Criptografia
MICEI / MSDPA - 2003/2004

Manuel Bernardo Barbosa

May 13, 2004

Manuel Bernardo Barbosa - mbb@di.uminho.pt

Parte I

Segurança: Utilização de Técnicas Criptográficas

Comunicação Utilizando Cifras Simétricas

- A comunicação utilizando cifras simétricas exige os seguintes passos:

  1. A Alice e o Bob combinam uma cifra.
  2. A Alice e o Bob acordam uma chave.
  3. A Alice cifra a mensagem e envia-a ao Bob.
  4. O Bob decifra a mensagem.

- Como a criptoanálise de uma cifra é geralmente muito difícil, os ataques centram-se muitas vezes nos dois primeiros passos.

- É por este motivo que a gestão de chaves é um problema muito importante em Criptografia. A chave tem de permanecer secreta antes, durante e depois da comunicação.

- Num universo de N agentes é necessário acordar $N(N-1)/2$ chaves: uma chave para cada par de utilizadores. Ou não?

Pré-distribuição de Chaves Secretas

- A aproximação mais directa à comunicação segura entre N agentes, utilizando cifras simétricas é a geração e distribuição de todas as chaves secretas necessárias antes de se iniciar a troca de informação.

- A principal desvantagem desta aproximação é o enorme número de chaves que é necessário gerar, armazenar e proteger: $N(N-1)/2$.

- A única forma racionalizar a utilização de recursos é transformar um dos agentes num servidor central de chaves secreta e responsável por armazenar todas as chaves secretas existentes, e de as disponibilizar aos outros agentes quando são necessárias. Como?

- Isto traz diversas vantagens imediatas: a redução do espaço necessário para o armazenamento, a centralização da gestão das chaves, a redução da exposição das chaves, etc.

Chaves de Sessão

- No entanto, esta aproximação tem limitações que é impossível transpor:
  - uma vez que as chaves permanecem válidas durante a actividade do sistema elas estão expostas a ataques durante um tempo prolongado.
  - se uma chave for corrompida, e isso não for detectado, a comunicação entre o par de agentes correspondente deixa de ser segura.

- O conceito de **chave de sessão** resolve estes problemas:
  - É gerada uma chave secreta para cada troca de informação.
  - Esta chave é estabelecida entre emissor e receptor, utilizada para a troca de informação e depois é destruída.

- O estabelecimento das chaves de sessão pode ser feito de duas formas: a **distribuição de chaves**, e o **acordo de chaves**.

Distribuição de Chaves

- Num esquema de distribuição de chaves um dos agentes é responsável por gerar a chave de sessão. O problema a resolver é a transmissão da chave de sessão para o interlocutor através de um canal seguro.

- Isto coloca diversas questões: como é que se implementa um canal seguro? Como é que se obtém a garantia de que se está efectivamente a falar com o interlocutor, e não com um intruso?

- Em geral a solução adoptada consiste na introdução no sistema de um **Agente de Confiança** ou Trusted Agent (TA), e na adopção de uma classificação das chaves secretas em dois tipos: **chaves para cifragem de chaves** e **chaves para cifragem de dados**.

- **Nota:** Na comunicação agente/TA podem também ser usadas cifras assimétricas. Voltaremos a este assunto mais tarde.

Distribuição de Chaves$_{cont}$

- As **chaves para cifragem de chaves** são geradas e pré-distribuidas por um canal seguro exterior ao sistema (e.g. em mão numa disquete, etc.) N chaves deste tipo que permitem ao TA comunicar com todos os agentes de forma segura.

- As **chaves para cifragem de dados** correspondem às chaves de sessão. São geradas por um dos interlocutores ou pelo TA e transmitidas, através do TA, utilizando as chaves anteriores.

- Como grande desvantagem, este esquema tem a necessidade de pré-distribuir, armazenar e proteger as N chaves secretas utilizadas para comunicação segura entre o TA e os agentes.

- O Kerberos é um exemplo de um protocolo comercial baseado nesta filosofia.

Acordo de Chaves

- Os protocolos de acordo de chaves são mecanismos que derivam da criptografia assimétrica. O seu princípio de funcionamento é o seguinte:
  - Dois agentes A e B pretendem comunicar utilizando uma cifra simétrica e, logo, estabelecer uma chave de sessão.
  - Existe um canal de comunicação aberto entre os dois agentes.
  - Os dois agentes seguem um conjunto de passos que lhes permitem identificar-se mutuamente.
  - Os dois agentes cooperam para gerar uma chave de sessão trocando informação através do canal aberto. Na composição da chave entra informação aleatória gerada por ambas as partes, por forma a ambos os agentes terem a garantia da **frescura da chave**.
  - Os dois agentes executam um conjunto de passos que lhes permitem confirmar que foi estabelecida uma chave de sessão adequada.

Acordo de Chaves$_{cont}$

- Em nenhum dos passos anteriores pode circular no canal aberto informação que permita a um intruso adivinhar a chave secreta acordada.

- Como é que isto se consegue? O segredo está na utilização de funções matemáticas com características especiais: as chamadas **funções one-way**.

- O que se consegue na prática é eliminar a necessidade de se armazenar informação secreta, bem como a necessidade da presença de um Agente de Confiança durante o funcionamento do sistema. (Em geral é necessário um TA para garantir as questões de identificação. Voltaremos a este assunto mais tarde.).

- O SSL é um exemplo de um protocolos comercial que recorre a este tipo de tecnologia.

Exemplo: Diffie-Hellman

- Foi inventado em 1976 e abriu o caminho para a criptografia de chave pública. É considerado o primeiro algoritmo desta categoria.

- É exclusivamente um protocolo de acordo de chaves. Não pode ser utilizado para cifragem e decifragem.

- A sua segurança baseia-se na dificuldade em resolver o problema do logaritmo discreto em corpos finitos. Por exemplo:
  - Considere-se o conjunto $\{x \in Z : 0 \leq x < 11\}$, sobre o qual se definem a multiplicação e a adição módulo 11.
  - É fácil calcular a exponencial $3^4 \pmod{11} = 81 \pmod{11} = 4$.
  - No entanto, não há nenhuma forma eficiente de calcular o inverso: $\log_3 4 \pmod{11}$.

Exemplo: Diffie-Hellman$_{cont}$

1. A Alice e o Bob acordam os parâmetros públicos do protocolo:
   - um número primo grande n, e
   - um número primo g, menor que n (e com algumas outras restrições que garantem a segurança do protocolo).
2. A Alice gera um número aleatório grande x.
3. A Alice calcula $X = g^x \pmod{n}$ e envia-o ao Bob.
4. O Bob gera um número aleatório grande y.
5. O Bob calcula $Y = g^y \pmod{n}$ e envia-o à Alice.
6. Ambos conseguem calcular $K = X^y \pmod{n} = Y^x \pmod{n}$.

Comunicação Utilizando Cifras Assimétricas

- A comunicação utilizando cifras assimétricas exige os seguintes passos:

  1. A Alice e o Bob combinam uma cifra assimétrica.
  2. O Bob envia à Alice, por canal aberto, a sua Chave Pública..
  3. A Alice cifra a mensagem com a Chave Pública e envia-a ao Bob.
  4. O Bob decifra a mensagem com a sua Chave Privada.

- Fica resolvido o problema da partilha e gestão de chaves secretas, uma vez que a Chave Privada permanece na posse de um único agente.

- Num universo de N agentes, cada qual terá o seu par de chaves, podendo as chaves públicas ser armazenadas num servidor central no qual **não tem de haver o mesmo tipo de confiança absoluta**.

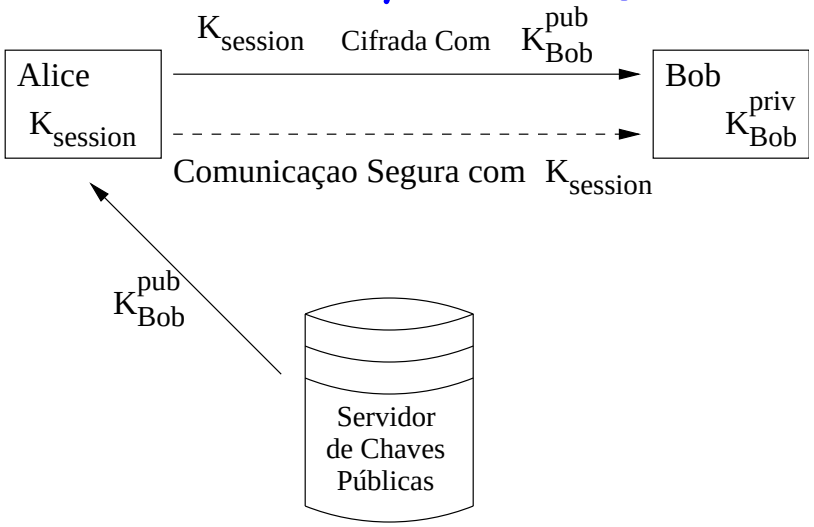
- Resta o problema de garantir que uma chave pública pertence a um determinado agente. Porquê?

Cifras Simétricas vs Cifras Assimétricas

- No mundo real as cifras assimétricas não são um substituto para as cifras simétricas. Isto por duas razões:
  - Os algoritmos assimétricos são pelo menos 1000 vezes mais lentos que os algoritmos simétricos que fornecem a mesma segurança. Devido à existencia de $atalhos$ nos ataques a estas cifras, as chaves têm de ter no mínimo 1024 bits, o que torna os cálculos muito pesados.
  - É sempre possível efectuar um ataque por texto limpo conhecido:
    * Se conhecemos um criptograma e a chave pública podemos, teóricamente, testar todas as possibilidades para o texto limpo até obtermos uma correspondência.
    * Quando os textos limpos possíveis são poucos, isto é um problema.

- As aplicações mais comuns das cifras assimétricas vêm complementar a funcionalidade das cifras simétricas e.g. **os sistemas de híbridos de distribuição de chaves de sessão**.

Sistemas Híbridos/Envelopes Digitais



- Neste esquema de distribuição de chaves desaparece a necessidade de um TA e do estabelecimento de chaves para cifragem de chaves:

  1. A Alice obtém a Chave Pública do Bob a partir do servidor.
  2. A Alice cifra a Chave de Sessão com essa Chave Pública.
  3. O Bob decifra a Chave de Sessão com a sua Chave Privada.

Autenticação de Origem de Mensagens

- A autenticação de origem de mensagens permite fornecer a uma entidade que recebe uma mensagem garantias sobre a identidade do seu emissor.

- Este tipo de autenticação pode ser necessária quando:
  - a troca de informação entre dois agentes se processa sem que ambos estejam on-line simultâneamente e.g. e-mail.
  - a verificação da autenticidade dos dados é feita posteriormente e.g. um recibo.
  - a comunicação entre dois agentes se processa através de intermediários e.g. internet.

- A autenticação da origem de mensagens implica garantias de integridade: quem altera uma mensagem passa a ser o seu emissor.

Message Authentication Codes

- Um Message Authentication Code (MAC) é uma função de hash criptográfica cujo resultado depende, não só do conteúdo da mensagem que é processada, mas também de uma chave secreta.

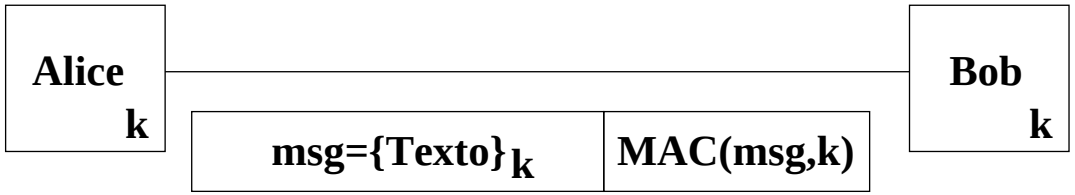
- Para gerar o MAC é necessário conhecer o algoritmo e a chave secreta. O mesmo acontece para verificar se o MAC é válido.

- Pode ser utilizado como uma impressão digital que garante que o checksum da mensagem foi calculado na sua origem: torna impossíveis alterações do checksum para mascarar uma modificação da mensagem.

- Outra aplicação possível é a protecção de ficheiros contra ataques de virus, uma vez que o virus seria incapaz de produzir um MAC válido para esconder as alterações que introduzisse.

Message Authentication Codes: Utilização

- A confidencialidade implica a autenticação da origem dos dados?
  - Poder-se-ia pensar que se apenas Alice e Bob conhecem k, então nenhum intruso consegue alterar msg sem ser detectado na decifragem.
  - **Isto não é sempre verdade**: implica a existência de algum tipo de semântica ou redundância no texto limpo. E se for aleatório?
  - Incluindo um MAC na mensagem, é possível comprovar a origem da mensagem e a sua integridade, antes de a decifrar.
  - É mesmo possível utilizar MACs sem confidencialidade.



- Que garantias tem Bob quando recebe a mensagem?

Assinaturas Digitais

- A assinatura manuscrita é há muito utilizada como prova de autoria ou, pelo menos, de concordância com o conteúdo de um documento. Quais são as propriedades que se esperam de uma assinatura escrita?
  - A assinatura é **autêntica**: convence o receptor do documento de que o signatário explicitamente assinou o documento i.e. que conhecia o seu conteúdo, por ser o seu autor ou, simplesmente, por o aceitar.
  - A assinatura **não é falsificável**: prova que o signatário, e não outra pessoa, assinou o documento.
  - A assinatura **não é reutilizável**: faz parte do documento e não se pode transpor para outro documento.
  - A assinatura garante a **integridade do documento**: o documento permaneceu inalterado desde que a assinatura lá foi colocada.
  - A assinatura **não é repudiável**: o signatário não pode, à posteriori, negar que assinou o documento.

Assinaturas Digitais$_{cont}$

- **Uma assinatura digital é uma sequência de bytes cujo valor não tem significado fora do contexto de um algoritmo específico**. É gerada pelo signatário e acompanha o documento assinado, como anexo.

- Associados a um esquema de assinaturas digitais há sempre dois procedimentos complementares a considerar:
  - **Geração da assinatura**. Procedimento segundo o qual o signatário produz a sequência de bytes a partir do documento e de informação secreta ou privada (uma chave). Este procedimento deve assegurar as propriedades requeridas para a assinatura.
  - **Verificação da assinatura**. Procedimento segundo o qual o destinatário do documento assinado consegue assegurar-se de que a sequência de bytes que recebe como assinatura é um valor válido, gerado pelo signatário para esse fim.

Assinaturas Digitais: Notas Importantes

- As assinaturas digitais baseadas na criptografia de chave pública dominam a criptografia actual, principalmente por não requererem um TA on-line.
- **Nem todos os algoritmos de assinatura digital são adaptações directas de algoritmos de cifra assimétrica**.
- Por esta razão é muito importante **não confundir** uma assinatura digital, que confere **autenticação** com uma cifra assimétrica.
- **Deve evitar-se** referir as operações de assinar e verificar uma assinatura como *cifrar com chave privada* ou *decifrar com chave pública*.
- O objectivo das assinaturas digitais **não é conferir confidencialidade**: acompanham o documento a que se referem, que pode ou não ser cifrado.

