



现代密码学

第七章 密码协议

信息与软件工程学院

7.1 密码协议概述

- 协议是一种有序的过程，每一步必须依次执行；需要至少两个或两个以上的参与者，且最终要完成某种任务，即实现某种功能
- 协议对参与者一定是完全公开的，且按照规范动作
- 协议的基本要求是有效性、公平性和完整性
- 密码协议也叫安全协议，是建立在密码体制的基础上的一种交互式通信协议，借助于密码算法来达到安全功能
- 密码协议所能实现的安全功能是多种多样的
 - 身份认证、密钥协商、消息鉴别、公平抛币
 - 多方计算、秘密共享、不经意传输
 - 零知识证明...
- 密码协议的应用：
 - 安全通信(通信的保密认证等)、电子选举、电子拍卖、公平电子交易...

认证协议

Authentication Protocols

7.2.1 认证协议概述

- 认证：一个实体向另一个实体证明某种声称的东西的过程
- 认证协议：主要目标是确认某个主体的真实性，确保信息的安全性
- 安全可靠的通信除需进行消息的认证外，还需建立一些规范的协议对数据来源的可靠性、通信实体的真实性加以认证，以防止欺骗、伪装等攻击
- 认证的分类
 - 身份认证：也称实体认证。验证消息发送者所声称的身份，发起通信或访问的实体是否具有合法身份和相应权限
 - 密钥建立认证：生成、获得加解密密钥
 - 数据源认证：验证消息与其发送主体的一致性，数据从哪儿来
 - 消息完整性认证：验证消息是否被篡改
- 这些认证常常一起进行，也有时单独进行
 - 即通信之前首先进行实体认证(身份认证)，身份认证之后会协商出会话密钥用于对每个传输数据的数据源认证和完整性认证

7.2.1 认证协议概述

- 身份认证有两个层次：身份证实和身份识别，二者差别是是否出示身份
- 身份证实：你是否是你宣称的你，一般的身份认证都是指这种情况
 - 验证方首先知道要验证的身份是谁，进一步证实来访或与之通信的人是否具有该身份
 - 一般用于A和B确定通信时所用，通常的网络认证协议都是身份证实
 - 具体技术：输入个人信息，经公式或算法运算所得结果与卡中或数据库中存储信息经公式运算所得结果比较
- 身份识别：我是否知道你是谁
 - 验证方不知道来访人是否为合法身份，没有比较确定的目标，只要满足某个条件就可判定身份的特点。验证者一般为权威机构
 - 一般在实体认证中需要，比如判断来访者是否是在逃犯，是否为密码开启者，是否为本公司员工。通常用指纹、虹膜技术
 - 具体技术：输入个人信息，经过处理提取模版信息，试着在存储数据库中找到一个与之匹配的模版而后得出结论

7.2.1 认证协议概述

- 数据源认证和消息完整性认证的区别
- (1) 数据源认证必然涉及通信，是一种安全服务
 - 消息完整性认证不一定包含通信，可用于存储的数据
- (2) 数据源认证一定涉及消息源识别，而消息完整性不一定涉及该过程
 - 比如用户A将一个由用户C签名的消息文件发给用户B，那么用户B能够通过验证C的签名判断消息的完整性，但是消息的来源却不是C而是A
- (3) 数据源认证必然涉及确认消息的新鲜性，而数据完整性却无此必要
 - 一组老的数据，或者重放的数据可能有完善的数据完整性。而数据源认证要求确认收到的消息是新鲜的，即使当前新发送的，而不是旧消息的重放

7.2.1 认证协议概述

- 在认证的几个类别里，消息完整性认证是最普通的层次，在前几种认证中一般都包含了消息完整性认证
- 认证需求也不同
 - 有的协议只需要认证身份
 - 有的协议除了认证身份还需要建立之后通信的密钥以保护通信的安全，即身份认证与密钥协商(密钥建立认证协议)
 - 有的协议主要是考虑对消息源的认证，这时一般首先要认证身份并建立密钥，然后再基于该会话密钥对传输的每个消息进行认证，以使收方确信收到的消息来自发方，而且不是重放、在顺序、时间等方面都是正确的，即完整性
- 对于身份认证，主要验证用户知道什么(如口令、密钥、私钥、秘密等)、验证用户拥有什么(如IC卡等)、验证用户具有什么特征(如指纹、掌纹、虹膜、DNA等)
- 其中基于口令的身份认证是比较重要的一类，具有广泛的应用，因为用户不需要携带或记忆复杂的密钥或者认证信息

A decorative blue horizontal bar with white horizontal stripes is positioned to the left of the main title.

常见攻击

- 常见的针对认证和密钥建立协议的攻击
 - 重放攻击 Replay attack
 - 中间人攻击 Man-in-the-middle
 - 已知密钥攻击：从以前用过的密钥确定新密钥
 - 假冒攻击：Impersonation attack
 - 篡改或替换
 - 字典攻击Dictionary attack：针对口令的一类攻击
 - 并行会话攻击
 - 反射攻击
 - 拒绝服务攻击
 -
-

7.2.3 基本认证技术

- 本节介绍在各种认证协议实现中被认为很好的结构
 - 证明消息的新鲜性和主体活现性的标准机制
 - 双向认证和单向认证
 - 包含可信第三方的认证
- (一) 消息的新鲜性和主体的活现性
 - 消息的新鲜性即实时性是数据源认证所不可或缺的一部分，而在这一过程中主体也要考虑通信对端的身份的真实性及是否在线，即主体活现性，因此证明消息的新鲜性或主体的活现性就成为认证协议的最基本组成部分
 - 消息新鲜性经常伴随着主体活现性的认证，因此我们就以新鲜性的讨论为主，重点考虑消息新鲜性的实现
 - 消息的新鲜性也称为实时性

7.2.3 基本认证技术

- 实时性(新鲜性): 对防止消息的重放攻击极为重要, 一种方法是对交换的每一条消息都加上一个序列号, 一个新消息仅当它有正确的序列号时才被接收。这种方法的困难性是要求每个用户分别记录与其他每一用户交换的消息的序列号, 从而增加了用户的负担, 所以序列号方法一般不用于认证和密钥交换。
- 实现消息新鲜性有两种技术: 时间戳和询问应答机制
 - 时戳
 - 如果A收到的消息包括一时戳, 且在A看来这一时戳充分接近自己的当前时刻, A才认为收到的消息是新的并接受之。这种方案要求所有各方的时钟是同步的
 - 询问-应答
 - 用户A向B发出一个一次性随机数作为询问, 如果收到B发来的消息(应答)也包含一正确的一次性随机数, A就认为B发来的消息是新的并接受之。

7.2.3 基本认证技术

- 时戳法不能用于面向连接的应用过程
 - 这是由于时戳法在实现时固有的困难性
 - 首先是需要不同的处理器时钟之间保持同步，那么所用的协议必须是容错的以处理网络错误，并且是安全的以对付恶意攻击
 - 第二，如果协议中任一方的时钟出现错误而暂时地失去了同步，则将使敌手攻击成功的可能性增加
 - 最后还由于网络本身存在着延迟，因此不能期望协议的各方能保持精确的同步。所以任何基于时戳的处理过程、协议等都必须允许同步有一个误差范围。考虑到网络本身的延迟，误差范围应足够大；考虑到可能存在的攻击，误差范围又应足够小

7.2.3 基本认证技术

- 询问-应答方式则不适合于无连接的应用过程
 - 这是因为在无连接传输以前需经询问-应答这一额外的握手过程，这与无连接应用过程的本质特性不符。
 - 对无连接的应用程序来说，利用某种安全的**时间服务器**保持各方时钟同步是防止重放攻击最好的方法
- 关于有状态和无状态的协议
 - 如果**协议的运行需要一些历史和当前的状态作为上下文来维护，则称为有状态的协议**，这样一次协议的运行要考虑历史的状态，会造成负担或者失同步等问题
 - 如果**每次协议运行独立于历史状态参数，则是无状态的**，实现和资源的占用更少。

7.2.3 基本认证技术

- (二) 双向认证与单向认证
 - A和B是网络的两个用户，他们想通过网络先建立安全的共享密钥再进行保密通信。**A(B)如何确信自己正在和B(A)通信而不是和C通信呢？**即双方的身份认证
 - 这种通信方式为**双向通信**，此时的认证称为**相互认证**
 - 双向认证也叫相互认证、双方认证等
 - 类似地，对于**单向通信**来说，认证称为**单向认证**

A decorative graphic consisting of several horizontal blue bars of varying lengths, located to the left of the section header.

7.2.3 基本认证技术

- (三)包含可信第三方的认证
 - A和B直接进行认证的前提是二者之间有共享的密钥，有时在实际应用中，这种方式并不实用，因为用户多时两两用户之间都要共享密钥，因此一种更为常见的方式是基于可信第三方的认证
 - 这时，双方都与可信第三方共享初始密钥，系统扩展性好，便于密钥管理，借助可信第三方提供的帮助来实现认证和密钥协商
 - 可信第三方可以在线也可以离线
-

7.3 密钥建立认证协议

- A、B两个用户在建立共享密钥时需要考虑的核心问题是保密性和实时性
- 保密性：为防止会话密钥的伪造或泄露，会话密钥在通信双方之间交换时应为密文形式，所以通信双方事先就应有密钥或公开钥
- 实时性即新鲜性，可使用前述的基本认证技术
- 通信双方建立共享密钥时可采用单钥加密体制和公钥加密体制
- 本节主要介绍以下几种认证协议
 - 1. Needham schreoder密钥分配协议(基于对称密钥技术)
 - 2. 基于公钥加密的密钥分配协议
 - 3. KERBEROS认证协议

7.3.1 NS密钥分配协议

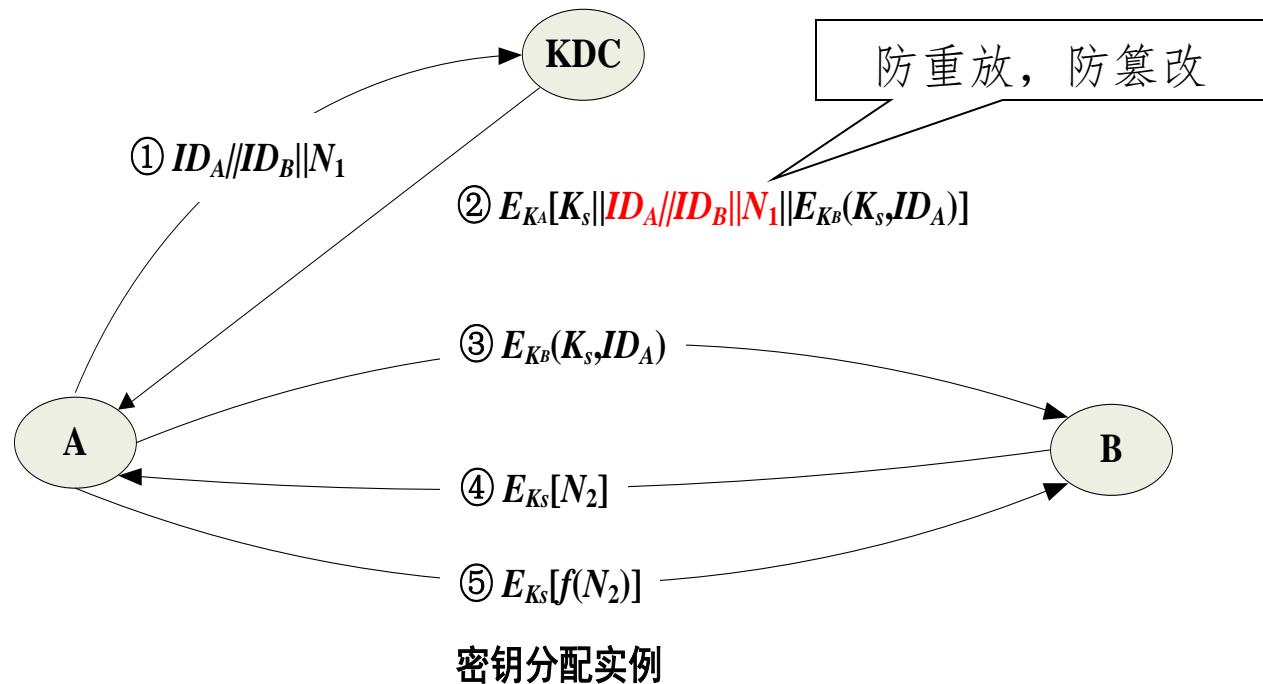
- 基于单钥加密体制设计
 - 需要有一个称为鉴别服务器的可信权威机构（密钥分发中心KDC），网络中每一用户都与KDC有一共享密钥，称为**主密钥**
 - KDC为通信双方建立一个短期内使用的密钥，称为**会话密钥**，并用主密钥加密会话密钥后分配给两个用户
 - 这种分配密钥的方式在实际应用中较为普遍采用，如Kerberos系统采用的就是这种方式
- Needham-Schroeder协议
 - 1978年，Needham和Schroeder设计的著名的采用KDC的密钥分配过程
 - 若用户A欲与用户B通信，则用户A向鉴别服务器申请会话密钥。在会话密钥的分配过程中，双方身份得以鉴别

7.3.1 NS密钥分配协议

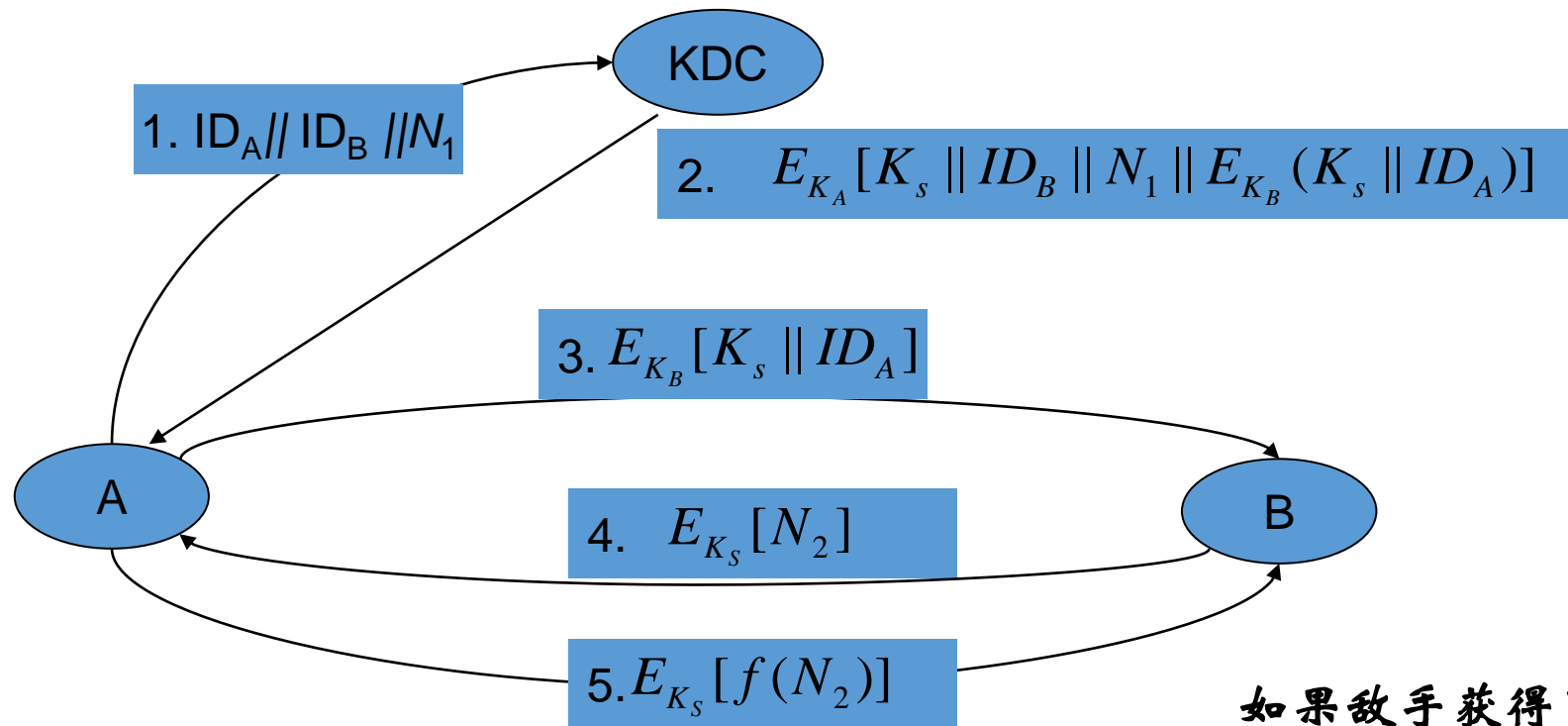
- 假定两个用户A、B分别与密钥分配中心KDC有一个共享的主密钥 K_A 和 K_B ，A希望与B建立一个共享的一次性会话密钥

- ① $A \rightarrow KDC$: $ID_A || ID_B || N_1$ // A向KDC发出会话密钥请求
- ② $KDC \rightarrow A$: $E_{K_A}[K_S || ID_B || N_1 || E_{K_B}[K_S || ID_A]]$
- ③ $A \rightarrow B$: $E_{K_B}[K_S || ID_A]$
- ④ $B \rightarrow A$: $E_{K_S}[N_2]$
- ⑤ $A \rightarrow B$: $E_{K_S}[f(N_2)]$

- N_1 为一次性随机数，可以是时戳、计数器或随机数
- 每次的 N_1 都应不同，且为防止假冒，应使敌手对 N_1 难以猜测。因此用随机数作为这个识别符最为合适。



