Serviços de Segurança

- O objetivos os serviços de segurança é oferecer uma proteção específica a recursos de um sistema
- Serviços de Segurança
  - Confidencialidade
  - Autenticação
  - Controle Acesso
  - Integridade
  - Não-Repúdio
  - Disponibilidade
- Serviços de segurança utilizam mecanismos de segurança

Autenticação x Ataques

- Basicamente a autenticidade de mensagens consiste em verificar que as mensagens recebidas de uma parte, devidamente reconhecida, não foram modificadas.
- Entre os ataques possíveis que podem comprometer a autenticidade de mensagens pode-se considerar:
  - Representação ou Impersonificação (Masquerade) – Neste caso as mensagens vindas de uma fonte fraudulenta são incluídas na comunicação.
  - Modificação do Conteúdo ou de parte do mesmo
  - Modificação na Seqüência das Mensagens – Através da inserção, remoção ou reordenação
  - Atraso ou reenvio de mensagens

Autenticação

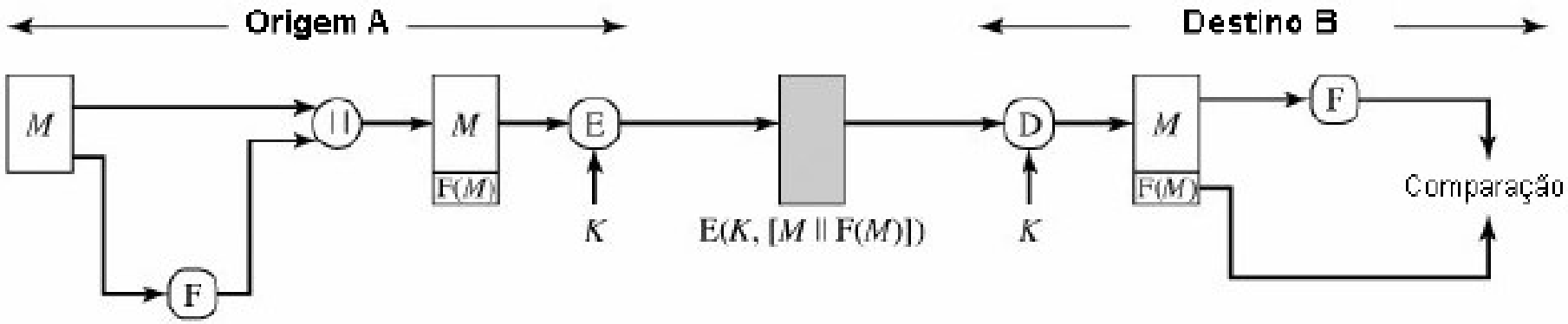
- A autenticidade é um importante serviço de segurança e a mesma pode ser considerada sob dois aspectos:
  - Autenticação da Origem da Entidade
  - Autenticação da Origem dos Dados
- Os requisitos necessários para a autenticidade de mensagens são os seguintes:
  - Garantia a integridade da mensagem
  - Garantir a identidade do remetente
  - Não-repúdio da origem
- As mecanismos de segurança utilizados para garantir a autenticidade de mensagens são:
  - Cifragem das Mensagens
  - Código de Autenticação de Mensagem (Messsage Authentication Code – MAC)
  - Funções Hash

Autenticação com Cifragem Cifragem da Mensagens

- A cifragem de mensagens pode, dependendo da forma como é utilizada, prover um meio de autenticação
- Neste caso podem ser utilizados
  - Algoritmos Simétricos
  - Algoritmos Assimétricos
- Algoritmos Simétricos
  - Considerando que somente o transmissor e o receptor a possuem a chave. (Este argumento porém é frágil!)
  - O conteúdo da mensagem não pode ter sido alterado, garantindo assim sua integridade, bem como a confidencialidade
  - O simples fato do receptor conseguir decifrar a mensagem não garante a autenticidade, pois o significado não pode ser comprovado
  - Para garantir a autenticidade deve ser utilizado por exemplo um FCS (Frame Check Sequence) ou CheckSum e cifrar o mesmo juntamente com a mensagem

Autenticação com Cifragem Algoritmos Simétricos

- Controle Interno de Erros
  - C = E [K,M||F(M)]
  - F(M) – FCS é função da mensagem
  - Se ambos os FCS são iguais o recebido e o calculado então a mensagem pode ser considerada autêntica
  - Problemas:
    - Chave e Método de cálculo do FCS conhecidos por um oponente

