

# Criptografia

## Módulo III – Aplicações Correntes da Criptografia

M. B. Barbosa

mbb@di.uminho.pt

Departamento de Informática  
Universidade do Minho

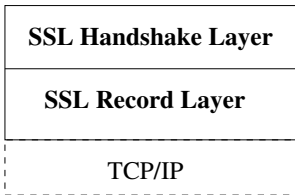
2006/2007

# Introdução

- A Secure Sockets Layer está para o TCP como o IPsec está para o IP. É um *upgrade* da camada de transporte para incluir segurança nas comunicações.
- O SSL foi desenvolvido pela Netscape, e a sua versão 3 foi adoptada pela IETF sob a designação **Transport Layer Security** (TLS). O TLS está definido no RFC2246.
- Os serviços fornecidos pelo SSL incluem:
  - Confidencialidade baseada em cifras simétricas.
  - Autenticação baseada em criptografia de chave pública.
  - Integridade baseada em Message Authentication Codes.

# Estrutura do SSL

- O SSL está estruturado em duas sub-camadas:



- A **Handshake Layer** permite a autenticação mútua entre clientes e servidores, e a negociação de algoritmos e chaves criptográficas antes de se iniciar a troca de dados através da Record Layer.
- A **Record Layer** encapsula a informação correspondente às camadas superiores.

# Sessões SSL

- O funcionamento do SLL baseia-se em **sessões** estabelecidas entre um **cliente** e um **servidor**.
- Cada sessão SSL pode incluir várias ligações seguras, e cada nó pode manter diversas sessões SSL. Durante o seu estabelecimento e operação, as sessões e ligações SSL atravessam uma sequência de estados.
- Cliente e Servidor mantêm uma máquina de estados para cada sessão e ligação. A camada de Handshake sincroniza os estados no cliente e no servidor.
- As transições entre estados efectuam-se em duas fases:
  - Primeiro constrói-se/negoceia-se um **pending state**.
  - Depois substitui-se o **operating state** pelo pending state.

## Estado de uma Sessão SSL

- **Session identifier** Uma sequência arbitrária de bytes escolhida pelo servidor para identificar a sessão.
- **X509 certificate of the peer** Certificado do interlocutor.
- **Compression method** Algoritmo de compressão da informação antes de ser cifrada.
- **Cipher spec** Algoritmo de cifra simétrica (e algoritmo de hash criptográfico para utilização em MACs).
- **Master secret** Chave secreta partilhada por Cliente e Servidor e da qual são derivados todos os segredos utilizados na sessão (chaves e IVs).
- **Is resumable** Indica se a sessão pode ser utilizada para novas ligações.

## Estado de uma Ligação SSL

- **Server/Client random** Números aleatórios escolhidos por Cliente e Servidor para estabelecimento da ligação.
- **Server/Client write MAC secret** Chaves utilizadas por Cliente e Servidor para efectuar MACs sobre dados transmitidos.
- **Server/Client write key** Chaves utilizadas por Cliente e Servidor para cifrar dados transmitidos.
- **Initialization vectors** Vectores de inicialização (IV) para os modos de cifra simétrica que os utilizam.
- **Sequence numbers** Contadores sequenciais das mensagens enviadas e recebidas.

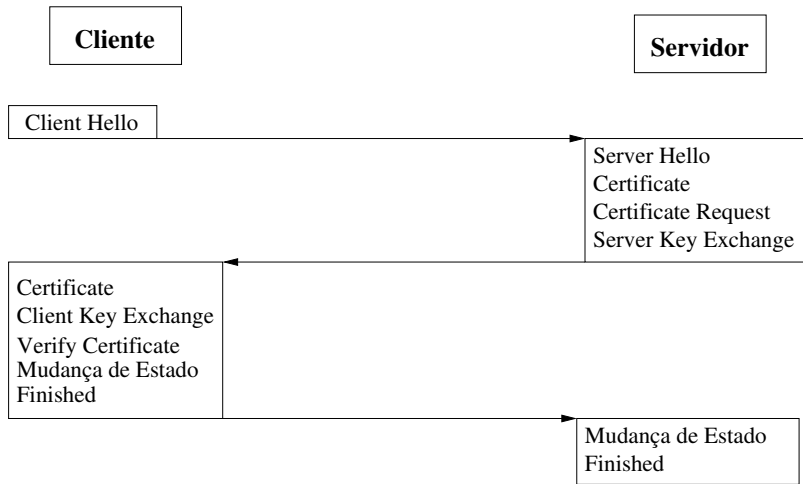
# Record Layer

- Recebe informação arbitrária das camadas superiores, em blocos de dados de tamanho variável.
- Os dados são **fragmentados** em blocos com um máximo de  $2^{14}$  bytes denominados **SSL Plaintext**.
- Os blocos SSL Plaintext são comprimidos com o algoritmo da sessão, originando blocos **SSL Compressed**.
- Os dados SSL Compressed são protegidos com a cifra e algoritmo de MAC definidos na CipherSpec da sessão (o MAC é calculado antes da cifragem). O resultado é um bloco do tipo **SSL Ciphertext**.
- Estes blocos são trocados entre Cliente e Servidor que têm de reverter estas transformações para obter o texto limpo.

