第7章 Web安全

第 7章 Web安全

- 7.1 Web安全概述
- 7.2 传输层安全--TLS
  - 7.2.1 TLS概述
  - 7.2.2 TLS握手协议
  - 7.2.3 TLS记录协议
  - 7.2.4 TLS更改密码和警告协议
  - 7.2.5 TLS应用
- 7.3 HTTPS

7.1 Web安全概述

- Web应用--使用TCP/IP协议栈在Internet上运行的客户端/服务器应用程序
  - Web浏览器易于使用和配置，但底层软件很复杂。这种复杂性可能隐藏了潜在的安全缺陷
  - Web服务器对外提供服务，容易受到来自外部环境的攻击。一旦Web服务器受到破坏，攻击者就可以获得与之相连的整个LAN中的其他服务器和系统的访问权限
  - 普通用户没有意识到安全风险或不知道如何采取有效的应对方法

Web安全概述

Web安全威胁—从安全需求角度分类

	威胁	后果	对策
完整性	用户数据修改 木马浏览器 内存修改 更改传输中的消息	丢失消息 设备受损 易受其他威胁	密码校验和
机密性	网上窃听 窃取服务器信息 窃取客户信息 窃取网络配置信息 窃取客户与服务器传输信息	信息丢失 秘密丢失	加密 Web委托代理
拒绝服务	破坏用户线程 用假消息使机器溢出 填满硬盘或内存 用僵尸网络攻击DNS服务器	中断 干扰 阻止用户完成任务	难以防止
认证	假冒合法用户 伪造数据	用户错误 相信虚假信息	密码技术

Web流量安全方法

- 方法一：使用IPSec
  - 终端用户和应用透明，并能提供一种通用的解决方案，具有过滤功能，使得只用IPSec处理所选的流量。
- 方法二：在TCP上实现安全
  - 提供端到端的安全服务，对路由器透明，对高层应用也透明
  - 可以嵌入到某些软件中，如浏览器软件
- Internet安全架构

7.2 运输层安全协议TLS

- 1994年，为了保护Web通信协议HTTP，网景公司开发了SSL,后来成为传输层通用的标准。
  - 安全套接层 SSL 广泛应用于基于万维网的各种网络应用（但不限于万维网应用）,1996年SSL 3.0成为 Web 安全的事实标准.
- 现在广泛使用的有以下两个协议：
  - 安全套接字层 SSL (Secure Socket Layer)
  - 运输层安全 TLS (Transport Layer Security)：IETF基于SSLv3制定，几乎完全一致