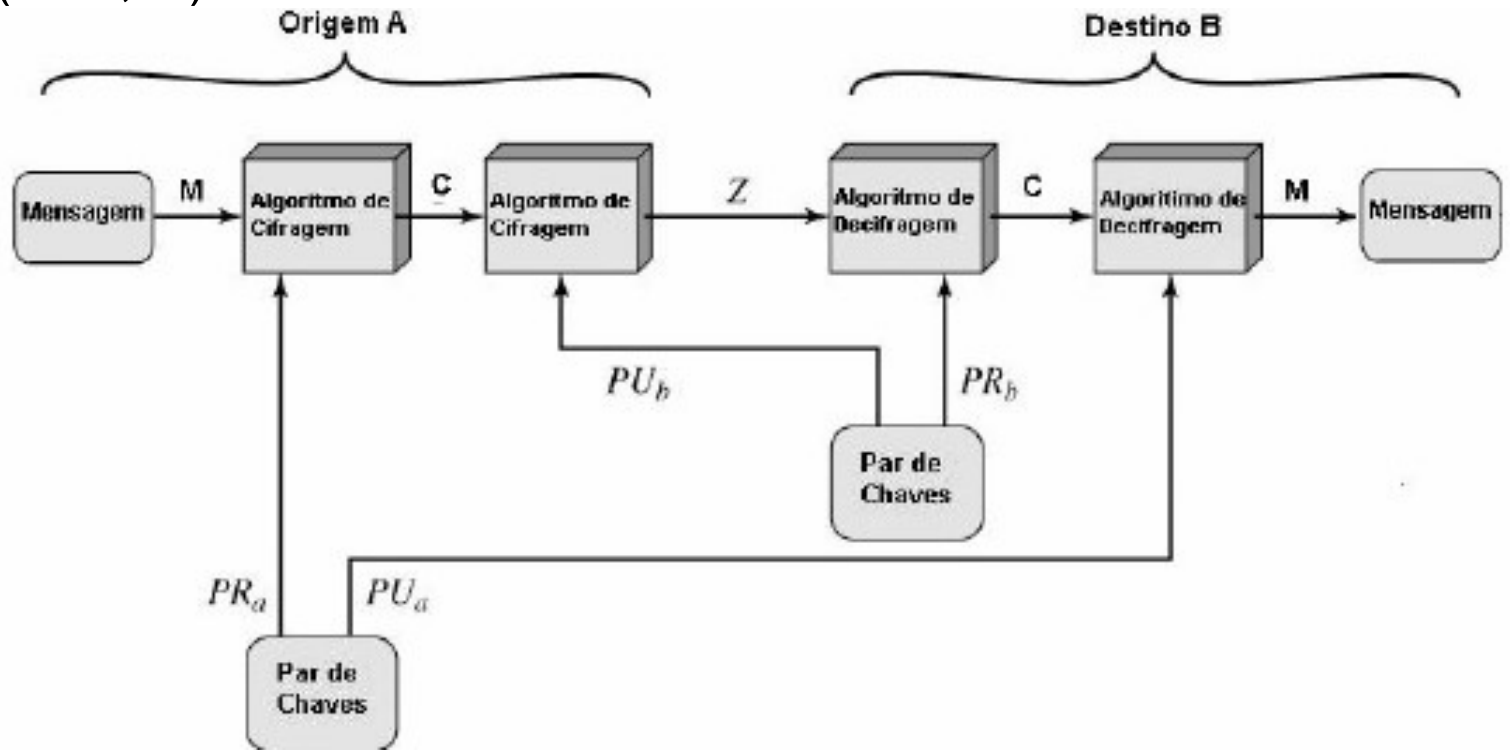


Autenticação com Cifragem Algoritmos Assimétricos

- A cifragem de mensagens pode, de certa forma, prover um meio de autenticação
- Algoritmos Assimétricos
  - Confidencialidade - Neste Caso é necessário que o transmissor (A) realize a cifragem da mensagem com a chave pública (PU_B) do receptor (B), sendo que a mesma somente poderá ser decifrada pela chave privada do receptor (PR_B)
  - Autenticidade - Neste Caso é necessário cifrar a mensagem com a chave privada (PR_A) do transmissor (A) e a mensagem deve ser decifrada, pelo receptor (B) com a chave pública de A (PU_A).

Autenticação com Cifragem Algoritmos Assimétricos

- Sistema utilizado para confidencialidade e autenticação
  - C = E(PRA, M)
  - Z = E(PUB, C)
  - C = D(PRB, E(PUB, C))
  - M = D(PUA, C)



Código de Autenticação de Mensagem (Messsage Authentication Code – MAC)

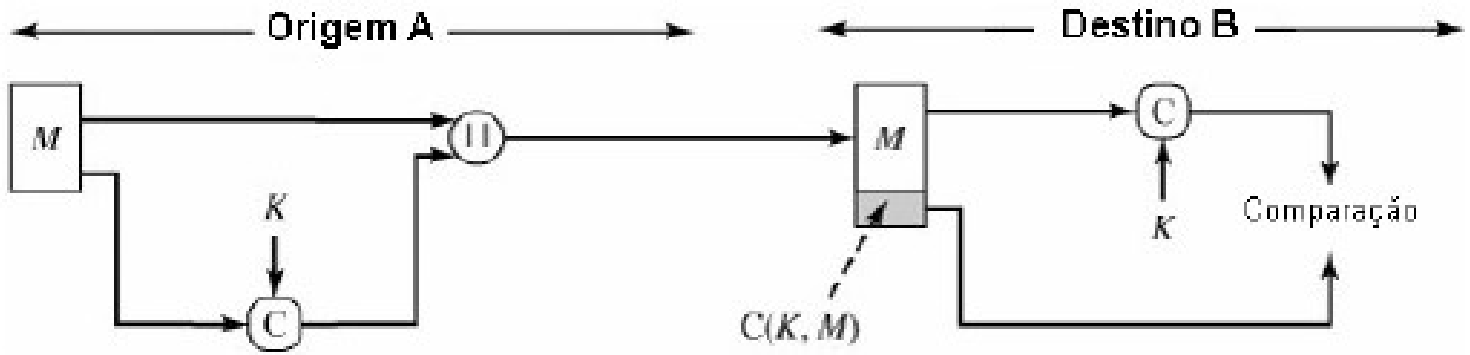
- O MAC é também conhecido como um checksum criptográfico
  - MAC = C(K,M)
  - M é uma mensagem de tamanho variável e K é uma chave secreta conhecida apenas pelo transmissor e receptor
  - A função de cifragem C produz como resultado um bloco de bits de tamanho fixo (MAC)
  - Somente é utilizado o processo de cifragem, logo pode ser utilizado uma função não-inversível (irreversível)
- O MAC é então adicionado à mensagem na origem
- No destino o MAC é recalculado e comparado como MAC recebido. A igualdade indica que a mensagem é autêntica, garantido a integridade da mesma
- A identidade da origem é garantida pelo uso da chave K
- O MAC não oferece o recurso de assinatura digital visto que ambas as partes possuem a chave

Código de Autenticação de Mensagem (Messsage Authentication Code – MAC)