# Handshake Layer

- Os parâmetros de sessão e ligação utilizados pela Record Layer são estabelecidos pela Handshake Layer.
- As mensagens da Handshake Layer viajam elas próprias sob o controlo da Record Layer. Inicialmente não há qualquer protecção: é utilizada uma *cipher spec* nula até que a primeira negociação seja concluída.
- Uma negociação é iniciada pelo Cliente com uma mensagem **Client Hello**. O Servidor deve responder com uma mensagem equivalente. Ficam acordados:
  - A versão do protocolo SSL a utilizar
  - O identificador da sessão e os números aleatórios.
  - Os algoritmos criptográficos a utilizar (os mais fortes dos suportados).
  - O algoritmo de compressão a utilizar





- Caso seja utilizada autenticação do Servidor, este envia o seu certificado X.509 ao Cliente, que o valida. Além da validação habitual, o Cliente assegura-se de que o nome de domínio do Servidor, indicado no certificado, está correcto.
- Parâmetros do Servidor específicos para acordo de chaves podem também ser enviados nesta fase (**Server Key Exchange**), se o seu certificado não incluir informação suficiente para esta funcionalidade.
- Caso o Servidor autentique o Cliente, solicita o certificado correspondente (**Certificate Request**). Este pedido inclui um desafio para ser utilizado na autenticação do cliente.
- O Servidor termina esta fase da negociação enviando uma mensagem **Server Hello Done**.

- Caso tenha recebido um pedido de certificado, o Cliente tem de envia-lo ou a negociação falha.
- Conjuntamente com o certificado o Cliente tem de enviar uma assinatura digital do desafio que recebeu, comprovando assim a posse da chave privada associada ao certificado.
- Finalmente, o Cliente envia os seus parâmetros para acordo de chaves (**Client Key Exchange**), altera o seu estado de sessão, e envia uma primeira mensagem cifrada que indica o seu estado de prontidão (**finished**).
- O Servidor efectua o mesmo procedimento e a negociação termina tendo sido acordado o Master Secret da sessão.

- A autenticação do servidor fica implícita pelo sucesso da comunicação cifrada nas mensagens **finished**, ou não?
- De facto, o servidor só fica autenticado se o protocolo de acordo de chaves implicar a utilização da sua chave privada.
- Isto acontece sempre:
  - No protocolo **RSAKeyExchange** o cliente gera um segredo e cifra-o com a chave pública do servidor. Para gerar o Master Secret, o servidor tem de decifrar este segredo com a sua chave privada.
  - Nos outros protocolos, os parâmetros públicos do servidor utilizados no protocolo de acordo de chave são assinados com a sua chave privada.

# Segurança

- A versão 3 do SSL é considerada um sistema seguro. É uma evolução em relação às versões anteriores, colmatando falhas de segurança importantes.
- Um dos problemas mais conhecidos na versão 2 do SSL era a vulnerabilidade ao ataque “ciphersuit rollback”:
  - Um intruso podia editar as mensagens de **hello** trocadas entre Cliente e Servidor de forma a que ambos pensassem que o outro apenas conseguia funcionar com um nível de segurança reduzido.
  - O resto da negociação decorria sem alterações e estabelecia-se uma ligação com um nível de segurança reduzido, mais vulnerável a ataques por parte do intruso.
- Este ataque era possível porque as mensagens de handshake não eram autenticadas!

- A versão 3 do SSL resolveu este problema obrigando a que todas as mensagens de handshake fossem utilizadas para gerar o valor cifrado nas mensagens **finished**.
- “Change cipher spec dropping” é outro ataque possível sobre uma implementação pouco cuidada:
  - Quando a sessão a ser negociada inclui apenas autenticação, i.e. não inclui cifragem, é possível eliminar das mensagens **finished** a informação de autenticação.
  - Interceptando as mensagens **change cipher spec**, impede-se a activação da autenticação. Fornecendo a Cliente e Servidor mensagens **finished** alteradas, estabelece-se uma sessão sem protecção.
- A solução para este ataque consiste em exigir uma mensagem de **change cipher spec** antes de uma mensagem **finished** nestas situações.

## Ficha Técnica

- **Cifras Simétricas:** DES, 3-DES, RC4
- **Algoritmos de Compressão:** ZLIB
- **Funções de Hash Criptográficas:** SHA-1, MD5
- **Cifras Assimétricas:** RSA
- **Assinaturas Digitais:** RSA, DSA
- **Acordo de Chaves:** Fortezza, Diffie-Hellmann, distribuição RSA.

# Introdução

- O SSH é um protocolo para o estabelecimento de serviços de *shell* seguros, nomeadamente de login remoto, sobre uma rede não segura.
- Foi desenvolvido para substituir os serviços **rlogin**, **rsh**, etc, incluídos nas *shell* Unix/Linux, e que não são satisfatórios ao nível da segurança.
- O SSH funciona numa filosofia Cliente/Servidor:
  - o Servidor é tipicamente uma máquina Unix/Linux que aceita o estabelecimento de sessões de *shell* seguras através da porta 22.
  - o Cliente pode ser qualquer tipo de máquina que corra software Cliente compatível com o SSH.
- O SSH foi normalizado pela IETF para utilização na Internet (Internet Drafts) e o seu uso é generalizado.

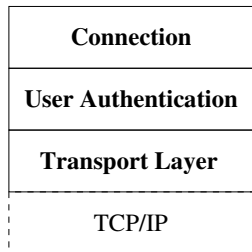


# Estrutura do SSH

A **Transport Layer** oferece autenticação do servidor, confidencialidade e integridade sobre uma rede insegura.