Assinaturas Digitais e Funções de Hash

- Os algoritmos assimétricos são geralmente muito pouco eficientes, o que torna pouco prática a assinatura de mensagens de tamanho realista.

- Por esta razão é mais comum a geração de assinaturas digitais a partir de impressões digitais das mensagens.

- O processo de geração da assinatura começa assim pelo cálculo de um valor de hash, que é depois assinado com a chave privada do signatário.

- A verificação da assinatura implica um novo cálculo do hash da mensagem. A validade da assinatura é avaliada através de uma algoritmo que processa este hash, a chave pública do signatário e a assinatura.

- Este método tem a vantagem adicional de tornar mais dificil o ataque à assinatura: o hash funciona como checksum da mensagem.

Exemplo: Protocolo de Acordo de Chaves Station-to-Station (STS)

- O protocolo de acordo de chaves Diffie-Hellman é vulnerável a um ataque chamado *man-in-the-middle*:
  - O intruso intercepta as mensagens trocadas entre os agentes, substituindo-as por mensagens suas.
  - Como não há autenticação, os agentes terminam o protocolo acreditando que acordaram uma chave entre si.
  - Na realidade, cada um dos agentes acordou uma chave secreta com o intruso, sem o saber.
  - Enquanto o intruso fizer o relay das mensagens entre os agentes, estes não se apercebem da sua presença.

- Outro problema com o protocolo DH é não incluir, explicitamente, um passo de confirmação da chave acordada.

Exemplo: Protocolo de Acordo de Chaves Station-to-Station (STS)$_{cont}$

- O protocolo STS corrige estes problemas acrescentando um novo passo ao protocolo DH, com base num esquema de assinaturas digitais:
  - Uma vez acordada a chave de sessão, os agentes assinam digitalmente o par ordenado (X, Y).
  - Estas assinaturas são trocadas entre os agentes, cifradas com a chave recém-acordada.
  - Só se considera que o protocolo terminou com sucesso se as assinaturas forem recuperadas e verificadas correctamente.

- Mais uma vez convém notar que, para este sistema ser seguro, é necessário estabelecer a validade das chaves públicas dos agentes: é necessário um sistema de certificação de chaves públicas.

Assinaturas Digitais e Cifragem

- Combinando a assinatura digital com a cifragem obtém-se um protocolo que combina confidencialidade e autenticação.

- Este conceito é intuitivo: as cartas são também assinadas (prova de autoria) e depois inseridas num envelope (privacidade).

- **Faz mais sentido assinar antes ou depois de fazer a cifragem**? A primeira hipótese é melhor por vários motivos:
  - Se a mensagem não é assinada antes de ser cifrada, pode haver dúvidas quanto ao conhecimento que o signatário tinha do seu conteúdo.
  - Um intruso não tem acesso à informação de autenticação, isto é, à assinatura, a não ser que quebre a cifra.
  - Um ataque de substituição ou reutilização da assinatura deixa de fazer sentido.

Autenticação de Entidades: Identificação

- O problema da identificação consiste em um agente (o **identificado** ou A) demonstrar a sua identidade a outro agente (o **identificador ou B**). Estes agentes podem, ou não, ser utilizadores humanos.

- A autenticação de entidades implica uma noção de tempo: é feita em tempo-real. Isto pode não acontecer para a autenticação de mensagens.

- Os objectivos dos protocolos de identificação são os seguintes:
  - No caso de entidades honestas, B aceita a identidade de A.
  - B não pode apropriar-se da identidade de A perante terceiros.
  - É muito pouco provável que uma entidade C consiga passar-se por A.
  - As propriedades anteriores mantêm-se verdadeiras mesmo depois de um número arbitrário de execuções do protocolo.

Autenticação de Entidades: Identificação$_{cont}$

- A identificação pode fazer-se de acordo com três princípios:
  - **Aquilo que se sabe** e.g. passwords.
  - **Aquilo que se possui** e.g. um cartão funcionando como passaporte.
  - **Aquilo que se é** e.g. propriedades biométricas inerentes ao indivíduo.

- Os esquemas de identificação biométricos são aqueles que apelam mais ao imaginário. São exemplos de mecanismos de identificação deste tipo:
  - a identificação através da impressão digital,
  - do reconhecimento das feições ou
  - do reconhecimento da voz.

- Há, no entanto, problemas importantes associados à utilização deste tipo de sistemas, tanto pelos níveis de segurança que é possível obter, como no que diz respeito à sua implementação prática, e mesmo a nível legal.

Autenticação de Entidades: Identificação$_{cont}$

- Em geral, os sistemas de identificação utilizados na prática não se baseiam na utilização de informação biométrica.

- Em alternativa, associa-se um $token$ de informação privada à identidade do agente. Idealmente, apenas o próprio agente conhecerá ou terá acesso a essa informação.

- A identificação efectua-se através da demonstração do conhecimento ou posse dessa informação privada.

- Exemplo: **Mecanismo Login/Password**
  - O utilizador identifica-se fornecendo a sua identificação e a sua informação privada (a password).
  - Problema: o identificador (ou intruso) conhece a informação privada !

Princípio do Conhecimento Zero

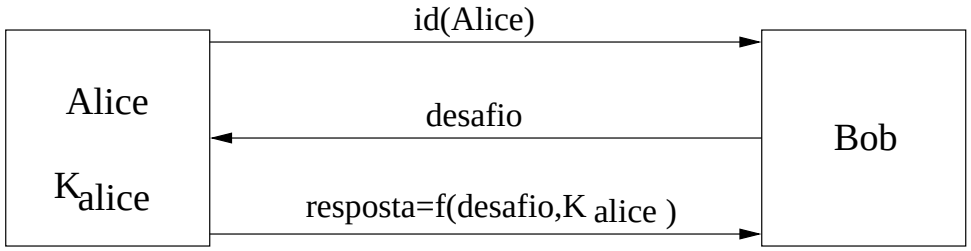
- Como demonstrar o conhecimento de informação privada ou secreta sem a divulgar explicitamente? Uma **prova de conhecimento zero** é um mecanismo que permite resolver este problema.

- É este tipo de protocolo que permite implementar um sistema de identificação ideal: demonstra-se o conhecimento do segredo associado à identidade sem revelar informação privada.

- Os protocolos que cumprem este princípio são probabilísticos:
  - É feita uma pergunta aleatória cuja resposta depende do segredo.
  - Não conhecendo o segredo, é possível acertar na resposta com 50% de probabilidade.
  - Fazendo uma série de perguntas, consegue-se estabelecer a identidade com uma probabilidade de erro arbitráriamente pequena.

Mecanismo de Desafio/Resposta

- Este mecanismo é o componente básico dos protocolos de identificação que seguem, ou aproximam, o princípio do conhecimento zero:



- Para o protocolo ser verdadeiramente de conhecimento zero, a resposta não pode implicar a transferência de informação da Alice para o Bob que permita a reconstrução do segredo !!!

- Note-se que qualquer mecanismo de assinatura digital pode ser utilizado como $f(desafio, K_{alice})$.

Exemplo: Protocolo de Identificação de Schnorr

- A Alice pretende identificar-se perante o Bob:

  1. Parâmetros públicos: um número primo grande p, um número primo q divisor de $p-1$ e um número g tal que $g^q \pmod p = 1$.
  2. A Alice gera um par de chaves: a chave privada é um número aleatório $0 < s < q$; a chave pública é $v = g^{-s} \pmod p$ e é enviada ao Bob.
  3. A Alice escolhe um número aleatório $0 < r < q$, calcula $x = g^r \pmod p$ e envia x ao Bob.
  4. O Bob envia à Alice um número aleatório $0 < e < 2^t - 1$: o desafio. t estabelece um nível de certeza, expresso em número de bits.
  5. A Alice calcula a resposta $y = (r + s * e) \pmod q$ e envia-a ao Bob.
  6. O Bob verifica que $x = g^y v^e \pmod p$.

- Mais uma vez convém notar que, para este sistema ser seguro, é necessário estabelecer a validade da chave pública da Alice: é necessário um sistema de certificação de chaves públicas.

Parte II

Certificados de Chave Pública X.509

Introdução

- Esforços recentes para melhorar a segurança na Internet levaram ao aparecimento de um grupo de protocolos (S/MIME, IPSec, etc.) que utilizam a criptografia de chave pública para garantir:
  - Confidencialidade
  - Integridade
  - Autenticação
  - Não repúdio.

- Esta utilização baseia-se no conceito de Certificado (de chave pública ou de atributos).

- A Public Key Infrastructure (PKI) tem como objectivo a gestão segura e eficiente de chaves e certificados para permitir essa mesma utilização.

Introdução$_{cont}$

- A utilização da criptografia de chave pública nas telecomunicações é regulamentada pela recomendação X.509 da International Telecommunications Union (ITU).

- A aplicação dessa recomendação à Internet está definida num conjunto de Requests For Comments (RFCs) publicados pela IETF.

- A Internet Engineering Task Force é uma comunidade internacional de produtores, operadores, vendedores e investigadores das tecnologias de redes, interessados no funcionamento e evolução da Internet.

- Dentro da IETF, o grupo que gere os RFCs relacionados com o X.509 chama-se "*PKIX Working Group*". Este grupo de trabalho mantém um conjunto de documentos que se denomina "*Internet X.509 Public Key Infrastructure*".

Certificados de Chave Pública

- Os utilizadores de um sistema baseado numa PKI devem poder confiar que, cada vez que utilizam uma chave pública, o agente com quem querem comunicar possui a chave privada associada.

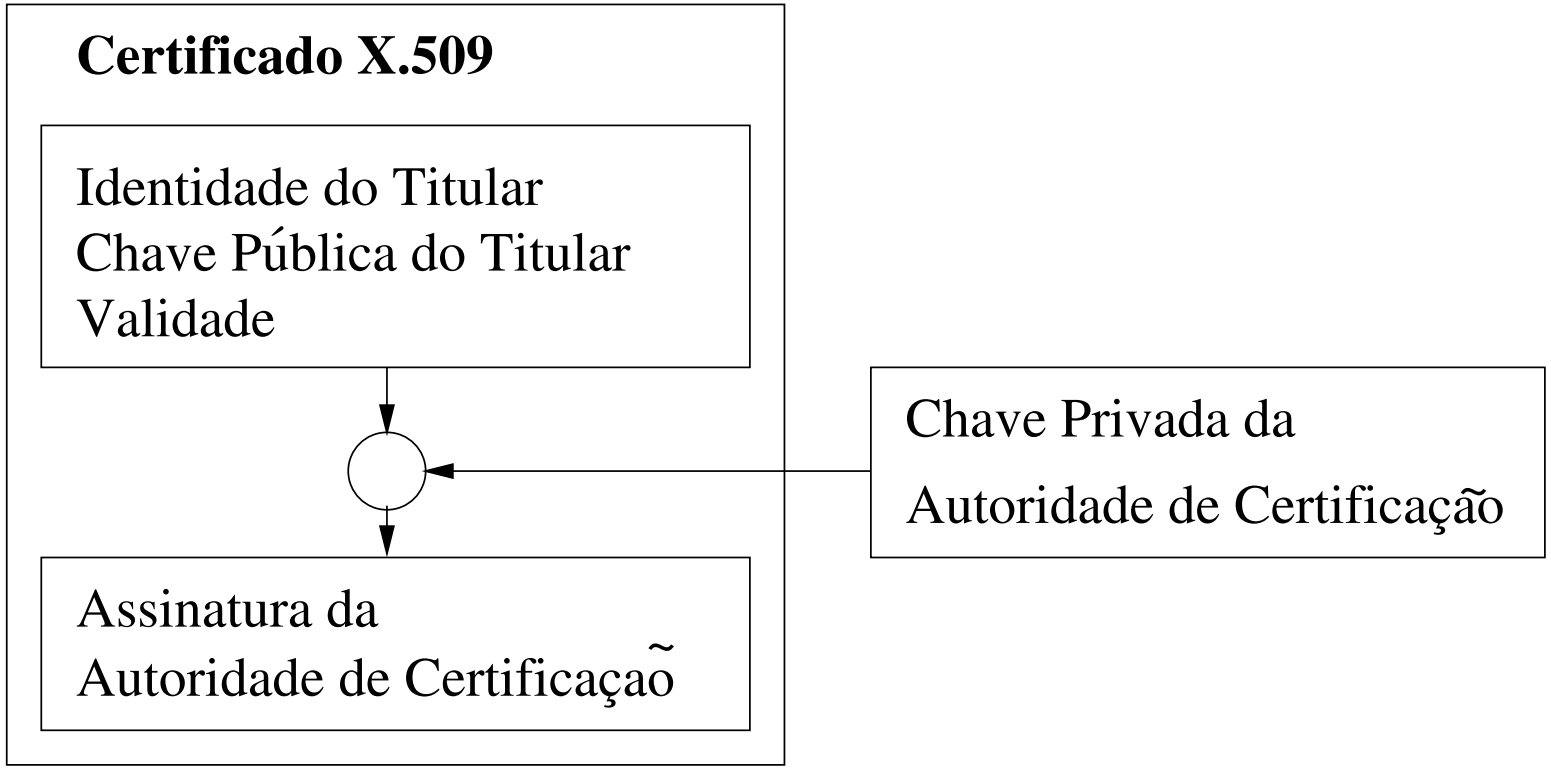
- Esta confiança é construída com base em Certificados de Chave Pública.

- Um Certificado de Chave Pública é uma estrutura de dados que associa uma chave pública a um determinado agente (a uma representação da sua identidade).

- A associação chave/agente é estabelecida por uma entidade terceira, uma Autoridade de Certificação, que assina digitalmente cada certificado.

- A utilidade de um certificado depende unicamente da **confiança** depositada na Autoridade de Certificação.

Certificados de Chave Pública$_{cont}$



Certificados de Chave Pública$_{cont}$

- O utilizador do certificado confia que a Autoridade de Certificação verificou que a chave pública contida no certificado pertence de facto ao titular do certificado.

- A assinatura da Autoridade de Certificação assegura a autenticidade e integridade do certificado.

- Um Certificado de Chave Pública é válido durante um período de tempo bem definido. Esse período vem especificado no conteúdo assinado.

- Como a assinatura e a validade temporal de um certificado podem ser verificados independentemente por um utilizador, os certificados podem ser distribuidos por canais inseguros. Será isto verdade para todos os certificados?