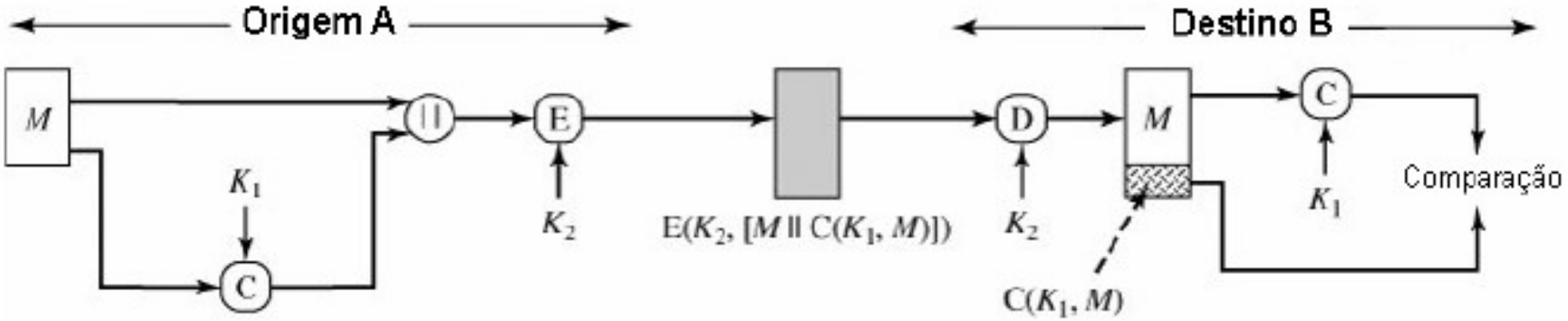
- O uso do MAC se justifica pelas seguintes situações:
  - Em certas aplicações pode não ser necessário que as mensagens sejam secretas, porém é importante autenticar as mesmas.
  - Um exemplo é o protocolo Simple Network Management Protocol Version 3 (SNMPv3), que separa as funções de confidencialidade e autenticação, desta forma mensagens que alteram parâmetros do sistema precisam ser autênticas, porém o tráfego SNMP pode ser aberto
  - A separação de autenticação e confidencialidade proporciona uma maior flexibilidade na arquitetura, permitindo por exemplo executar a autenticação na camada de aplicação e a confidencialidade no nível de transporte, por exemplo.
  - Permitir a recepção de mensagens autenticas e somente processar a mesma e realizar a decifragem em um momento posterior.
  - Um exemplo seria um arquivo enviando, cujas mensagens precisam ser autênticas, porém o mesmo deverá ser armazenado de forma cifrada

Código de Autenticação de Mensagem Funcionalidades

- Autenticação de Mensagens

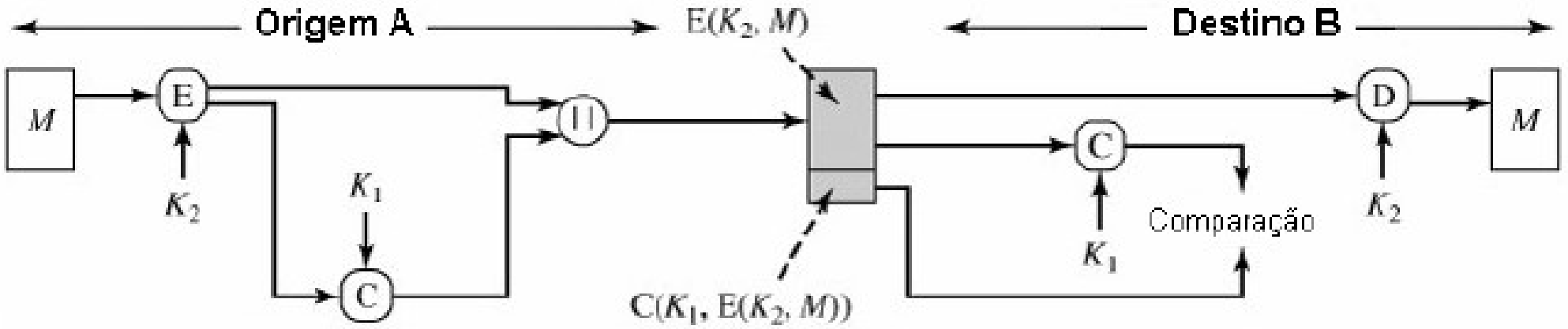


- Confidencialidade e Autenticação baseada no texto plano



Código de Autenticação de Mensagem Funcionalidades

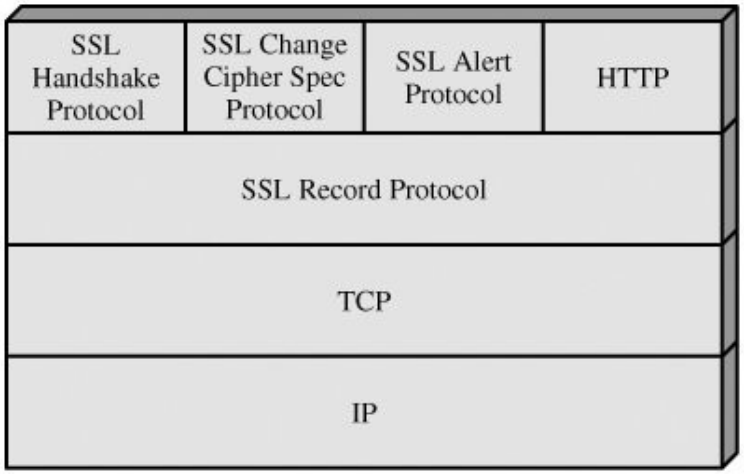
- Confidencialidade e Autenticação baseada na texto cifrado



- Nos casos de confidencialidade e autenticação São necessárias duas chaves uma para cifragem (K_2) e outra para a geração do MAC (K_1)
- Estas chaves precisam ser trocadas entre as partes, o que, quando não realizado de maneira conveniente pode provocar vulnerabilidades
- A Autenticação baseada no texto puro porporciona um maior grau de segurança

SSL (Secure Sockets Layer)

- Utiliza protocolo TCP para criar uma conexão segura, confiável, fim-a-fim
  - Segura
    - Integridade
    - Confidencialidade
    - Autenticação
- Não é um protocolo mas sim uma camadas com vários protocolos