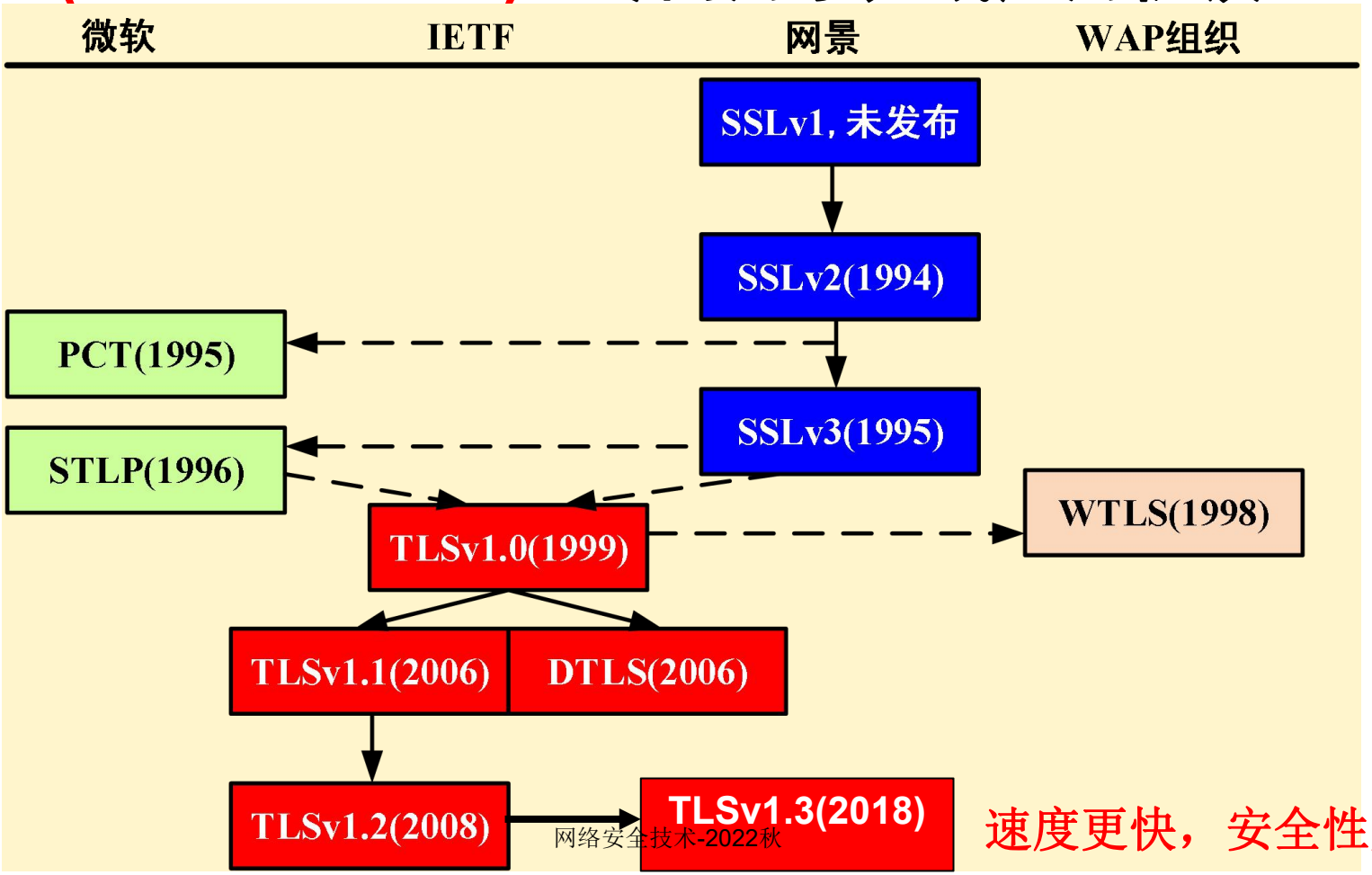
7.2.1 TLS概述

- 1999年，IETF 在 SSL 3.0 基础上推出了传输层安全标准 TLS，为所有基于 TCP 的网络应用提供安全数据传输服务。

- TLS建立在可靠的 TCP 之上，与应用层协议独立无关。
- 目标：实现两个应用实体之间的安全可靠通信。

SSL与TLS

- DTLS(Datagram TLS)：IETF推出的基于UDP的TLS协议
- WTLS(Wireless TLS)：专门用于无线应用协议WAP领域



速度更快，安全性更高

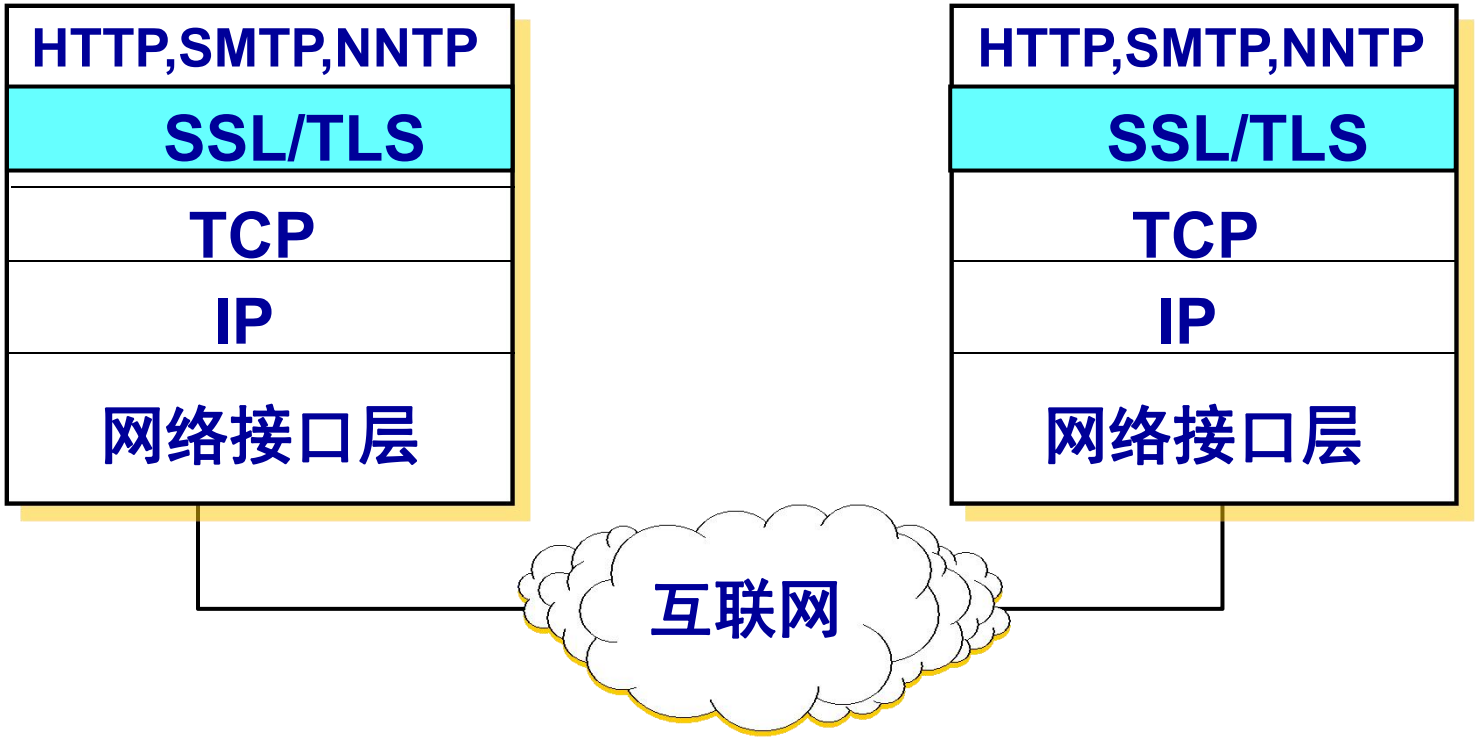
TLS安全需求分析

- **安全风险、安全服务与安全功能需求**

安全风险	安全服务
网站是否合法	服务器实体身份认证
支付密码泄露	数据机密性
支付金额不被篡改	数据完整性
客户的身份合法（有些情况需要）	客户端实体身份认证

- 身份认证
- 数据加密
- 数据认证
- 算法/密钥协商与分发

SSL / TLS 的位置



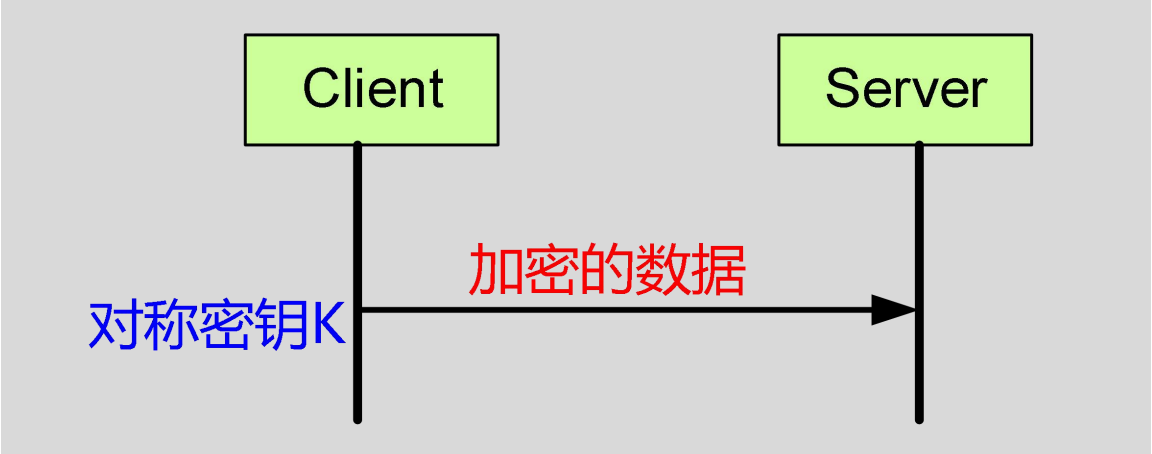
在发送方，TLS接收应用层的数据，对数据进行加密，然后把加了密的数据送往 TCP 套接字。在接收方，TLS从 TCP 套接字读取数据，解密后把数据交给应用层。

TLS方案设计

- TCP是互联网广泛使用的协议，但存在不足
  - 窃听：通信使用明文，内容可能被监听
  - 篡改：无法验证报文的完整性，内容可能被篡改
  - 冒充：不验证通信双方的身份，可能遭遇身份伪装
- 安全改进方案
  - 加密：　对称加密+密钥分配
  - 完整性保护：消息完整性+源鉴别+密钥分配
  - 验证身份
    - 验证服务器的身份（必选）
    - 验证客户端的身份（可选）

TLS方案设计--仅考虑加密

- 方案1：为了防止信息泄露，对传输的信息加密



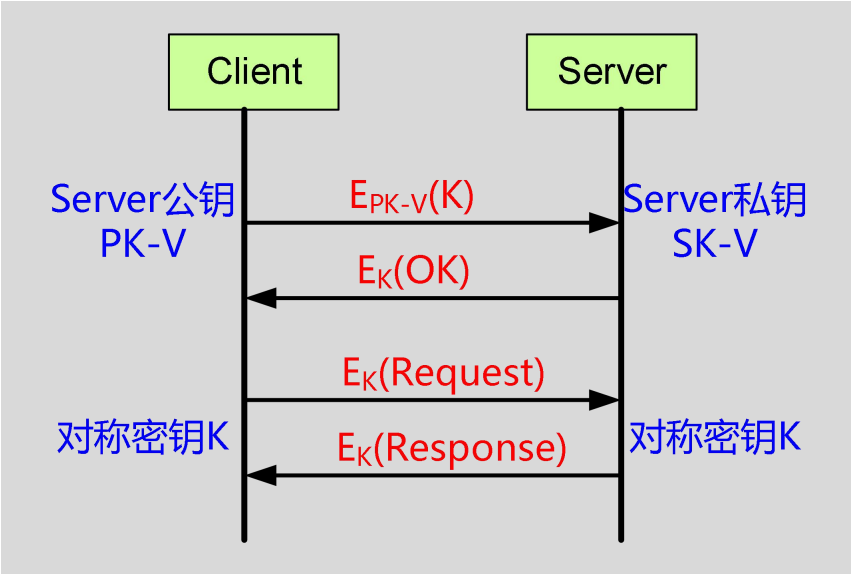
- 存在的问题：**对称密钥分配问题**
  - 点到点方式
    - **效率---**不同的客户端、服务器数量庞大，双方都需要维护大量的密钥，维护成本很高；
    - **安全性---**每个客户端、服务器的安全级别不同，密钥易泄露
  - KDC方式，互联网开放环境，管理复杂