Certificados de Chave Pública$_{cont}$

- Os Certificados de Chave Pública são utilizados maioritáriamente na validação de informação assinada digitalmente. Este processo consiste geralmente nos seguintes passos:

  1. O destinatário verifica que a identidade indicada pelo emissor está de acordo com a identidade indicada no certificado.
  2. O destinatário verifica que o certificado é válido:
     - que a assinatura do certificado é válida;
     - que foi efectuada por uma autoridade de certificação de confiança;
     - que o certificado está dentro do seu período de validade.
  3. O destinatário verifica que a informação que recebe está de acordo com as permissões/previlégios do emissor.
  4. O destinatário utiliza a chave pública contida no certificado para verificar a assinatura da informação recebida.

Certificados de Chave Pública$_{cont}$

- Se todos os passos anteriores forem executados sem problemas, o destinatário aceita que a informação foi assinada pelo emissor, e que essa informação permanece inalterada.

- Como é que se utiliza um certificado para proteger informação ao nível da confidencialidade?

- O certificado passa a ser utilizado pelo emissor, e contém informação relativa ao destinatário:

  1. o emissor valida o certificado e a identidade do destinatário;
  2. o emissor utiliza a chave pública contida no certificado para cifrar a informação;
  3. o emissor envia a informação cifrada ao destinatário que a decifra com a sua chave privada.

Certificados X.509 (V1)

- Surgiram em 1988 com a primeira versão do X.509. Um certificado deste tipo contém os seguintes campos:
  - **Version** version
  - **CertificateSerialNumber** serialNumber
  - **AlgorithmIdentifier** signature
  - **Name** issuer
  - **Validity** validity
  - **Name** subject
  - **SubjectPublicKeyInfo** subjectPublicKeyInfo

- A assinatura do certificado é efectuada pela Autoridade de Certificação (CA) sobre uma codificação DER da representação desta estrutura de dados em ASN.1.

Certificados X.509 (V2)

- Surgiram na revisão de 1993 do X.509. Não chegaram a ser muito utilizados porque pouco tempo depois surgiu a versão actual (V3).

- Foram introduzidos dois novos campos:
  - **UniqueIdentifier** issuerUniqueID
  - **UniqueIdentifier** subjectUniqueID

- Estes campos tentam rectificar o problema de ser muito difícil garantir que os campos do tipo **Name** tenham valores únicos.

- A IETF recomenda que as CAs não utilizem estes campos e garantam, na medida do possível, a unicidade dos nomes. Por outro lado recomenda que estes campos, caso existam, não devem ser ignorados.

Certificados X.509 (V3)

- Esta é a versão utilizada hoje em dia. Foi standardizada em 1996 e é compatível com as versões anteriores.

- Esta versão veio colmatar as deficiências que as versões anteriores apresentavam para algumas aplicações e que consistiam básicamente na necessidade de mais atributos.

- Como inovação, esta versão introduziu um novo campo do tipo **Extensions**: uma colecção de elementos do tipo **Extension**.

- Estas extensões permitem a associação de atributos genéricos a um agente ou à sua chave pública, de forma muito flexível.

- Cada extensão é ela própria uma estrutura de dados com um identificador e um valor adequado ao tipo do atributo que representa.

Certificados X.509 (V3): Atributos

- **version** Tem de estar de acordo com o conteúdo do certificado.
- **serialNumber** Número único atribuido pela CA.
- **signature** Estrutura que identifica o algoritmo utilizado para gerar a assinatura da CA que acompanha o certificado.
- **validity** Estrutura com as duas datas que delimitam o período de validade do certificado.
- **subjectPublicKeyInfo** Estrutura contendo a chave pública do titular do certificado e identificação do algoritmo correspondente.

Certificados X.509 (V3): Atributos$_{cont}$

- Os atributos **issuer** e **subject** identificam a CA e o titular do certificado respectivamente. Ambos são do tipo **Name**.

- O tipo **Name** provém da norma X.501 e é utilizado porque permite a compatibilidade com os sistemas de directório definidos nas normas X.500 (e.g. DAP e LDAP).

- O tipo **Name** é uma colecção de atributos, geralmente $strings$ da forma "$< nome > = < valor >$". Estes atributos definem um **Distinguished Name** para o agente titular.

- O **Distinguished Name** tem uma estrutura hierárquica. Inclui por exemplo, o país, a organização e o nome próprio do agente ou entidade.

Certificados X.509 (V3): Atributos$_{cont}$

- A norma X.520 standardiza alguns dos componentes de um Distinguished Name. Os seguintes são de reconhecimento obrigatório e são muito utilizados:
  - country (C)
  - organization (O)
  - organizational-unit (OU)
  - common name (CN)
  - serial number (SN)

- Algumas aplicações importantes utilizam também o endereço de e-mail como um dos atributos centrais da construção do Distinguished Name.

- Exemplo: C=PT, O=UMINHO, OU=DI, CN=MBB

Certificados X.509 (V3): Extensões

- As extensões são marcadas como **Critical** ou **Non Critical**.
- Uma aplicação que encontre uma extensão crítica que não reconheça tem de rejeitar o certificado.
- Não são permitidas várias instâncias da mesma extensão.
- O RFC3280 da IETF normaliza as extensões recomendadas para utilização na Internet, definindo o identificador (OBJECT IDENTIFIER) e o tipo de dados associado.
- São desaconcelhados desvios desta recomendação, nomeadamente no que diz respeito a extensões críticas, apesar de não haver qualquer limitação a nível do standard.

Certificados X.509 (V3): Extensões$_{cont}$

- **Subject Key Identifier** Serve para identificar o certificado que contém uma determinada chave pública e.g. quando um agente tem várias. É, em geral, um valor de hash derivado da chave pública que, caso esta extensão não conste do certificado, pode ser calculado em run-time.

- **Authority Key Identifier** Serve para identificar a chave pública da CA que assinou o certificado, caso existam várias, o que facilita a verificação de cadeias de certificação. Isto pode ser feito
  - identificando o certificado da CA através do seu **Subject** e **Serial Number**;
  - ou através da extensão **Subject Key Identifier** do certificado da CA.

- **Subject/Issuer Alternative Name** Permitem associar formas de identificação alternativas ao titular do certificado ou à CA que o emitiu, e.g. e-mail, nome DNS, endereço IP, URI, etc.

Certificados X.509 (V3): Extensões$_{cont}$

- **Basic Constraints** Permite assinalar um certificado como pertencendo a uma CA e limitar o comprimento de cadeias de certificados.

- **Certificate Policies** Permite incluir informação relativa às políticas de certificação aplicáveis ao certificado:
  - Para certificados de utilizador, permite especificar em que condições o certificado foi emitido e quais as restrições associadas à sua utilização.
  - Para certificados de CAs, permite definir as políticas de certificação aplicaveis por CAs hierarquicamente inferiores.

- **Policy Mappings** Permite a uma CA declarar que algumas das suas políticas são equivalentes às políticas de certificação de outra CA.

  (As hierarquias de CAs serão estudadas mais tarde.)

Certificados X.509 (V3): Extensões$_{cont}$

- **Key Usage** Esta extensão permite restringir as utilizações do par de chaves associado ao certificado e.g. quando uma chave apenas pode ser utilizada para verificar assinaturas digitais. Esta extensão contempla as seguintes utilizações:
  - **digitalSignature** Assinaturas digitais para autenticação e protecção da integridade de dados, excepto certificados e CRLs.
  - **nonRepudiation** Assinaturas digitais para não repúdio.
  - **keyEncipherment** Protecção da confidencialidade de chaves.
  - **dataEncipherment** Protecção da confidencialidade de dados.
  - **keyAgreement** Protocolos de acordo de chaves.
  - **keyCertSign** Assinatura de certificados.
  - **cRLSign** Assinatura de CRLs.
  - **encipherOnly**/**decipherOnly** Restringem a funcionalidade **keyAgreement**.

Certificados X.509 (V3): Extensões$_{cont}$

- **Extended Key Usage** Permite especificar ou restringir as utilizações previstas para o par de chaves associado ao certificado, em adição ou em alternativa à extensão **Key Usage**. Estão definidas diversas utilizações, bem como a sua relação com as especificadas na extensão **Key Usage**:
  - WWW server authentication
  - WWW client authentication
  - Signing of downloadable executable code
  - E-mail protection
  - . . .

- **CRL Distribution Points** Serve para indicar ao utilizador de um certificado onde pode obter informação quanto à revogação do certificado na forma de **Certificate Revocation Lists** (CRLs).

Certificados X.509 (V3): Codificação

- Os certificados, como todas as estruturas de dados na PKI, são definidos e representados em ASN.1. A codificação é feita utilizando as Dinstinguished Encoding Rules (DER).

- O ficheiro que contém um certificado X.509 consiste na codificação DER da seguinte estrutura ASN.1:

```
Certificate  ::=  SEQUENCE  {
        tbsCertificate          TBSCertificate,
        signatureAlgorithm      AlgorithmIdentifier,
        signatureValue          BIT STRING  }
```

- Além da estrutura de atributos apresentada nos slides anteriores (**tbsCertificate**), aparece também a assinatura da Autoridade de Certificação (**signatureAlgorithm** e **signatureValue**).

Parte III

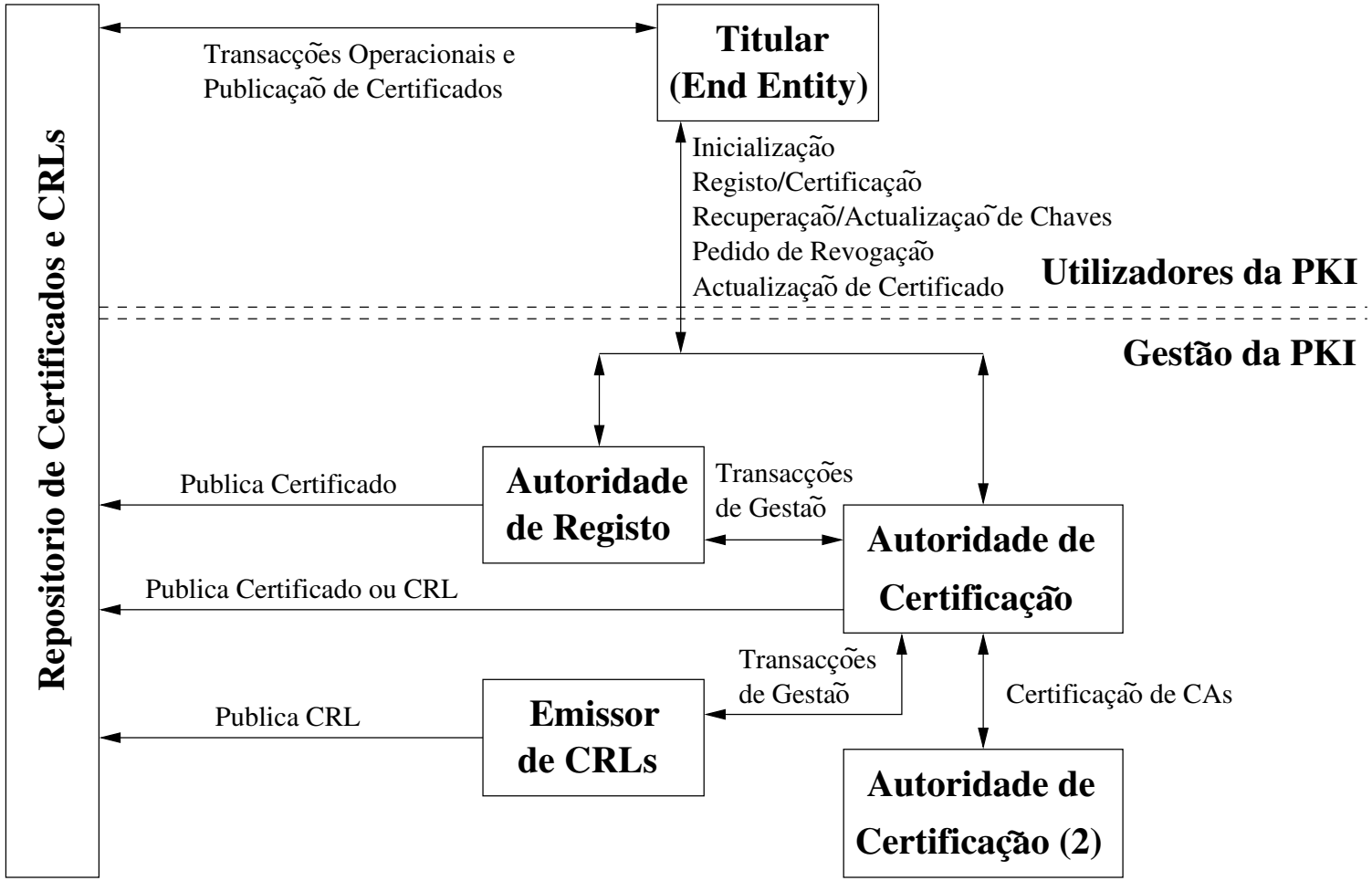
Infraestrutura de Chave Publica (PKI)

Introdução

- Uma Public Key Infrastructure define-se como o conjunto de hardware, software, pessoas, políticas e procedimentos necessários para criar, gerir, armazenar, distribuir e revogar Certificados de Chave Pública.

- Uma PKI é composta por cinco tipos de componentes:
  - **Titulares** de Certificados de Chave Pública, que possuem as chaves privadas e as utilizam para decifrar mensagens e assinar documentos.
  - **Clientes** que utilizam a chave pública contida num certificado para cifrar mensagens e verificar assinaturas.
  - **Autoridades de Certificação**, que emitem e revogam certificados.
  - **Autoridades de Registo** que garantem a associação entre chaves públicas e identidades de titulares (são opcionais).
  - **Repositórios** que armazenam e disponibilizam certificados e CRLs.

Arquitectura



Arquitectura$_{cont}$

- O funcionamento de uma PKI baseia-se em dois tipos de protocolos:
  - **Protocolos Operacionais** Estes protocolos são necessários para entregar certificados e CRLs aos sistemas que os utilizam. Estas operações podem ser efectuadas de diversas formas, incluindo o LDAP, HTTP e FTP. Para todos estes meios estão especificados protocolos operacionais que definem, inclusivamente, os formatos das mensagens.
  - **Protocolos de Gestão** Estes protocolos são necessários para dar suporte às interacções entre os utilizadores e as entidades de gestão da PKI, nomeadamente:
    * Inicialização.
    * Registo e Certificação.
    * Recuperação e Actualização de pares de chaves.
    * Pedido de revogação.
    * Certificação de CAs.

PKI: Operações

- **Inicialização** Consiste no processo inicial que permite ao utilizador comunicar com a PKI: o utilizador toma conhecimento das CAs em que confia e adquire as chaves públicas e certificados correspondentes, gera o seu par de chaves, etc.