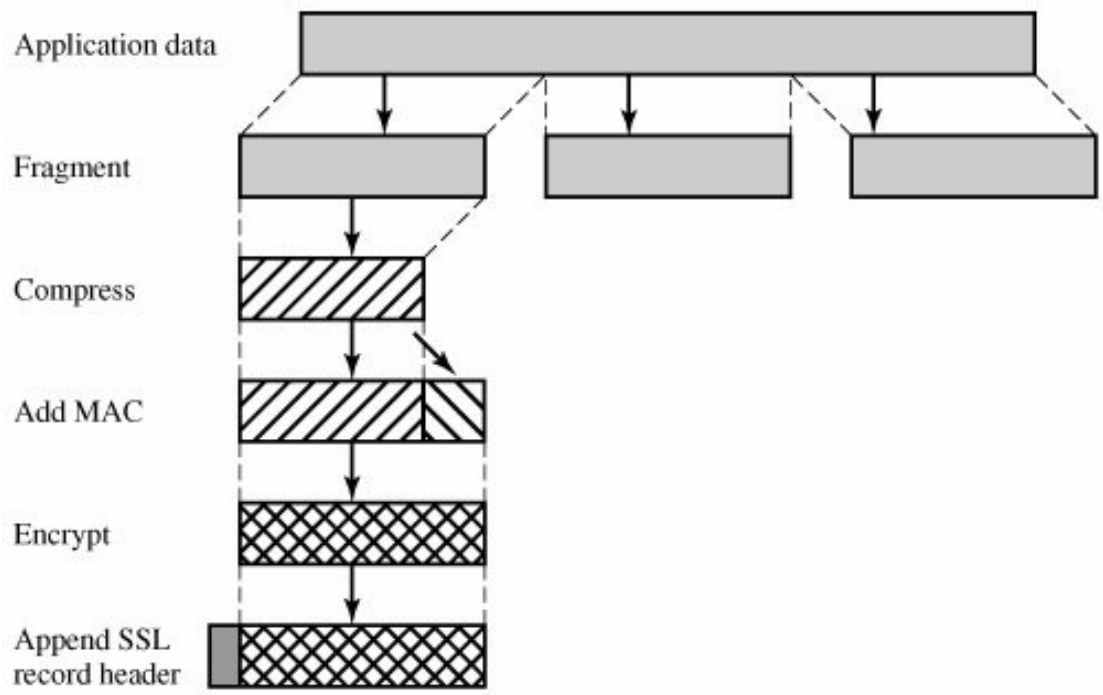


SSL Arquitetura

- Nível Básico
  - SSL Record Protocol
    - Provê serviços básicos de segurança para as camadas superiores
      - Confidencialidade
        - Utiliza chave secreta definida pelo Handshake Protocol
      - Integridade
        - Utiliza chave secreta a fim de criar um MAC (Message Autentication Code) das mensagens
    - Utilizado como serviço de transferência de dados para o protocolo HTTP
- Nível Superior
  - Handshake Protocol
    - Provê autenticação
  - Change Cipher Spec Protocol
  - Alert Protocol

SSL
Record Protocol

- Processamento das Mensagens



SSL Record Protocol

- Processamento das Mensagens
  - Fragmentação da mensagem da camada superior
    - Bloco de 2^{14} bytes (16384 bytes)
  - Compactação
    - Etapa é opcional
    - SSLv3 assim como no TLS nenhum algoritmo é especificado
  - Adição MAC
    - hash(MAC_write_secret || pad_2 || hash(MAC_write_secret || pad_1 || seq_num || SSLCompressed.type || SSLCompressed.length || SSLCompressed.fragment))
  - Cifgragem
    - Fragmento com o MAC adicionado é cifrado utilizado um algoritmo simétrico
  - Adição do Cabeçalho
    - Cabeçalho é adicionado a cada fragmento

SSL Record Protocol - MAC

- Adição do MAC
  - hash(MAC_write_secret || pad_2 || hash(MAC_write_secret || pad_1 || seq_num || SSLCompressed.type || SSLCompressed.length || SSLCompressed.fragment))
    - MAC_write_secret – chave secreta compartilhada
    - pad_1 – byte 0x36 (0011 0110) repetido 48 vezes (384 bits) caso seja utilizado MD5 ou 40 vezes (320 bits) no caso de SHA-1
    - pad_2 – byte 0x5C (0101 1100) repetido 48 vezes (384 bits) caso seja utilizado MD5 ou 40 vezes (320 bits) no caso de SHA-1
    - seq_num – número de sequência da mensagem
    - SSLCompressed.type – Tipo de Compressão utilizada
    - SSLCompressed.length – Comprimento do segmento compactado
    - SSLCompressed.fragment – Fragmento compactado ou o original caso não tenha sido compactado

SSL
Record Protocol - Cifragem

- Cifragem fragmento
  - Pode utilizar os seguintes algoritmos

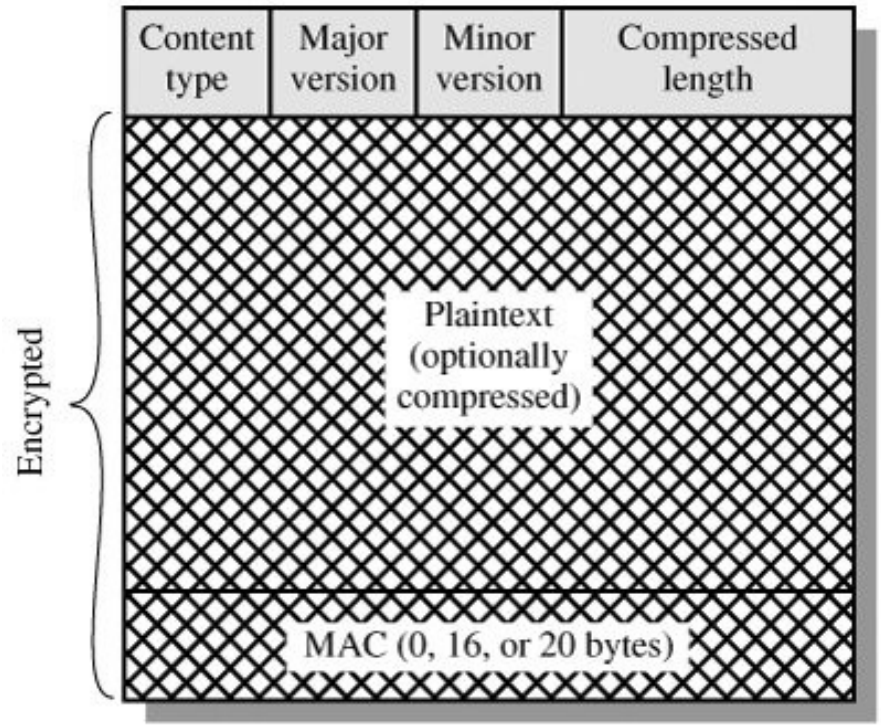
Block Cipher		Stream Cipher	
Algoritmo	Tamanho Chave	Algoritmo	Tamanho Chave
AES	128,256	RC4-40	40
IDEA	128	RC4-128	128
RC2-40	40		
DES-40	40		
DES	56		
3DES	168		
Fortezza	80		