A **User Authentication Layer** oferece serviços de validação de utilizadores perante um servidor.

A **Connection Layer** oferece a multiplexagem de um canal seguro a por vários canais lógicos.



# Políticas de Segurança

- Num servidor que utilize o SSH têm de ser definidas as seguintes políticas de segurança:
  - Quais os algoritmos de cifragem, compressão e autenticação utilizáveis para envio e recepção de dados; e, desses algoritmos, quais são as soluções preferenciais.
  - Quais os algoritmos de Chave Pública utilizados para acordo de chaves e autenticação do Servidor.
  - Que tipo de autenticação é requerida aos utilizadores que acedem a partir de um determinado Cliente.
  - Quais as operações que um utilizador pode efectuar, dependendo da sessão que estabeleceu.
- As implementações SSH permitem geralmente definir estas políticas, através da manipulação de parâmetros de configuração mais ou menos uniformes.

## Transport Layer

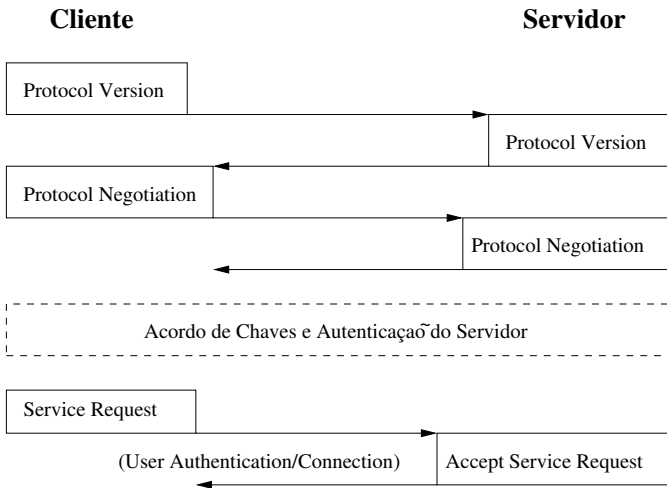
- A camada de transporte do SSH utiliza a infraestrutura de rede subjacente para transferir *streams* de bytes, geralmente com informação puramente binária.
- Exceções são as mensagens de gestão da própria camada de transporte, que são simplesmente *strings* de caracteres ASCII.
- Os pacotes trocados ao nível desta camada têm a seguinte estrutura:

Packet Length	Padding Length	Payload	Random Padding	MAC
---------------	----------------	---------	----------------	-----

- O processamento aplicado ao pacote segue a sequência habitual: compressão (**payload**), autenticação (MAC), **padding** e cifragem.

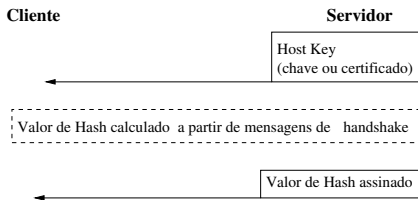
- O padding serve, não só para dar ao pacote um tamanho adequado para a cifragem por blocos, mas também para esconder o verdadeiro tamanho dos dados.
- O MAC é calculado sobre todos os bytes do pacote (antes da cifragem) e um número de sequência de pacote, utilizando um segredo pré-negociado. O número do pacote não é incluído no próprio pacote.
- O estabelecimento de um canal seguro começa pela negociação dos parâmetros do protocolo.
- As primeiras mensagens trocadas por Cliente e Servidor permitem escolher a versão do SSH utilizada: a versão mais recente suportada pelas duas máquinas (de preferência a última – actualmente a V2).

- Estabelecida a versão, Cliente e Servidor trocam mensagens em que indicam os algoritmos que implementam, e aqueles que são de utilização preferencial.
- É escolhido um algoritmo para cada funcionalidade, procurando na lista de algoritmos implementados pelo Cliente o primeiro que também é suportado pelo Servidor.
- Antes de se iniciar a comunicação segura Cliente e Servidor executam o protocolo de acordo de chaves negociado, protocolo esse que inclui uma componente de autenticação do servidor.
- Este processo termina com o estabelecimento de uma chave secreta partilhada e de um identificador de sessão (gerado a partir de um valor de hash).



- Os protocolos de acordo de chaves utilizados pela camada de transporte do SSH incluem uma componente de identificação do Servidor.
- Cada Servidor tem uma **Host Key**: um par de chaves que é utilizado na fase de acordo de chaves para autenticação do Servidor.
- Daí que, para haver segurança no estabelecimento de uma sessão:
  - ou o Cliente tem conhecimento prévio da chave pública do Servidor (distribuição manual da chave pública),
  - ou recorre-se a um esquema de certificação X.509, no qual o Cliente apenas tem de conhecer e confiar numa CA que permita validar os certificados dos Servidores.
- Ambos os modos de funcionamento são permitidos no SSH.

- Cada algoritmo de acordo de chaves especifica uma função de hash criptográfica que é utilizada, entre outras coisas, na geração de mensagens de autenticação.
- Como passo intermédio do protocolo de acordo de chaves temos:



- A autenticação do Servidor assegura também o Cliente de que as mensagens de handshake que recebeu provieram do Servidor.