TLS方案设计--仅考虑加密

- 方案2：

公钥分发对称密钥

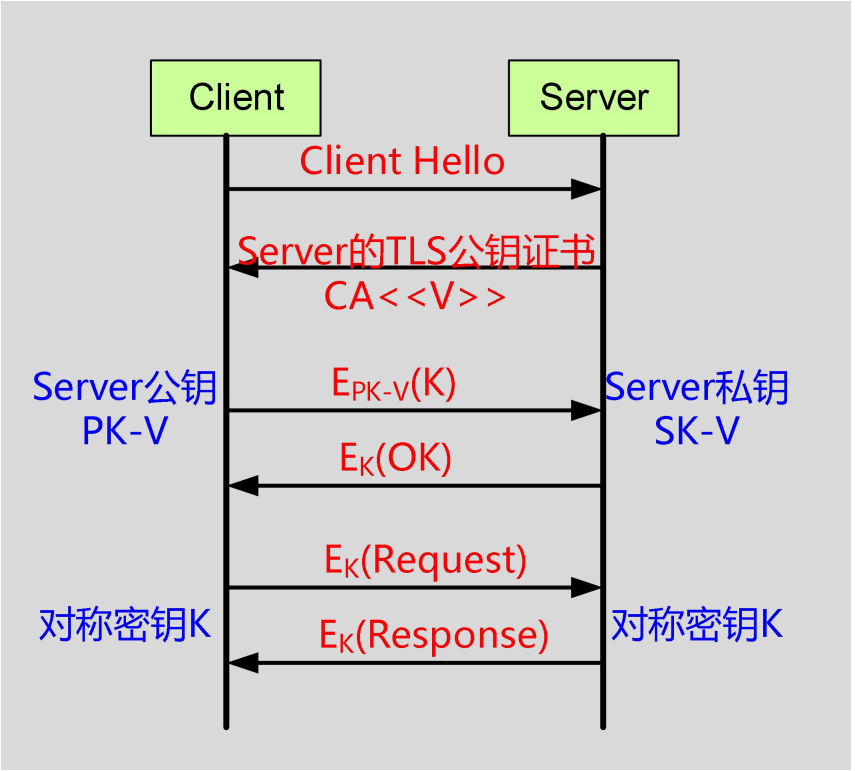
+对称密钥加密数据



- 存在的问题：公钥分发问题
  - 客户端如何获得服务器公钥PK-V

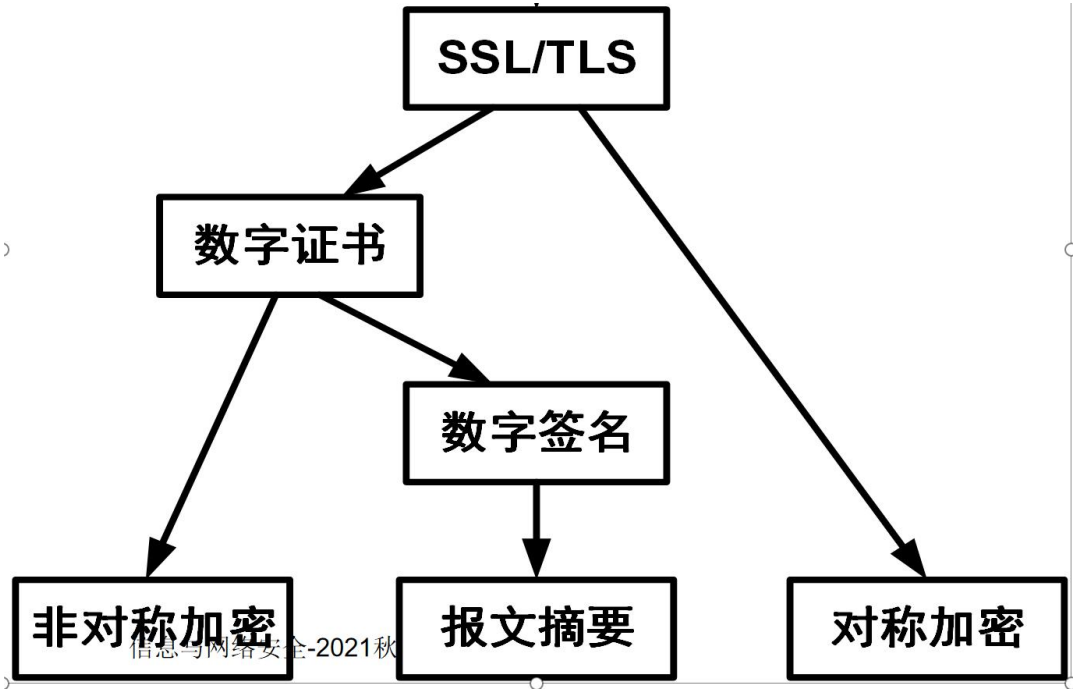
TLS方案设计--仅考虑加密

- 方案3：数字证书+公钥分发对称密钥+对称加密数据



TLS方案设计

- 数据完整性--消息认证码MAC
- TLS的作用
  - 建立安全通道，保证数据传输的安全；
  - 确认网站的真实性
- TLS安全架构



TLS的使用

- TLS可用于任何应用层协议，目前应用最多的是HTTP
- 应用程序HTTP调用TLS对整个网页进行加密时，网页上会提示用户，在网址栏原来显示http的地方变成https
- TLS/SSL了使用非对称加密、对称加密以及Hash算法
  - 非对称加密算法：RSA，DSA/DSS等(用来获得加密密钥和数字签名，算法使用服务器TLS数字证书中的公用密钥)
  - 对称加密算法：RC2, RC4,AES，3DES等(加密客户端与服务器间传输的数据)
  - HASH算法：SHA-1，SHA-256

TLS 提供的安全服务

- **(1) 服务器鉴别，** 允许用户证实服务器的身份。支持 SSL 的客户端通过验证来自服务器的证书，来鉴别服务器的真实身份并获得服务器的公钥。
- **(2) 客户鉴别（可选），** SSL 的可选安全服务，允许服务器证实客户的身份。
- **(3) 加密的 TLS会话，** 对客户和服务器间发送的所有报文进行加密，并检测报文是否被篡改。

TLS协议体系结构

- 协议栈结构（2层协议）

TLS握手协议	TLS修改密码协议	报警协议	应用
TLS记录协议			
TCP协议			
IP协议			

说明：TLS协议在应用层通信之前就已经完成加密算法、通信密钥的协商以及Server认证工作，在此之后，应用层协议所传送的数据都被加密

TLS协议体系结构

- 握手协议
  - 服务器认证、客户端认证（可选）
  - 算法协商：协商压缩算法、加密算法和消息认证算法
  - 密钥生成：生成会话密钥
- 记录协议：数据承载协议，握手报文和应用数据都要封装成“记录”的形式投递。
- 修改密码协议：客户端和服务器端交换该消息通告对方启用协商好的安全参数
- 警告协议：提供报错机制和安全断连机制

基本概念

- **TLS连接**：传输层TCP的概念，提供合适服务类型的一种传输，每个连接与一个会话相关。
- 连接状态由以下参数定义
  - 服务器和客户端的随机数(i.e. nonce)
  - 服务器写MAC密钥
  - 客户端写MAC密钥
  - 服务器写密钥
  - 客户端写密钥
  - 初始化向量(i.e. IVs for CBC encryption)
  - 序列号

基本概念

- **TLS会话**
  - 客户端和服务器之间的一个关联
  - 通过握手协议创建
  - 定义了密码安全参数集合，该参数集合可以在多个安全连接之间共享
  - 用来减少每次连接建立时进行安全参数协商的昂贵开销
  - 实体间可以有多个会话，一个**TLS**会话可以包括多个安全的连接

基本概念

- **TLS会话状态由以下参数定义**
  - 会话标识符
  - 对等实体证书**(X.509v3)**
  - 压缩方法
  - 密码规格(i.e. null, DES, MD5, SHA-1...)
  - 主密钥
  - 可恢复性:一个标志位，说明会话是否可被用于初始化新连接

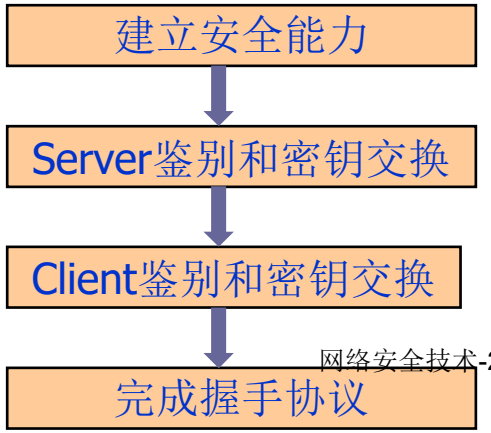
7.2.2 TLS握手协议

- 功能：在两个实体(Client与Server)建立安全TLS信道之前，该协议执行所有需要在它们之间达成的任务，具体包括
  - 协商用于建立安全信道的密码套件(*cipher suite*)
  - Client与Server互相鉴别对方
  - 建立确保信道安全所需的密钥