SSL
Record Protocol - Cabeçalho

- Adição Cabeçalho
  - Ultima fase do processamento pelo Record Protocol
  - Contém os seguintes campos:
    - Content Type (8 bits)
      - Utilizado pelo protocolo do nível superior para processar o fragmento
      - Pode assumir os seguintes valores: change_cipher_spec, alert, handshake, and application_data
    - Major Version (8 bits)
      - Indica a versão em uso. Para SSLv3 o valor é 3
    - Minor Version (8 bits)
      - Indica e versão em uso. Para SSLv3 o valor é 0
    - Compressed Length (16 bits)
      - O comprimento em bytes do fragmento que pode estar compactado ou não. O valor máximo é 2^{14} + 2048 bytes

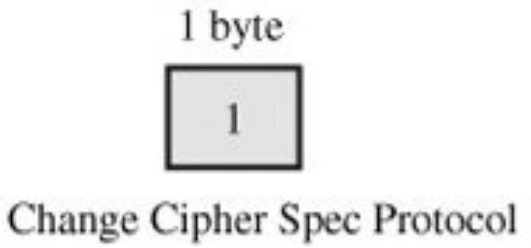
SSL – Record Protocol Formato Dados

- Formato dos dados utilizado pelo protocolo



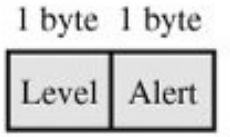
SSL
Change Cipher Spec Protocol

- Protocolo de nível superior, específico do SSL que utiliza o Record Protocol para transporte dos dados
- Contém uma única mensagem de um único byte com o valor 1
- Mensagem indica que um estado pendente pode ser tratado como o estado corrente
- Com isto ocorre a troca dos algoritmos de cifragem a serem utilizado em uma conexão



SSL
Alert Protocol

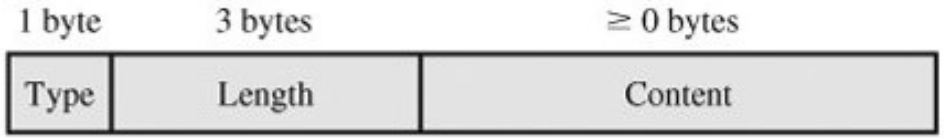
- Cada mensagem consiste de dois bytes.
  - Primeiro byte
    - Nivel da mensagem de alerta – Warning (1) ou Fatal (2), que neste caso finaliza a conexão e nao permite a criação de novas conexões na seção
  - Segundo byte
    - Indica o tipo de alerta ocorrido
      - unexpected_message; bad_record_mac; decompression_failure; handshake_failure; illegal_parameter
      - close_notify; no_certificate; bad_certificate; unsupported_certificate; certificate_revoked; certificate_expired; certificate_unknown



(b) Alert Protocol

SSL Handshake Protocol

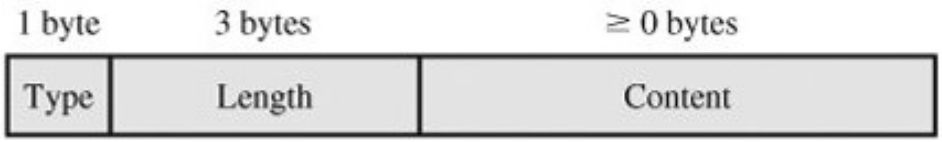
- Protocolo mais complexo
- Permite ao servidor e cliente se autenticarem
- Na sequencia permite a negociação dos algoritmos de cifragem e MAC que serão utilizados bem como de suas chaves
- É o protocolo que permite o estabelecimento de uma sessão entre o cliente e servidor