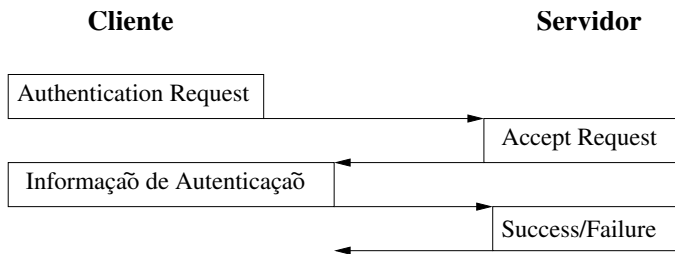


- O funcionamento da camada de transporte baseia-se em seis segredos derivados da chave secreta acordada por Cliente e Servidor:
  - IV Cliente-Servidor =  $HASH(K|H|A|sessionid)$
  - IV Servidor-Cliente =  $HASH(K|H|B|sessionid)$
  - Cifragem Cliente-Servidor =  $HASH(K|H|C|sessionid)$
  - Cifragem Servidor-Cliente =  $HASH(K|H|D|sessionid)$
  - MAC Cliente-Servidor =  $HASH(K|H|E|sessionid)$
  - MAC Servidor-Client =  $HASH(K|H|F|sessionid)$
- Em que *HASH* representa a função de hash associada ao protocolo de acordo de chaves, *H* é o valor de hash acordado nesse protocolo e *sessionid* é o valor de hash acordado no primeiro acordo de chaves.

## User Authentication Layer

- Quando o Cliente invoca com sucesso os serviços desta camada, ao nível da camada de transporte, pode proceder a um pedido de autenticação de um utilizador.
- Um pedido de autenticação enviado pelo Cliente inclui os seguintes parâmetros:
  - **User Name** Identificação do utilizador a autenticar.
  - **Service Name** O serviço a que pretende aceder.
  - **Authentication Method** O método de autenticação a utilizar.
- Caso o Servidor aceite o pedido, o que depende do método de autenticação solicitado (bem como do utilizador e do serviço indicados), seguem-se mensagens específicas do processo de autenticação.

- Métodos de autenticação:
  - **Chave Pública** O Cliente envia a chave pública do utilizador ao Servidor, juntamente com uma assinatura do identificador de sessão da camada de transporte. Também aqui a confiança na chave pública do cliente pode ser estabelecida de forma manual ou através de um esquema de certificação.
  - **Password** O Cliente envia simplesmente uma password que permite validar o utilizador no Servidor.
  - **Host Based** O Servidor não autentica o utilizador, mas sim a máquina Cliente, com base numa chave pública. A validação depende não só do *User Name* do utilizador no Servidor, mas também do seu *User Name* no Cliente. Este método de autenticação, apesar de conveniente, não é recomendado.



- Caso a autenticação falhe, o Servidor indica ao Cliente se o processo pode continuar, e com que métodos de autenticação.
- Caso a autenticação tenha sucesso, essa informação fica disponível para a camada superior (*Connection*) para que possam ser estabelecidas ligações de *shell*.

## Connection Layer

- Os serviços desta camada utilizam a confidencialidade e autenticação fornecida pelas camadas inferiores para oferecer os seguintes serviços:
  - login remoto.
  - execução remota de comandos.
  - reencaminhamento de portas TCP/IP
  - reencaminhamento de ligações X11
- Para uma determinada sessão, esta camada permite estabelecer múltiplos canais de comunicação paralelos, através dos quais podem ser invocados serviços independentes.
- Os detalhes do funcionamento desta camada não são relevantes para a segurança do sistema e ficam, portanto, fora do âmbito desta disciplina.

# Ficha Técnica

- **Cifras Simétricas:** 3DES, Blowfish, Twofish, AES, Serpent, IDEA, CAST (as cifras por blocos funcionam em modo CBC).
- **Algoritmos de Compressão:** ZLIB
- **Funções de Hash Criptográficas:** SHA-1, MD5
- **Message Authentication Codes:** HMAC
- **Cifras Assimétricas:** RSA
- **Assinaturas Digitais:** DSA
- **Acordo de Chaves:** Diffie-Hellmann

# Introdução

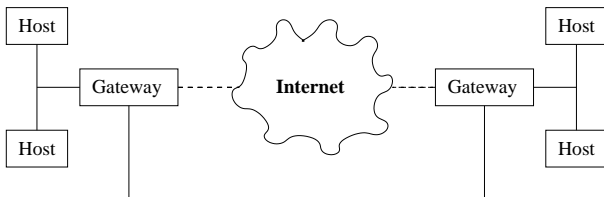
- O IP (Internet Protocol) está ao nível da camada de rede do Modelo OSI. Fornece serviços de encaminhamento de pacotes através de redes heterogéneas.
- A maior parte das infraestruturas de comunicação na Internet são baseadas neste protocolo, conjuntamente com o TCP (TCP/IP).
- O IPsec fornece o mesmo conjunto de serviços, mas inclui funcionalidade extra ao nível da segurança.
- Estes serviços são oferecidos ao nível da camada de rede, oferecendo protecção não só a esse nível, mas também a todas as camadas superiores.
- O IPsec está definido nas especificações RFC2401, e seguintes, da IETF.

- O IPsec oferece os seguintes serviços seguros ao nível da camada de rede:
  - Controlo de acessos
  - Integridade ao nível do pacote
  - Autenticação da origem de dados
  - Protecção contra pacotes repetidos
  - Confidencialidade
  - Confidencialidade de parte do tráfego
- Estes serviços permitem proteger ligações de rede entre nós IP, entre gateways seguras, ou entre um nó IP e uma gateway segura.
- Não substituem os serviços IP. São módulos adicionais que podem ser implementados e utilizados consoante o contexto e as necessidades das aplicações.