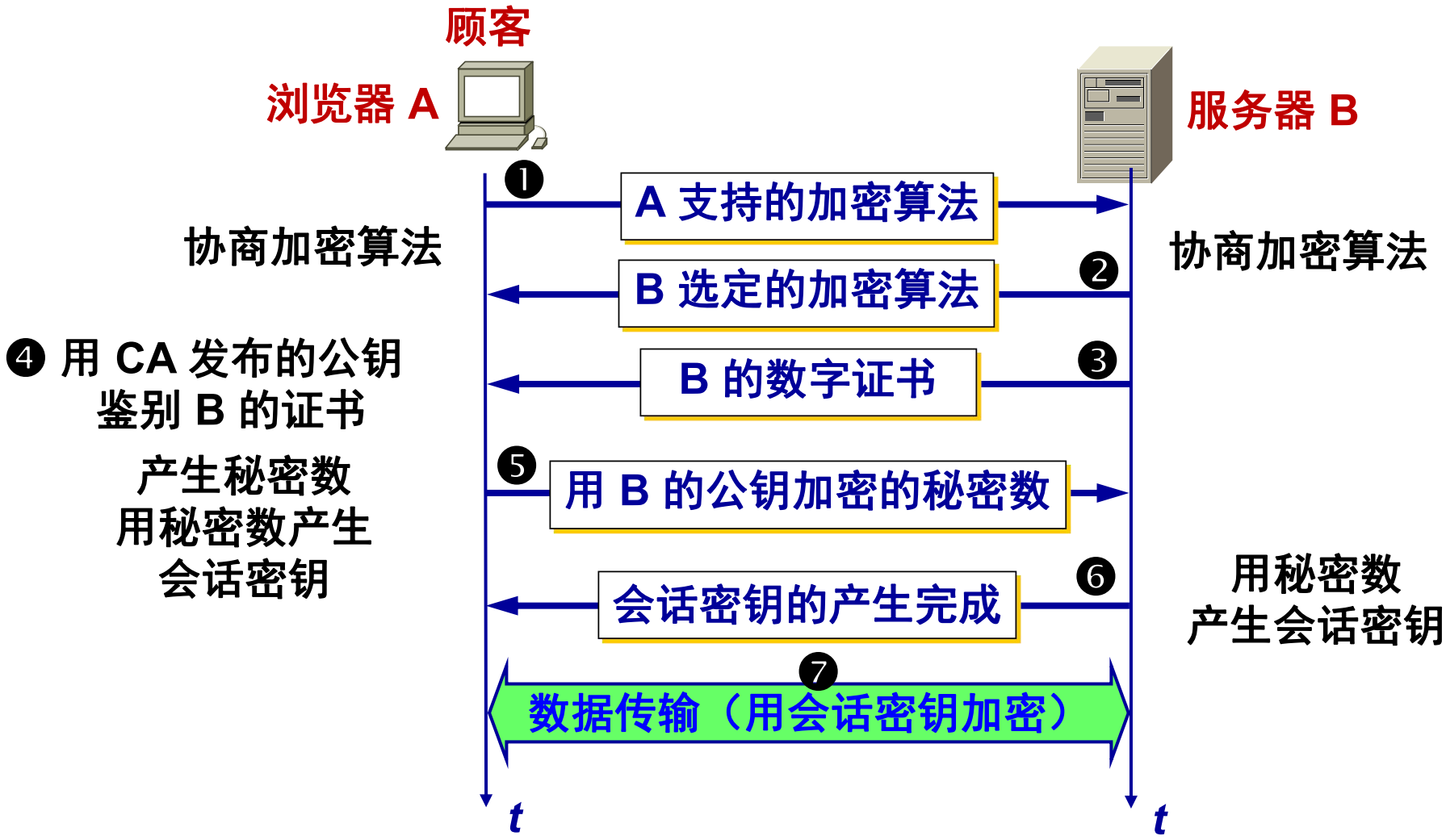
包含加密算法组合的列表

- 工作过程：

4个阶段



TLS 安全会话建立过程

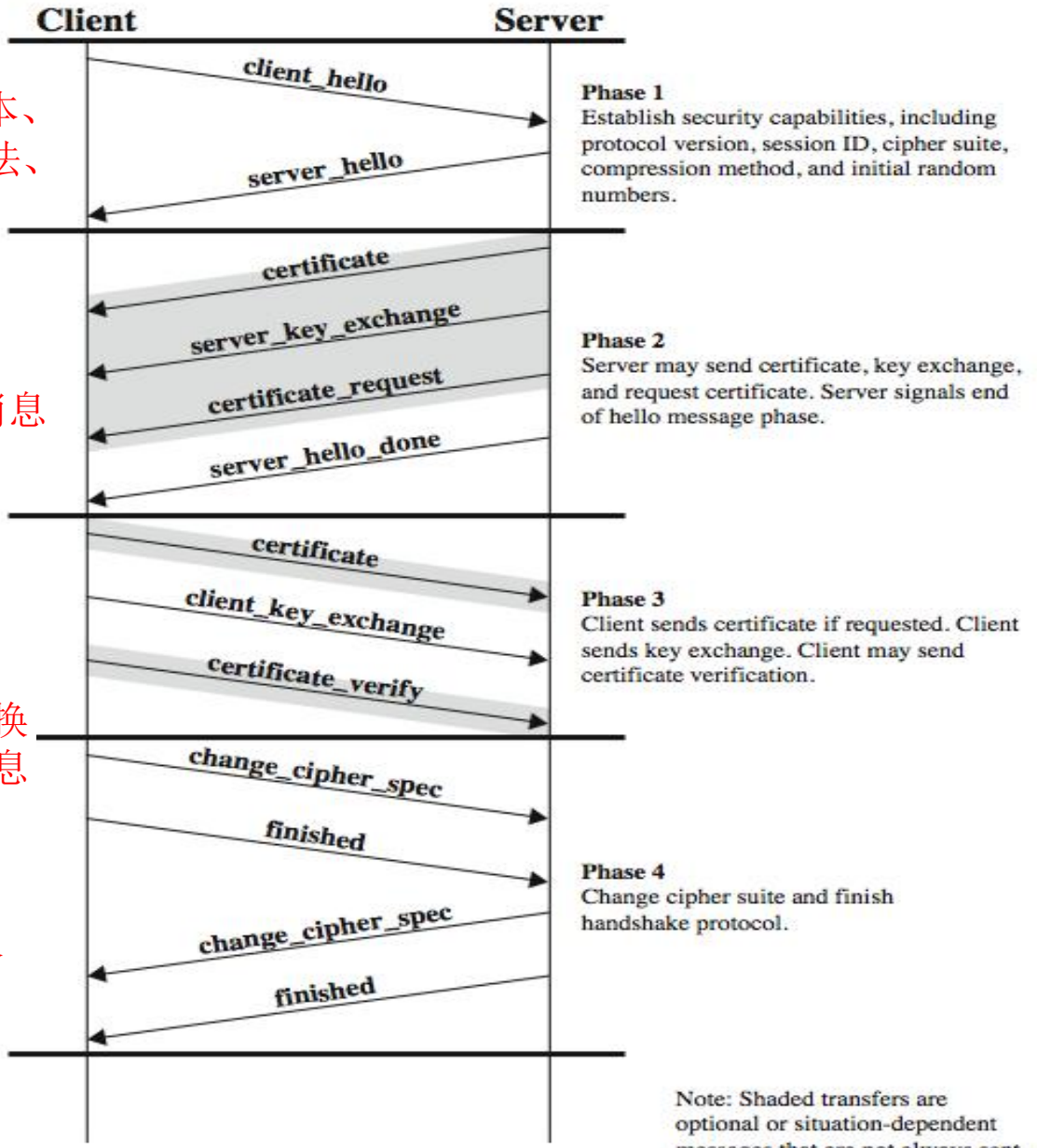


阶段1
建立安全能力：包括协议版本、会话ID、密码套件、压缩方法、初始随机数

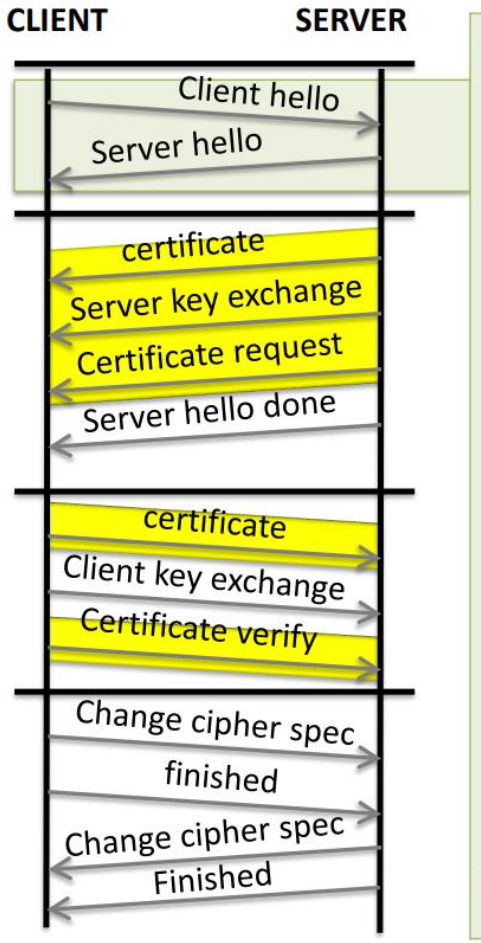
阶段2
服务器发送证书，交换密钥、请求证书。服务器发送hello消息阶段的结束信号

阶段3
如果有证书请求，客户端发送证书，之后客户端发送密钥交换数据，也可以发送证书验证消息

阶段4
变更密码套件并结束握手协议



Handshake Protocol: Phase 1



Establish Security Capabilities

Client hello = Version (The highest TLS version understood by the client),
Nonce,
Session ID,
Compression method
Cipher Suite
Key Exchange
RSA
Fixed Diffie-Hellman
Ephemeral Diffie-Hellman
Anonymous Diffie-Hellman
CipherSpec
Cipher Algorithm (RC4,3DES, AES..)
*MAC algorithm (SHA)
*Cipher Type (block, stream)
*IsExportable
*Hash size
*Key Material (data used in generating write keys)
*IV size

Server hello = replies choosing from the client list a set of algorithms and parameters.