- **Registo** Uma end entity (utilizador) dá-se a conhecer perante uma CA (directamente, ou através de uma RA) para que a CA possa emitir um certificado para essa entidade. O utilizador fornece informação de identificação que deve ser verificada pela CA (RA).

- **Geração de Par de Chaves** Em algumas implementações, as CAs encarregam-se de gerar o par de chaves da entidade.

- **Certificação** A CA recebe a chave pública do utilizador e a sua identificação e emite o respectivo certificado, segundo regras internas.

PKI: Operações$_{cont}$

- **Publicação de Certificados e CRLs** Esta tarefa pode ser feita directamente pela CA, ou indirectamente por entidades como RAs. Além de colocar os certificados e CRLs em repositórios é muitas vezes necessário fazer estes documentos chegar aos utilizadores finais por outros meios (on-line ou não).

- **Revogação** Quando um certificado é emitido o seu período util de vida está pré-definido. No entanto, pode haver a necessidade de invalidar o certificado antes do fim desse período por diversos motivos (e.g. um despedimento, o comprometimento da chave privada, etc.). A revogação de certificados faz-se através de CRLs. As CRLs vão ser analizadas em detalhe mais tarde.

PKI: Operações$_{cont}$

- **Recuperação de um Par de Chaves** Em algumas implementações as CAs armazenam o par de chaves da entidade como $back\text{-}up$ e protecção e.g. no caso de uma empresa e os seus empregados.Nestes casos o par de chaves pode ser restaurado em caso de extravio ou danificação do seu suporte.

- **Actualização de Par de Chaves** Todos os pares de chaves precisam de ser alterados, periódicamente por razões de segurança, ou simplesmente porque a segurança da chave privada foi corrompida.

- **Certificação de CAs** Os certificados das CAs chamam-se **cross certificates**. São utilizados para a validação de cadeias de certificados, mas também podem ser utilizados para outros fins e.g. comunicação segura entre uma entidade e a CA.

Cadeias de Certificação e Confiança

- Para utilizar um serviço que requeira o conhecimento de uma chave pública, é necessário obter e validar um certificado que a contenha.

- A validação do certificado implica, por sua vez, o conhecimento da chave pública da Autoridade de Certificação que o emitiu e, consequentemente, a obtenção e validação do certificado que a contém.

- A validação do certificado da CA poderá implicar o conhecimento da Chave Pública de outra CA que o tenha emitido, e assim sucessivamente.

- Chama-se a esta sequência uma **Cadeia de Certificação**.

- Em geral, para se conhecer e confiar numa chave pública é necessário validar o certificado dessa chave, e zero ou mais certificados de CAs.

Cadeias de Certificação e Confiança$_{cont}$

- A validação de um certificado segue o seguinte algoritmo:
  - Para todo o $x = 1, ..., n-1$, o `subject` do certificado x é o emissor do certificado $x+1$;
  - O certificado 1 é emitido pela **raiz da relação de confiança**;
  - O certificado n é o certificado a ser validado; e . . .
  - Para todo o $x = 1, ..., n$, verifica-se que o certificado é válido na altura da sua utilização.

- Perguntas:
  - Onde termina a validação de uma cadeia de certificados?
  - O que é a raiz da relação de confiança?
  - Se cada chave pública implica um certificado, e vice-versa, o que aparece primeiro?

Cadeias de Certificação e Confiança$_{cont}$

- As cadeias de certificação reflectem uma **hierarquia** de Autoridades de Certificação.

- As CAs hierarquicamente superiores emitem os certificados das CAs hierarquicamente inferiores.

- No topo da hierarquia reside uma CA denominada **Root** ou raiz. O certificado desta CA é emitido e assinado por ela própria: os campos `subject` e `issuer` do seu certificado são iguais.

- A confiança na chave pública de uma Root CA **não depende** de outra CA. É estabelecida por um meio externo à PKI.

- Por exemplo, a utilização de uma instalação comum do MS Windows implica a "confiança" em dezenas de Root CAs!

Cadeias de Certificação e Confiança$_{cont}$

- Um utilizador conhece um número limitado de chaves públicas pertencentes a CAs (em geral Root CAs) e que funcionam como raizes das relações de confiança.

- Isso significa que o utilizador aceitará um certificado emitido por uma dessas CAs e que depositará um determinado nível de confiança no seu conteúdo.

- A validação de uma cadeia de certificados terminará quando for encontrado um certificado com essa característica.

- Conclusão: o grau de confiança depositada num certificado validado baseia-se apenas na confiança depositada na CA que funcionou como raiz da relação de confiança.

Políticas de Certificação

- De uma forma geral, a confiança que é depositada numa CA depende, em última instância, da sua política de certificação, e da forma como essa política é implementada.

- Essa confiança é influenciada por diversos factores internos e externos à PKI. Factores externos, como a credibilidade da instituição ou empresa que suporta a CA e o seu país de origem são obviamente importantes.

- No entanto, o conceito de PKI prevê uma forma de "ancorar" a confiança que se deposita numa CA, naquilo que de facto importa: as leis da sociedade em que a PKI opera.

- Isto é feito através das chamadas **Certificate Policies** ou CP.

Certificate Policies

- Uma Certificate Policy é um conjunto de regras que define a aplicabilidade de certificados a uma determinada comunidade ou classe de aplicações:
  - Qual a legislação em que se baseará a emissão e utilização dos certificados.
  - Quais os requisitos e as responsabilidades (nomeadamente legais e financeiras) associados a CAs e RAs.
  - Quais os requisitos e as responsabilidades associados a Titulares e Clientes.
  - Restrições ao conteúdo e utilização dos certificados e.g. apenas autenticação, somas máximas envolvidas numa transacção, etc.
  - Procedimentos a serem implementados relativamente a diversos aspectos do funcionamento de CAs e RAs.

- Cada CP é identificada por um `ObjectIdentifier` que pode ser incluido na extensão **Certificate Policies** de um certificado.

Certification Practice Statements

- Cada CA publica uma ou mais **Certification Practice Statements** (CPS), nas quais publicita as suas normas de operação internas. Uma CPS explica a forma como uma CA implementa um determinado conjunto de **Certificate Policies** (CP).

- Em geral, a acreditação de uma CA de acordo com uma determinada CPS implica uma auditoria efectuada por (ou em nome de) uma entidade denominada **Policy Management Authority**.

- Por exemplo, a PKI Governamental do Canadá define oito CPs correspondentes a quatro níveis de segurança na utilização de certificados em assinaturas digitais e protecção de dados.

- Uma CA que pretenda emitir certificados que em conformidade com estas políticas tem de ser acreditada pelo estado Canadiano.

Políticas de Certificação na Prática

- Uma parte significativa do RFC3280 é dedicada às políticas de certificação e ao efeito de uma política de certificação imposta num determinado ponto da hierarquia de certificação.

- Como foi já referido, esta especificação define também as extensões que permitem incluir este tipo de informação nos certificados X.509.

- De facto, associada a cada certificado pode estar uma lista de políticas aplicáveis à sua utilização ou, no caso do certificado de uma CA, uma lista das políticas aceitáveis para os certificados hierarquicamente inferiores.

- Durante a validação de um certificado é necessário propagar as políticas impostas desde o topo da hierarquia até à sua base.

Políticas de Certificação na Prática$_{cont}$

- A informação contida nos certificados inclui uma componente processável num algoritmo de resolução deste problema, algoritmo esse que está também definido no RFC3280.

- A política em vigor na base da hierarquia de certificação resulta da reunião das políticas em vigor nos níveis superiores, com a ressalva de que uma política inserida num determinado nível não pode contradizer uma política de nível superior.

- Compete ao utilizador determinar se a política associada a um determinado certificado é aceitável ou não.

- É também possível incluir num certificado uma CPS, de forma directa ou referenciada, bem como informação dirigida ao utilizador final sobre as condições de emissão do certificado.

Parte IV

Certificate Revocation Lists (CRL)

Introdução

- As **Certificate Revocation Lists (CRL)** são o canal previsto no X.509 para a revogação de certificados dentro do período de validade. Uma CRL diz-se:
  - **Base CRL** quando lista todos os certificados revogados por uma CA que ainda estão no seu período de validade.
  - **Delta CRL** quando apenas lista os certificados revogados desde a publicação de uma Base CRL referenciada.

- As CRLs são emitidas, em geral, pelas próprias CAs. No entanto é possível que a CA delegue esta função numa outra autoridade denominada **CRL Issuer**. Uma CRL emitida nestas condições denomina-se **CRL Indirecta**.

- Cada CRL tem um contexto específico: o conjunto de certificados passíveis de aparecerem no seu conteúdo. Para que uma CRL possa ser utilizada eficientemente, este contexto deve estar bem definido.

CRLs: Estrutura e Atributos

- Uma CRL contém os seguintes campos:
  - **Version** version
  - **AlgorithmIdentifier** signature
  - **Name** issuer
  - **Time** thisUpdate
  - **Time** nextUpdate
  - Lista de certificados revogados.

- A assinatura da CRL é efectuada pela Autoridade de Certificação (CA) sobre uma codificação DER da representação desta estrutura de dados em ASN.1.

- O campo **version** é opcional e deve ter o valor 2 uma vez que as CRLs foram introduzidas na versão 2 do X.509. O campo **signature** tem o mesmo significado que nos certificados.

CRL: Estrutura e Atributos$_{cont}$

- O campo **thisUpdate** indica a data de publicação da CRL.
- O campo **nextUpdate** é muito importante uma vez que permite ao utilizador conhecer uma data a partir da qual é garantido ter sido publicada uma nova CRL.
- A lista de certificados revogados é representada por uma sequência de registos que indicam o titular e número de série de cada certificado, bem como a data em que foi revogado.
- A definição de CRL contempla também a inclusão de extensões, tanto dos atributos globais da CRL, como dos atributos associados a cada certificado revogado.

CRL: Extensões Globais

- **Authority Key Identifier** Semelhante à utilizada nos certificados, permite identificar o certificado que contém a chave pública da CA que permite validar a CRL.

- **Issuer Alternative Name** Semelhante à utilizada nos certificados, permite indicar identificações alternativas para a CA.

- **CRL Number** Permite incluir um número sequencial que serve como número de ordem da CRL dentro do seu contexto específico.

- O CRL Number serve também para as Delta CRLs referenciarem as Base CRLs correspondentes.

CRL: Extensões Globais$_{cont}$

- **Delta CRL Indicator** Marca a CRL como sendo uma Delta CRL, e referencia a Base CRL correspondente.

- **Issuing Distribution Point** Identifica o ponto de distribuição da CRL, bem como o seu contexto e.g. o tipo de certificados que revoga, as possíveis razões de revogação, etc.

- **Delta CRL Distribution Point** Permite, nas Base CRLs, indicar onde pode ser obtida a Delta CRL correspondente mais recente.

- Os pontos de distribuição são indicados como URIs, e apontam sempre para a CRL mais recente, num dado contexto.

CRL: Extensões às Entradas

- **Reason Code** Permite indicar uma razão para o aparecimento do certificado na CRL.

- Em geral, a razão será a revogação de um certificado por um determinado motivo e.g. chave privada comprometida, segurança da CA comprometida, alteração de filiação ou de previlégios, existência de uma versão mais recente do certificado, fim da operação da CA.

- No entanto, é possível um certificado não ser imediatamente revogado, mas ser suspenso. O valor `certificateHold` está definido para esta extensão precisamente para este efeito.

- Um certificado suspenso numa CRL pode ser reactivado numa CRL posterior atribuindo o valor `removeFromCRL` a esta extensão.

CRL: Extensões às Entradas$_{cont}$

- **Hold Instruction Code** Quando um certificado é suspenso, é possível indicar a acção que deve ser tomada quando a utilização do certificado for necessária: rejeitar o certificado ou contactar a CA.

- **Invalidity Date** Permite indicar a data em que se suspeita que o certificado deixou de ser válido.

- **Certificate Issuer** Nas CRLs Indirectas utiliza-se esta extensão para identificar a CA que emitiu o certificado revogado.

CRL: Publicação periódica e segurança

- A segurança de uma PKI depende da eficácia com que são revogados os certificados que se tornaram inválidos. Este facto sugere que assim que um certificado se torna inválido uma nova CRL deva ser publicada.

- No entando, desta forma, um utilizador nunca saberia qual a CRL mais recente. Isto possibilitaria ataques do tipo:
  - Um intruso controla o meio de comunicação que liga o utilizador ao ponto de publicação de uma CRL.
  - O utilizador pretende utilizar um certificado cuja chave privada foi comprometida, e que é conhecida pelo intruso.
  - O utilizador tenta obter a CRL mais recente, que revogaria o certificado. O intruso fornece-lhe uma versão antiga da CRL.
  - O utilizador aceita o certificado porque não tem como saber que a CRL que utilizou estava desactualizada.

CRL: Publicação periódica e segurança$_{cont}$

- De facto, a utilidade de uma CRL depende do facto de ela ser publicada periódicamente e.g. diáriamente, semanalmente, mensalmente, etc.

- Isto permite também que a CRL seja pública, e distribuida por canais não seguros.

- Compete ao utilizador estar ao corrente da frequência de publicação das CRLs, e definir uma política sobre o que é uma CRL "suficientemente recente".

- O atributo `nextUpdate` permite indicar na própria CRL a altura a partir da qual é garantida a publicação de uma nova CRL.

CRL: Publicação periódica e segurança$_{cont}$

- O utilizador está consciente de que, a menos que obtenha a última versão da CRL, estará a correr o risco de aceitar certificados inválidos.

- Isto não quer dizer que não possam ser publicadas CRLs extraordinárias, fora da frequência normal de publicação.

- Isto pode ocorrer, por exemplo, se um certificado importante tem de ser revogado porque a chave privada correspondente foi comprometida e.g. o certificado de uma CA hierarquicamente inferior.

- No entanto, a granularidade garantida nunca é inferior ao período de publicação da CRL: não é possível garantir que os utilizadores obtenham a CRL extraordinária antes da data de publicação da próxima CRL periódica.

CRLs: Codificação

- As CRLs, como todas as estruturas de dados na PKI, são definidos e representados em ASN.1. A codificação é feita utilizando as Dinstinguished Encoding Rules (DER).

- O ficheiro que contém uma CRL consiste na codificação DER da seguinte estrutura ASN.1:

```
CertificateList  ::=  SEQUENCE  {
        tbsCertList           TBSCertList,
        signatureAlgorithm    AlgorithmIdentifier,
        signatureValue        BIT STRING  }
```

- Além da estrutura de atributos apresentada nos slides anteriores (**tbsCertificate**), aparece também a assinatura da Autoridade de Certificação (**signatureAlgorithm** e **signatureValue**).

Parte V

Problemas do X.509

Principais Críticas

- O X.509 baseia-se num sistema de directório. A chave primária desse directório é o Distinguished Name (DN): um DN tem de ser único. Mas como é que se garante esta unicidade?