# Revisão do Protocolo IP

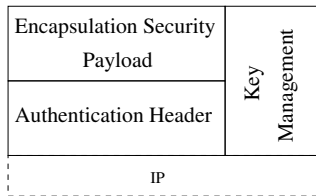
- Funcionamento:



- Dentro de uma rede local, cada nó constrói um pacote IP incluindo os endereços de origem e destino no cabeçalho.
- A comunicação com redes remotas é feita passando os pacotes a uma gateway: o endereço da gateway encapsula o verdadeiro.
- A gateway substitui o encapsulamento reencaminhando o pacote. A gateway da rede remota retira o encapsulamento.

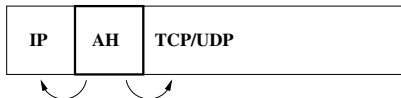
# Estrutura do IPsec

- O IPsec está estruturado em duas sub-camadas:



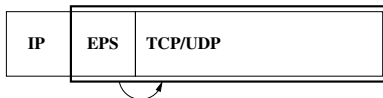
- As duas sub-camadas são apoiadas por procedimentos e protocolos de gestão de chaves criptográficas (manuais ou automáticos).
- Os protocolos estão especificados de forma a serem independentes de algoritmos criptográficos. No entanto, alguns destes algoritmos estão pré-definidos.

# IP Authentication Header



- A sub-camada **IP Authentication Header** (AH) inclui serviços de integridade ao nível dos pacotes, autenticação da origem de dados e, opcionalmente, protecção contra a repetição de pacotes.
- Recorre-se ao AH quando se pretende autenticação da informação correspondente à camada de rede (cabeçalho IP), ou quando a confidencialidade não é necessária (ou permitida).

# Encapsulating Security Payload



- A sub-camada **Encapsulating Security Payload (ESP)** fornece confidencialidade (cifragem) da totalidade ou de apenas parte do tráfego.
- Como opções, o ESP oferece autenticação, verificação de integridade, e protecção contra pacotes repetidos. No entanto, esta funcionalidade abrange apenas informação correspondente às camadas de transporte e superiores (não trata o cabeçalho IP).
- No entanto pode ser, geralmente é, usado isoladamente.

# Modos de Funcionamento

- Consoante o tipo de nós envolvidos, pode funcionar em:
  - **Transport Mode** – Sobre os cabeçalhos IP originais.
    - Apenas serve para ligações host-host
    - Com ESP não há protecção dos cabeçalhos IP (interferiria com a infraestrutura IP).
    - Com AH, há uma protecção parcial desses cabeçalhos.
  - **Tunnel Mode** – Sobre cabeçalhos IP encapsulados.
    - Corresponde a um túnel IP (caminho virtual entre nós).
    - Típicamente utilizado em ligações com/entre gateways
    - A protecção alcança todo o pacote original.
    - Diferenças entre AH e ESP mantêm-se, mas apenas para o cabeçalho exterior.
    - Para as camadas superiores, estas ligações aparecem como interfaces de rede adicionais.

# Security Associations e SADs

- A gestão do IPsec baseia-se em **Security Associations**.
- Uma SA é uma ligação simplex identificada por três parâmetros incorporados nos cabeçalhos IPsec:
  - **IP Destination** Define o endereço de destino dos pacotes.
  - **IPsec Protocol** O protocolo (AH ou ESP) utilizado pela SA.
  - **Security Parameter Index (SPI)** Número de 32 bits que distingue SAs do mesmo tipo.
- Associados a este identificador estão todos os parâmetros de funcionamento necessários para a codificação e descodificação dos pacotes enviados através da ligação segura representada pela SA.
- Cada nó IPsec regista esta informação numa Security Association Database (SAD).

## Security Policy Database

- O IPsec funciona num nó (máquina ou gateway) de uma rede IP. Cada pacote que passa num nó IPsec pode ser tratado de acordo com uma de três políticas:
  - proteger o pacote com segurança IPsec,
  - enviar o pacote com IP simples, ou
  - descartar o pacote (no caso de gateways que limitam o tráfego entre redes).
- A política aplicada a cada pacote específico está armazenada numa Security Policy Database (SPD) que é gerida por um utilizador ou administrador do sistema.
- A pesquisa na tabela baseia-se em informação contida nos cabeçalhos de rede e transporte do pacote em questão: endereços IP de origem e destino, protocolo e portas de transporte (TCP/UDP).

# Configuração do IPsec

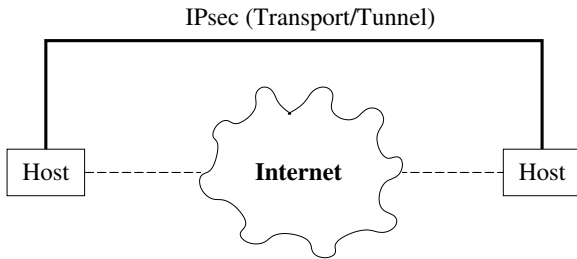
- O IPsec permite grande flexibilidade na configuração:
  - dos serviços seguros que são utilizados e em que combinações;
  - da granularidade com que um determinado serviço é aplicado; e
  - dos algoritmos criptográficos utilizados em cada serviços.
- A configuração da granularidade de um serviço consiste em definir com que detalhe se distinguem os pacotes. Por exemplo:
  - podem-se cifrar todos os pacotes entre duas gateways, criando um canal seguro para todas as máquinas que utilizem essa ligação, ou ...
  - podem cifrar-se apenas os pacotes que circulam entre duas máquinas protegendo apenas essa ligação.