(c) Handshake Protocol

SSL - Handshake Protocol Mensagens

- Formato da Mensagem



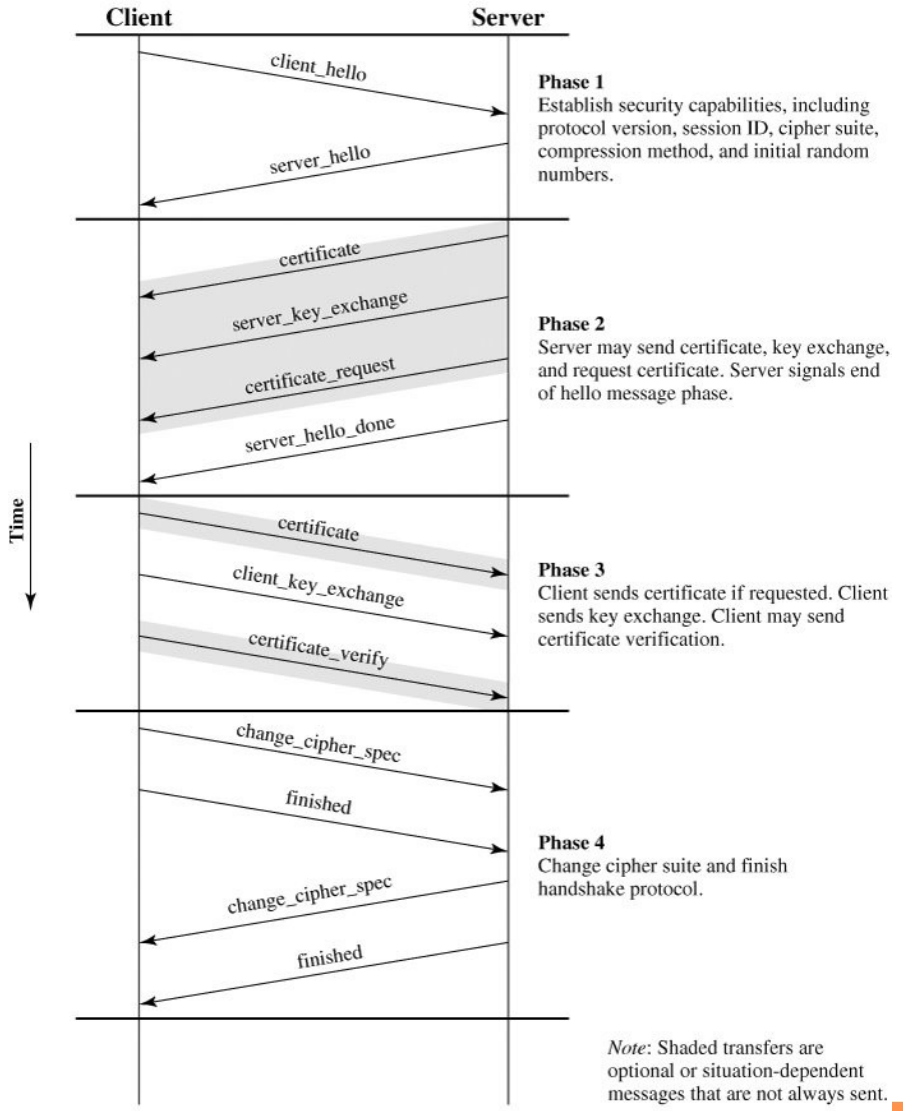
(c) Handshake Protocol

- Tipos de Mensagens e Parâmetros

Message Type	Parameters
hello_request	null
client_hello	version, random, session id, cipher suite, compression method
server_hello	version, random, session id, cipher suite, compression method
certificate	chain of X.509v3 certificates
server_key_exchange	parameters, signature
certificate_request	type, authorities
server_done	null
certificate_verify	signature
client_key_exchange	parameters, signature
finished	hash value

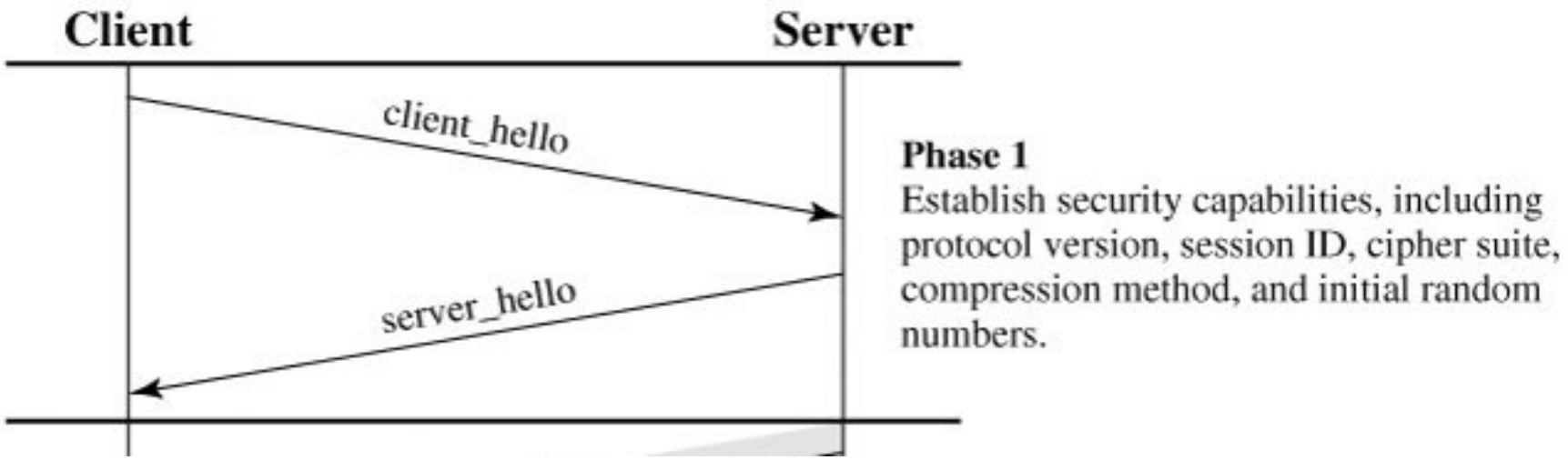
SSL Handshake Protocol

- Troca de Mensagens



SSL Handshake Protocol

- Troca de Mensagens
  - Fase 1



SSL – Handshake Protocol Fase 1- client_hello

- Inicia a conexão lógica e estabelece parâmetros de segurança
- Cliente envia mensagem client_hello
  - Version – Maior versão do SSL entendida pelo cliente
  - Random – Valores aleatórios que consistem:
    - timestamp(32 bits) + secure_random_number(28 bits) – nonces (number used once)
    - Valores serão utilizados para prevenir ataques do tipo replay durante troca de chaves
  - Session ID – Identificador da sessão
    - 0 – cliente deseja estabelecer uma nova conexão em uma nova sessão
    - Diferente zero – cliente deseja estabelecer uma nova conexão na sessão ou uma atualização nos seus parâmetros
  - CipherSuite – Lista com as combinações de criptografia suportadas pelo cliente em order decrescente de preferência
    - Define algoritmo para troca de chaves (Key Exchange) e cifragem (CipherSpec)
  - Compression Method – Lista de métodos de compressão suportados pelo cliente