Needham-Schroeder协议



如果敌手获得了旧会话密钥，则可以冒充A重放3，并且可回答5，成功的欺骗B

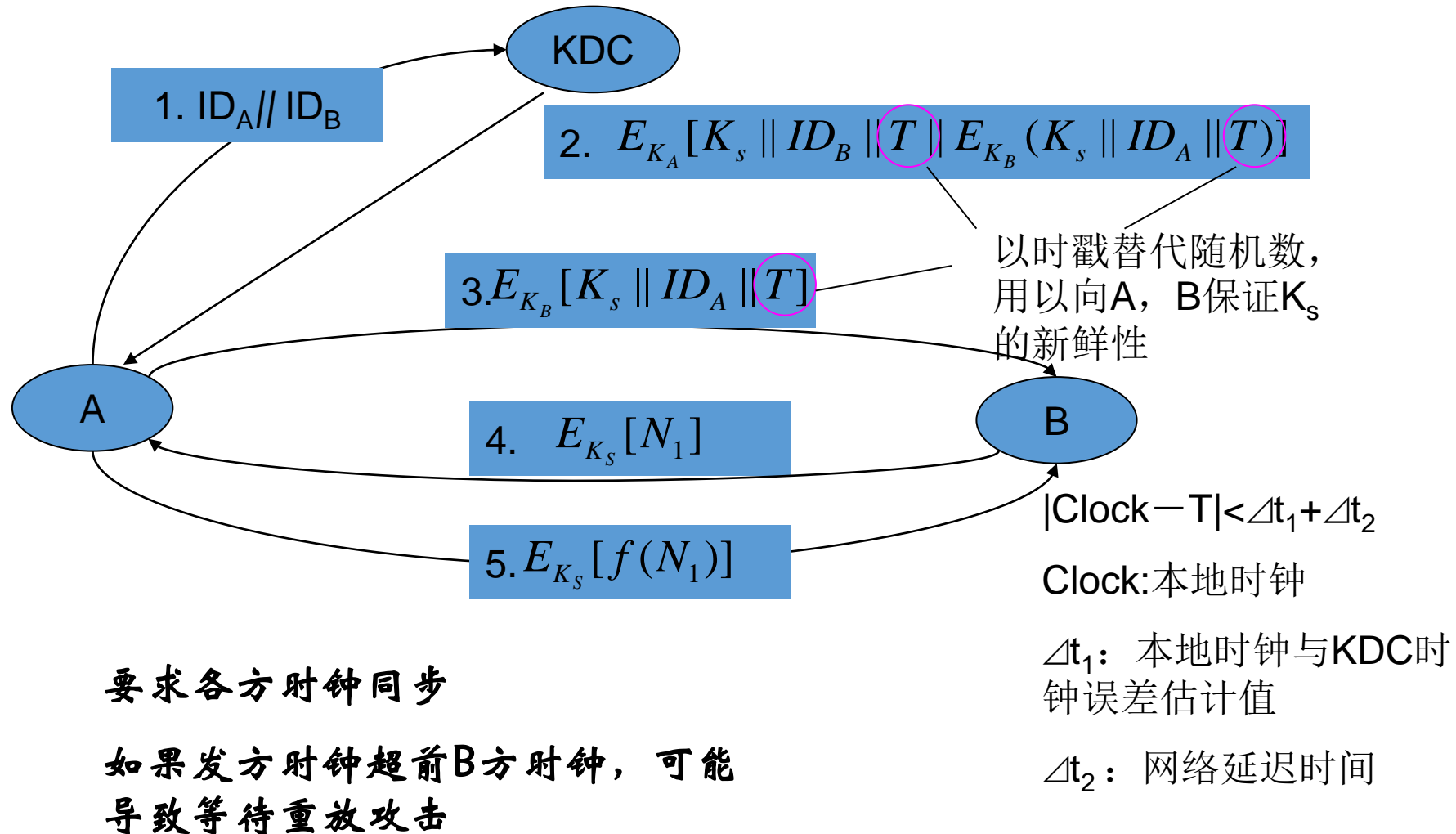
7.3.1 NS密钥分配协议

- 协议的目的是由**KDC**为**A**、**B**安全地分配会话密钥 K_S
 - **A**在第②步安全地获得了 K_S
 - 第③步的消息仅能被**B**解读，因此**B**在第③步安全地获得了 K_S
 - 第④步中**B**向**A**示意自己已掌握 K_S ， N_2 用于向**A**询问自己在第③步收到的 K_S 是否为一新会话密钥，
 - 第⑤步**A**对**B**的询问作出应答，一方面表示自己已掌握 K_S ，另一方面由 $f(N_2)$ 回答了 K_S 的新鲜性。
 - 第④、⑤两步用于防止对第③步的重放攻击
- 以上协议易遭受另一种重放攻击
 - 假定敌手能获取旧会话密钥，则冒充**A**向**B**重放第③步的消息后，就可欺骗**B**使用旧会话密钥
 - 敌手进一步截获第④步**B**发出的询问后，可假冒**A**作出第⑤步的应答
 - 敌手就可冒充**A**使用经认证过的会话密钥向**B**发送假消息

7.3.1 NS密钥分配协议

- (2) Needham-Schroeder协议的改进方案之一
- 在第②步和第③步加上一时戳，改进后的协议如下：
 - ① $A \rightarrow KDC: ID_A \| ID_B$
 - ② $KDC \rightarrow A: E_{KA}[K_S \| ID_B \| T \| E_{KB}[K_S \| ID_A \| T]]$
 - ③ $A \rightarrow B: E_{KB}[K_S \| ID_A \| T]$
 - ④ $B \rightarrow A: E_{KS}[N_1]$
 - ⑤ $A \rightarrow B: E_{KS}[f(N_1)]$
- 其中 T 是时戳，用以向A、B双方保证 K_S 的新鲜性。A和B可通过下式检查 T 的实时性：
 - $|\text{Clock} - T| < \Delta t_1 + \Delta t_2$
 - Clock为用户（A或B）本地的时钟
 - Δt_1 是用户本地时钟和KDC时钟误差的估计值
 - Δt_2 是网络的延迟时间
- T 是经主密钥加密的，所以敌手即使知道旧会话密钥，并在协议的过去执行期间截获第③步的结果，也无法成功地重放给B
 - 因B对收到的消息可通过时戳检查其是否为新的

Needham-Schroeder改进协议 (1)



7.3.1 NS密钥分配协议

- 以上改进还存在以下问题：
 - 方案主要依赖网络中各方时钟的同步，这种同步可能会由于系统故障或计时误差而被破坏
 - 等待重放攻击：如果发送方的时钟超前于接收方的时钟，敌手就可截获发送方发出的消息，等待消息中时戳接近于接收方的时钟时，再重发这个消息
 - 这种重放可造成重复接收等问题

7.3.1 NS密钥分配协议

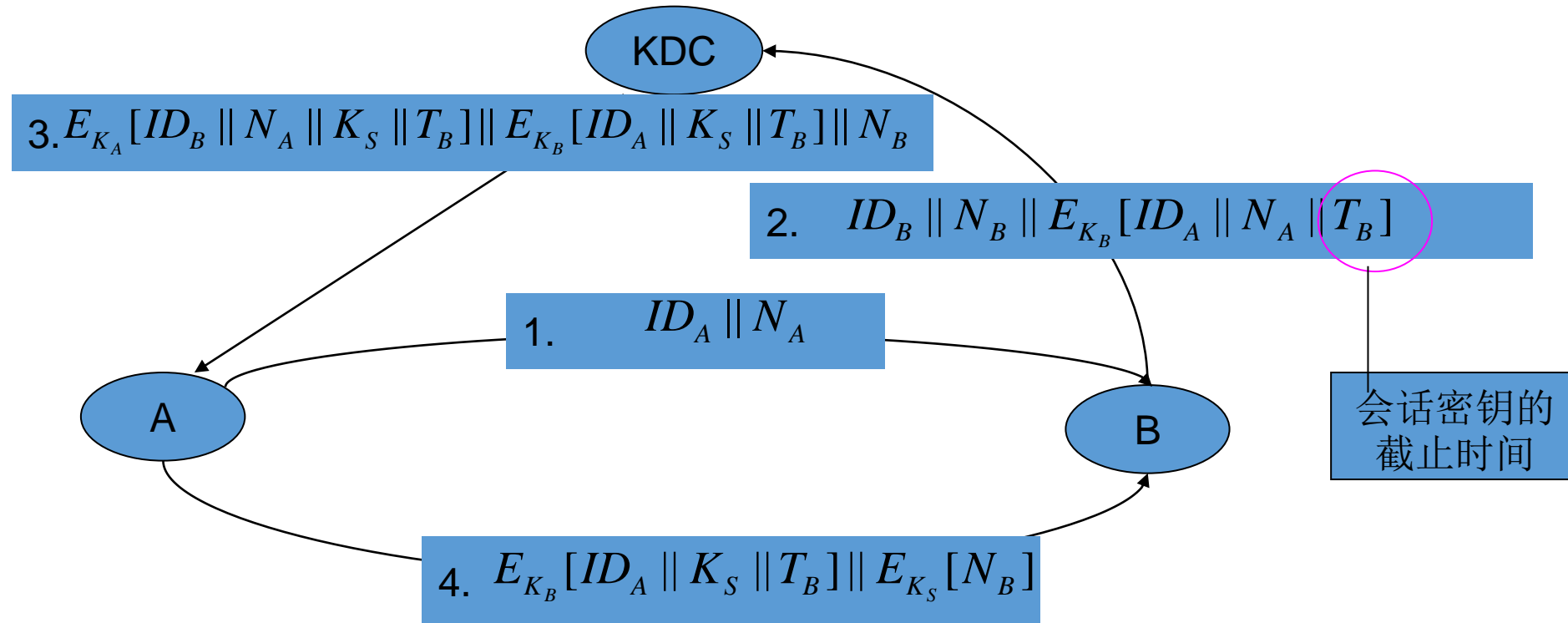
- 抗等待重放攻击

- 方法一：要求网络中各方以KDC的时钟为基准定期检查并调整自己的时钟
- 方法二：使用一次性随机数的握手协议
 - 这就是NS协议的另一种改进方法

- (3) Needham-Schroeder协议的改进方案之二

- ① $A \rightarrow B: ID_A \| N_A$
- ② $B \rightarrow KDC: ID_B \| N_B \| E_{KB}[ID_A \| N_A \| T_B]$
- ③ $KDC \rightarrow A: E_{KA}[ID_B \| N_A \| K_S \| T_B] \| E_{KB}[ID_A \| K_S \| T_B] \| N_B$
- ④ $A \rightarrow B: E_{KB}[ID_A \| K_S \| T_B] \| E_{KS}[N_B]$
 - T_B 是B建议的证书(会话密钥)截止时间，用于截止时间前再次发起会话时 K_S 是否可用的判别时间

Needham-Schroeder改进协议 (2)



7.3.1 NS密钥分配协议

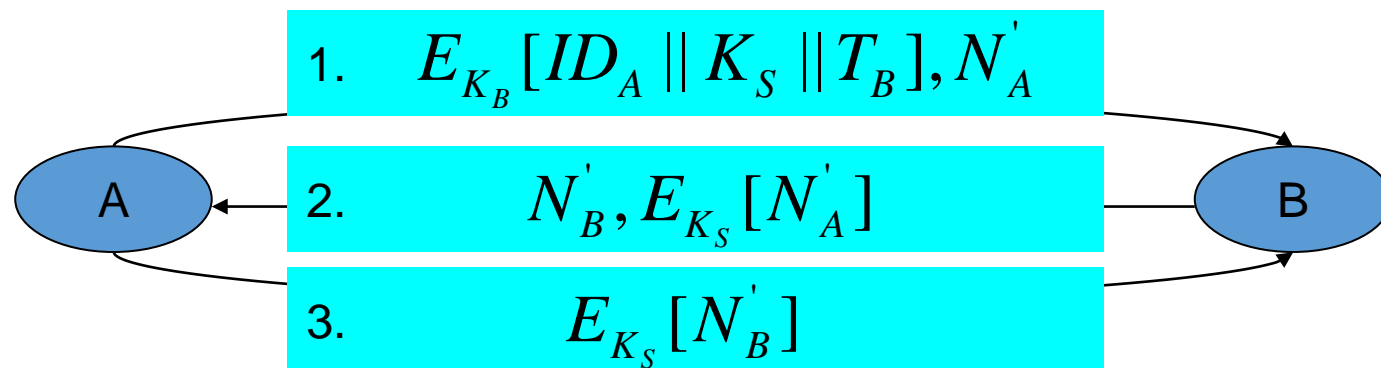
- 协议的具体含义如下：
 - ① A发出的新的一次性随机数 N_A 以后将与会话密钥 K_S 一起以加密形式返回给A，以保证A收到的会话密钥的新鲜性。
 - ② B向KDC发出与A建立会话密钥的请求，发出的一次性随机数 N_B 以后将与会话密钥一起以加密形式返回给B以向B保证会话密钥的新鲜性
 - ③ KDC将B产生的 N_B 连同由KDC与B共享的密钥 K_B 加密的 $ID_A \| K_S \| T_B$ 一起发给A，其中 K_S 是KDC分配的会话密钥
 - $E_{KB}[ID_A \| K_S \| T_B]$ 由A当作票据用于以后的认证。
 - ④ A将票据 $E_{KB}[ID_A \| K_S \| T_B]$ 连同由会话密钥加密的一次性随机数 N_B 发往B，B由票据得到会话密钥 K_S ，并由 K_S 得 N_B 。 N_B 由会话密钥加密的目的是B认证了自己收到的消息不是一个重放，而的确是来自于A。

7.3.1 NS密钥分配协议

- 如果A保留由协议得到的票据 $E_{KB}[ID_A \| K_S \| T_B]$ ，就可在有效时间范围内不再求助于认证服务器而由以下方式实现双方的新认证：
 - ① $A \rightarrow B$: $E_{KB}[ID_A \| K_S \| T_B], N_A'$
 - ② $B \rightarrow A$: $N_B', E_{KS}[N_A']$
 - ③ $A \rightarrow B$: $E_{KS}[N_B']$
- B在第①步收到票据后，可通过 T_B 检验票据是否过时，而新产生的一次性随机数 N_A' ， N_B' 则向双方保证了没有重放攻击
- 以上协议中时间期限 T_B 是B根据自己的时钟定的，因此不要求各方之间的同步

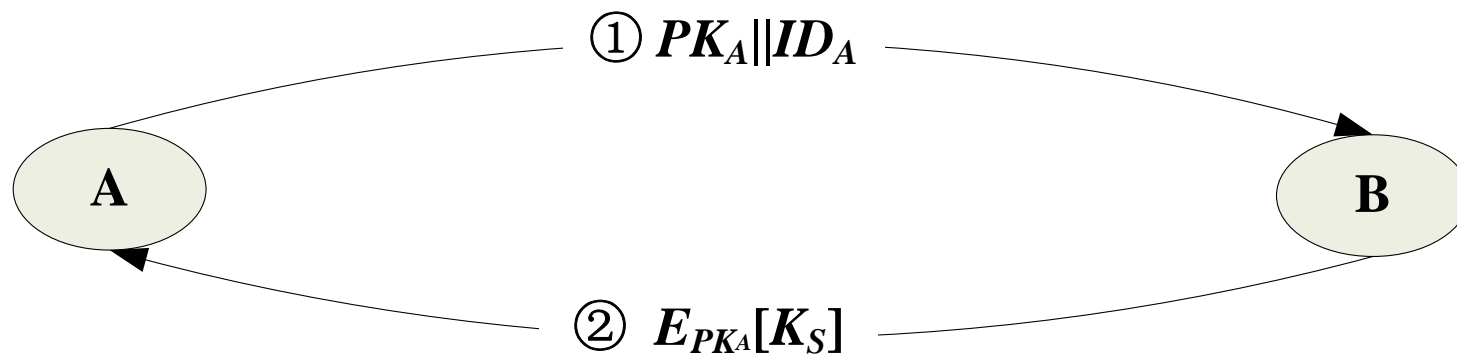
Needham—Schroeder改进协议 (2)

有效期内可不通过KDC直接认证



7.3.2 基于公钥加密的会话密钥分配协议

- 由于公钥加密的速度过慢，因此进行保密通信不太合适，但用于分配单钥密码体制的密钥却非常合适
- 1. 简单分配
 - 下图表示简单使用公钥加密算法建立会话密钥的过程，如果A希望与B通信，可通过以下步骤建立会话密钥



简单使用公钥加密算法建立会话密钥

7.3.2 基于公钥加密的会话密钥分配协议

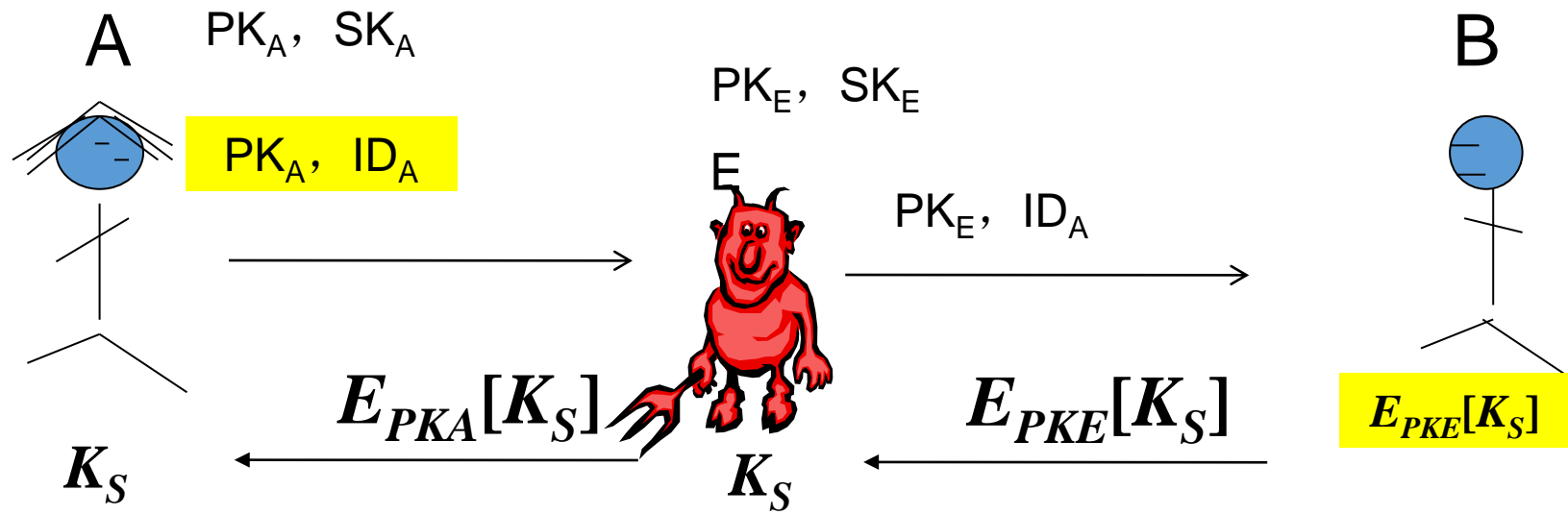
- ① A产生自己的一对密钥 $\{PK_A, SK_A\}$ ，并向B发送 $PK_A || ID_A$ ，其中 ID_A 表示A的身份
- ② B产生会话密钥 K_S ，并用A的公开钥 PK_A 对 K_S 加密后发往A
- ③ A由 $D_{SK_A}[E_{PK_A}[K_S]]$ 恢复会话密钥。因为只有A能解读 K_S ，所以仅A、B知道这一共享密钥。
- ④ A销毁 $\{PK_A, SK_A\}$ ，B销毁 PK_A 。
- A、B现在可以用单钥加密算法以 K_S 作为会话密钥进行保密通信，通信完成后，又都将 K_S 销毁
- 这种分配法尽管简单，但却由于A、B双方在通信前和完成通信后，都未存储密钥，因此，密钥泄露的危险性为最小，且可防止双方的通信被敌手监听，
 - 每次公私钥由发方临时产生
- 但由于公钥缺少证书管理机构认证且非物理传输容易受到主动攻击