# Gestão de Chaves e Algoritmos Criptográficos

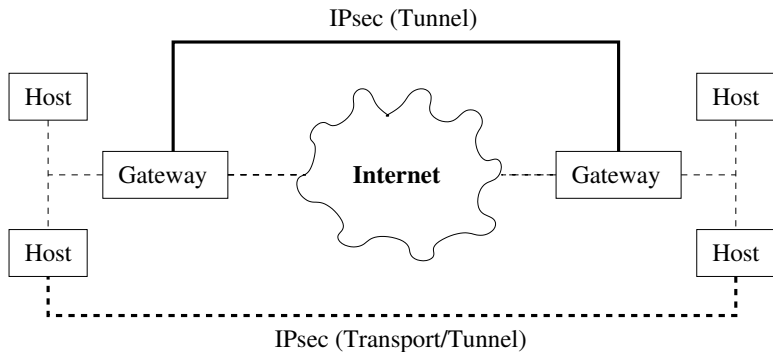
- O IPsec permite o funcionamento baseado em técnicas manuais de gestão de chaves.
- No entanto, este tipo de operação não é adequado em sistemas com muitos nós, onde um sistema de criação dinâmica de SAs é preferível.
- Este tipo de funcionamento implica um sistema automático de gestão de algoritmos e chaves criptográficos.
- O sistema recomendado é o **Internet Key Exchange** (IKE), especificado nos RFCs 2407, 2408, 2409 e 2412.
- O IKE permite a construção automática de SAs com negociação de parâmetros de comunicação e segurança, autenticação e protocolos de geração e acordo de chaves.

# Ligações Host-to-Host



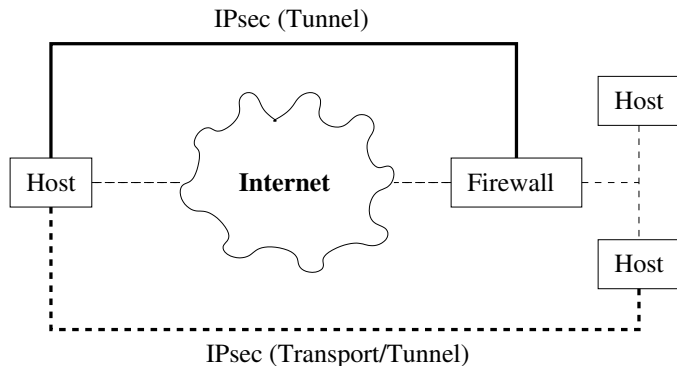
- Os pacotes trocados entre as duas máquinas são protegidos com IPsec, com base em Transport SAs.
- Pode ser utilizado o modo Tunnel, mas isso é redundante: o cabeçalho externo terá os mesmos endereços que o interno.

# Virtual Private Networks (VPN)



- As gateways tornam a troca de pacotes entre redes totalmente transparente. Todo o tráfego é protegido.

# Ligações remotas seguras (Road Warrior)



- Depois de a máquina externa se validar perante a firewall, passa a funcionar como se estivesse dentro da rede.

# Críticas ao IPsec

- O IPsec é muito criticado pela sua complexidade (o seu desenvolvimento esteve a cargo de um comité).
- Os principais problemas apontados são:
  - A complexidade do IPsec dificulta a sua implementação e a sua aplicação. Além disso torna virtualmente impossível uma avaliação cabal da sua segurança.
  - A documentação é muito dispersa e difícil de ler.
  - Existe demasiada flexibilidade e funcionalidades aparentemente redundantes. Por exemplo:
    - Porquê dois modos de funcionamento (transporte e túnel): se o que se pretende é mais segurança, porque não utilizar apenas o túnel?
    - porquê dois protocolos (AH e ESP) quando seria muito mais simples especificar um protocolo único, mais simples e mais consistente?

- Um tão elevado grau de flexibilidade torna a configuração demasiado complexa para um utilizador comum, o que pode levar a instalações com graves falhas de segurança.
- Uma questão que levantou também alguma discussão foi o facto de a autenticação no IPsec ser feita depois da cifragem. Isto contraria o chamado “princípio de Horton”: *deve autenticar-se a informação necessária e suficiente para a interpretação de uma mensagem.*
- O sistema de gestão de chaves recomendado (IKE) também contribui para as críticas ao IPsec: é demasiado genérico, as suas especificações estão mal escritas, etc.
- Conclusão: a segurança de um sistema IPsec depende demasiado das escolhas de implementação e configuração.

# Ficha Técnica

- Algoritmos obrigatórios:
  - **Cifras Simétricas:** DES (CBC)
  - **Funções de Hash Criptográficas:** SHA-1, MD-5
  - **Message Authentication Codes:** HMAC
- Os protocolos são especificados de forma independente dos algoritmos, e há uma grande flexibilidade na utilização de outros algoritmos criptográficos.
- Está prevista por exemplo a utilização de diversas cifras simétricas (3-DES, IDEA, Blowfish), do RSA, do DSA, etc.

