo a um segredo que é partilhado entre A e B
  - Tipicamente uma palavra passe
- Este tipo de autenticação pode ser usado em sistemas que usam canais seguros baseados em chaves siméticas ou em chaves assimétricas

Palavra passe: desafio

- Problemático se usamos muitos serviços (por exemplo, aplicações web)
- Má ideia deixar a gestão ao utilizador:
  - Usam senhas fracas
  - Tomam nota e deixam à vista
  - Usam as mesmas em várias aplicações


Hacked French network exposed its own passwords during TV interview

Post-it note on wall revealed network's passwords for YouTube, Instagram.

SAM MACHKOVECH - 4/10/2015, 2:37 AM


Forbes

EDITORS' PICK | 98,759 views | Apr 28, 2020, 06:46am EDT

Zoom Gets Stuffed: Here's How Hackers Got Hold Of 500,000 Passwords

Davey Winder Senior Contributor

Cybersecurity

I report and analyse breaking cybersecurity and privacy stories

Solução: Single Sign-On (SSO)

- Utilizador pode autenticar-se em apenas um **serviço de autenticação (SSO)**
  - Autenticação é baseada num único conjunto de credenciais (p.ex., apenas uma senha)

- Várias aplicações **partilham** esse serviço de autenticação
  - O cliente não precisa de partilhar segredos com todos os serviços; apenas tem de partilhar um segredo com o servidor de autenticação.
  - Os serviços não precisam de partilhar segredos com todos os clientes; apenas com o servidor de autenticação.

Needham-Schroeder

Operating Systems

R. Stockton Gaines Editor

Using Encryption for Authentication in Large Networks of Computers

Roger M. Needham and
Michael D. Schroeder
Xerox Palo Alto Research Center

Use of encryption to achieve authenticated communication in computer networks is discussed.

authentication in such a network that are integrated with the allied subject of naming. There is minimal reliance on network-wide services; in particular there is no reliance on a single network clock or a single network name management authority.

Three functions are discussed:

(1) Establishment of authenticated interactive communication between two principals on different machines. By interactive communication we mean a series of messages in either direction, typically each in response to a previous one.

(2) Authenticated one-way communication, such as is found in mail systems, where it is impossible to require protocol exchanges between the sender and the recipient while sending an item, since there can be no guarantee that sender and recipient are simultaneously available.

(3) Signed communication, in which the origin of a communication and the integrity of the content can be