7.3.2 基于公钥加密的会话密钥分配协议

• 中间人攻击

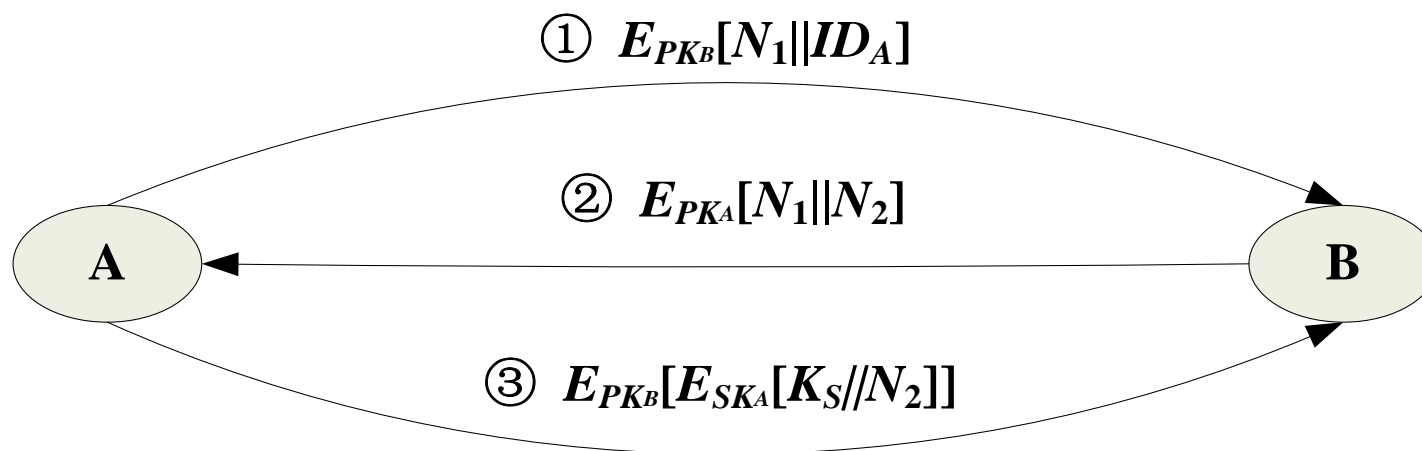
- 如果敌手E已接入A、B双方的通信信道，就可通过以下不被察觉的方式截获双方的通信：
 - ① 与上面的步骤①相同
 - ② E截获A的发送后，建立自己的一对密钥 $\{PK_E, SK_E\}$ ，并将 $PK_E \parallel ID_A$ 发送给B。
 - ③ B产生会话密钥 K_S 后，将 $E_{PKE}[K_S]$ 发送出去。
 - ④ E截获B发送的消息后，由 $D_{PKE}[E_{PKE}[K_S]]$ 解读 K_S 。
 - ⑤ E再将 $E_{PKA}[K_S]$ 发往A。
- 现在A和B知道 K_S ，但并未意识到 K_S 已被E截获。A、B在用 K_S 通信时，E就可以实施监听

中间人攻击



7.3.2 基于公钥加密的会话密钥分配协议

- 2. 具有保密性和认证性的密钥分配
 - 既可防止被动攻击，又可防止主动攻击
 - 假定A、B双方已完成公钥交换，可按以下步骤建立共享会话密钥：
 - ① A用 PK_B 的公开钥加密A的身份 ID_A 和一个一次性随机数 N_1 后发往B，其中 N_1 用于唯一地标识这一业务



具有保密性和认证性的密钥分配

7.3.2 基于公钥加密的会话密钥分配协议

- ② B用 PK_A 加密 N_1 和B新产生的一次性随机数 N_2 后发往A。B发来的消息中 N_1 的存在可使A相信对方的确是B
- ③ A选一会话密钥 K_S ，然后将 $M=E_{PKB}[E_{SKA}[K_S||N_2]]$ 发给B，其中用B的公开钥加密是为保证只有B能解读加密结果，用A的秘密钥加密是保证该加密结果只有A能发送
- ④ B以 $D_{PKA}[D_{SKB}[M]]$ 恢复会话密钥

7.3.2 基于公钥加密的会话密钥分配协议

- 以上是假设收发双方已经互相知道对方公钥)，以下协议也用于此目的(但事先不知道彼此公钥)

• ① $A \rightarrow AS: ID_A \parallel ID_B$ A发送证书请求

• ② $AS \rightarrow A: E_{SK_{AS}}[ID_A \parallel PK_A \parallel T] \parallel E_{SK_{AS}}[ID_B \parallel PK_B \parallel T]$

AS恢复双方证书

B的证书可省去

• ③ $A \rightarrow B: E_{SK_{AS}}[ID_A \parallel PK_A \parallel T] \parallel E_{SK_{AS}}[ID_B \parallel PK_B \parallel T] \parallel E_{PK_B}[E_{SK_A}[K_S \parallel T]]$

- 其中 SK_{AS} 、 SK_A 分别是AS和A的秘密钥， PK_A 、 PK_B 分别是A和B的公开钥，E为公钥加密算法，AS是认证服务器， K_S 是要协商的会话密钥

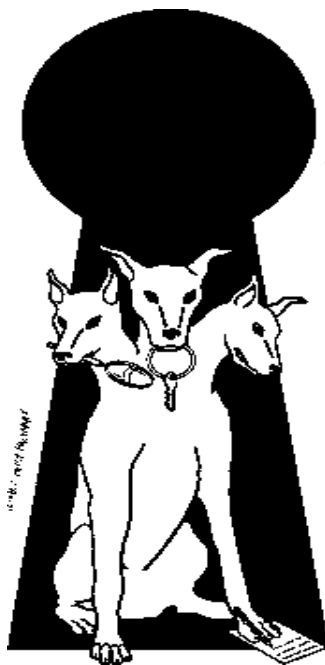
- 时戳T用以防止重放攻击，所以需要各方时钟同步

7.3.2 基于公钥加密的会话密钥分配协议

- 下一协议使用一次性随机数保护新鲜性，因此不需要时钟的同步：
 - ① $A \rightarrow KDC: ID_A \| ID_B$
 - ② $KDC \rightarrow A: E_{SK_{AU}}[ID_B \| PK_B]$
 - ③ $A \rightarrow B: E_{PK_B}[N_A \| ID_A]$
 - ④ $B \rightarrow KDC: ID_B \| ID_A \| E_{PK_{AU}}[N_A]$
 - ⑤ $KDC \rightarrow B: E_{SK_{AU}}[ID_A \| PK_A] \| E_{PK_B}[E_{SK_{AU}}[N_A \| K_S \| ID_B]]$
 - ⑥ $B \rightarrow A: E_{PK_A}[E_{SK_{AU}}[N_A \| K_S \| ID_B] \| N_B]$
 - ⑦ $A \rightarrow B: E_{K_S}[N_B]$
 - 其中 SK_{AU} 和 PK_{AU} 分别是 KDC 的秘密钥和公开钥
- 协议可进一步做如下改进：
在第⑤、⑥两步出现 N_A 的地方加上 ID_A ，以说明 N_A 的确是由 A 产生的而不是其他人产生的，这时 $\{ID_A, N_A\}$ 就可惟一地识别 A 发出的连接请求

A decorative blue horizontal bar with a series of parallel lines is positioned to the left of the title.

KERBEROS认证协议



- 在希腊神话中，有一个多头的狗，它有一条毒蛇的尾巴，它是地狱之门的守护者。
-

A decorative blue horizontal bar with white horizontal stripes is positioned on the left side of the slide.

Kerberos

- 是美国麻省理工学院（MIT）开发的一种身份鉴别服务。
 - “Kerberos”的本意是希腊神话中守护地狱之门的守护者。
 - Kerberos提供了一个集中式的认证服务器结构，认证服务器的功能是实现用户与其访问的服务器间的相互鉴别。
 - Kerberos建立的是一个实现身份认证的框架结构。
 - 其实现采用的是**对称密钥加密技术**，而未采用公开密钥加密。
 - 公开发布的Kerberos版本包括版本4和版本5。
-

A decorative blue horizontal bar with white horizontal stripes is positioned on the left side of the slide.

Kerberos设计目标

- 安全性
 - 能够有效防止攻击者假扮成另一个合法的授权用户。
 - 可靠性
 - 分布式服务器体系结构，提供相互备份。
 - 对用户透明性
 - 可伸缩
 - 能够支持大数量的客户和服务。
-

Kerberos设计思路

- 基本思路：
 - 使用一个（或一组）独立的认证服务器（AS — Authentication Server），来为网络中的用户（C）提供身份认证服务；
 - 认证服务器（AS），用户口令由 AS 保存在数据库中；
 - AS 与每个服务器（V）共享一个唯一保密密钥（ K_v ）（已被安全分发）。
- 会话过程：

(1) $C \rightarrow AS: ID_C \parallel P_C \parallel ID_v$

(2) $AS \rightarrow C: Ticket$

(3) $C \rightarrow V: ID_C \parallel Ticket$

■ 其中：

$Ticket = E_{K_v}[ID_C \parallel AD_C \parallel ID_v]$

Kerberos设计思路

- 会话过程:

(1) ID_C, P_C, ID_V

AS

认证服务器

搜索数据库看用户是否合法
如果合法，验证用户口令是否正确
如果口令正确，检查是否有权限访问服务器V

Ticket

C

$ID_C, Ticket$

V

应用服务器

用与AS共享密钥解密票据
检查票据中的用户标识与网络地址是否与用户发送的标识及其地址相同
如果相同，票据有效，认证通过

$Ticket = E_{K_v}[ID_C, AD_C, ID_v]$ 用户

ID_C : 用户C的标识

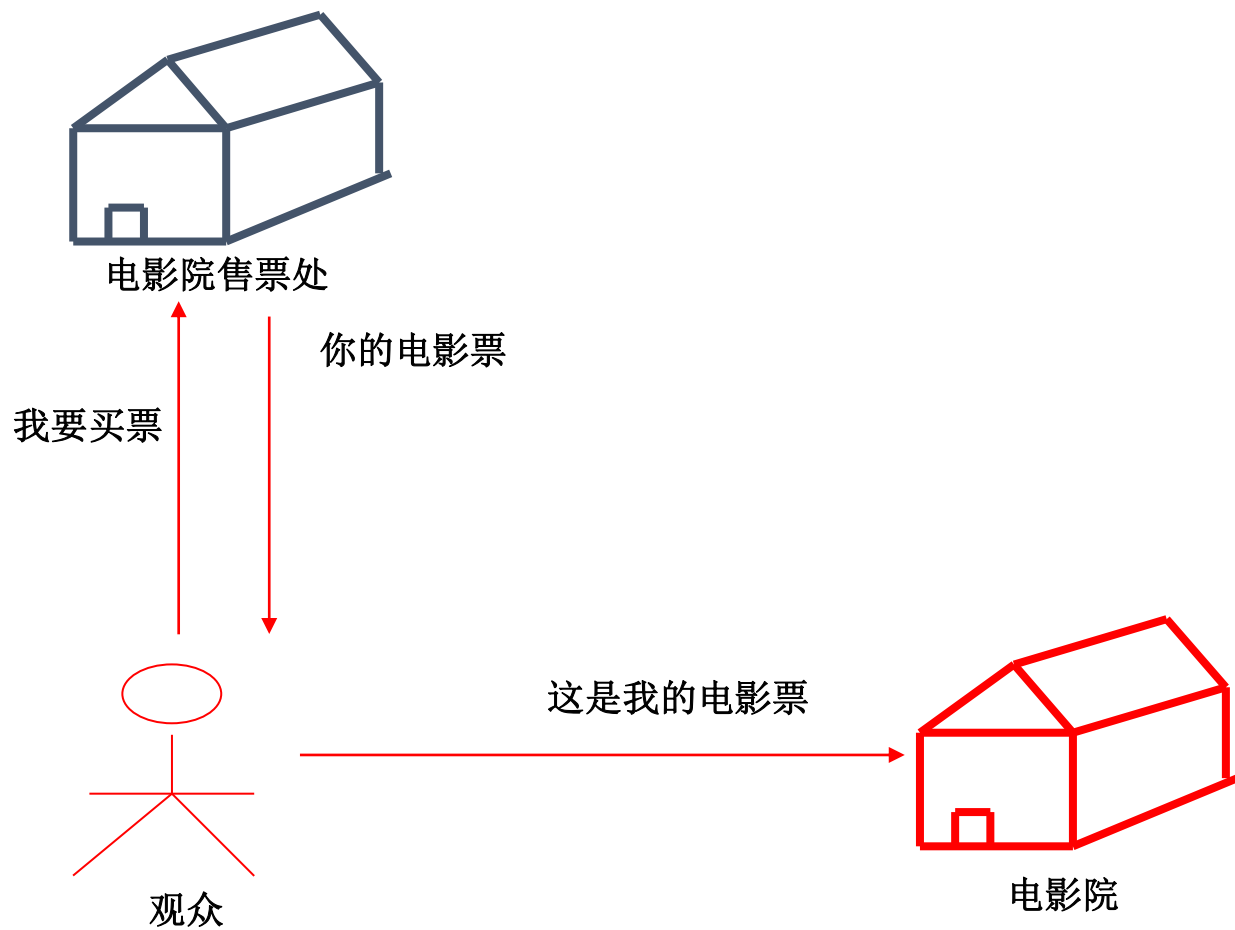
P_C : 用户口令

ID_v : 服务器标识

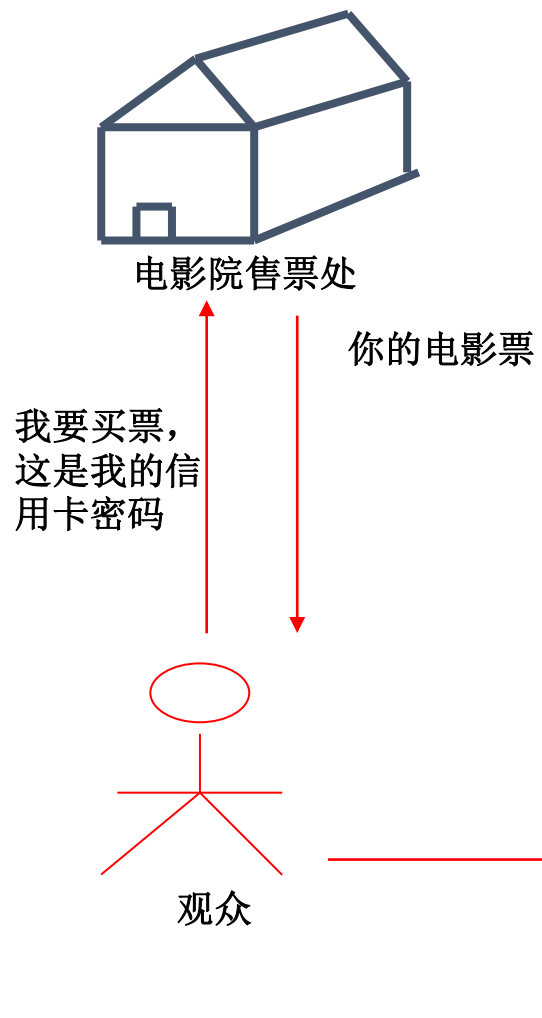
AD_C : 用户网络地址



Kerberos设计思路



Kerberos设计思路

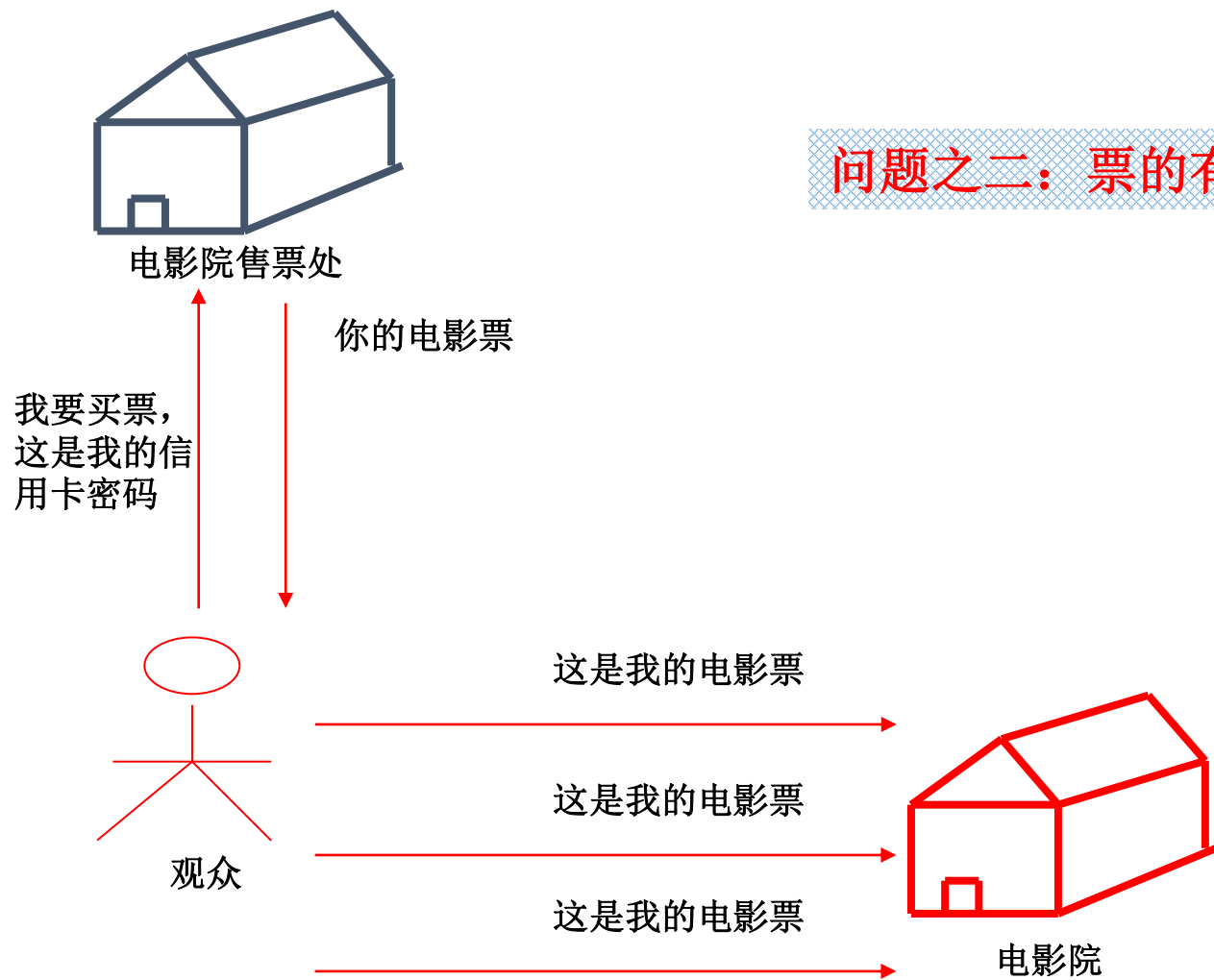


问题：如何买票

答案：出示信用卡卡号和密码

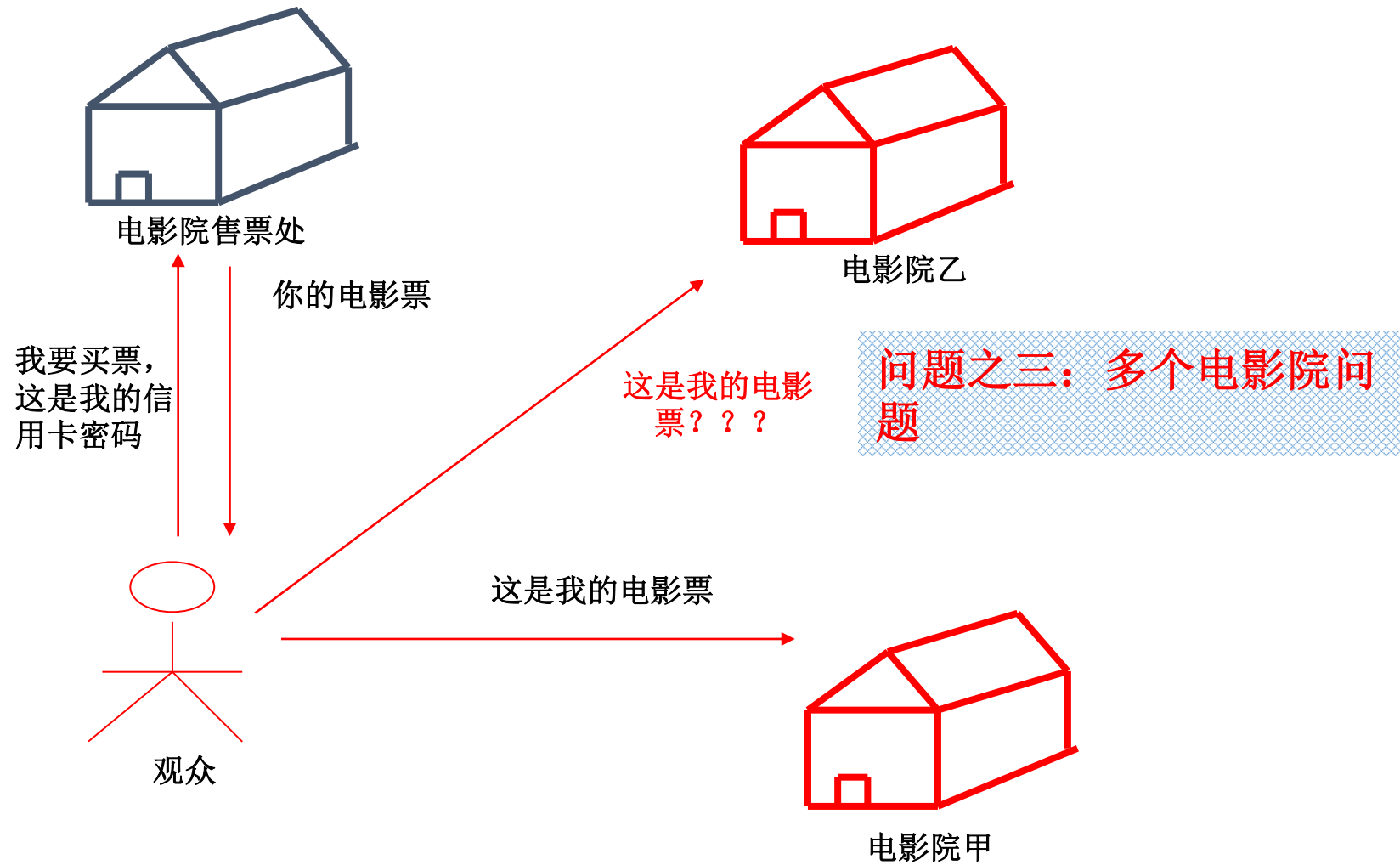
问题之一：信用卡问题

Kerberos设计思路



问题之二：票的有效性问题

Kerberos设计思路





Kerberos设计思路



上述协议问题？

上述协议的问题：

- (1) 口令明文传送
- (2) 票据的有效性（多次使用）
- (3) 访问多个服务器则需多次申请票据（即口令多次使用）

如何解决

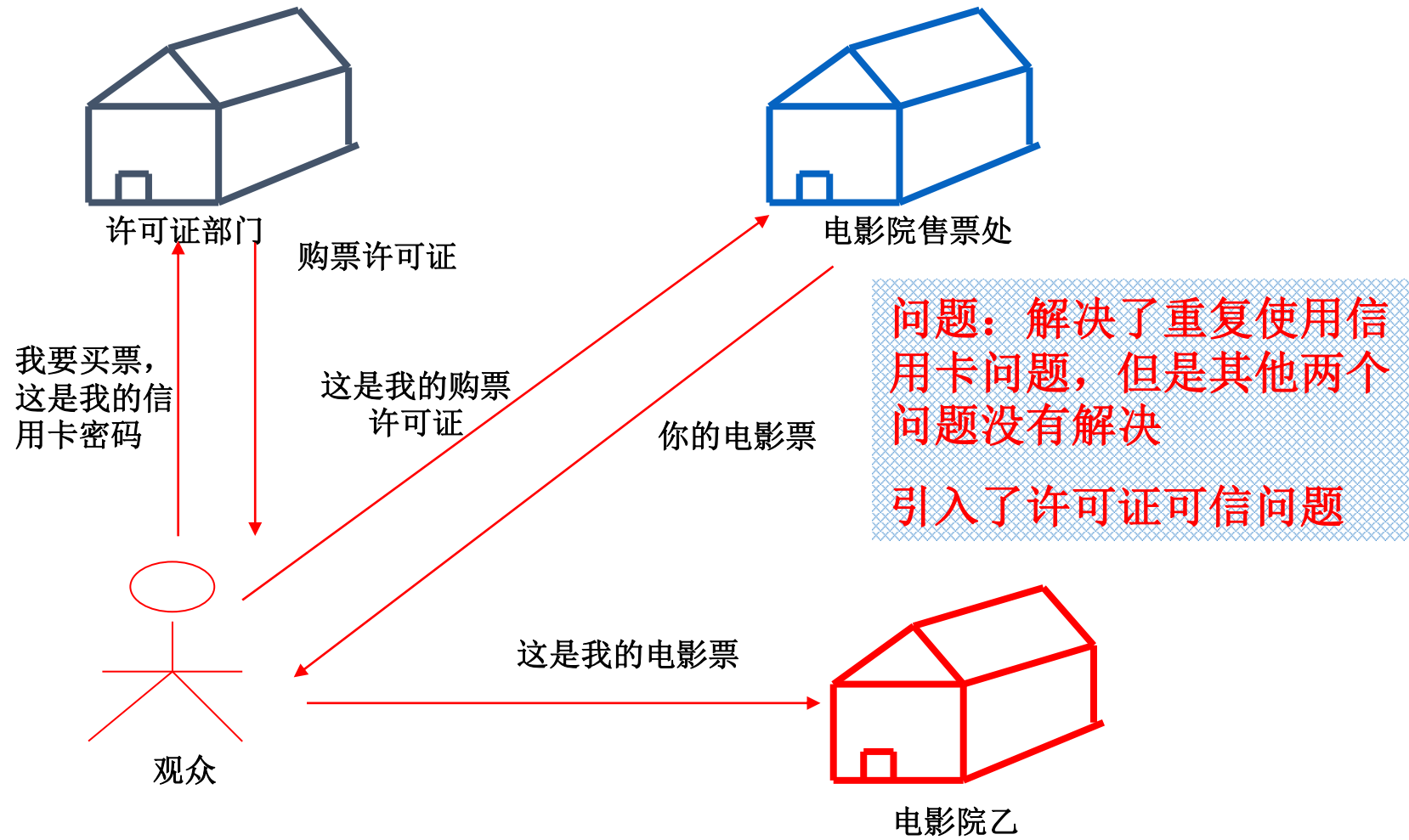


Kerberos设计思路



- 问题：
 - 用户希望输入口令的次数最少。
 - 口令以明文传送会被窃听。
 - 解决办法
 - 票据重用 (ticket reusable) 。
 - 引入票据许可服务器 (TGS - ticket-granting server)
 - 用于向用户分发服务器的访问票据；
 - 认证服务器 AS 并不直接向客户发放访问应用服务器的票据，而是由 TGS 服务器来向客户发放。
-

Kerberos设计思路



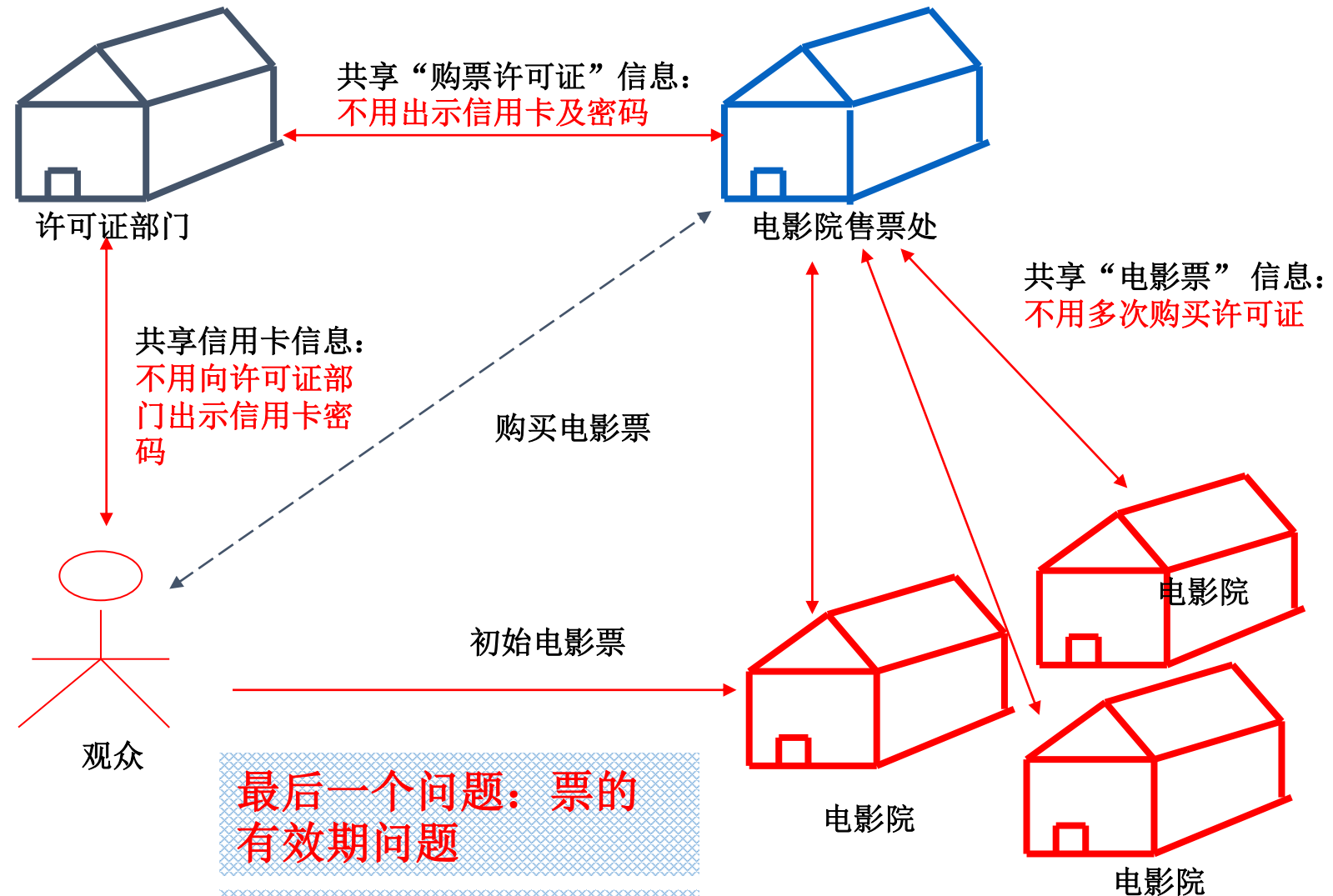


Kerberos设计思路



- 两种票据
 - 票据许可票据 (Ticket granting ticket)
 - 客户访问 TGS 服务器需要提供的票据，目的是为了申请某一个应用服务器的“服务许可票据”；
 - 票据许可票据由 AS 发放；
 - 用 $\text{Ticket}_{\text{tgs}}$ 表示访问 TGS 服务器的票据；
 - $\text{Ticket}_{\text{tgs}}$ 在用户登录时向 AS 申请一次，可多次重复使用；
 - 服务许可票据 (Service granting ticket)
 - 是客户时需要提供的票据；
 - 用 Ticket_V 表示访问应用服务器 V 的票据。