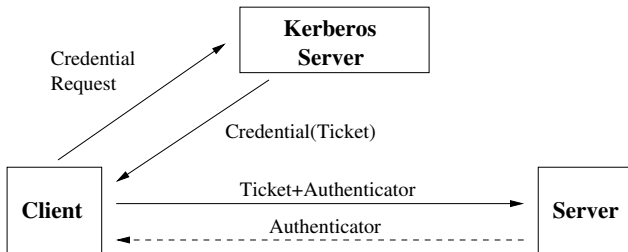
# Introdução

- O Kerberos é um protocolo para identificação de **principals** (agentes: utilizadores, aplicações, serviços) sobre uma rede insegura, em que os pacotes podem ser lidos, modificados e inseridos por intrusos.
- O sistema não baseia a sua segurança nos endereços de rede das máquinas envolvidos, não exigindo segurança física em todas as máquinas, e não impõe restrições ao sistema operativo.
- Actualmente na versão 5, o Kerberos é utilizado na Internet com base em Internet Standards e RFCs publicados pela IETF.
- Os serviços Kerberos são oferecidos às aplicações através de uma API definida no RFC1964.



- O Kerberos baseia-se em Criptografia Simétrica e num sistema de autenticação por um agente de confiança, com pré-distribuição de chaves.
- É atribuída uma chave secreta a todas os agentes que utilizam o sistema (para os utilizadores as chaves são derivadas de passwords).
- O Kerberos mantém uma base de dados com as identidades e chaves secretas de cada agente.
- O Kerberos permite utilizar estas chaves secretas para estabelecer uma chave de sessão entre um agente Cliente e um agente Servidor.
- A chave de sessão é utilizada para autenticação do Cliente perante o Servidor e, opcionalmente, para autenticação do Servidor e comunicação segura entre os dois.

# Autenticação Básica



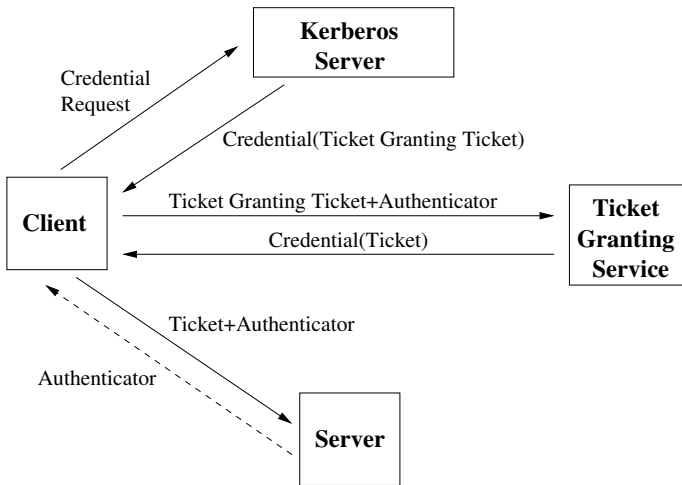
- **Cliente:** utilizador, aplicação.
- **Servidor:** serviço perante o qual se faz a autenticação.
- Todas as mensagens são definidas/codificadas utilizando ASN.1/DER.

- Em termos genéricos, uma **Credential** contém um **Ticket** e uma **Session Key** cifrados com a chave secreta pertencente ao Cliente.
- Um **Ticket** contém a identificação do Cliente e a mesma **Session Key** cifrados com a chave secreta pertencente ao Servidor.
- A chave de sessão é gerada pelo Kerberos, e é transmitida ao Cliente numa Credential.
- O Servidor obtém a mesma chave de sessão, inserida no Ticket, via Cliente. Para o Cliente, o conteúdo do Ticket é desconhecido.
- Conjuntamente com o Ticket, o Cliente envia um **Authenticator**.

- A função do Authenticator é demonstrar o conhecimento da chave de sessão, e assegurar a frescura e integridade do pedido de autenticação.
- Um Authenticator é uma mensagem autenticada por um MAC, gerado com a chave de sessão, e que contém a identidade do Cliente e um timestamp indicando o instante da sua geração.
- Um Ticket pode ser reutilizado. Um Authenticator não pode ser reutilizado.
- O processo de autenticação pode, opcionalmente, incluir a autenticação do Servidor perante o Cliente.
- Neste caso, o Servidor gera e envia ao Cliente um Authenticator semelhante ao que recebeu.

# Servidor Kerberos

- Num servidor Kerberos distinguem-se dois serviços: o **Authentication Server** e o **Ticket Granting Server**.
- A obtenção de uma Credential para aceder a um qualquer Servidor é, geralmente, uma negociação com duas fases:
  - O Cliente solicita primeiro uma Credential contendo um **Ticket Granting Ticket** ao Authentication Server.
  - Um Ticket Granting Ticket é um Ticket especial que permite ao Cliente aceder ao Ticket Granting Server de forma segura.
  - Utilizando o Ticket Granting Ticket, o Cliente pode obter a Credential que pretende junto do Ticket Granting Server.
- Em casos especiais a obtenção do Ticket pode ser feita numa só fase, directamente junto do Authentication Server.



# Chave de Sessão

- A chave de sessão estabelecida entre um Cliente e um Servidor que utilizam Kerberos tem diversas finalidades:
  - Autenticação do Cliente perante o Servidor. O MAC incluído no Authenticator enviado pelo Cliente demonstra ao Servidor que o Cliente conhece a chave de sessão estabelecida.
  - É esta mensagem que implicitamente identifica o Cliente perante o Servidor: a confiança depositada no servidor Kerberos assegura o Servidor que apenas Cliente e Servidor conhecem a chave de sessão.
  - Autenticação do Servidor perante o Cliente (opcional).
  - Autenticação (MAC) de mensagens trocadas subsequentemente entre Cliente e Servidor (opcional).
  - Confidencialidade de mensagens trocadas subsequentemente entre Cliente e Servidor (opcional).