- Se cada DN apenas pode ser associado a uma entidade, nada impede uma entidade de ter vários DNs. Além disso, nada obriga a que o DN contenha elementos de identificação baseados na identidade real da entidade. Assim, um certificado associa um DN a uma chave pública, mas não se pode dizer que identifique a entidade.

- As garantias que se esperam de uma CA relativamente à verificação da informação constante de um certificado não são abrangidas pela norma!

Principais Críticas$_{cont}$

- Em resumo, o X.509 não foca o esforço necessário para validar a informação constante num certificado, nem define um significado global para essa informação, um significado absoluto, exterior às CAs.

- O X.509 remete para as CPSs das CAs todos os aspectos relacionados com confiança e com semântica da informação constante num certificado.

- Os certificados não são legíveis directamente pelo utilizador: quem garante que uma aplicação mostra ao utilizador toda a informação relevante acerca de um certificado?

- O X.509 implica a utilização de um directório ou repositório para distribuição de certificados e CRLs.

Principais Críticas$_{cont}$

- As CRLs são muito pouco utilizadas, e reconhecidas por poucos como a solução para o problema das revogações de certificados.

- O conceito de "raiz da relação de confiança" e Root CA transfere o problema da confiança de um nível local para um nível global, potenciando os efeitos da ignorância ou da fraude.

- O processo de inicialização e manutenção de uma lista de certificados, nomeadamente de Root CAs em que uma entidade confia, está mal definido.

- Qualquer assinatura torna-se efectivamente inválida se deixar de existir um certificado válido que permita verifica-la!

- Qual a duração ideal do período de validade de um certificado?

Parte VI

Online Certificate Status Protocol OCSP

Introdução

- Há aplicações em que os riscos associados à utilização indevida de um certificado revogado podem não ser aceitáveis.

- Em alternativa ou adição à consulta de uma CRL, pode ser necessária informação actual sobre o estado de revogação de um certificado.

- O OCSP permite a uma aplicação determinar o estado de um certificado, com uma frescura temporal maior do que é possível com uma CRL, ou mesmo para obter informação adicional sobre o certificado.

- O Cliente OCSP emite um pedido a um Responder OCSP (Servidor) e suspende a aceitação do certificado até que este forneça uma resposta.

- Quando uma CA suporta este serviço, isso deve ser indicado nos certificados através da extensão `AuthorityInfoAccess`.

Pedidos OCSP

- Um pedido OCSP contém a seguinte informação, podendo ser assinados:
  - Versão do protocolo
  - Identificação da entidade que emite o pedido
  - Identificador(es) do(s) certificado(s) alvo
  - Extensões opcionais que podem diferenciar o serviço requisitado.

- O processamento de um pedido pelo Responder passa pelas seguintes fases:
  - Verificação do formato da mensagem.
  - Verificação de que o servidor está configurado para fornecer o serviço requisitado.
  - Verficação de que o pedido contém toda a informação necessária.
  - Construção da resposta.

Serviços OCSP

- O OCSP é standardizado no RFC2560. Neste documento é apenas definido um serviço básico que iremos estudar nos próximos slides.

- A IETF publicou entretanto um Internet Draft definindo o OCSP V2, uma extensão à funcionalidade básica que define três serviços:
  - Online Revocation Status (ORS)
  - Delegated Path Validation (DPV)
  - Delegated Path Discovery (DPD)

- O serviço é indicado numa extensão incluída nos pedidos OCSP no campo `requestExtensions`.

- O ORS corresponde à funcionalidade básica e é o caso default i.e. quando nenhum serviço é indicado trata-se de uma invocação do ORS.

Conteúdo de um pedido OCSP

- Num pedido OCSP, os certificados para os quais é solicitada a informação de revogação são geralmente indicados numa lista de estruturas ASN.1 `CertID`:

```
CertID          ::=     SEQUENCE {
    hashAlgorithm       AlgorithmIdentifier,
    issuerNameHash      OCTET STRING,
                         -- Hash of Issuer's DN
    issuerKeyHash       OCTET STRING,
                         -- Hash of Issuers public key
    serialNumber        CertificateSerialNumber }
```

- A CA emissora de cada certificado é indicada através de valores de hash do seu *Distinguished Name* e da sua chave pública.

Respostas OCSP

- As respostas OCSP consistem num envelope contendo
  - um **código de status** que indica se o pedido foi processado correctamente ou não, e
  - uma **sequência de bytes com seu conteúdo**, no caso de sucesso.

- As respostas OCSP são assinadas digitalmente por uma das seguintes entidades:
  - A CA que emitiu o certificado em questão.
  - Um CA Designated Responder i.e. um Responder autorizado pela CA que emitiu o certificado em questão (detentor de um certificado com uma extensão específica para este fim – `extendedKeyUsage` – emitido pela mesma CA).

Respostas OCSP$_{cont}$

- O conteúdo da resposta consiste nos seguintes itens:
  - A versão do protocolo da resposta.
  - O nome do Responder e a data da geração da resposta.
  - **Respostas para cada um dos certificados contidos no pedido**.
  - Uma assinatura digital da resposta.

- Para cada certificado é enviada a seguinte informação:
  - O identificador do certificado alvo.
  - O estado do certificado: **good**, **revoked** ou **unknown**.
  - Período de validade da informação de estado do certificado.

- Uma resposta **good** significa apenas que não foi encontrado nenhum registo de revogação. Não indica que o certificado sequer exista!

Respostas OCSP$_{cont}$

- A indicação da validade da resposta pode ser feita de três formas diferentes:
  - **thisUpdate** Hora a que o Responder sabe que a resposta é correcta.
  - **nextUpdate** Hora a que o Responder sabe que vai ser capaz de fornecer uma resposta actualizada. Se não for indicada indica que qualquer nova resposta conterá informação actualizada.

- A validação de uma resposta OCSP por parte de um cliente implica verificar que:
  - há correspondência entre os certificados do pedido e da resposta.
  - a assinatura da resposta é válida e provém de um agente autorizado.
  - a validade da resposta é suficientemente recente.

Respostas OCSP$_{cont}$

- No serviço OCSP básico a resposta usa a seguinte estrutura ASN.1:

```
ResponseData ::= SEQUENCE {
   version                 [0] EXPLICIT Version DEFAULT v1,
   responderID                 ResponderID,
   producedAt                  GeneralizedTime,
   responses                   SEQUENCE OF SingleResponse,
   responseExtensions      [1] EXPLICIT Extensions OPTIONAL }

SingleResponse ::= SEQUENCE {
   certID                          CertID,
   certStatus                      CertStatus,
   thisUpdate                      GeneralizedTime,
   nextUpdate       [0]     EXPLICIT GeneralizedTime OPTIONAL,
   ...  }
```

Outros serviços: Delegated Path Validation

- Este serviço permite a uma aplicação transferir o processamento da validação de cadeias de certificados complexas para um servidor central.

- Isto permite simplificar as aplicações cliente, reduzindo ao mínimo a funcionalidade de validação de certificados a implementar.

- O protocolo permite que a aplicação cliente controle os aspectos essenciais do comportamento do servidor, por exemplo:
  - Restringir as cadeias de certificação que o servidor deve aceitar.
  - Se o servidor se deve basear em CRLs e/ou OCSP para fazer a validação.
  - Quais as políticas de certificação que são relevantes para a aplicação.

Outros serviços: Delegated Path Discovery

- Este serviço permite a uma aplicação que processe certificados obter do servidor a informação disponível sobre um certificado e a sua revogação: cadeias de certificados, CRLs, respostas de outros servidores OCSP, etc.

- Deste modo, a aplicação pode validar certificados obtendo informação de todos os pontos de acesso à PKI disponíveis no servidor OCSP e.g. X.500, LDAP, HTTP, FTP, etc.

- Tal como no serviço anterior, a aplicação pode controlar diversos aspectos da operação do servidor:
  - Restringir as cadeias de certificação que o servidor deve aceitar.
  - Quais as políticas de certificação que são relevantes para a aplicação.
  - Mecanismo interactivo de selecção de uma cadeia de validação aceitável para o cliente em termos de políticas de certificação.

Parte VII

Internet Protocol Security (IPsec)

Introdução

- O protocolo IP (Internet Protocol) está ao nível da camada de rede do Modelo OSI. Fornece essencialmente 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 seguraça.

- Estes serviços são oferecidos ao nível da camada de rede, oferecendo protecção não só esse nível, mas também a todas as camadas superiores.

- O IPsec está definido nas especificações RFC2401, e seguintes, da IETF.

Introdução$_{cont}$

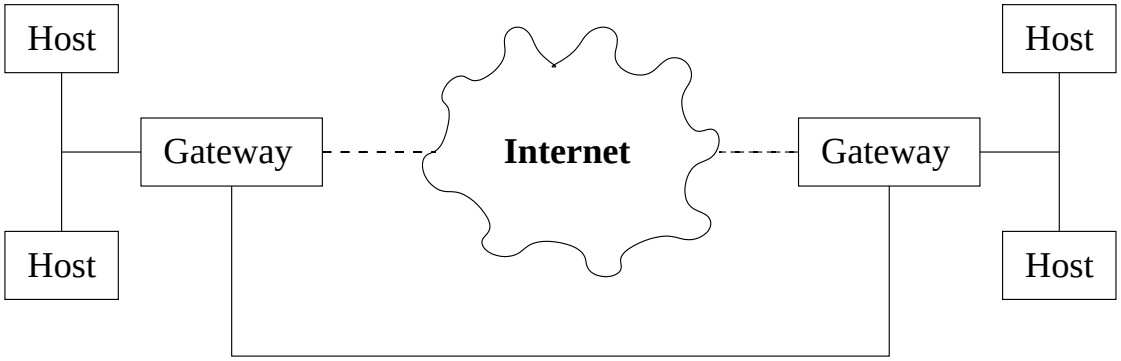
- O IPsec oferece os seguintes serviços seguros ao nível da camada de rede:
  - Controlo de acessos
  - Integridade ao nível do pacote
  - Autenticação da origem de dados
  - Protecção contra pacotes repetidos
  - Confidencialidade
  - Confidencialidade de parte do tráfego

- 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

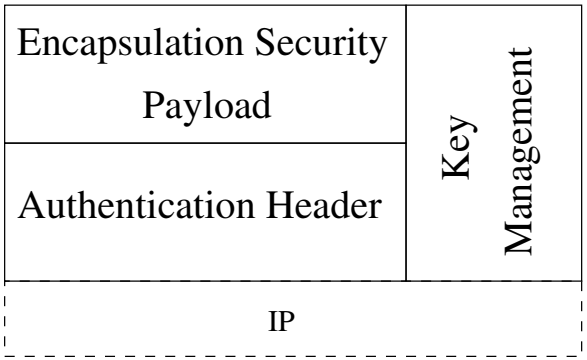
- Funcionamento:



  - Dentro de uma rede local, cada nó constroi 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

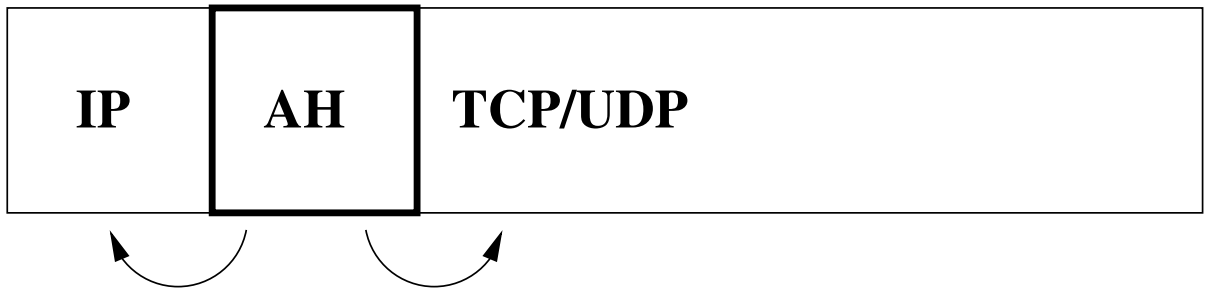
- O IPsec está estruturado em duas sub-camadas:



- 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.

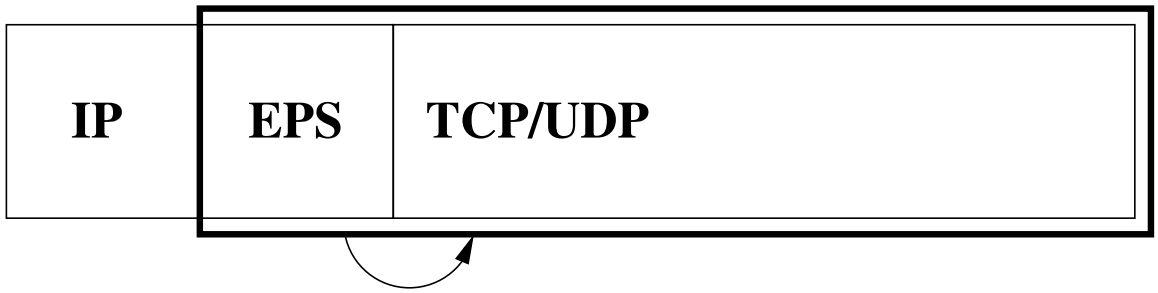
IP Authentication Header



- 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).

- Apesar disto, o ESP pode ser, e geralmente é, utilizado isoladamente.

Modos de Funcionamento

- Consoante o tipo de nós envolvidos, pode funcionar em dois modos:
  - **Transport Mode** – Operam 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** – Operam 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 no conceito de **Security Association** (SA).
- 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 IPsec (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 pelo IPsec 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 seguros de geração e acordo de chaves.

Tratamento de Pacotes Recebidos

1. A informação contida no cabeçalho IPsec indica uma entrada na SAD:

2. Se essa entrada não existir desencadeia-se, se possível, a negociação de chaves para configurar uma nova SA. Se isto não for possível, o pacote é descartado.