TLS握手协议-- ClientHello消息

Struct {

Protocol Version client_version;

Pseudorandom_random nonce_r_c ;

SessionID session_id;

CipherSuite cipher_suites<2..2^16-1>;

CompressionMethod compression_methods<1..2^8-1>;

} ClientHello

- 密码套件

密钥交换方法-密码算法-MAC算法-密码类型-可否出口-散列长度-密钥材料-IV大小

密钥交换方法

- RSA：对方公钥
- Hiffie-Hellman(全局密钥值q和a)
  - Fixed Hiffie-Hellman
  - Ephemeral Hiffie-Hellman
    - 用于创建暂态(临时或一次性)密钥，是三种DH方法中最安全的，临时的、被认证的密钥
    - 发送方公钥使用数字签名方式传输
    - 提供了完全的前向保密
  - Anonymous Hiffie-Hellman

TLS握手协议-- ServerHello消息

Struct {

Protocol Version client_version;

Pseudorandom_random nonce_r_s;

SessionID session_id;

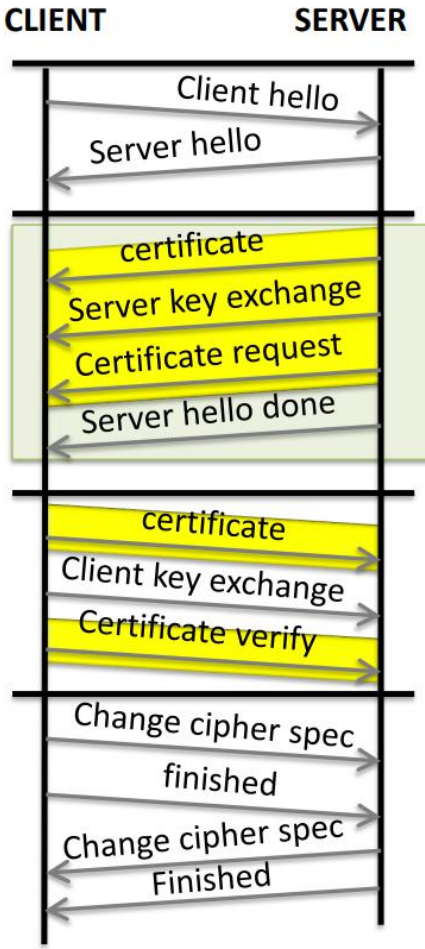
CipherSuite cipher_suite;

CompressionMethod compression_method;

} ClientHello

- 密码算法：RC4,RC5,DES,3DES,IDEA...
- MAC算法：MD5或SHA-1
- 密码类型：流密码或分组密码
- 可否出口：可以或不可以
- 散列长度：0、16(用于MD5)，20（用于SHA-1）
- 密钥材料：字节序列（包含用于产生写密钥的数据）
- IV大小：CBC模式中的初始向量

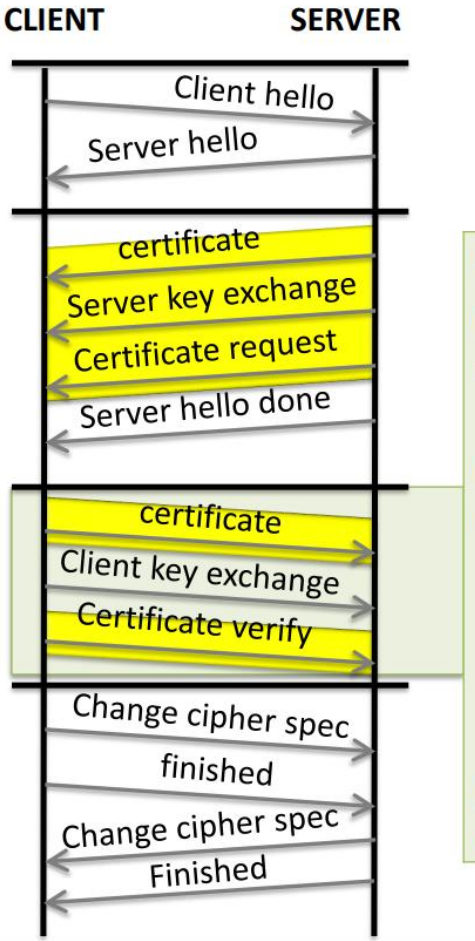
Handshake Protocol: phase 2



Authentication and Key exchange

Certificate	Contains Server's certificate (X.509)
Server key exchange	Needed if • Anonymous Diffie-Hellman • Ephemeral Diffie-Hellman • RSA key exchange (Signature only RSA key)
Certificate request	Server requests a certificate from client (optional) • Certificate type • Certificate authority
Server hello done	Indicate the end of Serve Hello and associated messages

Handshake Protocol: phase 3

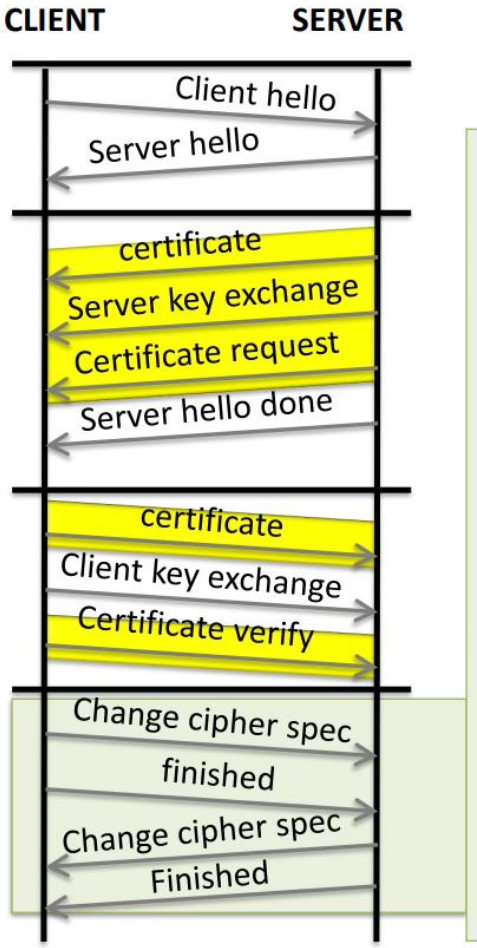


Client Authentication and Key exchange

Certificate message	Send the requested certificate
Client key exchange	Depending on the key exchange mechanism • RSA • Diffie-Hellman (ephemeral and Anonymous) • Fixed Diffie-Hellman
Certificate verify	Verification of clients certificate

23

Handshake Protocol: phase 4

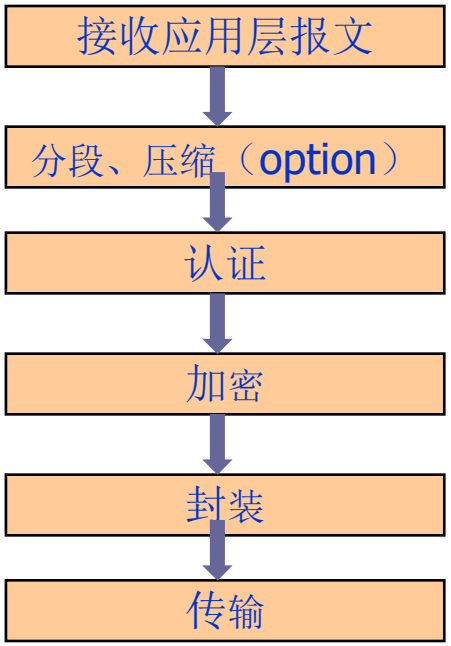


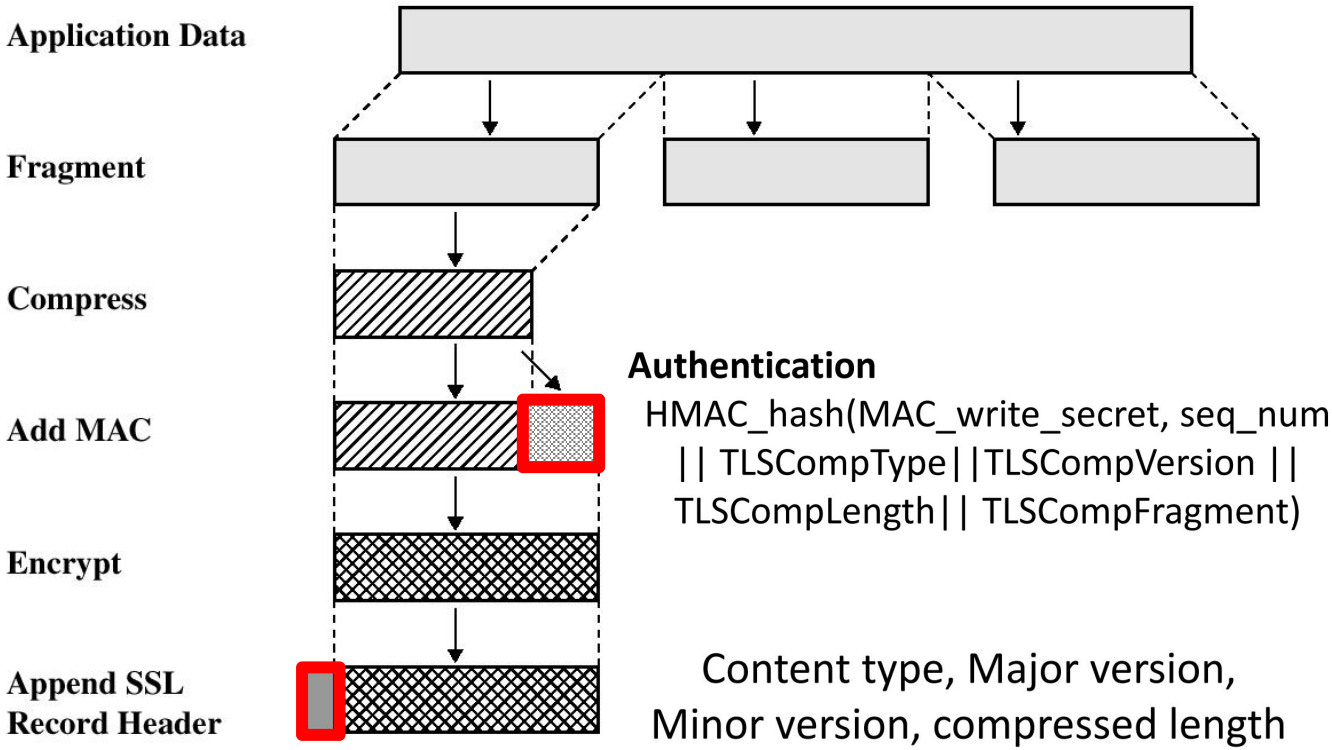
Finish (Change Cipher Spec)