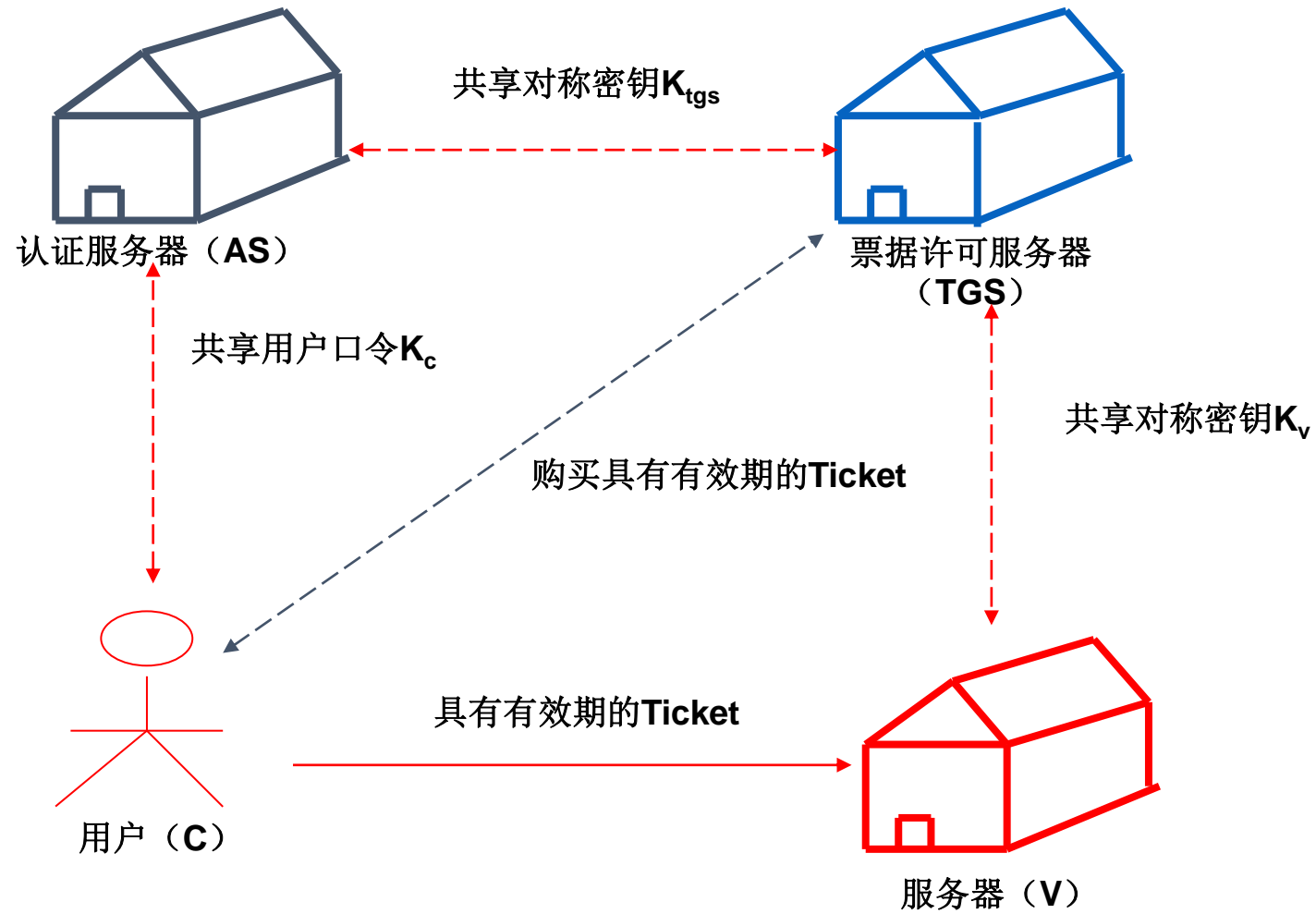
Kerberos设计思路



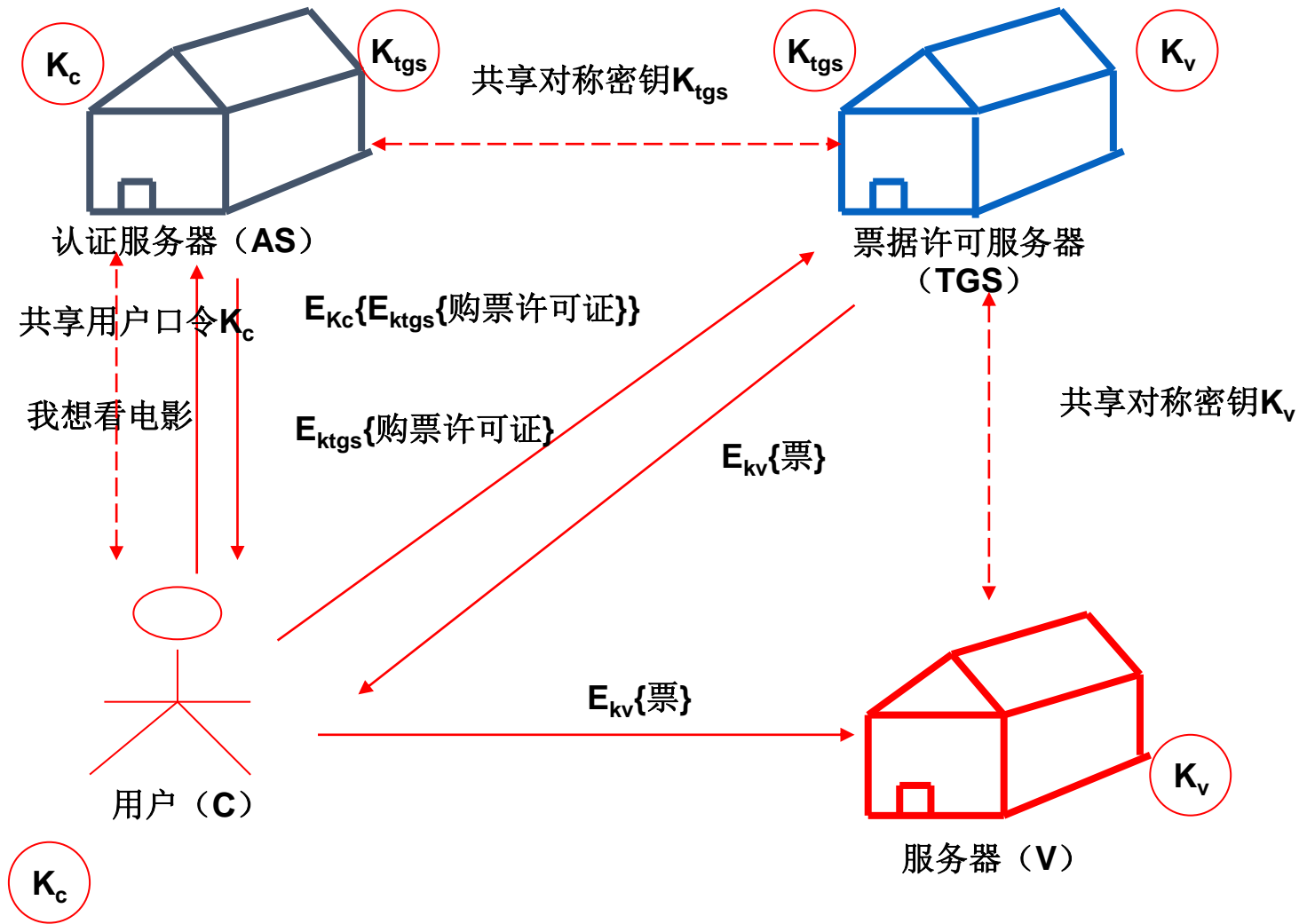
最后一个问题：票的有效期问题

解决方法：时间

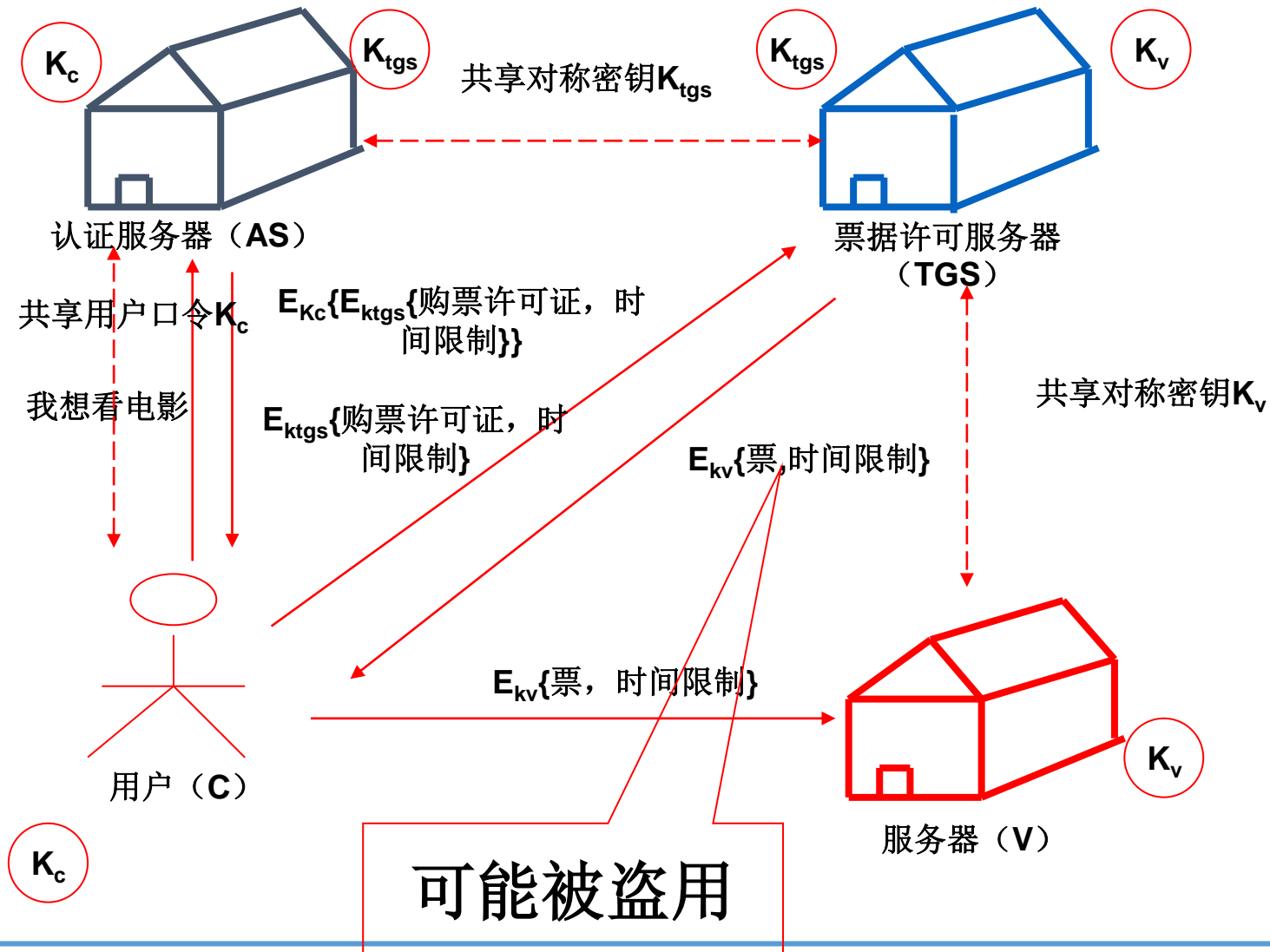
Kerberos设计思路



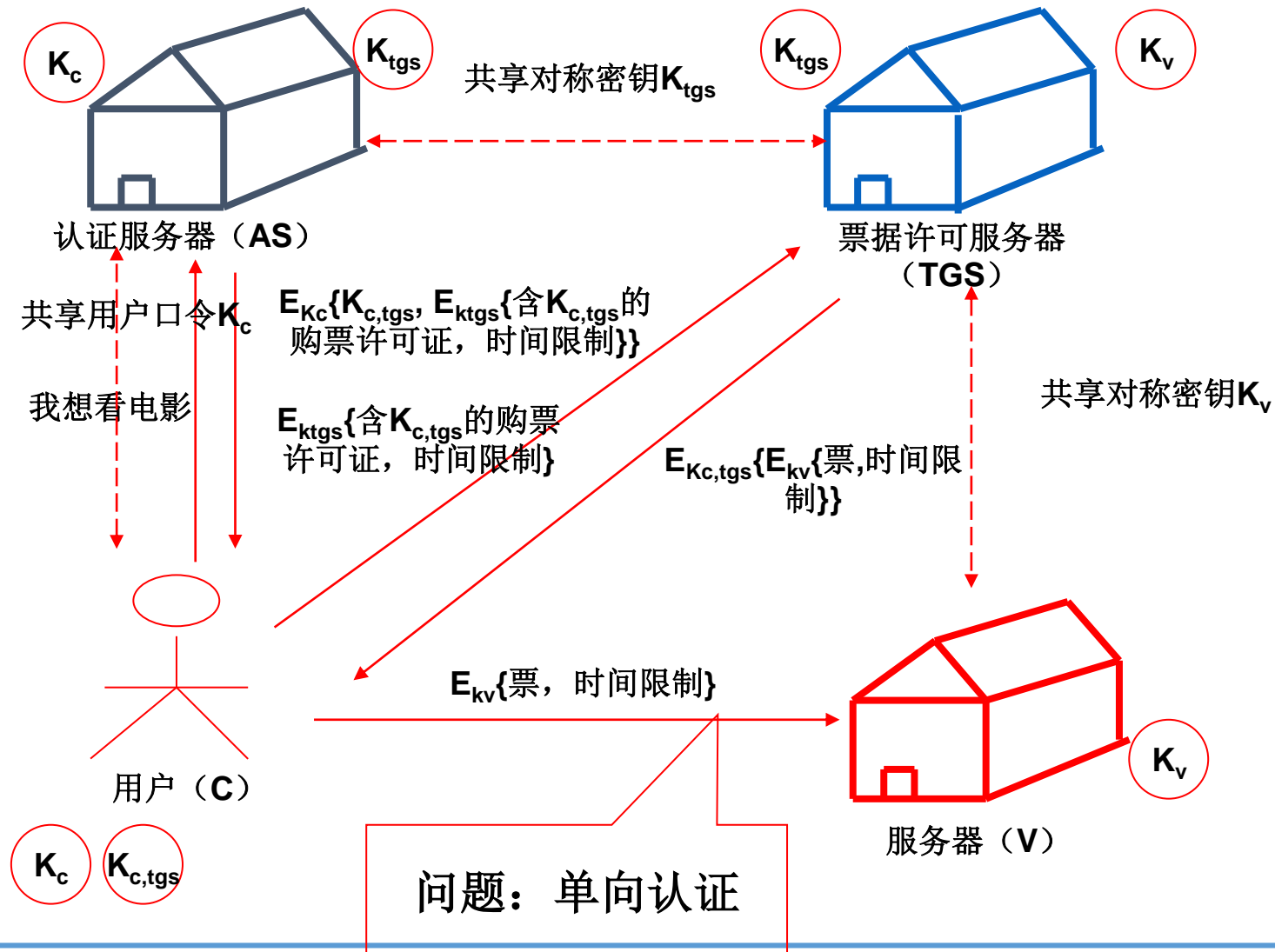
Kerberos设计思路



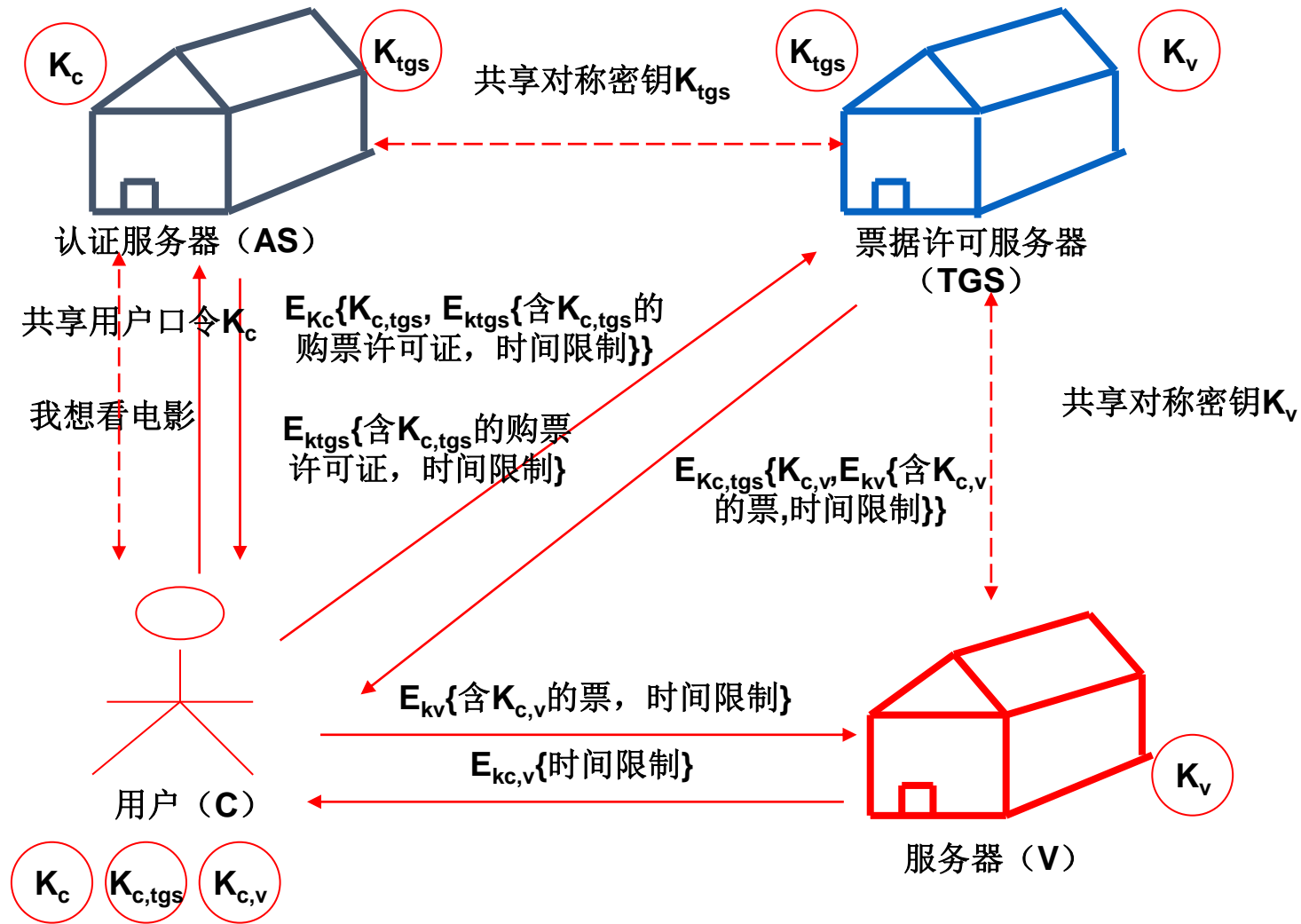
Kerberos设计思路



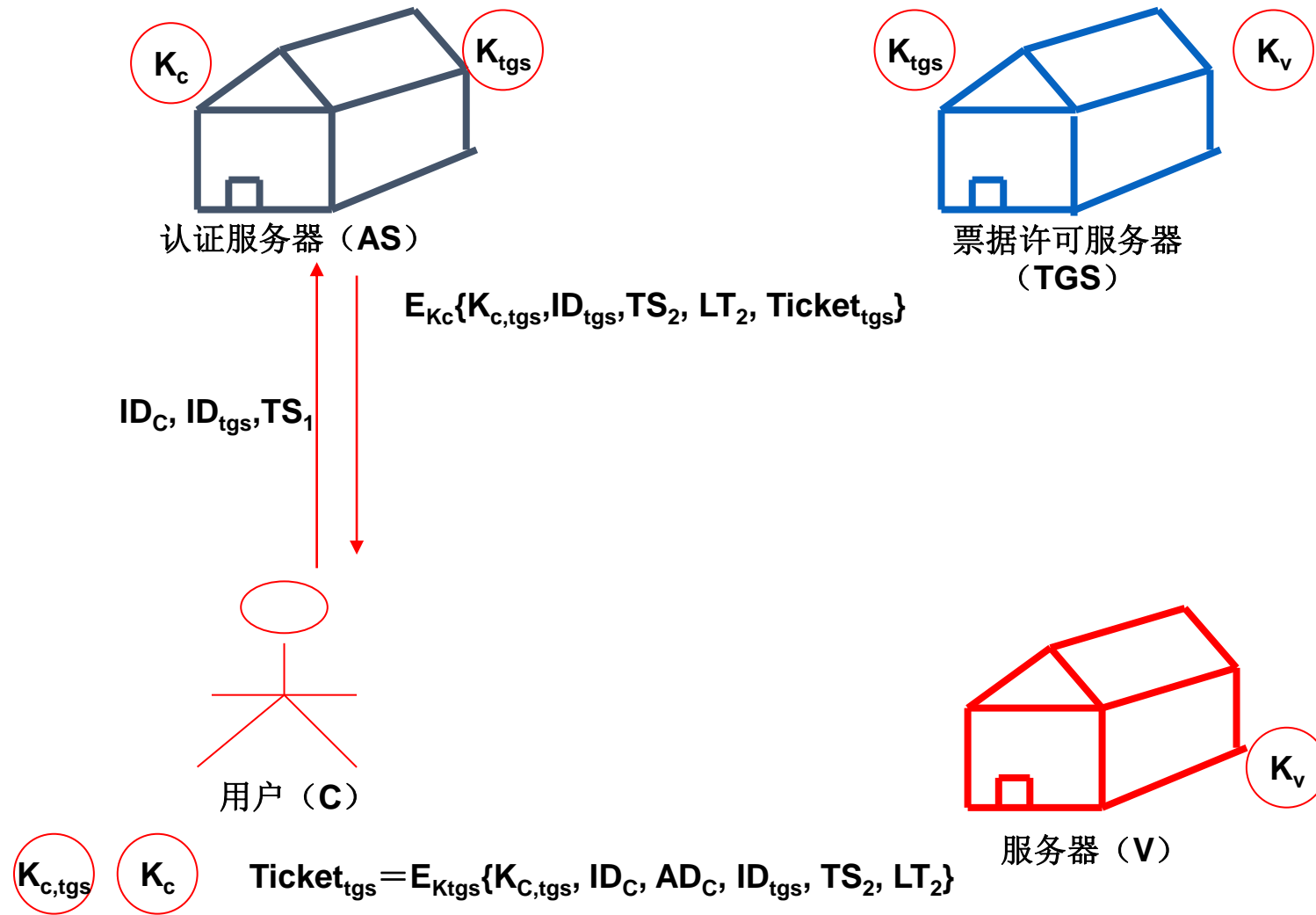
Kerberos设计思路



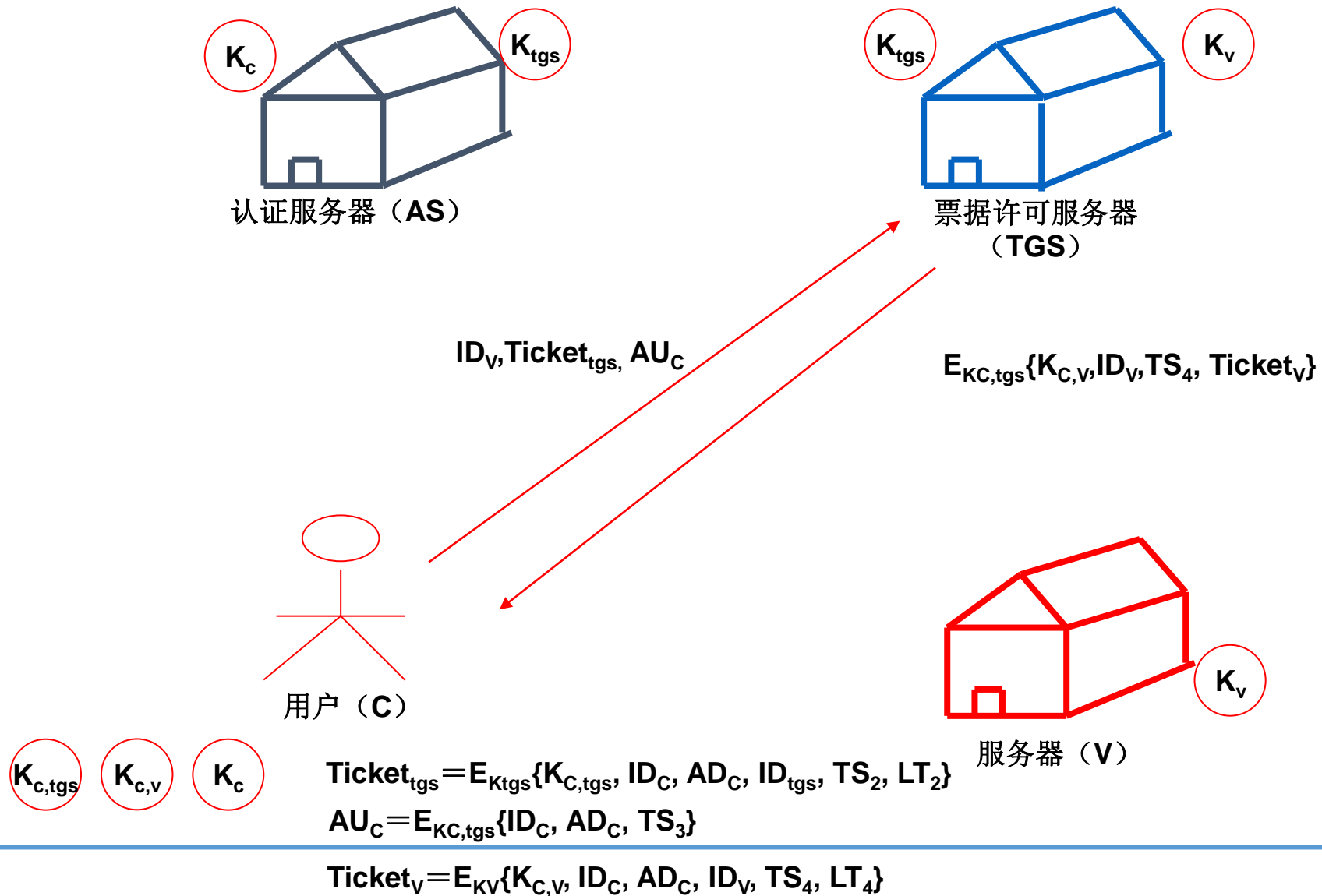
Kerberos设计思路



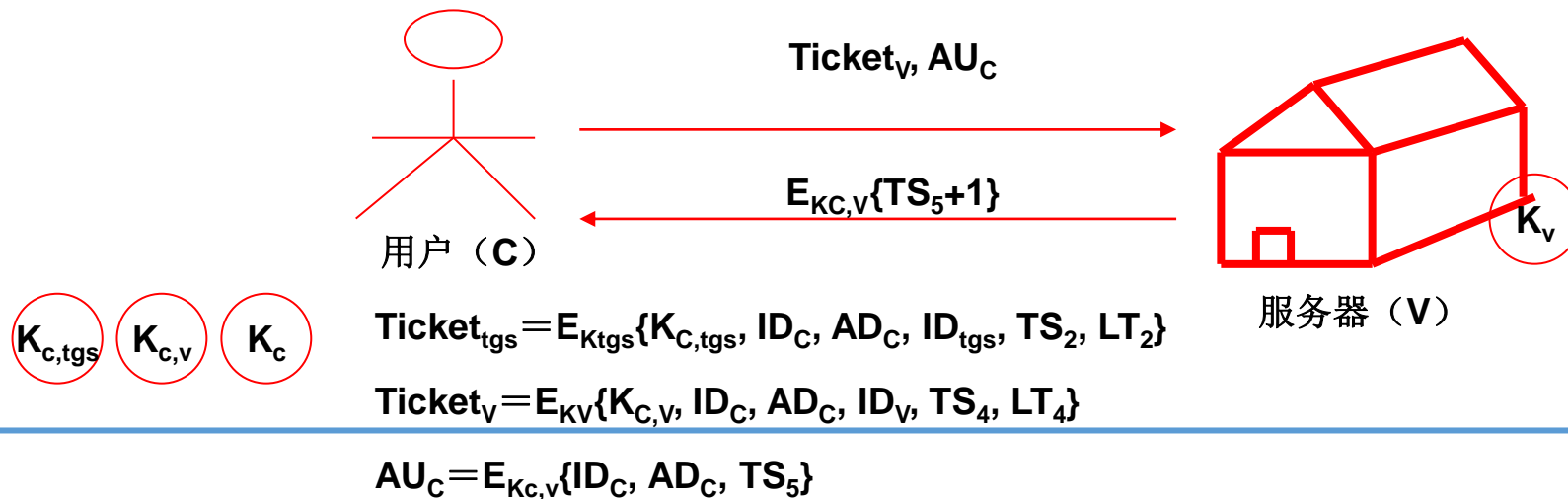
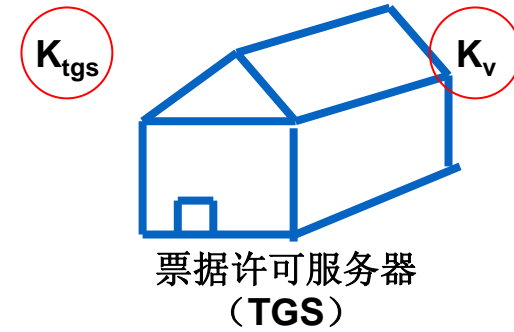
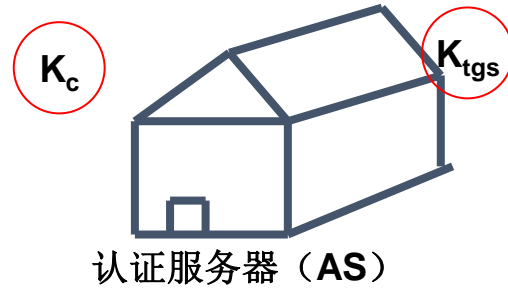
Kerberos设计思路



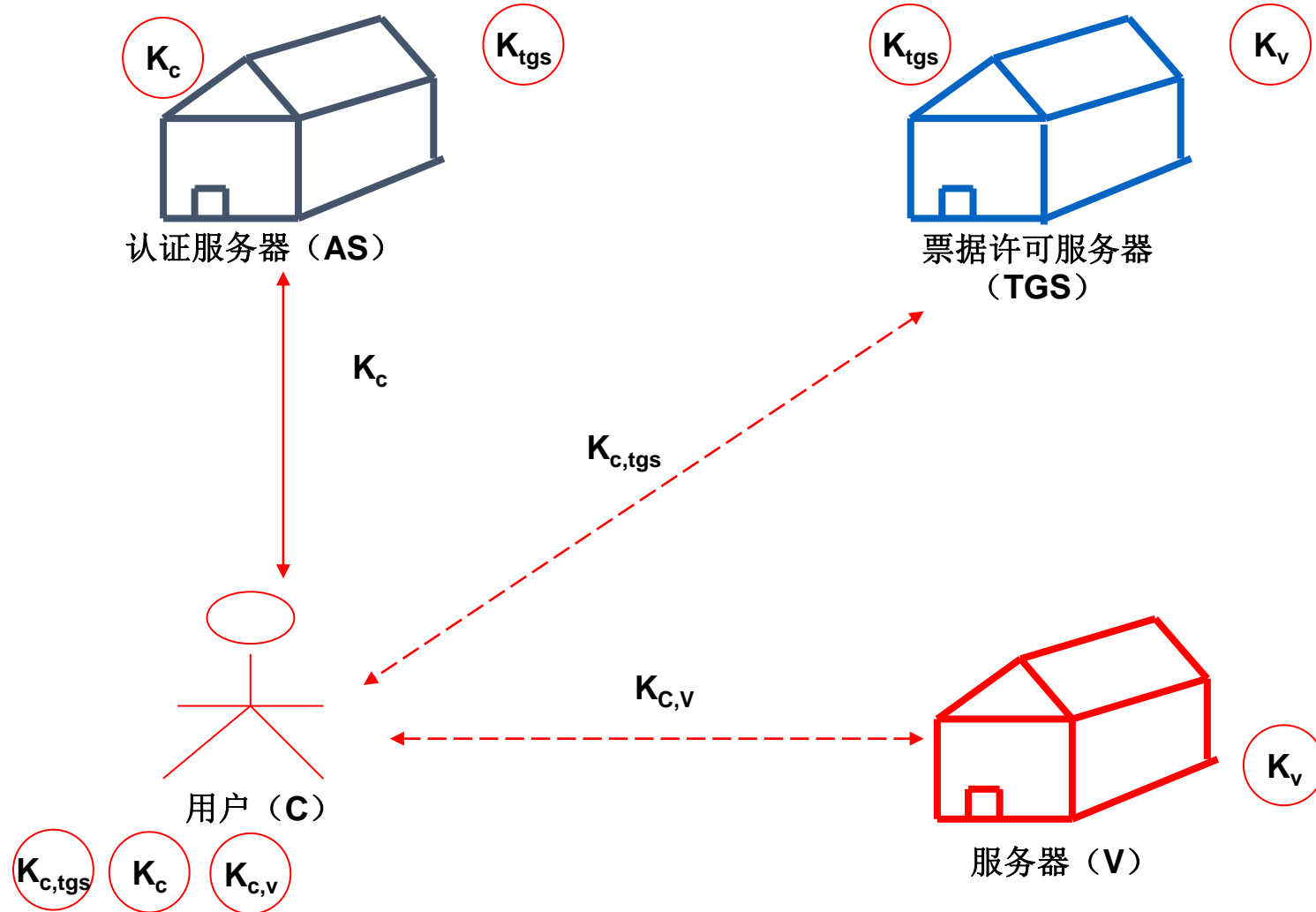
Kerberos设计思路



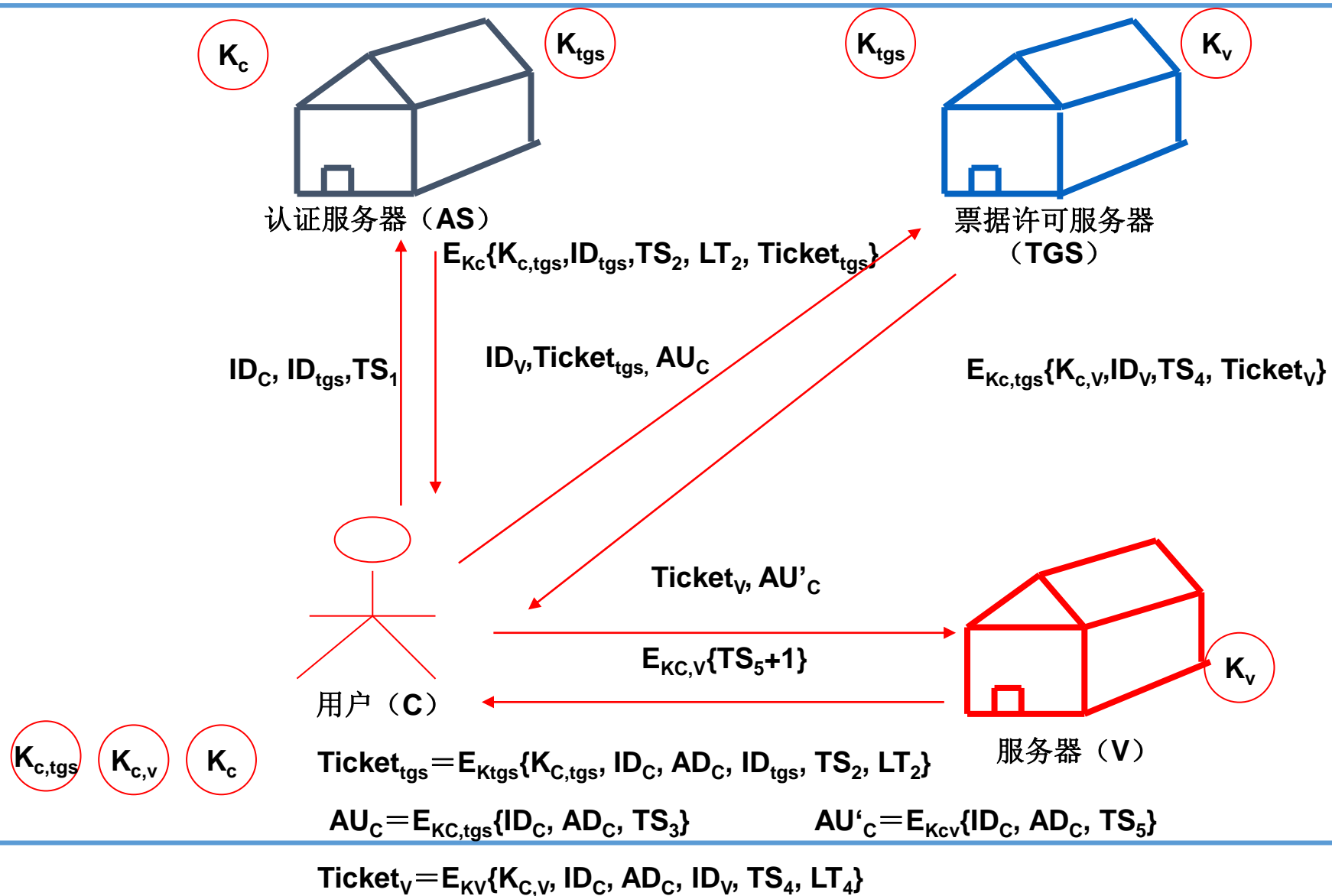
Kerberos设计思路



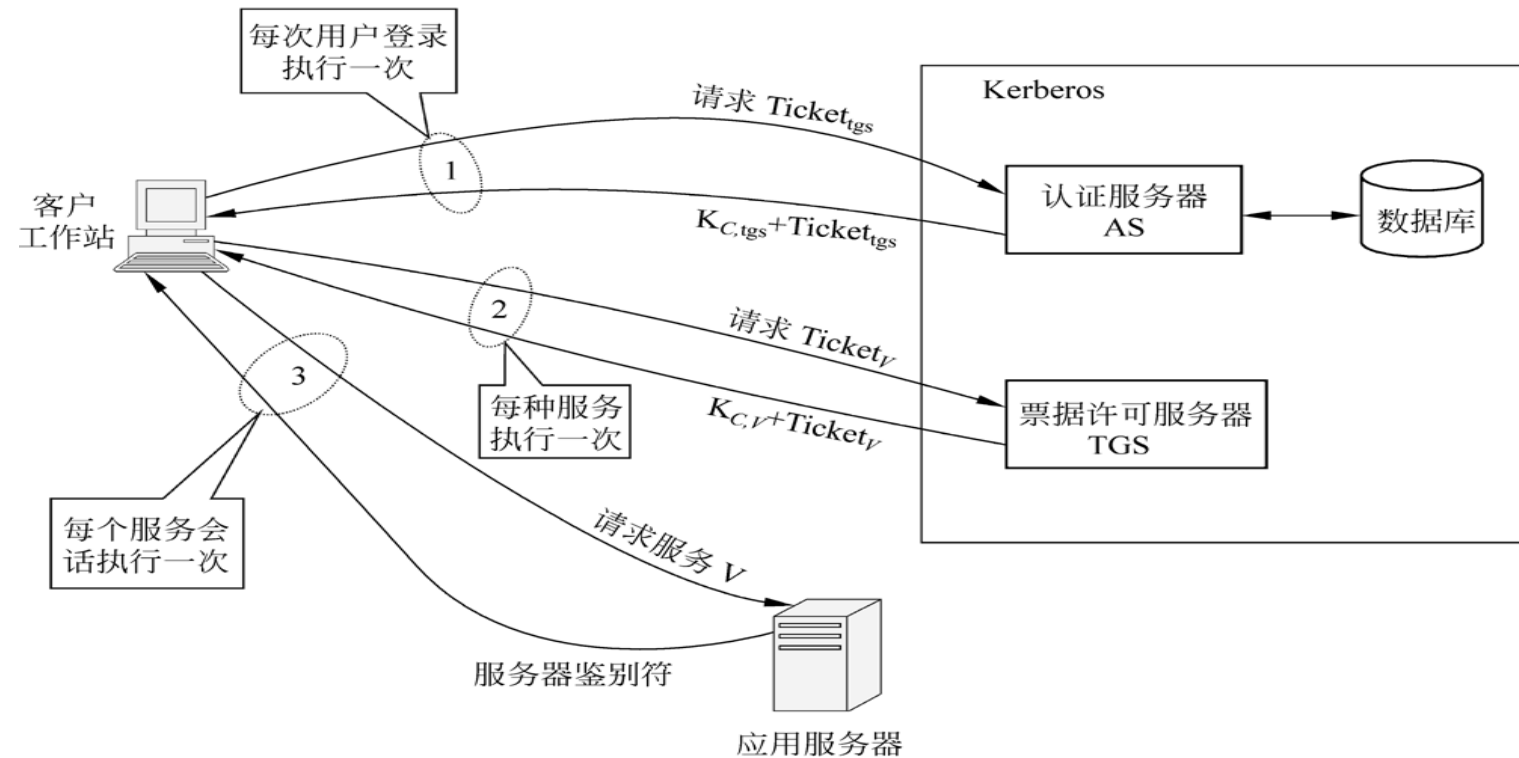
Kerberos V4协议描述：共享密钥及会话密钥




Kerberos设计思路




Kerberos (V4) 协议交互过程