SSL – Handshake Protocol Fase 1- server_hello

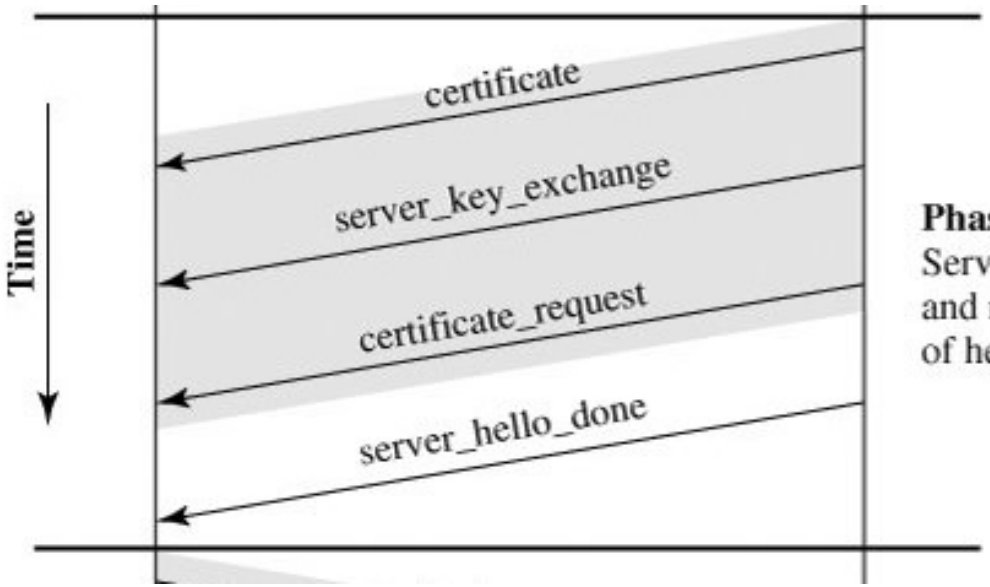
- Cliente espera pelo server_hello
- Contém os mesmos parâmetros do client_hello, indicando:
  - Version – máxima versão suportada pelo servidor
  - Random – Valores aleatórios que consistem:
    - timestamp(32 bits) + secure_random_number(28 bits) – nonces (number used once)
    - Valores serão utilizados para prevenir ataques do tipo replay durante troca de chaves, gerados agora, pelo servidor
  - Session ID – Identificador da sessão
    - 0 – Ao receber um zero o servidor gera um novo identificador de sessão
    - Diferente zero – valor é devolvido pelo servidor
  - CipherSuite – Contém um único conjunto de algoritmos que serão utilizados a partir da lista sugerida pelo cliente
  - Compression Method – Contém método selecionado pelo servidor a partir da lista proposta pelo cliente

SSL – Handshake Protocol Fase 1- server_hello

- Cipher Suite - Contém dois elementos: Método Troca de chave (Key Exchange) e especificação de algoritmos (CipherSpec)
  - Key Exchange (Troca de Chaves)
    - Método que será utilizado para troca de chaves:
      - RSA
      - Fixed Diffie-Hellman
      - Ephemeral Diffie-Hellman
      - Anonymous Diffie-Hellman
      - Fortezza
  - CipherSpec (Especificação dos algoritmos)
      - CipherAlgorithm - RC4, RC2, DES, 3DES, DES40, IDEA, Fortezza
      - MACAlgorithm - MD5 or SHA-1
      - CipherType: Stream or Block
      - IsExportable: True or False
      - HashSize: 0, 16 (MD5) ou 20 (SHA-1) bytes
      - Key Material – Material utilizado para geração de chaves
      - IV – Vetor de inicialização (Modo operação CBC)

SSL Handshake Protocol

- Troca de Mensagens
  - Fase 2



Phase 2
Server may send certificate, key exchange, and request certificate. Server signals end of hello message phase.

SSL – Handshake Protocol

Fase 2- Autenticação e Troca de Chaves

- Mensagem **“certificate”**
  - Opcional
  - Não é utilizada na seguinte situação
    - Troca de chaves “anonymous Diffie-Hellman”
  - Servidor envia seu certificado caso seja necessário sua atenticação
    - Mensagem contém um certificado X.509 ou uma cadeia de certificados
    - Caso seja utilizado o método “fixed Diffie-Hellman” esta mensagem funcionará como uma troca de chaves do servidor, pois conterá os parâmetros públicos conforme proposta de Diffie-Hellman

SSL – Handshake Protocol

Fase 2- Autenticação e Troca de Chaves

- Mensagem “server_key_exchange”
  - Não é necessária nas seguintes situações
    - Servidor está utilizando opção “Fixed Diffie-Hellman”
    - Será utilizado troca de chaves baseado em “RSA”
  - Utilizada nos seguintes casos:
    - Anonymous Diffie-Hellman
      - Conteudo da mensagem consiste dos dois valores globais (número primo e sua raiz primitiva) juntamente com a chave pública prevista em Diffie-Hellman
    - Ephemeral Diffie-Hellman
      - Além dos três valores enviados no método “Anonymous Diffie-Hellman” contém uma assinatura
    - Troca de chaves RSA
      - Neste caso o servidor cria um par de chaves RSA temporário para o cliente e envia a chave pública. Mensagem contém o expoente e o módulo da chave, juntamente com uma assintatura

SSL – Handshake Protocol

Fase 2- Autenticação e Troca de Chaves

- Mensagem “certificate_request message”
  - Servidor pode solicitar certificado do cliente
  - Contém as seguintes informações:
    - certificate_type e certificate_authorities
    - certificate_type (Indica o tipo de algoritmo e seu uso)
      - RSA
      - DSS
      - RSA para fixed Diffie-Hellman
      - DSS para fixed Diffie-Hellman
      - RSA para ephemeral Diffie-Hellman
      - DSS par ephemeral Diffie-Hellman
    - certificate_authorities
      - Lista de Distinguished Names de autoridades aceitáveis