Change cipher spec (Client)	Copies the pending CipherSpec into the current CipherSpec
Finished	Verifies the key exchange and authentication processes to be successful
Change cipher spec (Server)	Copies the pending CipherSpec into the current CipherSpec
Finished	Verifies the key exchange and authentication processes to be successful

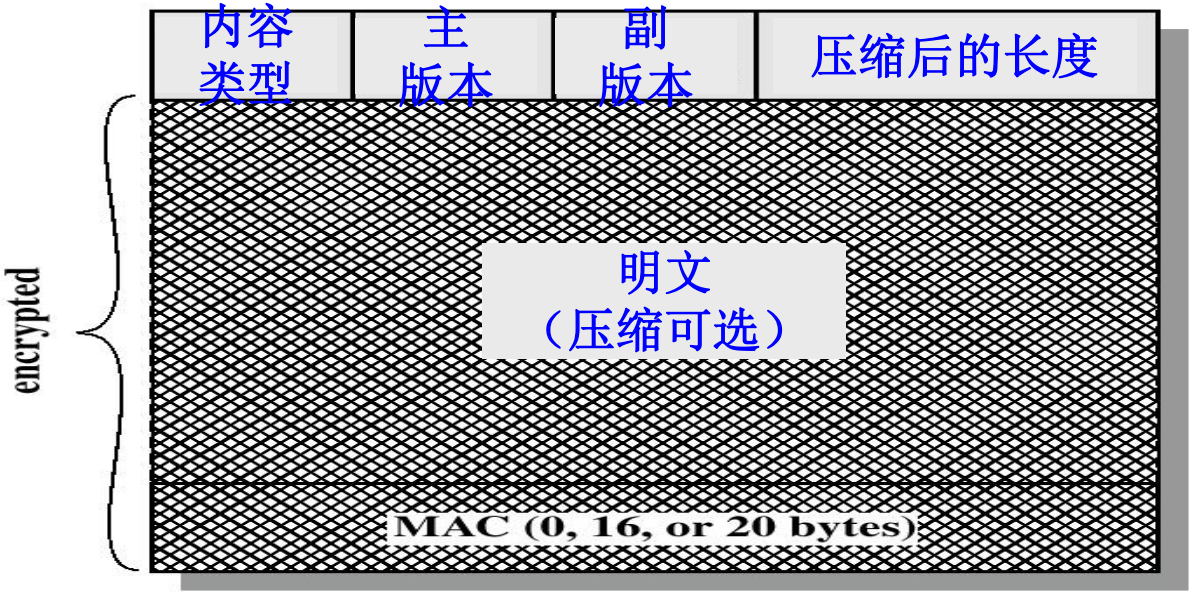
7.2.3 TLS记录协议

- 功能：为SSL连接提供两种服务--机密性、消息完整性
- 工作过程





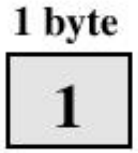
- SSL记录协议的格式



- 使用40 位关键字作为RC4流加密算法
- 支持使用X.509数字认证，如果需要的话用户可以确认发送者是谁

7.2.4 更改密码规格协议

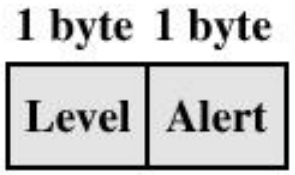
- 目的：为了表示密码策略的变化
- 作用：更新连接的密码组
- 协议：一个报文且由一个字节的单一消息组成，它告知记录层按照当前密码规范中所指定的方式进行加密和压缩
- 完成握手协议前，客户端和服务器端均需要发送该消息以便通知对方其后的记录用刚刚协商的密码规范以及相关的密钥来保护



(a) Change Cipher Spec Protocol

警告协议

- 用来向对等实体运送TLS相关的警告
- 每个报文由两个字节构成，一个指示严重程度（值 2 表示致命错误），一个指示具体警告.

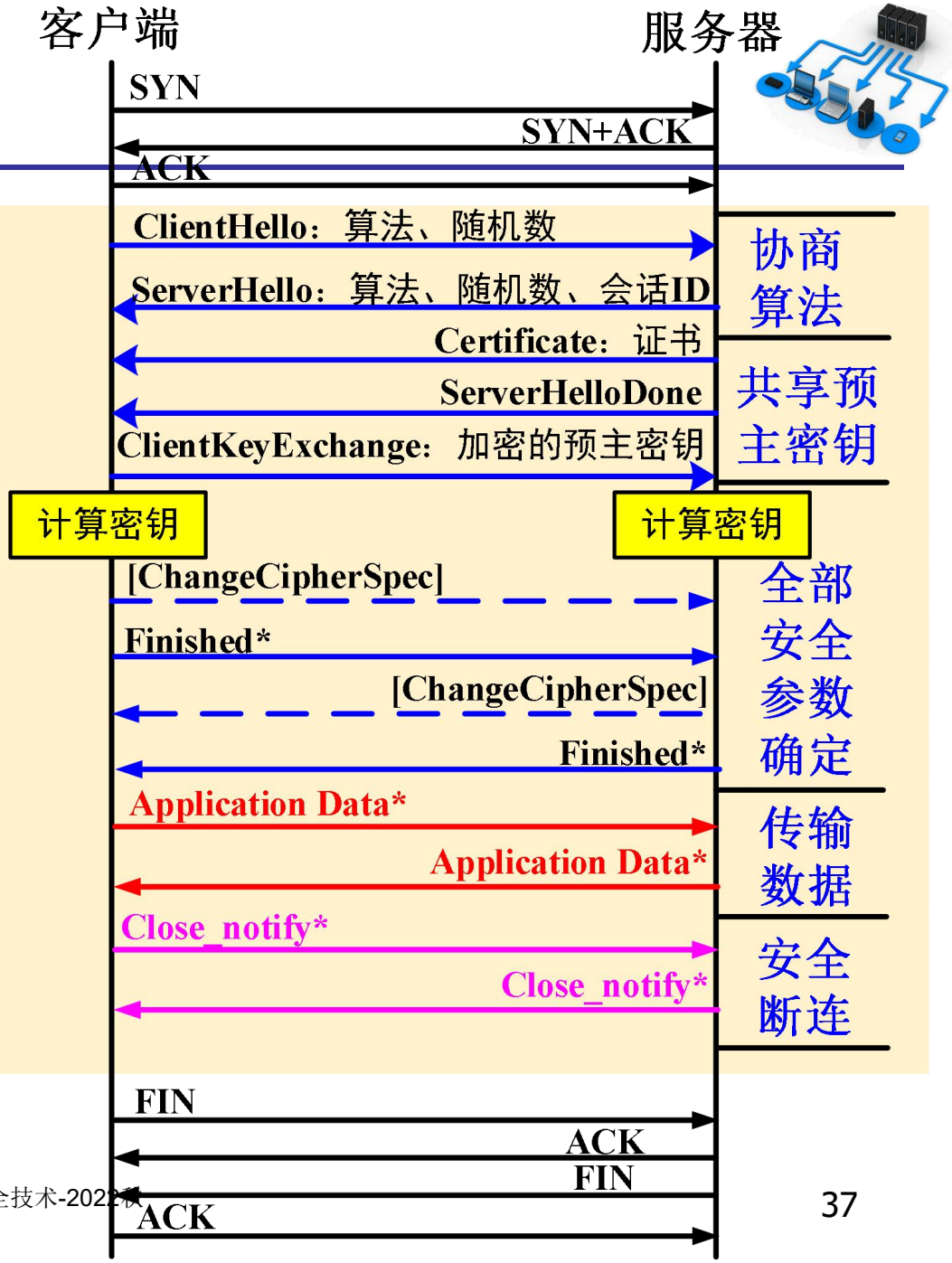


(b) Alert Protocol

- 任何一方检测到错误时，检测一方就向另一方发送消息
  - 如果警告消息有致命后果(14种，书P160)，则双方立即关闭连接并清除与该连接相关的会话标识符、密钥和秘密
  - 如果警告消息有非致命后果(9种，书P161)，则双方可以缓存信息以重用该连接

TLS工作过程小结

- 以握手协议开始，协商加密算法和密钥鉴别客户和服务器的身份；
- 握手完成,利用更改密码协议启用协商的安全参数，所有数据使用握手阶段协商的会话密钥进行加密
- 利用记录协议，加密并使用TCP管道发送数据，从TCP套接字读取数据，解密并把数据导向应用程序
- 数据传输完成后，利用警告协议，通过可认证方式断开连接。