A decorative blue horizontal bar with white horizontal stripes is positioned on the left side of the slide.

Kerberos (V4) 协议的缺陷

- 依赖性
 - 加密系统的依赖性 (DES)、对 IP 协议的依赖性和对时间依赖性。
 - 字节顺序：没有遵循标准
 - 票据有效期
 - 有效期最小为5分钟，最大约为21小时，往往不能满足要求
 - 认证转发能力
 - 不允许签发给一个用户的鉴别证书转发给其他工作站或其他客户使用
 - 领域间的鉴别
 - 管理起来困难
 - 加密操作缺陷
 - 非标准形式的 DES 加密（传播密码分组链接 PCBC）方式，易受攻击
 - 会话密钥
 - 存在着攻击者重放会话报文进行攻击的可能
 - 口令攻击
 - 未对口令提供额外的保护，攻击者有机会进行口令攻击
-

A decorative blue horizontal bar with white horizontal stripes is positioned on the left side of the slide.

Kerberos (V5) 协议的改进

- 加密系统
 - 支持使用任何加密技术。
 - 通信协议
 - IP 协议外，还提供了对其他协议的支持。
 - 报文字节顺序
 - 采用抽象语法表示 (ASN.1) 和基本编码规则 (BER) 来进行规范。
 - 票据的有效期
 - 允许任意大小的有效期，有效期定义为一个开始时间和结束时间。
 - 鉴别转发能力
 - 更有效的方法来解决领域间的认证问题
 - 口令攻击
 - 提供了一种预鉴别 (preauthentication) 机制，使口令攻击更加困难。
-

A decorative blue horizontal bar with white horizontal stripes is positioned to the left of the title.