3. Com base na informação associada à SA, processam-se os cabeçalhos e descodifica-se o pacote.

4. Consulta-se outra vez a SPD para saber se o pacote foi processado de acordo com a política correcta.

5. Se o processo decorrer sem problemas passa-se o pacote à camada de transporte.

Tratamento de Pacotes Enviados

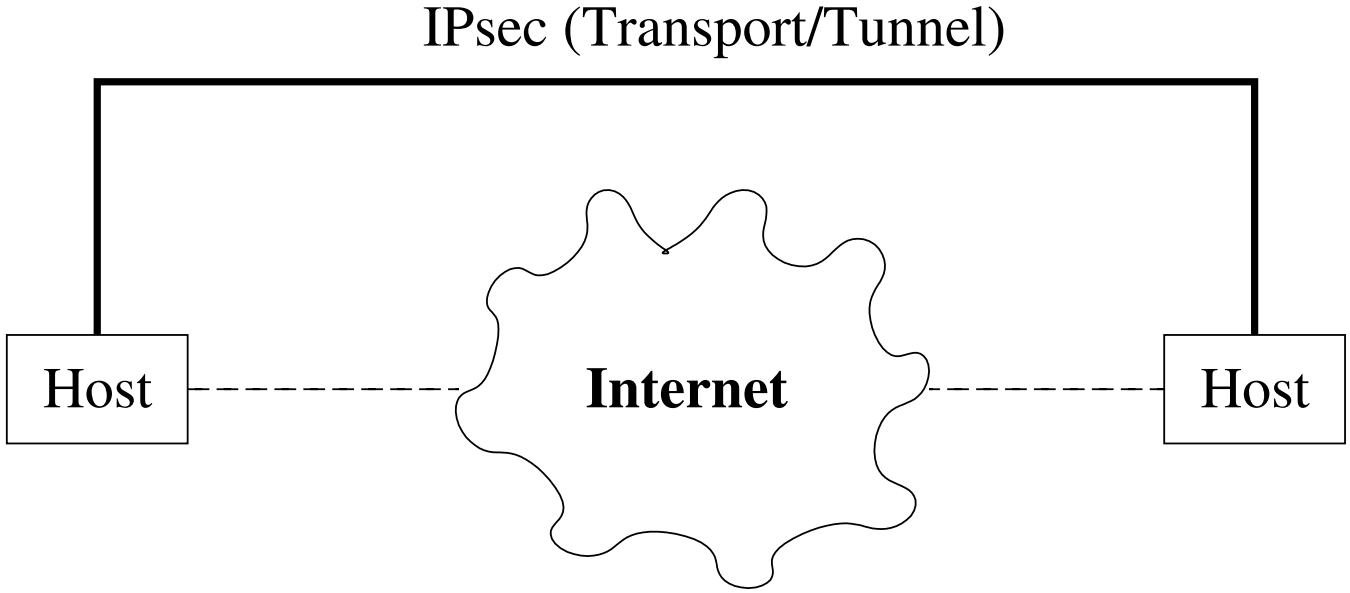
1. O pacote é procurado na SPD que indica se deve ser protegido, enviado sem protecção, ou descartado.

2. Se a opção for proteger o pacote, a SPD conterá ligações para as entradas da SAD que contêm a(s) SA(s) correspondente(s).

3. Se a SA existir, o pacote é processado em conformidade com a informação correspondente e enviado para o seu destino.

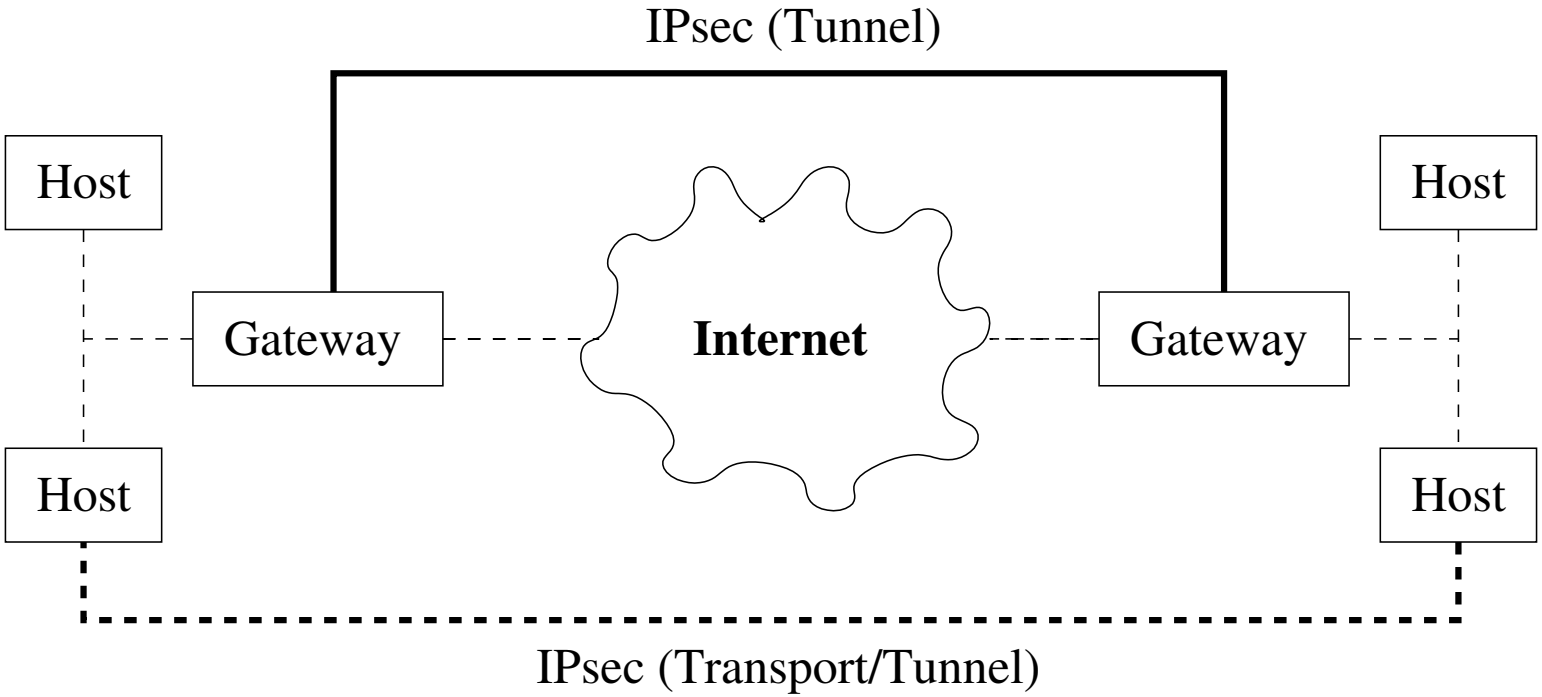
4. Se a SA não existir, e caso isso seja possível, cria-se uma nova SA com base num processo automático de negociação de chaves. Caso contrário o pacote é descartado.

Ligações Host-to-Host



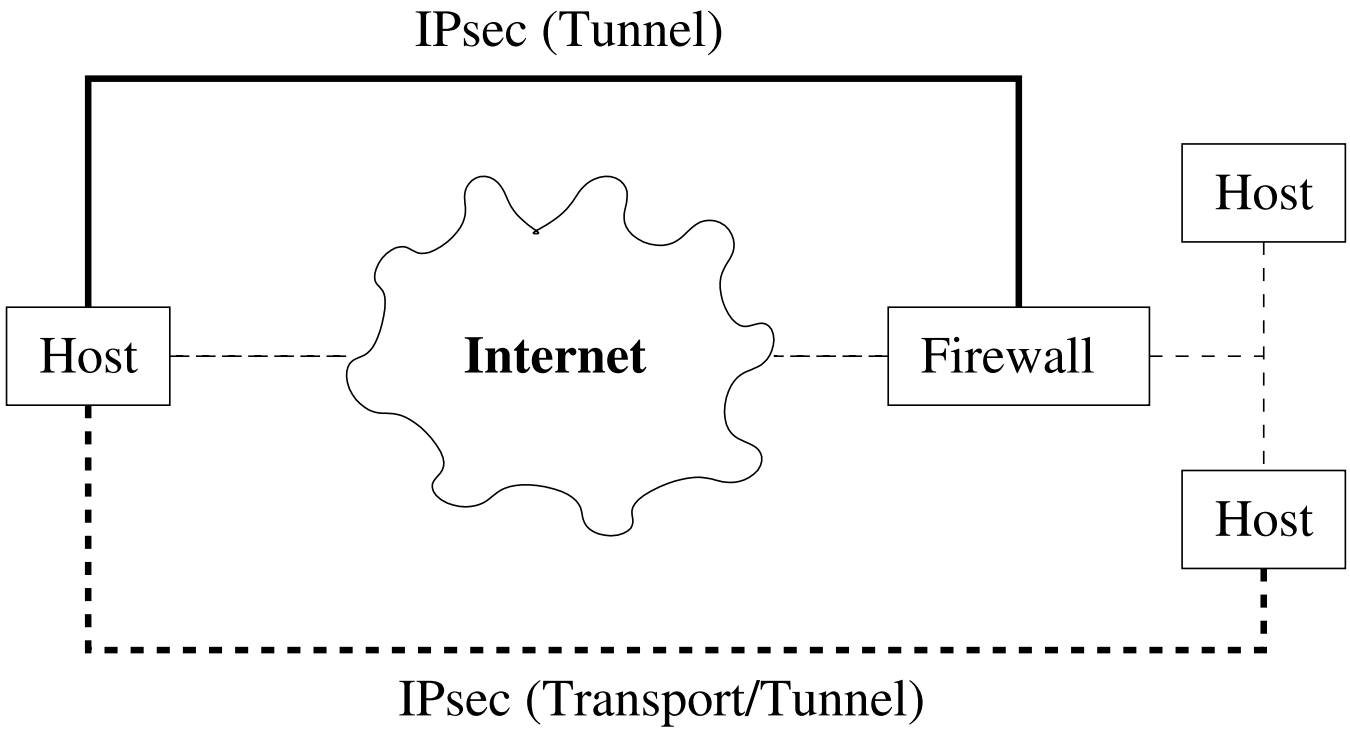
- Os pacotes trocados entre as duas máquinas são protegidos com IPsec, com base em Transport SAs.
- Pode ser utilizado o modo Tunnel, mas isso é redundante: o cabeçalho externo terá os mesmos endereços que o interno.

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, atribuída ao facto do seu desenvolvimento ter estado 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?

Críticas ao IPsec$_{cont}$

- 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.

Parte VIII

Pretty Good Privacy (PGP)

Introdução

- O PGP é uma aplicação freeware desenvolvida por Phil Zimmermann com o objectivo de colocar uma infraestrutura para protecção de informação, i.e. manutenção de privacidade, à disposição do cidadão comum.

- O lançamento deste software causou alguma polémica nos EUA devido às leis que restringiam a difusão e utilização generalizada da chamada *strong cryptography*.

- Até hoje prevalece a ideia, nomeadamente nos EUA, de que a utilização do PGP é ilegal.

- Um argumento apresentado a favor da utilização do PGP, e contra as restrições ao uso de sistemas de protecção da privacidade é: "se a protecção de informação é ilegalizada apenas os fora-da-lei conseguem ter privacidade!"

Funcionamento do PGP

- O PGP é um sistema com três vertentes principais: privacidade, integridade e autenticação, e certificação.

- **Privacidade** Define um conjunto de regras para a utilização de algoritmos de compressão, cifras simétricas e assimétricas na protecção de informação, nomeadamente ficheiros e mensagens de e-mail.

- **Integridade e Autenticação** Define um conjunto de regras para a utilização de funções de hash criptográficas e algoritmos de assinatura digital para a assinatura de mensagens e documentos.

- **Certificação** Define um conjunto de regras para o estabelecimento de relações de confiança e distribuição de chaves públicas com base num esquema certificação próprio, alternativo à PKI e ao X.509.

Cifragem

- Quando um utilizador cifra um texto limpo utilizando o PGP, o primeiro passo é a compressão dessa informação.

- A compressão tem duas vantagens:
  - Poupa espaço e largura de banda.
  - Esconde padrões existentes no texto limpo aumentando a segurança.

- O segundo passo consiste na criação de uma chave de sessão. A chave secreta é criada utilizando um gerador de números (pseudo-)aleatórios.

- A aleatoriedade deste gerador baseia-se na monitorização de movimentos do rato.

Cifragem$_{cont}$

- A mensagem comprimida é depois cifrada com a chave de sessão utilizando um algoritmo de cifra simétrica.

- Finalmente, a chave de sessão é cifrada com a chave pública do destinatário utilizando um algoritmo de cifra assimétrica.

- A mensagem PGP é constituida pelos dois criptogramas: é um envelope digital.

- O processo de decifragem é inverso:
  - Decifragem da chave secreta utilizando a chave privada do destinatário.
  - Recuperação da mensagem comprimida utilizando a chave de sessão.
  - Descompressão do texto limpo.

Assinaturas Digitais

- O PGP permite efectuar assinaturas digitais sobre documentos, de duas formas:
  - Assinaturas que permanecem ligadas aos documentos a que correspondem, na forma de attachments.
  - Assinaturas que existem isoladamente dos documentos a que correspondem. Para quê?

- As assinaturas geradas pelo PGP seguem o procedimento standard:
  - Primeiro o documento é passado por uma função de hash criptográfica.
  - O valor de hash é então assinado com a chave privada do utilizador utilizando um algoritmo de assinatura digital.

Chaves

- O PGP armazena as chaves que um utilizador conhece em dois ficheiros denominados **keyrings**:
  - **Private Keyring** Ficheiro onde o PGP guarda as chaves privadas do utilizador (cifradas com PBE utilizando uma $passphrase$).
  - **Public Keyring** Ficheiro onde o PGP guarda as chaves públicas dos interlocutores do utilizador.

- As chaves públicas conhecidas por um utilizador pertencem a outros utilizadores do sistema PGP.

- A introdução destas chaves públicas no sistema do utilizador pode ser feita directamente, apresentando o seu valor numérico, ou através de certificados PGP trocados com outros utilizadores.

Certificação

- Um certificado PGP contém os seguintes itens de informação:
  - A versão do PGP utilizada.
  - Uma chave pública e informação sobre o seu tipo.
  - A identificação do detentor da chave privada correspondente à chave pública. Esta informação pode incluir um User ID, endereço de e-mail, número do ICQ, uma fotografia, etc.
  - O período de validade do certificado.
  - O algoritmo de cifra simétrica preferido pelo titular para receber informação cifrada.
  - A assinatura do detentor da chave pública, que atesta a associação identidade/chave pública, e demonstra o conhecimento da chave privada (**self-signature**).
  - Assinaturas de outros utilizadores PGP que corroboram a informação patente no certificado, ou parte dela.

Certificação$_{cont}$

- Um certificado PGP pode incluir diversas instancias/versões da identidade do utilizador assinadas separadamente por diversos utilizadores.

- As grandes diferenças entre os certificados PGP e os certificados X.509 são evidentes:
  - Os emissores de certificados PGP são utilizadores comuns do sistema PGP.
  - A confiança não é transitiva: A confia em B e B confia em C, não implica que A confia em C.
  - Não existe a figura de $agente\ de\ confiança$ ou de hierarquia/cadeia de certificação.

- Cada certificado pode ser assinado por vários utilizadores PGP. Pode dizer-se que a credibilidade de um certificado PGP pode ser reforçada desta forma.