心跳(Heartbeat)协议

- 心跳是一种周期信号，用来表示正常运行或同步
- 心跳协议通常用于监视协议实体的可用性
  - 2012年(RFC6250)--TLS和DTLS心跳扩展中定义了心跳协议
  - 运行在TLS记录协议之上
  - heartbeat_request、heartbeat_response
- 目的
  - 向发送方确保收收方活着，即使双方一段时间没有活动
  - 避免被不允许空闲连接的防火墙关闭

7.2.5 TLS应用

- 针对TLS的攻击
  - 攻击握手协议
  - 攻击记录和应用数据协议
    - 会话劫持
    - 选择明文攻击
  - 攻击PKI
  - 其他攻击

TLS1.3

- 针对TLS的各种攻击被设计出来
- 为了增强TLS的安全性，2018年8月发布TLS 1.3 (RFC8446)
- TLS1.3的改进
  - 取消了对一些操作和函数的支持（如压缩, RC4, MD5...）
  - 客户端和服务器之间只需要一次完整的往返
  - 不允许使用RSA
  - 加密必须使用块密码的认证加密模式进行

TLS的应用

- 利用TLS保护高层应用安全：Web访问、新闻组访问、文件传输和邮件传输等标准应用都可以用TLS保护，其服务器软件都实现了SSL服务器的功能，并支持普通和安全两种服务访问方式。
- 两种策略
  - 分设端口
  - 向上协商

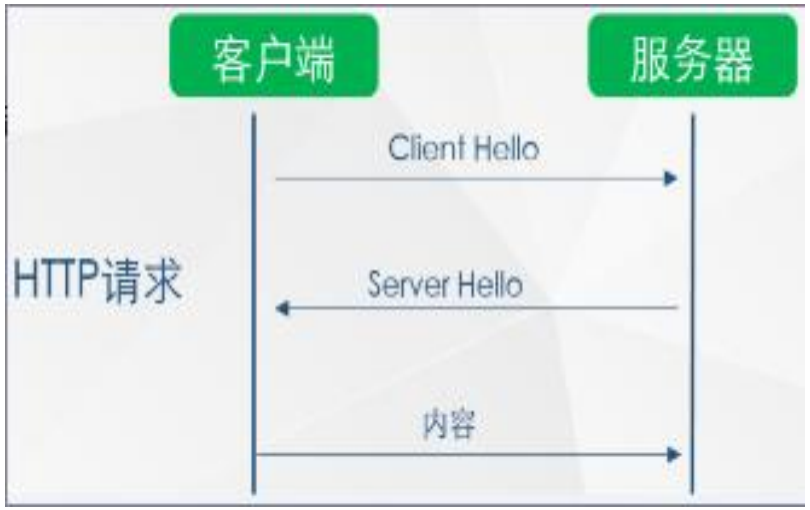
分设端口

- 为不同的访问方式提供不同的监听端口，IETF为一些常用的应用层协议指定了安全访问端口。

普通协议	普通端口	安全协议	安全端口	备注
HTTP	80	HTTPS	443	超文本传输协议
FTP	21	FTPS	990	文件传输协议
FTP-data	20	FTPs-data	989	FTP数据连接
NNTP	119	NNTPS	563	网络新闻传输协议
Telnet	23	Telnets	992	远程登录协议
IMAP	143	IMAPS	993	Internet邮件访问协议
SMTP	25	SMTPS	465	简单邮件传输协议
POP3	110	POP3S	995	邮局协议3
LDAP	389	LDAPS	636	轻量级目录访问协议

7.3 HTTPS

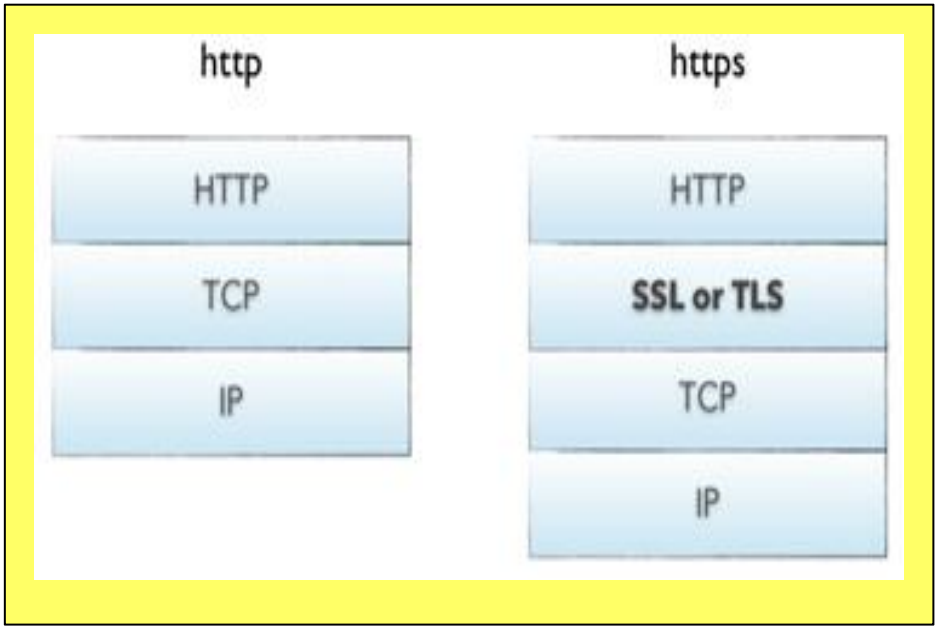
- HTTP的工作原理



- HTTP劫持攻击

HTTPS

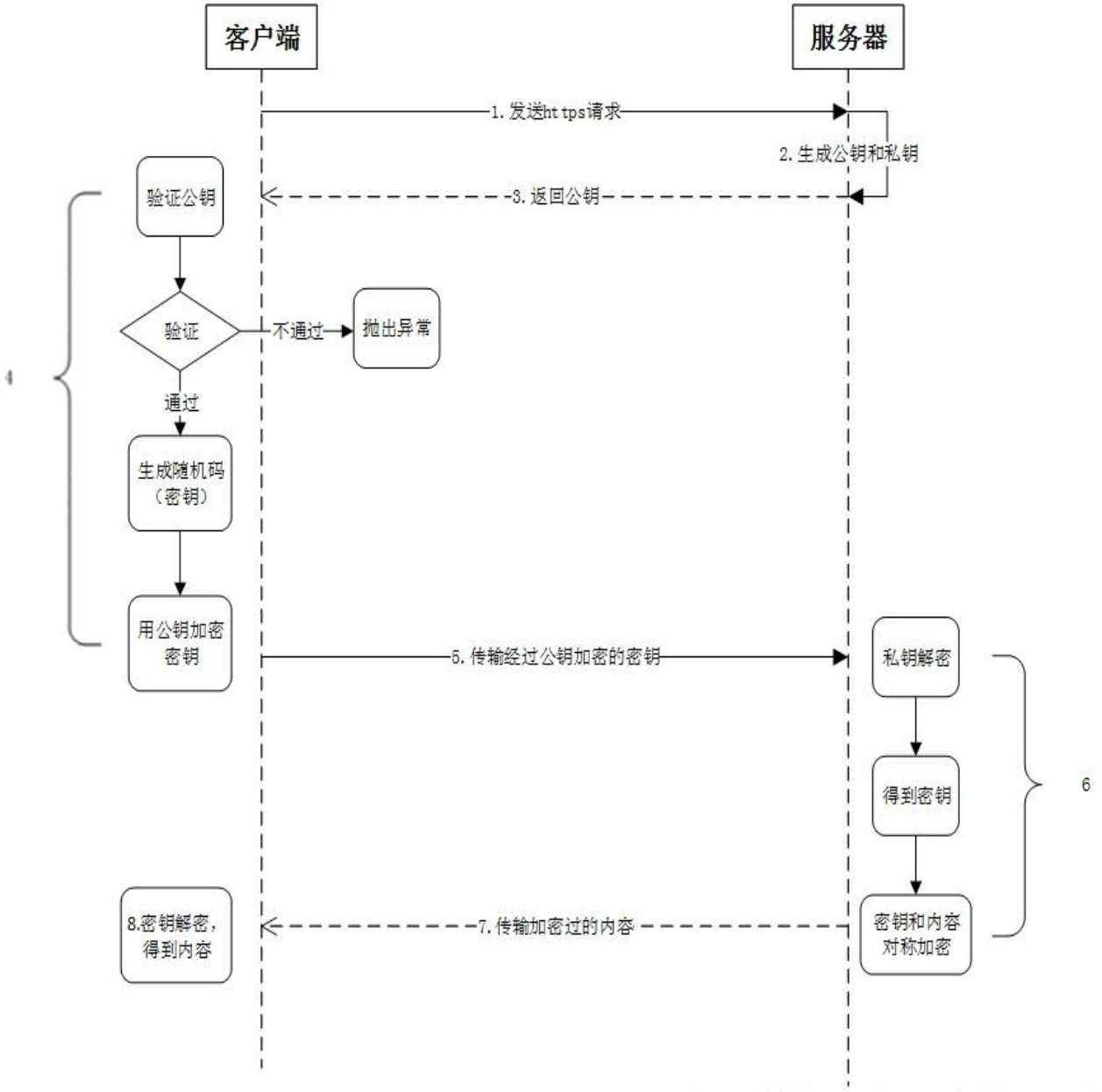
- HTTPS（HTTP over Secure Socket Layer）是以安全为目标的HTTP通道(即HTTP的安全版)，是指用HTTP和TLS（或SSL）的结合来实现网络浏览器和服务器之间的安全通信。