Kerberos 领域(realm)

- 构成：一个完整的 Kerberos 环境包括一个 Kerberos 服务器，一组工作站和一组应用服务器。
 - Kerberos 服务器数据库中拥有所有参与用户的 UID 和口令散列表。
 - Kerberos服务器必须与每一个服务器之间共享一个保密密钥。
 - 所有用户均在 Kerberos 服务器上注册。
 - 所有服务器均在 Kerberos 服务器上注册。
 - 领域的划分是根据网络的管理边界来划定的。
-

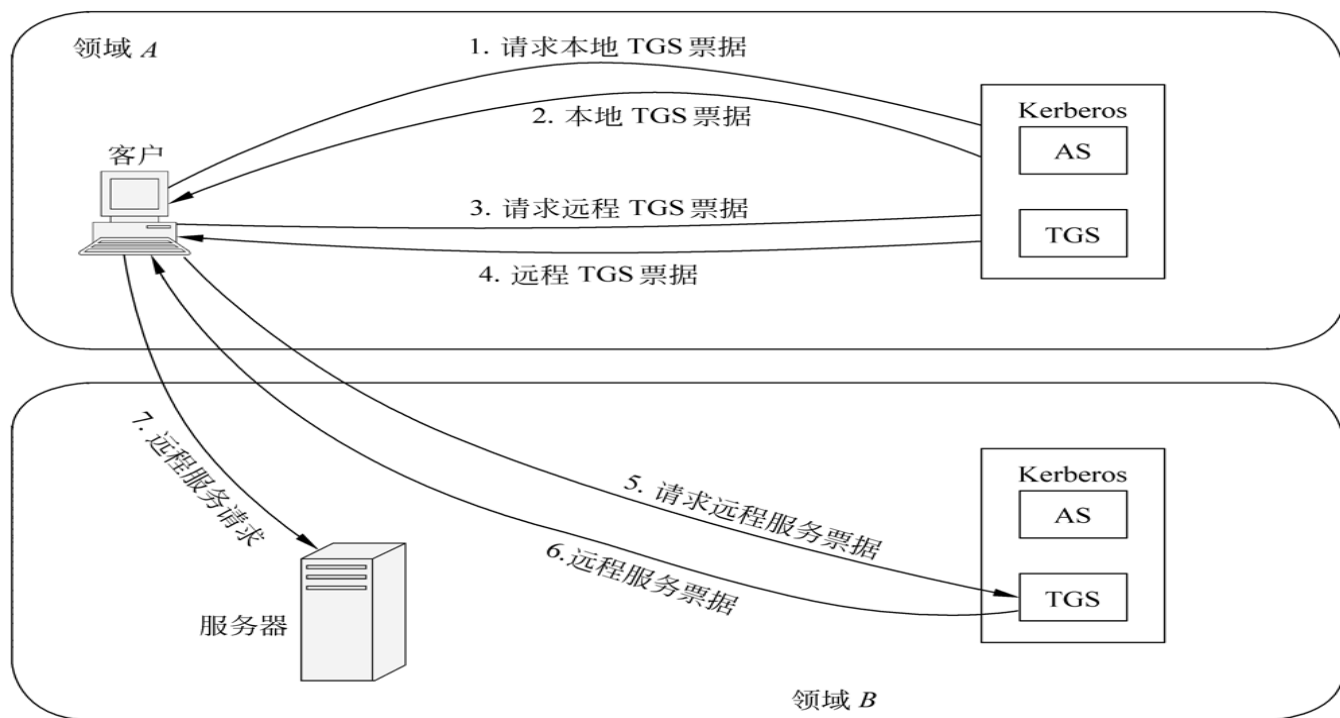
A decorative blue horizontal bar with white horizontal stripes is positioned on the left side of the slide.

Kerberos 领域间的互通

- 跨领域的服务访问
 - 一个用户可能需要访问另一个 Kerberos 领域中应用服务器;
 - 一个应用服务器也可以向其他领域中的客户提供网络服务。
 - 领域间互通的前提
 - 支持不同领域之间进行用户身份鉴别的机制;
 - 互通领域中的 Kerberos 服务器之间必须共享一个密钥;
 - 同时两个 Kerberos 服务器也必须进行相互注册。
-

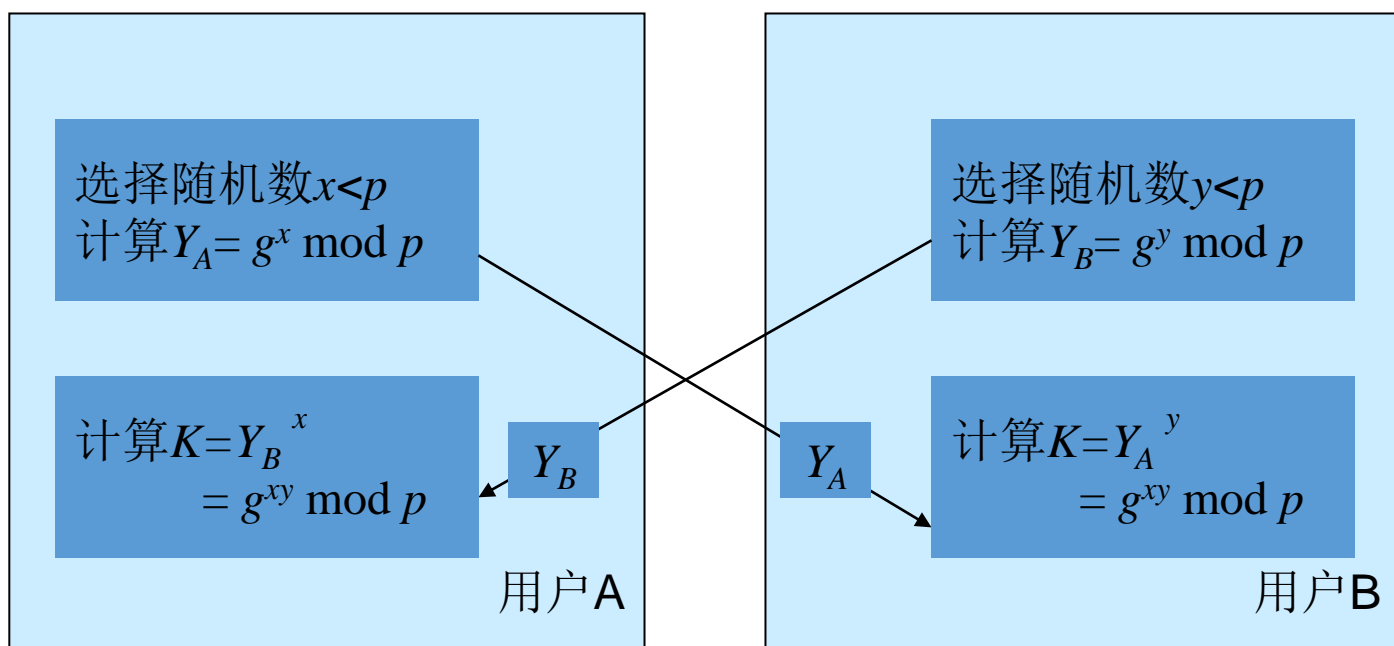


远程服务访问的认证过程

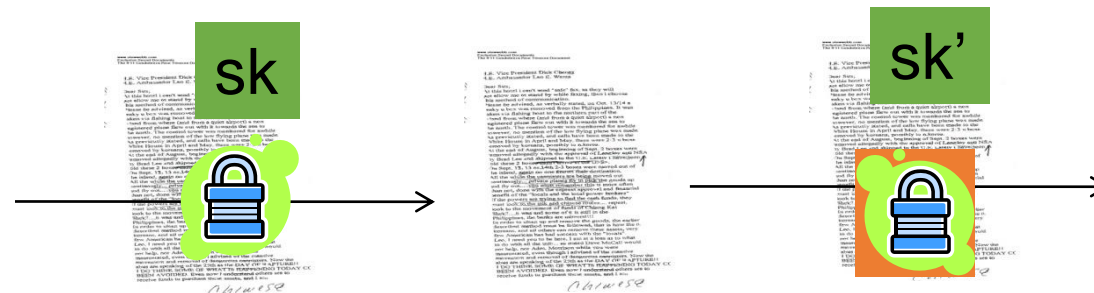
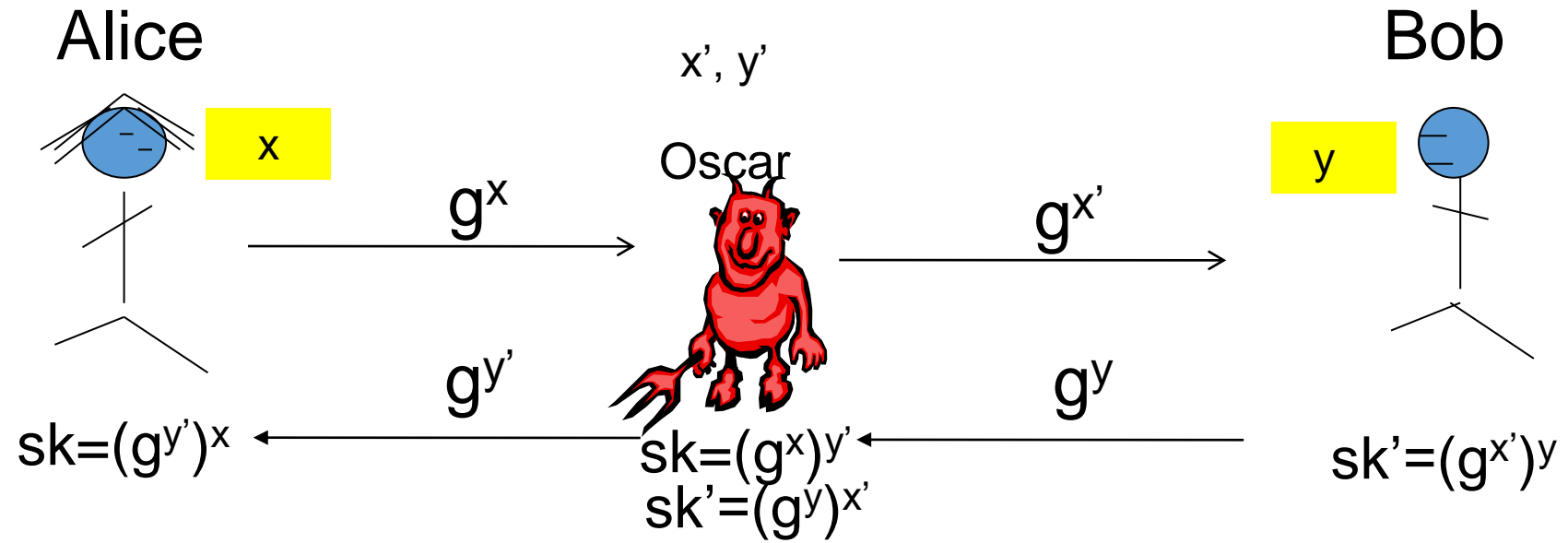


Diffie-Hellman 密钥交换

- W.Diffie和M.Hellman1976年提出
- 算法的安全性基于求离散对数的困难性

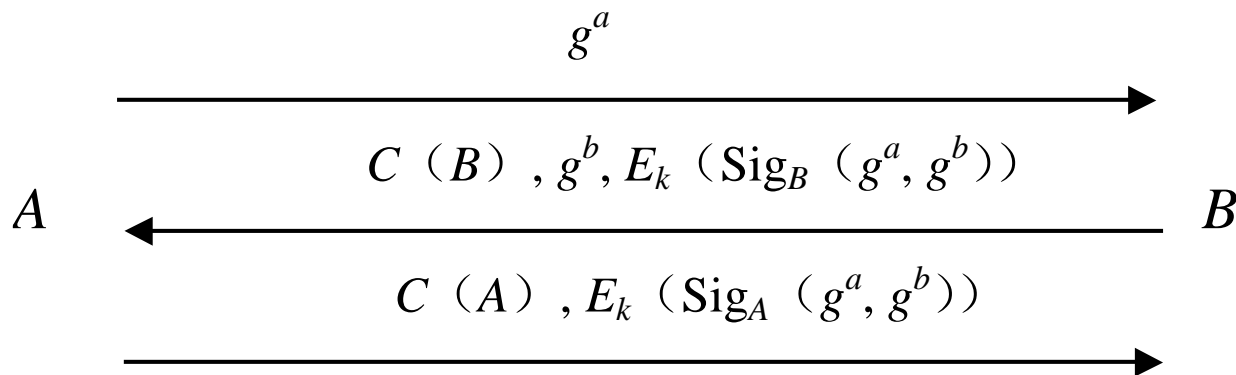


中间人攻击



端到端协议

- 1992年, Diffie、Oorschot和Wiener提出了一个端到端协议 (station-to-station protocol)



三方Diffie-Hellman密钥交换

选取一个大素数 p , $g \in Z_p$ 是一个本原元, 用户 A, B, C 执行以下步骤:

- (1) 用户 A 随机选取 $a(2 \leq a \leq p-2)$, 计算 $y_A \equiv g^a \bmod p$ 发送给 B;
- (2) 用户 B 收到 y_A 后, 随机选取 $b(2 \leq b \leq p-2)$, 计算 $y_B \equiv g^b \bmod p$ 和 $y_{AB} \equiv g^{ab} \bmod p$ 发送给 C;
- (3) 用户 C 收到 y_{AB} 和 y_B 后, 随机选取 $c(2 \leq c \leq p-2)$, 计算 $y_C \equiv g^c \bmod p$ 和 $y_{BC} \equiv g^{bc} \bmod p$ 发送给 A;
- (4) 用户 A 收到 y_{BC} 和 y_C 后, 计算 $y_{AC} \equiv g^{ac} \bmod p$ 发送给 B;
- (5) 用户 A 计算 $k = (y_{BC})^a \equiv g^{abc} \bmod q$;
- (6) 用户 B 计算 $k = (y_{AC})^b \equiv g^{abc} \bmod q$;
- (7) 用户 C 计算 $k = (y_{AB})^c \equiv g^{abc} \bmod q$ 。

至此, 用户 A, B, C 共享了密钥 k 。

秘密共享

Secrete Sharing

A decorative blue horizontal bar with white horizontal stripes is positioned to the left of the section header.

问题1

- 保险柜的开启
 - 保险柜中存放有7个人的共有财产
 - 要从保险柜中取出物品，必须有半数以上的人在场才可取出，半数一下则不行
 - 如何构造锁的设计方案？
-

A decorative blue horizontal bar with white horizontal stripes is positioned to the left of the section header.

问题2

- 导弹控制发射，重要场所通行检验，通常需要多人同时参与才能生效，需要将秘密分为多人掌管，并且由一定掌管秘密的人数同时到场才能恢复秘密。
-

门限方案的一般概念

- 秘密 s 被分为 n 个部分, 每个部分称为shadow, 由一个参与者持有, 使得
- 由 k 个或多于 k 个参与者所持有的部分信息可重构 s 。
- 由少于 k 个参与者所持有的部分信息则无法重构 s 。
- 称为 (k, n) 秘密分割门限方案, k 称为 门限值。
- 少于 k 个参与者所持有的部分信息得不到 s 的任何信息称该门限方案是完善的。



Shamir门限方案

Shamir门限方案是基于多项式的Lagrange插值公式的。插值是古典数值分析中的一个基本问题，问题如下：

A decorative blue horizontal bar with white horizontal stripes is positioned to the left of the title.

Shamir门限方案

Shamir门限方案是基于多项式的Lagrange插值公式的。插值是古典数值分析中的一个基本问题，问题如下：

已知一个函数 $\varphi(x)$ 在 k 个互不相同的点的函数值 $\varphi(x_i), i = 1, \dots, k$ 寻求一个满足 $f(x_i) = \varphi(x_i), i = 1, \dots, k$ 的函数 $f(x)$ ，用来逼近 $\varphi(x)$ 。

Shamir门限方案

Shamir门限方案是基于多项式的Lagrange插值公式的。插值是古典数值分析中的一个基本问题，问题如下：

已知一个函数 $\varphi(x)$ 在 k 个互不相同的点的函数值 $\varphi(x_i), i = 1, \dots, k$ 寻求一个满足 $f(x_i) = \varphi(x_i), i = 1, \dots, k$ 的函数 $f(x)$ ，用来逼近 $\varphi(x)$ 。

$f(x)$ 称为 $\varphi(x)$ 的插值函数， $f(x)$ 可取自不同的函数类，既可为代数多项式，也可可为三角多项式或有理分式。

若取 $f(x)$ 为代数多项式，则称插值问题为代数插值， $f(x)$ 称为 $\varphi(x)$ 的插值多项式。

常用的代数插值有Lagrange插值、Newton插值、Hermite 插值。

Shamir门限方案

- 基于多项式Lagrange插值公式

设 $\{(x_1, y_1), \dots, (x_k, y_k)\}$ 是平面上 k 个点构成的点集，其中 $x_i (i=1, \dots, k)$ 各不相同，那么在平面上存在**唯一**的 $k-1$ 次多项式 $f(x)$ 通过这 k 个点。若把秘密 s 取做 $f(0)$ ， n 个shadow取做 $f(x_i) (i=1, \dots, n)$ ，那么利用其中任意 k 个shadow可以重构 $f(x)$ ，从而可以得到秘密 s 。

上述问题也可认为是已知 $k-1$ 次多项式 $f(x)$ 的 k 个互不相同的点的函数值 $f(x_i) (i=1, \dots, k)$ ，构造多项式 $f(x)$ 。

Shamir门限方案

- 有限域 $GF(q)$, q 为大素数, $q \geq n+1$ 。秘密 s 是 $GF(q) \setminus \{0\}$ 上均匀选取的随机数, 表示为 $s \in_R GF(q) \setminus \{0\}$. $k-1$ 个系数 a_1, a_2, \dots, a_{k-1} 选取 $a_i \in_R GF(q) \setminus \{0\}$. 在 $GF(q)$ 上构造一个 $k-1$ 次多项式 $f(x) = a_0 + a_1x + \dots + a_{k-1}x^{k-1}$
- N 个参与者 P_1, \dots, P_n , P_i 的Shadow为 $f(i)$ 。任意 k 个参与者得到秘密, 可使用 $\{(i_1, f(i_1)) \mid 1=1, \dots, k\}$ 构造方程组

$$\begin{cases} a_0 + a_1(i_1) + \dots + a_{k-1}(i_1)^{k-1} = f(i_1) \\ \vdots \\ a_0 + a_1(i_k) + \dots + a_{k-1}(i_k)^{k-1} = f(i_k) \end{cases}$$

Shamir门限方案

- 由Lagrange插值公式

$$f(x) = \sum_{j=1}^k f(i_j) \prod_{\substack{l=1 \\ l \neq j}}^k \frac{(x - i_l)}{i_j - i_l} \pmod{q}$$

$$s = (-1)^{k-1} \sum_{j=1}^k f(i_j) \prod_{\substack{l=1 \\ l \neq j}}^k \frac{i_l}{i_j - i_l} \pmod{q}$$

A decorative blue horizontal bar with white horizontal stripes is positioned to the left of the title.