Certificação$_{cont}$

- Cada utilizador do PGP tem de decidir quais são os utilizadores em que confia para assinar certificados. Por exemplo:
  - "Confio em B quando este assina um certificado que o indica como o titular da chave K_B", ou
  - "Confio em B quando este assina um certificado que indica C como o titular da chave K_C".

- Quando um utilizador se assegura (de uma forma que julgue segura) de que uma chave pública pertence a um utilizador, pode assinar a cópia dessa chave no seu Keyring.

- Se assim pretender, o utilizador pode exportar esse certificado PGP para um **PGP Key Server** (repositório de certificados PGP) para que outros utilizadores possam usufruir dessa informação.

Confiança e Validade de Chaves Públicas

- Uma forma de testar a validade de uma chave pública é através de um processo manual qualquer e.g. exigir que o destinatário de uma mensagem cifrada entregasse em mãos uma cópia da sua chave pública.

- Outra forma é confiar em alguém que afirma ter feito esse tipo de validação. Numa PKI, este é o papel de uma autoridade de certificação.

- O PGP permite estas duas formas de aceitar uma chave pública como válida e acrescenta-la ao Keyring do utilizador.

- Em termos de relações de confiança, diz-se que o PGP permite implementar três modelos: **confiança directa**, **confiança hierarquica** e **rede de confiança**.

Confiança Directa

- O modelo de confiança directa é o mais básico.
- Cada par de utilizadores troca as suas chaves públicas entre si, directamente.
- A validação das chaves públicas e o estabelecimento da confiança resultam sempre de uma relação em primeiro grau.
- Por outras palavras, nenhum utilizador confia em intermediários no estabelecimento de confiança numa chave pública.
- É equivalente à relação de confiança estabelecida entre uma entidade e uma Root CA dentro de uma PKI.

Confiança Hierárquica

- É equivalente às cadeias de certificação da PKI.

- No PGP existem dois tipos de agentes de confiança:
  - **Trusted Introducer** É um utilizador em que se confia para certificar chaves públicas de outros utilizadores.
  - **Meta Introducer** É um utilizador PGP no qual se confia, não só para certificar chaves públicas, mas também para indicar outros utilizadores PGP como **Trusted Introducers** válidos. É equivalente a uma Root CA.

- Com o PGP é possível implementar uma hierarquia de relações de confiança semelhante à existente numa PKI.

- Por exemplo, reconhecendo um Meta Introducer e certificando este alguns Trusted Introducers obtém-se uma "PKI" com dois níveis hierárquicos.

Rede de Confiança

- A Rede (Teia) de Confiança é um modelo misto, em que se utilizam os dois anteriores.

- É um modelo mais próximo da visão da confiança que existe no mundo real, e evidencia o facto de que a confiança é um conceito que depende da perspectiva do utilizador.

- A validação de uma chave pública pode ser obtida directamente, ou através de uma cadeia que termina com um Meta Introducer ou com um conjunto suficientemente grande de Trusted Introducers.

- O PGP introduz o conceito de "aritmética" nas relações de confiança através do qual é possível construir confiança de uma forma cumulativa.

Níveis de Confiança e Validade

- Armazenado no Public Keyring de um utilizador PGP está a seguinte informação:
  - Se o utilizador considera uma determinada chave pública válida.
  - O nível de confiança que o utilizador deposita nessa chave e, em particular no titular dessa chave, para certificar as chaves de outros utilizadores.

- O mais alto nível de confiança é a **Confiança Implícita** que é aquela que se deposita na nossa chave pública.

- Isto significa que o PGP assume que o utilizador confia em tudo o que foi assinado com a sua própria chave pública.

Níveis de Confiança e Validade$_{cont}$

- Às chaves públicas dos outros utilizadores podem ser atríbuidos três níveis de confiança:
  - **Completa** O titular da chave é aceite como um Trusted Introducer digno de confiança.
  - **Marginal** O titular da chave não é aceite como Trusted Introducer, mas a informação por ele fornecida pode ser utilizada na validação de chaves públicas.
  - **Nula** Não se confia neste utilizador para certificar chaves públicas de outros utilizadores.

- A validação automática de uma chave pública requer pelo menos:
  - Certificação utilizando uma chave pública com confiança completa, ou
  - Certificação por duas chaves públicas com confiança marginal.

Revogação de Certificados

- Um utilizador PGP pode, em qualquer altura, revogar uma assinatura (certificação) sua: basta que ainda possua a mesma chave prívada.

- Isto significa simplesmente que aquele utilizador já não acredita que a chave pública em questão seja válida. Mas . . .

- Se um utilizador reconhece uma chave como válida devido a assinaturas em que confia marginalmente, a revogação de uma dessas assinaturas pode não implicar a invalidação da chave.

- No entanto, o PGP permite sempre a um utilizador invalidar totalmente a sua própria chave directamente, ou através de outro utilizador: o **revoker**. O revoker é designado na altura da emissão do certificado.

- Uma revogação é geralmente distribuida via um PGP Key Server.

Ficha Técnica

- **Cifras Simétricas**: CAST, Triple-DES, IDEA, Two-Fish
- **Algoritmos de Compressão**: ZIP
- **Funções de Hash Criptográficas**: SHA-1, MD-5 (versões mais antigas)
- **Cifras Assimétricas**: RSA, El Gamal
- **Assinaturas Digitais**: RSA, Digital Signature Algorithm

Parte IX

Secure Sockets Layer (SSL)
Transport Layer Security (TLS)

Introdução

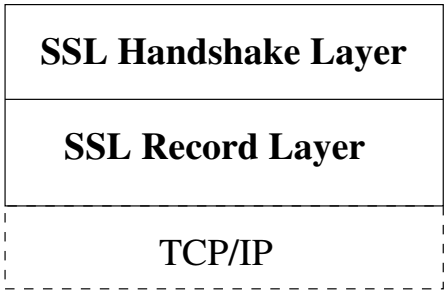
- A Secure Sockets Layer está para o TCP como o IPsec está para o IP. É um $upgrade$ da camada de transporte para incluir segurança nas comunicações.

- 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 mutua entre clientes e servidores, e a negociação de algoritmos e chaves criptográficos 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 Hanshake 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

- Esta camada recebe informação arbitrária das camadas superiores, em blocos de dados de tamanho variável.

- Os dados são **fragmentados** (ou aglutinados) em blocos com um máximo de 2^{14} bytes denominados **SSL Plaintext**.

- Os blocos SSL Plaintext são comprimidos com o algoritmo escolhido para a sessão, e transformados em 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.

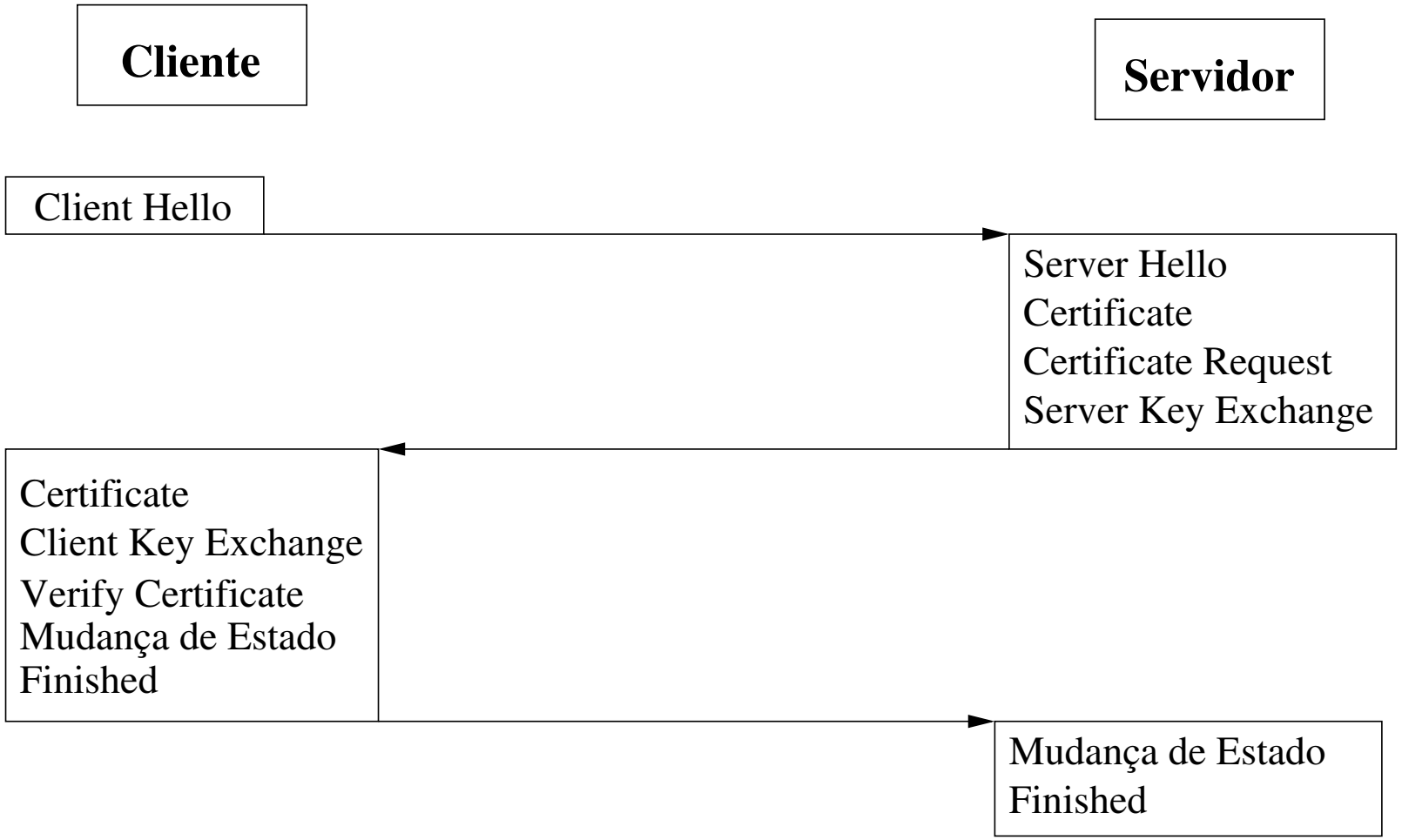
Handshake Layer

- Os parâmetros de sessão e ligação utilizados pela Record Layer são estabelecidos pela Hanshake 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 concluida.

- Uma negociação é iniciada pelo Cliente com uma mensagem **Client Hello**. O Servidor deve responder com uma mensagem equivalente. Após esta troca ficam acordados:
  - A versão do protocolo SSL a utilizar
  - O identificador da sessão
  - Os algoritmos criptográficos a utilizar (os mais fortes dos suportados).
  - O algoritmo de compressão a utilizar

Handshake Layer$_{cont}$



Handshake Layer$_{cont}$

- 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**.

Handshake Layer$_{cont}$

- 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.

Handshake Layer$_{cont}$

- A autenticação do servidor fica implicita 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. No entanto, esta versão é uma evolução em relação às versões anteriores, muito criticadas por conterem falhas de segurança importantes.

- Um dos problemas mais conhecidos na versão 2 do SSL era a vulnerabilidade a um ataque chamado "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!

Segurança$_{cont}$

- 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**.

- Outro ataque teóricamente possível sobre uma implementação pouco cuidada do SSL chama-se "change cipher spec dropping":
  - Quando a sessão a ser negociada inclui apenas autenticação, i.e. não inclui cifragem, as mensagens **finished** podem ser editadas retirando-se 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.

Parte X

Kerberos

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.

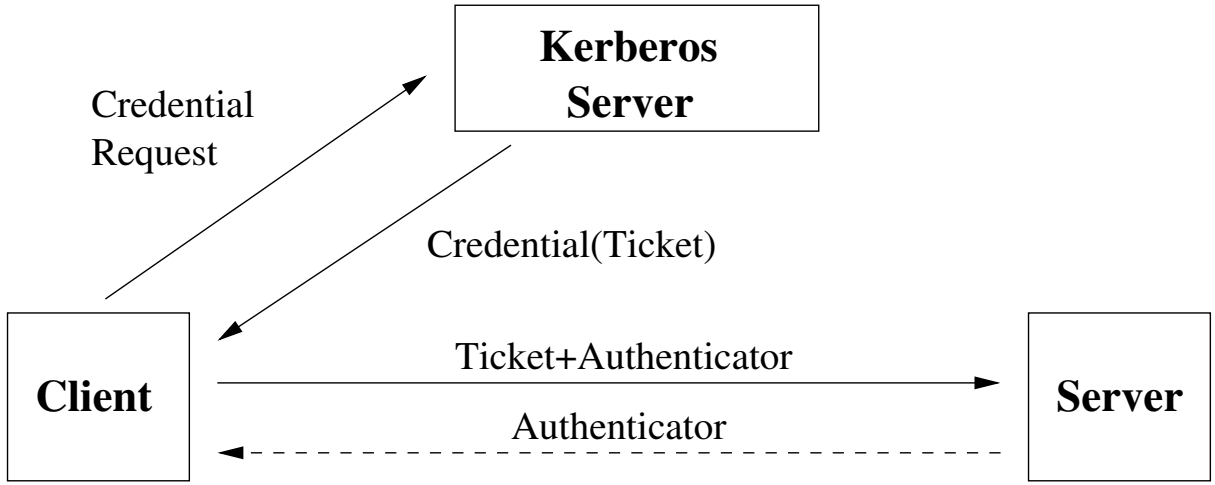
- Actualmente na versão 5, o Kerberos é utilizado na Internet com base em Internet Standards e RFCs publicados pela IETF.

- Os serviços Kerberos são oferecidos às aplicações através de uma API definida no RFC1964.

Introdução$_{cont}$

- O Kerberos baseia-se em Criptografia Simétrica e num sistema de autenticação por um agente de confiança, com pré-distribuição de chaves.
- É atribuida 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.
- Essa 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 efectua a autenticação.
- Todas as mensagens são definidas/codificadas utilizando ASN.1/DER.

Autenticação Básica$_{cont}$

- 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**.

Autenticação Básica$_{cont}$

- 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.