- HTTPS提供了浏览器和服务器之间的身份验证与加密通信，用于Web上安全敏感的通信(如金融交易支付)。
  - HTTPS已经融合到网络浏览器中
  - 网络服务器需要支持HTTPS通信（搜索引擎不支持）
- HTTPS由在2000年五月公布的RFC 2818正式确定

HTTPS

- 基本工作过程
  - 协商加密算法
  - 获得公钥证书
  - 验证公钥证书
  - 交换会话密钥
  - 加密信息传输



HTTPS

- 连接初始化
  - HTTP层：用户在合适的端口向服务器发起一个连接
  - TLS层：发送TLS ClientHello，开始交换TLS信号
  - TLS信号交换完毕，发起第一次HTTP请求
  - 所有HTTP数据都以TLS应用数据的形式发送
  - 说明：一个TLS连接请求的建立伴随着一个TCP连接的建立。

HTTPS

- 连接关闭
  - 关闭HTTPS连接要求关闭TLS与其对应的远程终端之间的连接，这意味着要求关闭相应的TCP连接
  - HTTP层：用户或者服务器在HTTP记录中加入“connection: close”指示，指示一个连接的关闭
  - TLS层：两端使用TLS警报通信协议发出一个“close_notify”警告来关闭连接
  - TCP层：释放响应的连接

HTTPS

- 使用HTTPS时，以下信息被加密
  - 文件的URL
  - 文件的内容
  - 浏览器表单的内容（由浏览器的使用者填写）
  - 浏览器和服务器之间发送的Cookie
  - HTTP报头的内容

HTTPS

- 特点
  - 保证传输信息安全：数据加密和完整性保护
  - 可防止劫持：身份认证
  - 需要申请证书
  - 由于增加了安全层，访问速度可能有所减慢

HTTPS

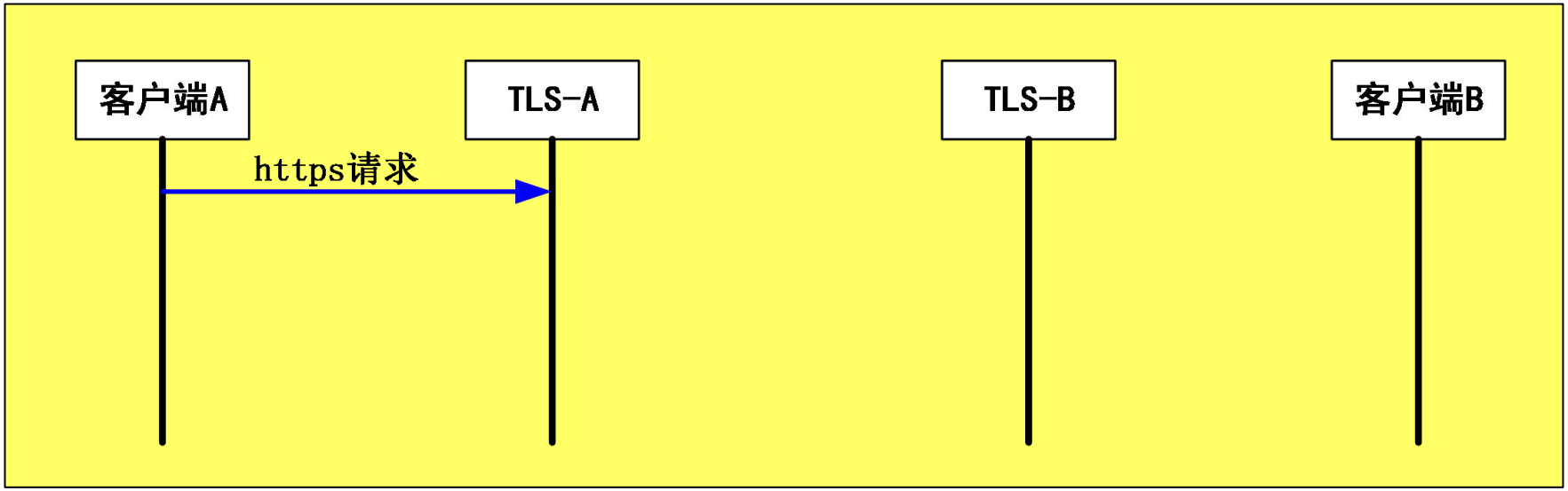
- HTTP与HTTPS两者的区别
  - 浏览器用户视角：HTTPS的URL地址是“https//”
  - HTTPS协议需要到CA申请证书
  - HTTP信息是明文传输，HTTPS是加密传输
  - HTTP不支持对服务器的认证，HTTPS可以对服务器认证
  - 连接端口号：HTTP用80端口，HTTPS用443端口
  - HTTP的低层是TCP连接, HTTPS的低层是SSL或TLS
  - HTTP的连接简单，是无状态的；HTTPS协议是由SSL+HTTP协议构建的可进行加密传输、身份认证的网络协议

HSTS

- HSTS(HTTP Strict Transport Security,HTTP严格安全传输)是国际互联网工程组织 IETF 正在推行一种新的 Web 安全协议
- HSTS的作用:强制客户端（如浏览器）使用HTTPS与服务器创建连接
- 采用HSTS协议的网站将保证浏览器始终连接到该网站的HTTPS加密版本
- 作用：用来抵御SSL剥离攻击

问题

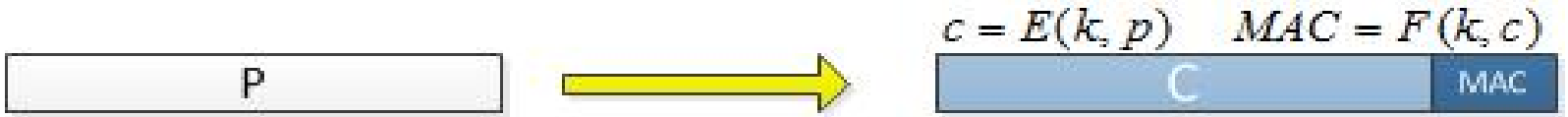
- HTTPS的低层是TLS协议，请将HTTPS工作过程与TLS协议工作过程相结合，描述完整的浏览器访问服务器网页的过程。



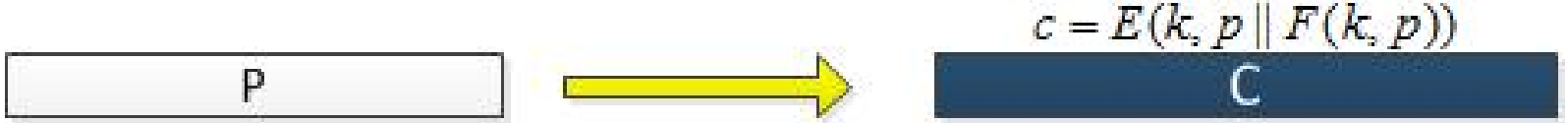
问题1：

- IPSec和TLS均对数据进行完整性和保密性的保护，但是其采用的认证加密顺序是不同的，为什么？请尝试查阅资料分析原因。

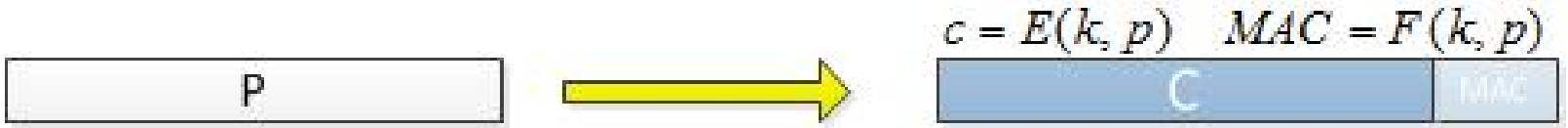
Encrypt-then-MAC (EtM)　IPSec



MAC-then-Encrypt (MtE)　SSL/TLS



Encrypt-and-MAC (E&M)　Secure Shell(SSH)



- 请查阅资料调研什么是AE(Authenticated Encryption)模式（即认证加密模式）

问题2：

- 请自学TLS1.3的工作原理，并论述说明TLS1.3在效率和安全性方面进行了什么样的改进，为什么？