Protocolo de Needham–Schroeder

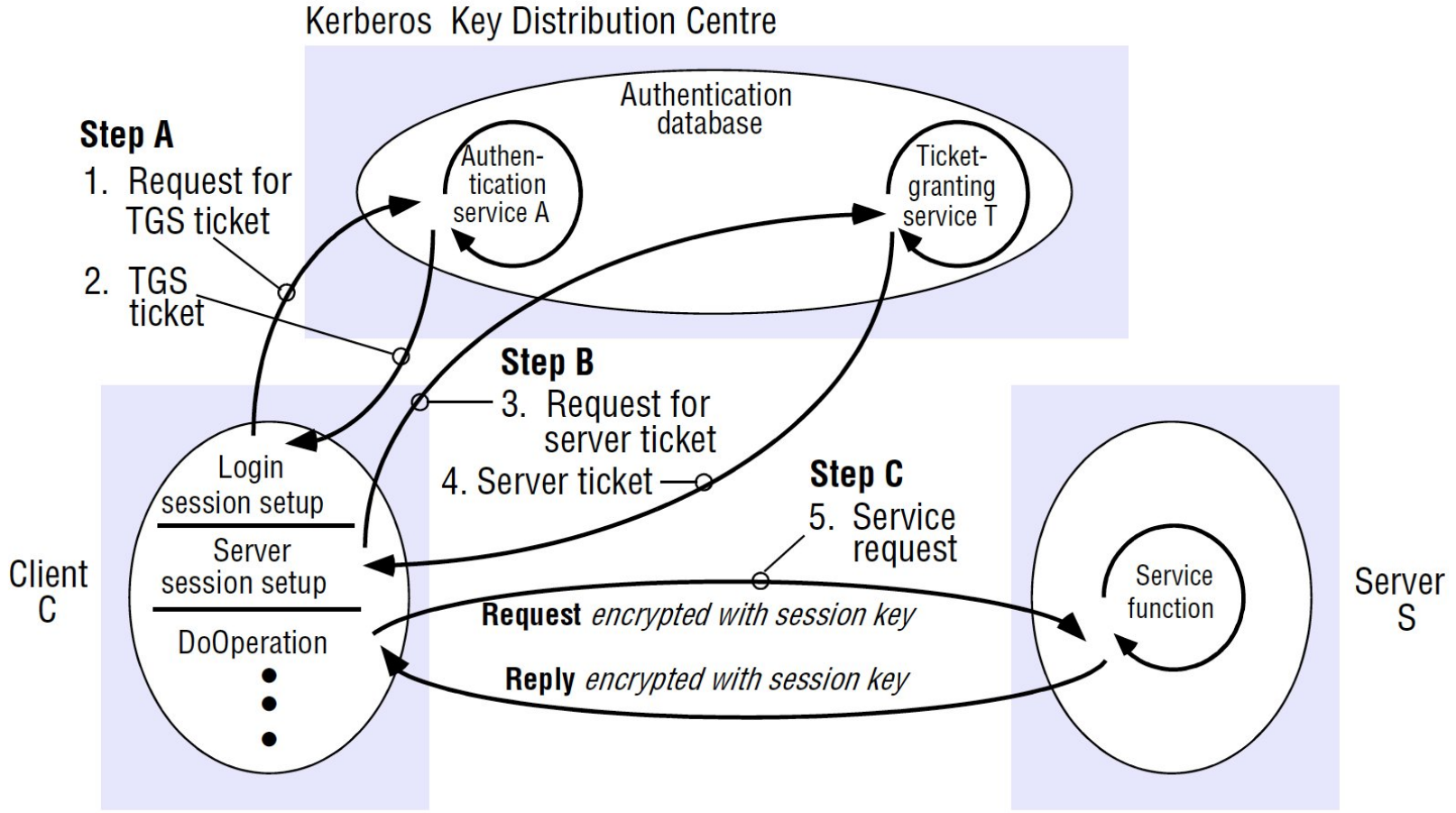
Retirado da página 521 do livro

Header	*Message*	*Notes*
1. A → S:	A, B, N_A	A requests S to supply a key for communication with B.
2. S → A:	$\{N_A, B, K_{AB}, \{K_{AB}, A\}_{K_B}\}_{K_A}$	S returns a message encrypted in A's secret key, containing a newly generated key K_{AB}, and a 'ticket' encrypted in B's secret key. The nonce N_A demonstrates that the message was sent in response to the preceding one. A believes that S sent the message because only S knows A's secret key.
3. A → B:	$\{K_{AB}, A\}_{K_B}$	A sends the ticket to B.
4. B → A:	$\{N_B\}_{K_{AB}}$	B decrypts the ticket and uses the new key, K_{AB}, to encrypt another nonce, N_B.
5. A → B:	$\{N_B - 1\}_{K_{AB}}$	A demonstrates to B that it was the sender of the previous message by returning an agreed transformation of N_B.

Nota: como descrito no livro, este algoritmo não verifica se a mensagem 3 não está ser replayed: um algorimo mais robusto passa por ter um timestamp físico no ticket

Kerberos: solução com chave simétrica

- Sistema de autenticação, originalmente desenvolvido no MIT, baseado no protocolo de **Needham–Schroeder**



Single sign-on com chaves assimétricas

- O mesmo princípio pode ser usado em sistemas com chaves assimétricas
- Isto é bastante usado na prática hoje em dia: por exemplo, quando um utilizador faz login num dado serviço usando as mesmas credenciais que usa nos serviços da Google ou do Facebook

Bibliografia recomendada

- [Coulouris et al]
  - Secções 11.1-11.6