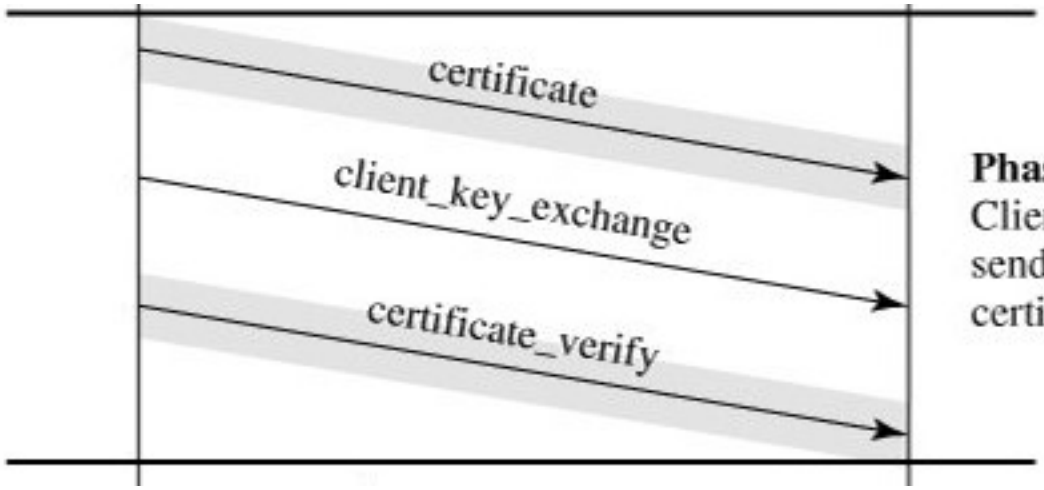
SSL – Handshake Protocol

Fase 2- Autenticação e Troca de Chaves

- Mensagem “server_done message”
  - Sempre é enviada
  - Não contém parâmetros
  - Após este ponto o servidor aguardará as respostas do cliente

SSL Handshake Protocol

- Troca de Mensagens
  - Fase 3



Phase 3
Client sends certificate if requested. Client sends key exchange. Client may send certificate verification.

SSL – Handshake Protocol

Fase 3- Autenticação e Troca de Chaves

- Após receber mensagem “server_done message” é necessário que o cliente:
  - Verifique se o certificado enviado pelo servidor é válido
  - Se parâmetros recebidos no “server_hello” são aceitáveis
- Mensagem “certificate”
  - Enviada caso o servidor tenha solicitado certificado
  - Caso não possua um certificado cliente envia um alerta “no_certificate”

SSL – Handshake Protocol

Fase 3- Autenticação e Troca de Chaves

- Mensagem “client_key_exchange message”
  - Sempre é enviada
  - Conteúdo depende do tipo de troca de chaves
  - RSA
    - Cliente produz um número de 48 bits e cifra com a chave pública do servidor
  - Ephemeral ou Anonymous Diffie-Hellman
    - Parâmetros públicos do cliente são enviados
  - Fixed Diffie-Hellman
    - Nulo, pois parâmetros públicos do cliente foram enviados na mensagem “certificate”

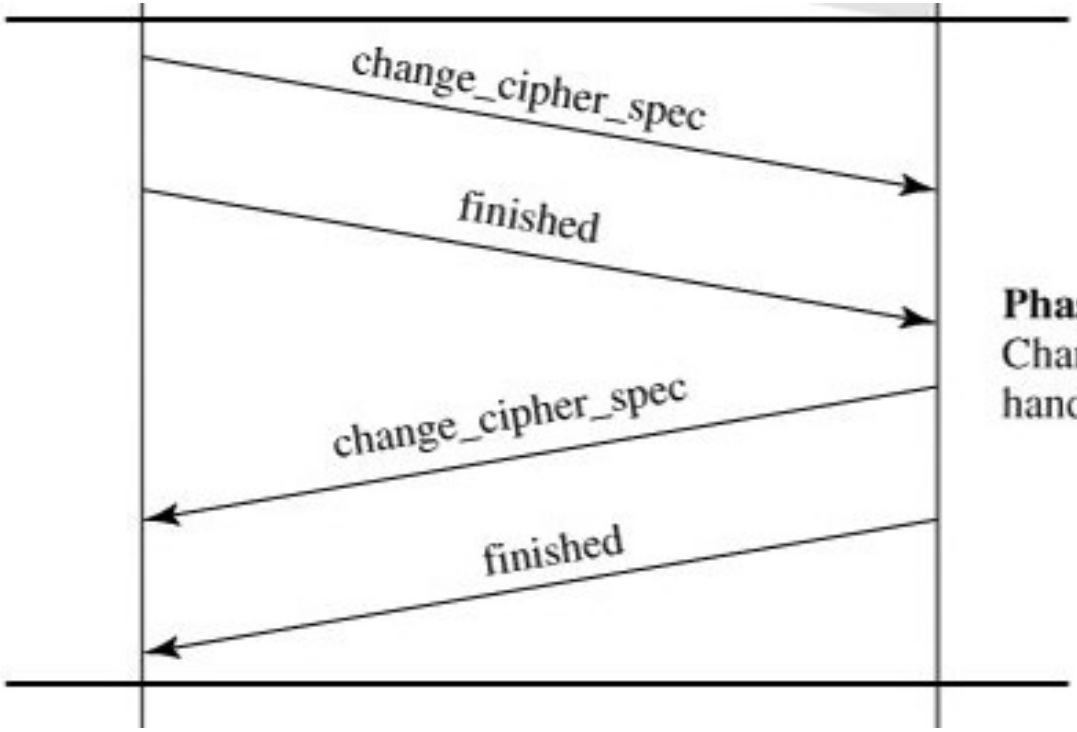
SSL – Handshake Protocol

Fase 3- Autenticação e Troca de Chaves

- Mensagem “certificate_verify message”

SSL Handshake Protocol

- Troca de Mensagens
  - Fase 4



Phase 4
Change cipher suite and finish handshake protocol.

SSL – Handshake Protocol
Fase 4 - Finalização

- Cliente
  - Envio da mensagem “change_cipher_spec”
    - Copia o CipherSpec pendente para o corrente
    - Esta mensagem não é parte o Handshake Protocol mas sim do “Change Chipher Spec Protocol”
  - Envio da mensagem “finished”
    - Utiliza novos algoritmos e chaves acorddas
    - Indica que processo de autenticação e troca de chaves foi bem sucedido
- Servidor
  - Realiza processo similar
- A partir deste ponto inicia-se a troca de dados entre cliente e servidor