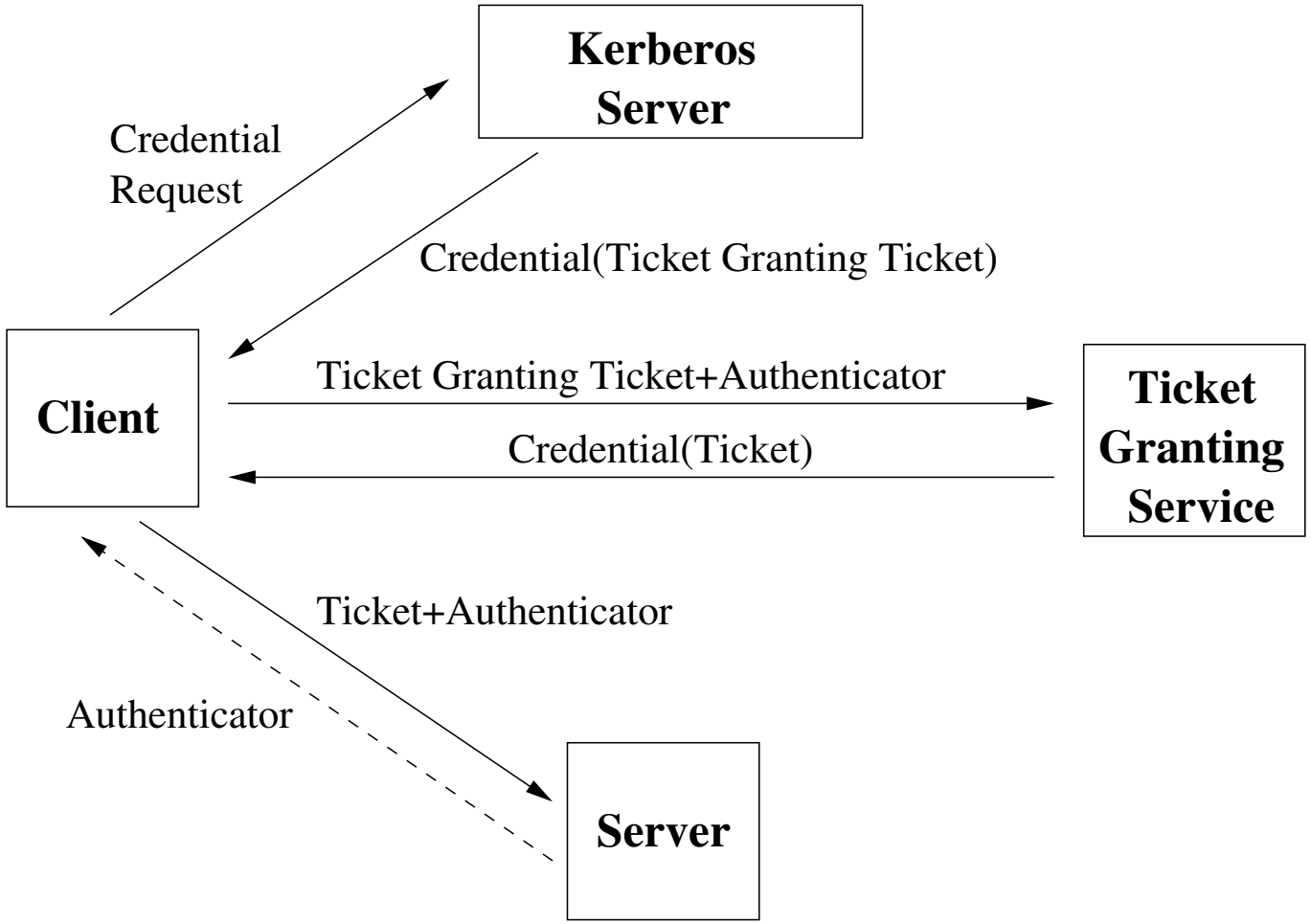
Servidor Kerberos

- 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.

Servidor Kerberos$_{cont}$



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 incluido 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 incluido no nome de todos os utilizadores nele registados, e pode servir para um Servidor Kerberos noutro 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.

Domínios (Realm) Kerberos$_{cont}$

- 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 então obter, no seu Domínio, um Ticket Granting Ticket para o Ticket Granting Server de outro 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 incluido na mensagem Ticket.

Tickets: 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. Implicitamente 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.

Tickets: Alguns Atributos/Flags$_{cont}$

- **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 substituida 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 substituida por um par de chaves.

- Estas extensões básicamente definem procedimentos de geração e formatos de transferencia alternativos para as mensagens Kerberos correspondentes a estes pontos de operação.

Extensões de Criptografia Chave Pública$_{cont}$

- Por exemplo, as alterações a um Pedido de Ticket básico são as seguintes:
  - O Cliente junta ao seu pedido de Ticket o seu Certificado de Chave Pública e uma assinatura digital do próprio pedido.
  - A Credential devolvida pelo Authentication Server passa a vir cifrada:
    * utilizando o RSA, caso a Chave Pública do Cliente o permita, ou
    * utilizando 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*, em que se tira partido da fraca qualidade das passwords auto-atribuidas.

- 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

Parte XI

Secure Shell (SSH)

Introdução

- O SSH é um protocolo para o estabelecimento de serviços de *shell* seguros, nomeadamente de login remoto, sobre uma rede não segura.

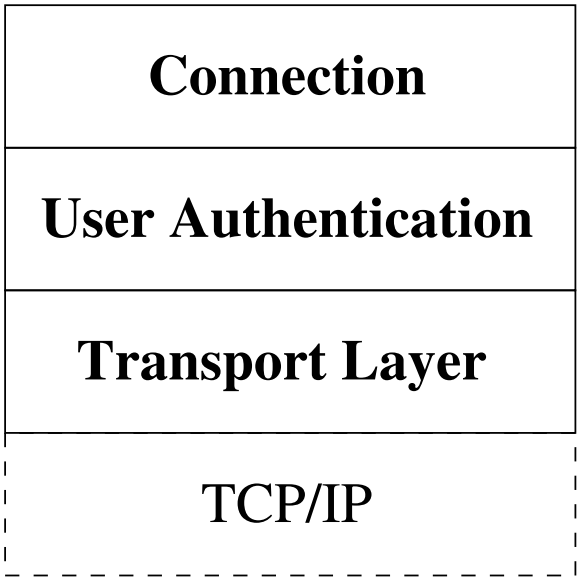
- Foi desenvolvido para substituir os serviços **rlogin**, **rsh**, etc, incluídos nas *shell* Unix/Linux, e que não são satisfatórios ao nível da segurança.

- O SSH funciona numa filosofia Cliente/Servidor:
  - o Servidor é típicamente 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 (está publicado na forma de Internet Drafts) e o seu uso é generalizado.

Estrutura do SSH

- A **Transport Layer** oferece autenticação do servidor, confidencialidade e integridade de dados 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 perante o Cliente.
  - Que tipo de autenticação é requerida pelo Servidor aos utilizadores que acedem ao sistema 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.

- Excepções 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.

Transport Layer$_{cont}$

- 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 incluido 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).

Transport Layer$_{cont}$

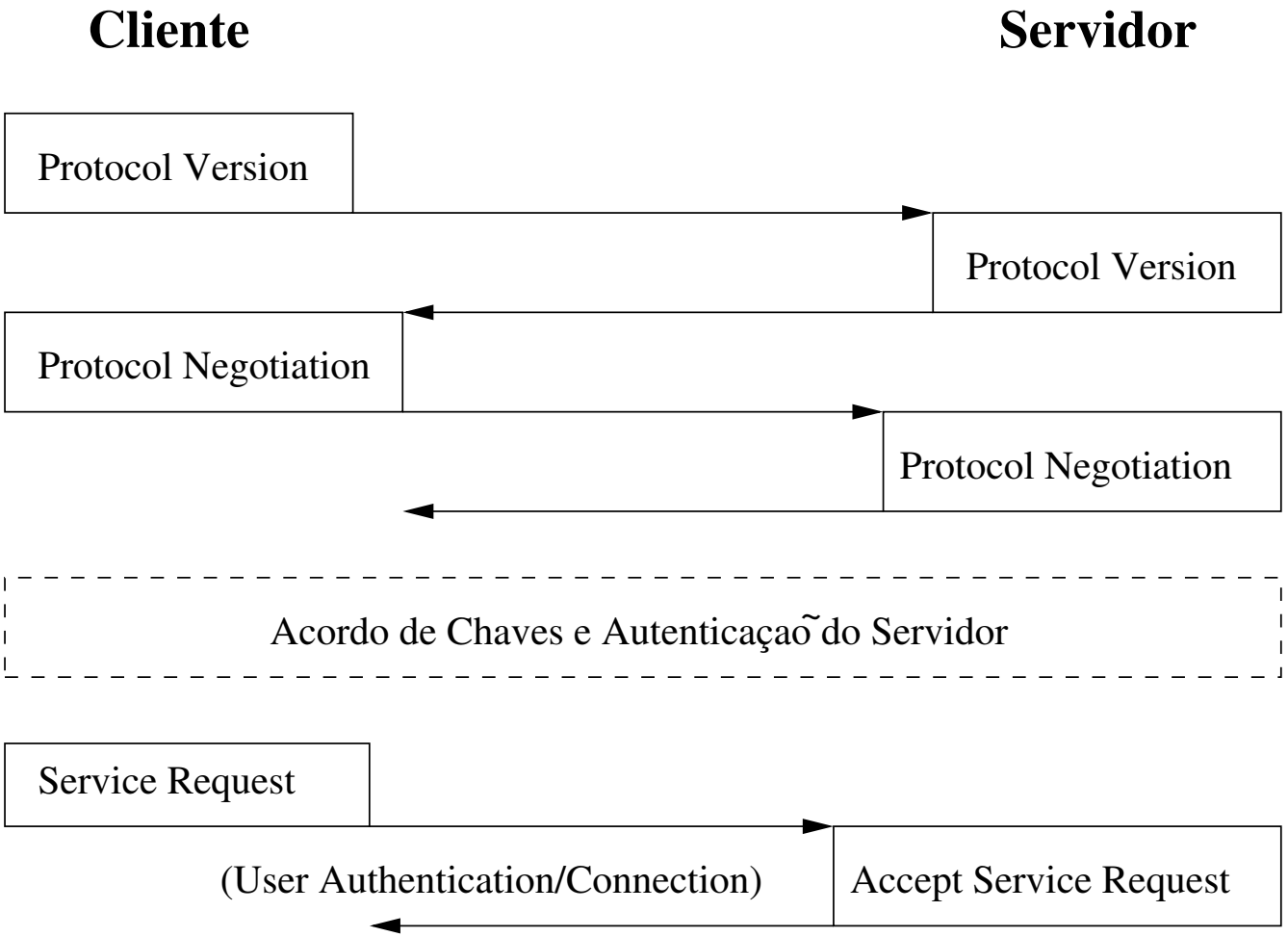
- Estabelecida a versão, Cliente e Servidor trocam mensagens em que indicam os algoritmos que implementam, e aqueles que são de utilização preferencial.

- 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).

Transport Layer$_{cont}$



Transport Layer$_{cont}$

- 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.

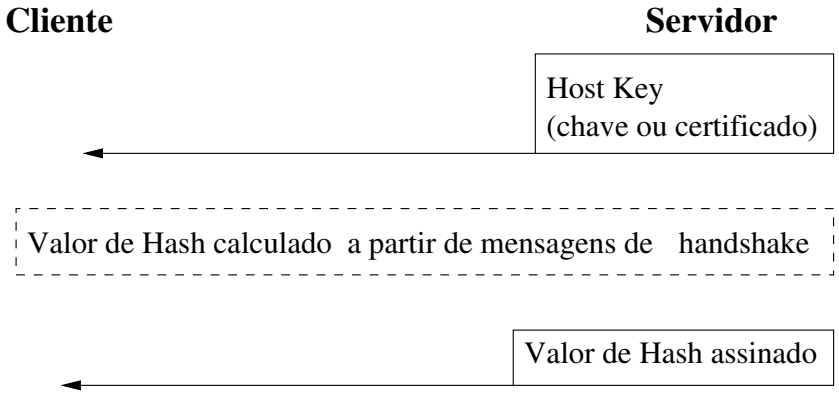
- Daí que, para haver segurança no estabelecimento de uma sessão:
  - ou o Cliente tem conhecimento prévio da chave pública do Servidor (distribuição manual da chave pública),
  - ou recorre-se a um esquema de certificação X.509, no qual o Cliente apenas tem de conhecer e confiar numa CA que permita validar os certificados dos Servidores.

- Ambos os modos de funcionamento são permitidos no SSH.

Transport Layer$_{cont}$

- 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.

Transport Layer$_{cont}$

- 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 $= HASH(K|H|'B'|sessionid)$
  - Chave de cifragem Cliente-Servidor $= HASH(K|H|'C'|sessionid)$
  - Chave de cifragem Servidor-Cliente $= HASH(K|H|'D'|sessionid)$
  - Chave de MAC Cliente-Servidor $= HASH(K|H|'E'|sessionid)$
  - Chave de 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.

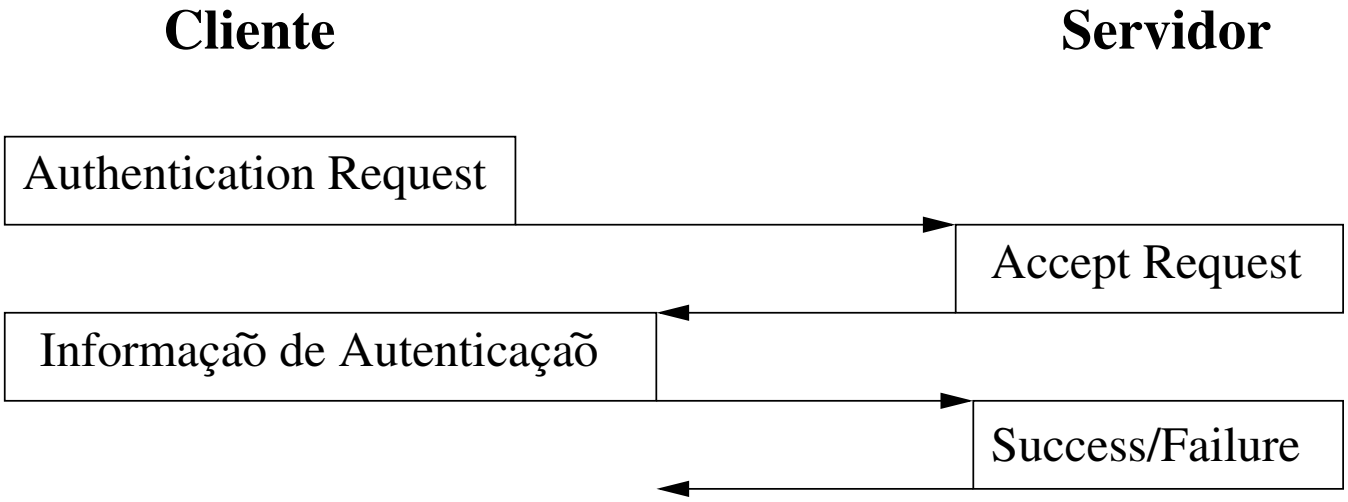
- Um pedido de autenticação enviado pelo Cliente inclui os seguintes parâmetros:
  - **User Name** Identificação do utilizador a autenticar.
  - **Service Name** O serviço a que pretende aceder.
  - **Authentication Method** O método de autenticação a utilizar.

- 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.

User Authentication Layer$_{cont}$

- 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.

User Authentication Layer$_{cont}$



- 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