## Domínios (Realm) Kerberos

- O Kerberos foi desenvolvido para ultrapassar fronteiras organizacionais: um Cliente numa organização pode ser autenticado perante um Servidor noutra organização.
- Cada organização implementa um ou mais Servidores Kerberos que constituem a infraestrutura do seu Domínio Kerberos.
- O nome do Domínio é incluído no nome de todos os utilizadores nele registados, e pode servir para um Servidor Kerberos noutra domínio “localizar” e validar esses utilizadores.
- A ligação entre Domínios consegue-se registando o Ticket Granting Service de uma organização no Domínio da outra organização, e vice-versa.



- Este registo consiste na criação de uma **Inter-Realm Key**: uma chave secreta que o Kerberos Server de um domínio utiliza para autenticar um Cliente local perante um Kerberos Server remoto.
- Um Cliente pode obter no seu Domínio um Ticket Granting Ticket para o Ticket Granting Server noutro Domínio.
- Estas relações são transitivas, i.e. se o Domínio A está ligado ao B, e o B ao C, então é possível autenticar utilizadores de A em C.
- Para evitar o estabelecimento de redes de Domínios, o que dificulta a identificação de um caminho de autenticação, em geral as ligações de Domínios estabelecem-se de forma hierárquica.
- O caminho de autenticação é também incluído na mensagem Ticket.

## Alguns Atributos/Flags

- **Initial** Indica a fase do processo de autenticação a que o Ticket pertence i.e. se foi obtido com base num Ticket Granting Ticket. Indica se o Cliente teve de apresentar recentemente a sua chave secreta para o conseguir.
- **Renewable** Indica um Ticket que é válido por um determinado período de tempo e renovável durante um período mais alargado. Evita a utilização frequente da chave secreta e mantém a frescura do ticket.
- **Post Dated** Permite a emissão de Tickets suspensos, para activação na altura da utilização.
- **Proxiable** Indica que um Servidor pode servir-se de um Ticket fornecido por um utilizador para adoptar a sua identidade perante outro Servidor.

- **Pre-authenticated** Indica que o Authentication Server autenticou o utilizador que pediu o Ticket de alguma forma e.g. login/password.
- **Hardware Authenticated** Indica que o Authentication Server autenticou o utilizador que pediu o Ticket utilizando um token de hardware e.g. um smartcard.
- **Anonymous** Permite a emissão de Tickets para uma entidade genérica dentro do Domínio.
- **Transited Policy Checked** Indica que o Servidor Kerberos do Domínio verificou a validade do caminho de autenticação indicado no Ticket (válido apenas para autenticações inter-domínio).

## Extensões de Criptografia Chave Pública

- O IETF define dois Draft Standards com extensões ao Kerberos que utilizam técnicas de Criptografia de Chave Pública e Certificação a dois níveis:
  - **Autenticação Inter-Domínio** A Inter-Realm Key é substituída por dois pares de chaves que passam a suportar a comunicação entre Servidores Kerberos em Domínios diferentes.
  - **Pedido de Ticket básico** A chave secreta que um Cliente utiliza para solicitar um Ticket (Granting Ticket) perante um Authentication Server é substituída por um par de chaves.
- Estas extensões basicamente definem procedimentos de geração e formatos de transferência alternativos para as mensagens Kerberos correspondentes a estes pontos de operação.

- Por exemplo, as alterações a um Pedido de Ticket básico são as seguintes:
  - O Cliente junta ao seu pedido de Ticket o seu Certificado e uma assinatura digital do próprio pedido.
  - A Credential devolvida pelo Authentication Server passa a vir cifrada:
    - usando o RSA, caso a Chave Pública do Cliente o permita,
    - ou uma cifra simétrica e uma chave secreta negociada utilizando o protocolo Diffie-Hellman.
- A utilização de certificados não é obrigatória: é possível adicionar manualmente as chaves públicas dos agentes à base de dados do Kerberos Server, conferindo-lhes desta forma o nível de confiança necessário.
- As extensões definem também restrições aos Distinguished Names dos certificados que permitem utiliza-los como identificadores Kerberos.

# Segurança

- A utilização de *timestamps* como indicadores da frescura dos Authenticators pode trazer problemas:
  - obriga a uma sincronização próxima dos relógios das máquinas envolvidas – isto é uma brecha na segurança porque os protocolos de sincronização temporal são, geralmente, inseguros.
  - torna possível os ataques por repetição de pedidos – o standard obriga a armazenar todos os pedidos para impedir este tipo de ataque, mas isto nem sempre é implementado.
- A utilização de PBE para gerar as chaves dos utilizadores simplifica os ataques por *password-guessing*, que tiram partido da fraca qualidade de passwords auto-atribuídas.
- Apesar destes questões, o Kerberos é tido como um sistema seguro, e a sua utilização é generalizada.

# Ficha Técnica

- **Cifras Simétricas:** DES, AES
- **Algoritmos de MAC:** DES-MAC, H-MAC
- **Funções de Hash Criptográficas:** SHA-1, MD5
- **Cifras Assimétricas:** RSA
- **Assinaturas Digitais:** RSA, DSA
- **Acordo de Chaves:** Diffie-Hellmann