Shamir门限方案

- 如果 $k-1$ 个参与者想获得 s , 可构造 $k-1$ 个方程, 有 k 个未知量。对任一 s_0 , 设 $f(0) = s_0$. 这样可以得到第 k 个方程, 得到 $f(x)$ 。对每个 s_0 都有唯一的多项式满足, 所有由 $k-1$ 个shadow得不到任何 s 的信息。因此此方案是完善的。



【例5-4】

设 $k = 3, n = 5, q = 19, s = 11$

随机选取 $a_1 = 2, a_2 = 7$ ，得多项式为

$$f(x) = (7x^2 + 2x + 11) \bmod 19$$

分别计算

$$f(1) = (7 + 2 + 11) \bmod 19 = 20 \bmod 19 = 1$$

$$f(2) = (28 + 4 + 11) \bmod 19 = 43 \bmod 19 = 5$$

$$f(3) = (63 + 6 + 11) \bmod 19 = 80 \bmod 19 = 4$$

$$f(4) = (112 + 8 + 11) \bmod 19 = 131 \bmod 19 = 17$$

$$f(5) = (175 + 10 + 11) \bmod 19 = 196 \bmod 19 = 6$$

得5个子密钥。





如果我们知道其中的3个子密钥 $f(2)=5, f(3)=4, f(5)=6$,
就可按以下方式重构 $f(x)$:

$$\begin{aligned} 5 \frac{(x-3)(x-5)}{(2-3)(2-5)} &= 5 \frac{(x-3)(x-5)}{(-1)(-3)} = 5 \frac{(x-3)(x-5)}{3} = 5 \cdot (3^{-1} \bmod 19) \cdot (x-3)(x-5) \\ &= 5 \cdot 13 \cdot (x-3)(x-5) = 65(x-3)(x-5) \end{aligned}$$

$$\begin{aligned} 4 \frac{(x-2)(x-5)}{(3-2)(3-5)} &= 4 \frac{(x-2)(x-5)}{(1)(-2)} = 4 \frac{(x-2)(x-5)}{-2} = 4 \cdot ((-2)^{-1} \bmod 19) \cdot (x-2)(x-5) \\ &= 4 \cdot 9 \cdot (x-2)(x-5) = 36(x-2)(x-5) \end{aligned}$$

$$\begin{aligned} 6 \frac{(x-2)(x-3)}{(5-2)(5-3)} &= 6 \frac{(x-2)(x-3)}{(3)(2)} = 6 \frac{(x-2)(x-3)}{6} = 6 \cdot (6^{-1} \bmod 19) \cdot (x-2)(x-3) \\ &= 6 \cdot 16 \cdot (x-2)(x-3) = 96(x-2)(x-3) \end{aligned}$$





所以

$$\begin{aligned} f(x) &= [65(x-3)(x-5) + 36(x-2)(x-5) + 96(x-2)(x-3)] \bmod 19 \\ &= [8(x-3)(x-5) + 17(x-2)(x-5) + (x-2)(x-3)] \bmod 19 \\ &= (26x^2 - 188x + 296) \bmod 19 \\ &= 7x^2 + 2x + 11 \end{aligned}$$

从而得秘密为 $s = 11$ 。



门限方案的实例

- 假定房间里有4个人，其中一个国外特务，其余3人拥有Shamir秘密分享方案的数对，任何两个人都能确定秘密。国外特务随机选择了一个数对，人员和数对如下。所有的数对都是模11的。
- A: (1, 4) B: (3, 7) C: (5, 1) D: (7, 2)
- 确定哪一个是特务，秘密是什么？

中国剩余定理

- 如果已知某个数关于一些两两互素的数的同余类集，就可以重构这个数
- 定理(中国剩余定理):

设 m_1, m_2, \dots, m_k 是两两互素的正整数,

则一次同余方程组

$$\begin{cases} x \bmod m_1 = a_1 \\ x \bmod m_2 = a_2 \\ \dots \\ x \bmod m_k = a_k \end{cases}$$

对模 M 有唯一解

$$x \equiv \left(\frac{M}{m_1} e_1 a_1 + \frac{M}{m_2} e_2 a_2 + \dots + \frac{M}{m_k} e_k a_k \right) \bmod M$$

$$M = \prod_{i=1}^k m_i$$

$$e_i \text{ 满足 } \frac{M}{m_i} e_i \equiv 1 \bmod m_i$$

中国剩余定理

- 中国剩余定理可以将一个很大的数 x 表示为一组较小的数 (a_1, \dots, a_k)
- 例: $x \equiv 1 \pmod{2}$, $x \equiv 2 \pmod{3}$, $x \equiv 3 \pmod{5}$, $x \equiv 5 \pmod{7}$, 求 x
- 解: $M = 2 \times 3 \times 5 \times 7 = 210$, $M_1=105$, $M_2=70$, $M_3=42$, $M_4=30$,
($M_i=M/m_i$), 可以求得 $e_1=1$, $e_2=1$, $e_3=3$, $e_4=4$, 所以 $x=105 \times 1 \times 1 + 70 \times 1 \times 2 + 42 \times 3 \times 3 + 30 \times 4 \times 5 \pmod{210} = 173$

基于中国剩余定理的门限方案

设 m_1, m_2, \dots, m_n 是 n 个大于1的整数, 满足

$$m_1 \leq m_2 \leq \dots \leq m_n \quad \gcd\{m_i, m_j\} = 1 (\forall i, j, i \neq j)$$

$$m_1 m_2 \cdots m_k > m_n m_{n-1} \cdots m_{n-k+2}$$

又设 s 是秘密数据, 满足 $m_n m_{n-1} \cdots m_{n-k+2} < s < m_1 m_2 \cdots m_k$ 。

计算 $M = m_1 m_2 \cdots m_n$, $s_i \equiv s \pmod{m_i}$ ($i = 1, \dots, n$)。以 (s_i, m_i, M) 作为一个子密钥, 集合 $\{(s_i, m_i, M)\}_{i=1}^n$ 即构成了一个 (k, n) 门限方案。

这是因为在 k 个参与者 (记为 i_1, i_2, \dots, i_k) 中, 每个 i_j 计算

$$\begin{cases} M_{i_j} = M / m_{i_j} \\ N_{i_j} \equiv M_{i_j}^{-1} \pmod{m_{i_j}} \\ y_{i_j} = s_{i_j} M_{i_j} N_{i_j} \end{cases}$$

基于中国剩余定理的门限方案

结合起来，根据中国剩余定理可求得方程组

$$\begin{cases} s \equiv s_{i_1} \pmod{m_{i_1}} \\ \vdots \\ s \equiv s_{i_k} \pmod{m_{i_k}} \end{cases} \quad (7.2)$$

的解

$$s = \sum_{j=1}^k y_{i_j} \pmod{\prod_{j=1}^k m_{i_j}} \quad (7.3)$$

基于中国剩余定理的门限方案

下面证明在方案的条件下, (7.3) 得到的 \mathbf{S} 的确是方案中被分割的秘密数据, 即 \mathbf{S} 也是方程组 (7.4) 的解。

$$\begin{cases} s \equiv s_1 \pmod{m_1} \\ \vdots \\ s \equiv s_n \pmod{m_n} \end{cases} \quad (7.4)$$

设 (7.4) 的解为 s' , 显然 s' 也满足 (7.2), 有 $s' - s \equiv 0 \pmod{m_{i_1}}, \dots, s' - s \equiv 0 \pmod{m_{i_k}}$, 所以 $m_{i_1} | s' - s, \dots, m_{i_k} | s' - s$, $\text{lcm}(m_{i_1}, \dots, m_{i_k}) = m_{i_1} \cdots m_{i_k} | s' - s$, 得 $s' \equiv s \pmod{m_{i_1} \cdots m_{i_k}}$ 。

又因 $s', s < m_1 m_2 \cdots m_k \leq m_{i_1} \cdots m_{i_k}$, 所以 $s' = s$ 。

基于中国剩余定理的门限方案

若参与者少于 k 个，不妨设为 $k-1$ ，则建立的方程组为

$$\begin{cases} s \equiv s_{i_1} \pmod{m_{i_1}} \\ \dots \\ s \equiv s_{i_{k-1}} \pmod{m_{i_{k-1}}} \end{cases}$$

方程组的解为 $s' = \sum_{j=1}^{k-1} y_{i_j} \pmod{\prod_{j=1}^{k-1} m_{i_j}}$ ，满足

$$s' < \prod_{j=1}^{k-1} m_{i_j} < m_n m_{n-1} \cdots m_{n-k+2} < S$$

得到不正确的结果。



【例5-5】

设 $k=3$, $n=5$, $m_1=97, m_2=98, m_3=99, m_4=101, m_5=103$,
秘密数据 $s=671875$, 满足

$$10403 = m_4 m_5 < s < m_1 m_2 m_3 = 941094 \quad .$$

计算 $M = m_1 m_2 m_3 m_4 m_5 = 9790200882$, $s_i \equiv s \pmod{m_i}$
($i=1, \dots, 5$) 得 $s_1=53, s_2=85, s_3=61, s_4=23, s_5=6$ 。5个子
密钥为 $(53, 97, 9790200882)$, $(85, 98, 9790200882)$,
 $(61, 99, 9790200882)$, $(23, 101, 9790200882)$, $(6, 103, 9790200882)$

现在假定 i_1, i_2, i_3 联合起来计算, 他们分别计算:

$$\begin{cases} M_1 = M/m_1 = 100929906 \\ N_1 \equiv M_1^{-1} \pmod{m_1} \equiv 95 \end{cases} \quad \begin{cases} M_2 = M/m_2 = 99900009 \\ N_2 \equiv M_2^{-1} \pmod{m_2} \equiv 13 \end{cases} \quad \begin{cases} M_3 = M/m_3 = 98890918 \\ N_3 \equiv M_3^{-1} \pmod{m_1} \equiv 31 \end{cases}$$





得到

$$\begin{aligned}s &\equiv s_1 M_1 N_1 + s_2 M_2 N_2 + s_3 M_3 N_3 \pmod{m_1 m_2 m_3} \\&\equiv 53 \cdot 100929906 \cdot 95 + 85 \cdot 99900009 \cdot 13 + 61 \cdot 98890918 \cdot 31 \pmod{97 \cdot 98 \cdot 99} \\&\equiv 805574312593 \pmod{941094} \\&\equiv 671875\end{aligned}$$

假定 i_1, i_4, i_5 联合起来计算 s ，他们分别计算：

$$\begin{cases} M_1 = M / m_1 = 100929906 \\ N_1 \equiv M_1^{-1} \pmod{m_1} \equiv 95 \end{cases} \quad \begin{cases} M_4 = M / m_4 = 96932682 \\ N_4 \equiv M_4^{-1} \pmod{m_4} \equiv 61 \end{cases} \quad \begin{cases} M_5 = M / m_5 = 95050494 \\ N_5 \equiv M_5^{-1} \pmod{m_5} \equiv 100 \end{cases}$$

得到





$$\begin{aligned}s &\equiv s_1 M_1 N_1 + s_4 M_4 N_4 + s_5 M_5 N_5 \pmod{m_1 m_4 m_5} \\&\equiv 53 \cdot 100929906 \cdot 95 + 23 \cdot 96932682 \cdot 61 + 6 \cdot 95050494 \cdot 100 \pmod{97 \cdot 101 \cdot 103} \\&\equiv 701208925956 \pmod{1009091} \\&\equiv 671875\end{aligned}$$

假定 i_1, i_4 联合起来计算 s ，则：

$$\begin{aligned}s &\equiv s_1 M_1 N_1 + s_4 M_4 N_4 \pmod{m_1 m_4} \\&\equiv 53 \cdot 100929906 \cdot 95 + 23 \cdot 96932682 \cdot 61 \pmod{97 \cdot 101} \\&\equiv 644178629556 \pmod{9791} \\&\equiv 5679\end{aligned}$$

得到一个不正确结果。



A decorative graphic consisting of several horizontal blue bars of varying lengths, stacked vertically, is positioned to the left of the title.

可验证秘密分享

- Feldman, 1987
 - 参数生成
 - 选取大素数 p , q , q 是 $p-1$ 的素因子
 - g 是 $GF(p)^*$ 中的一个 q 阶元
 - k 是门限值, n 是用户个数
-

可验证秘密分享（续）

- 份额分配

- 假设可信中心 T 欲将秘密 $s \in Z_q$ 在 n 个用户 U_i ($1 \leq i \leq n$) 之间进行分配，其步骤如下：
 - ① T 随机选择 $k-1$ 个独立的系数 $a_1, a_2, \dots, a_{t-1} \in Z_q$ ，定义 $a_0 = s$ ，建立一个多项式：
 - ② T 计算 $y_i = f(i) \bmod q$, $1 \leq i \leq n$ ，并将 (i, y_i) 分配给用户 U_i ，其中 i 公开， y_i 为 U_i 的秘密份额。
 - ③ 广播 $b_i = g^{a_i} \bmod p$, $i = 0, 1, \dots, k-1$ 。

A decorative blue horizontal bar with a series of horizontal lines is positioned to the left of the title.

可验证秘密分享（续）

- 份额验证算法

- 对于份额 y_i ($1 \leq i \leq n$), U_i 可以验证是否有:

$$g^{y_i} = \prod_{j=0}^{k-1} b_j^{i^j} \bmod p$$

- 如果等式不成立, 说明 U_i 收到的份额 y_i 是无效的, U_i 就可以广播一个对 T 的抱怨。
-

可验证秘密分享（续）

- 恢复算法

- 当 k 个用户 U_1, U_2, \dots, U_k 合作恢复秘密时，每一 U_i 向其他合作者广播自己的份额 y_i ，每一合作者都可以通过下式：

$$g^{y_i} = \prod_{j=0}^{k-1} b_j^{i^j} \bmod p$$

- 来验证 y_i 的有效性。当所有的 y_i ($1 \leq i \leq k-1$) 都被验证为有效时，每个合作者都可以通过拉格朗日插值法计算出秘密 s 。

基于口令的认证

A decorative blue horizontal bar with white horizontal stripes is positioned to the left of the title.

身份证明技术

传统的身份证明:

一般是通过检验“物”的有效性来确认持该物的的身份。徽章、工作证、信用卡、驾驶执照、身份证、护照等，卡上含有个人照片(易于换成指纹、视网膜图样、牙齿的X适用的射像等)。

信息系统常用方式:

用户名和口令

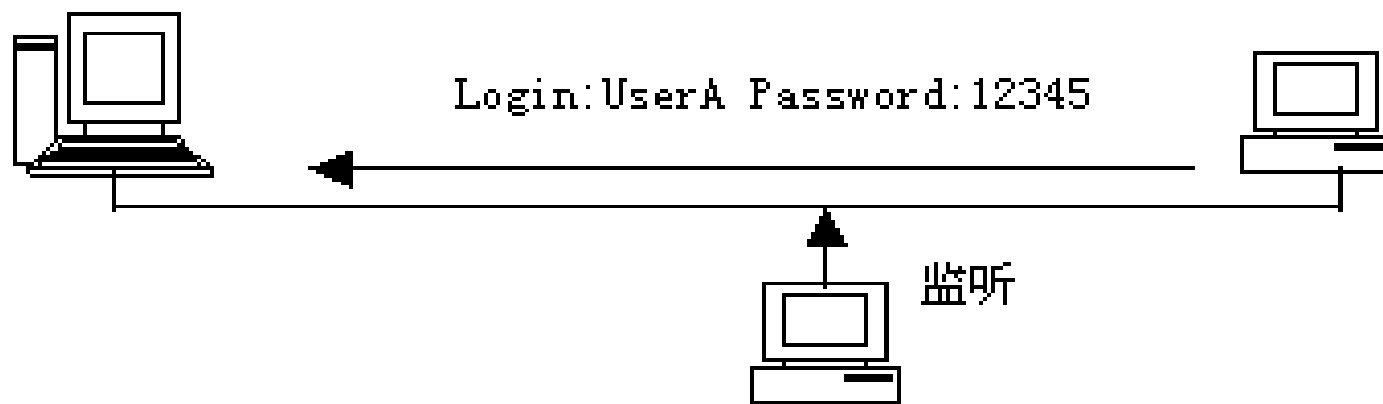
A decorative blue horizontal bar with a series of horizontal lines is positioned to the left of the title.

单向认证中的口令认证

- 通过用户ID和口令进行认证是操作系统或应用程序通常采用的。
 - 易猜的口令或缺省口令也是一个很严重的问题。
-

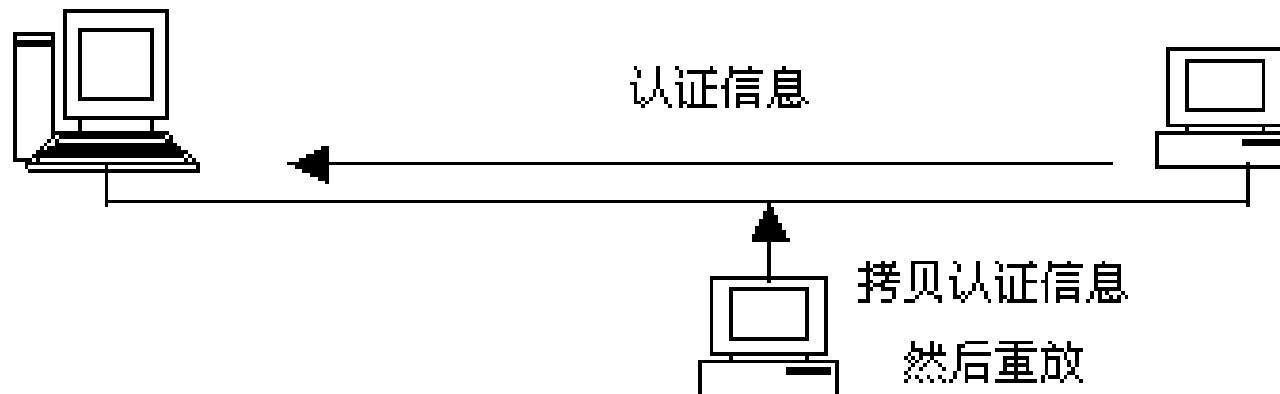
口令认证的攻击类型

- 目前各类计算资源主要靠固定口令的方式来保护。这种以固定口令为基础的认证方式存在很多问题，对口令的攻击包括以下几种：
- (1) 网络数据流窃听(Sniffer)：攻击者通过窃听网络数据，如果口令使用明文传输，则可被非法截获。大量的通讯协议比如Telnet、Ftp、基本HTTP都使用明文口令，这意味着它们在网络上是以未加密格式传输于服务器端和客户端，而入侵者只需使用协议分析器就能查看到这些信息，从而进一步分析出口令。



口令认证的攻击类型—截取/重放

- (2) 认证信息截取/重放(Record/Replay): 有的系统会将认证信息进行简单加密后进行传输, 如果攻击者无法用第一种方式推算出密码, 可以使用截取/重放方式, 需要的是重新编写客户端软件以使用加密口令实现系统登录。



A decorative blue horizontal bar with white horizontal stripes is positioned to the left of the title.

口令认证的攻击类型

- (3) 字典攻击：根据调查结果可知，大部份的人为了方便记忆选用的密码都与自己周遭的事物有关，例如：身份证字号、生日、车牌号码、在办公桌上可以马上看到的标记或事物、其他有意义的单词或数字，某些攻击者会使用字典中的单词来尝试用户的密码。所以大多数系统都建议用户在口令中加入特殊字符，以增加口令的安全性。
 - (4) 穷举攻击 (Brute Force)：也称蛮力破解。这是一种特殊的字典攻击，它使用字符串的全集作为字典。如果用户的密码较短，很容易被穷举出来，因而很多系统都建议用户使用长口令。
-

口令认证的攻击类型

- (5) 窥探：攻击者利用与被攻击系统接近的机会，安装监视器或亲自窥探合法用户输入口令的过程，以得到口令。
- (6) 社交工程：社会工程就是指采用非隐蔽方法盗用口令等，比如冒充是处长或局长骗取管理员信任得到口令等等。冒充合法用户发送邮件或打电话给管理人员，以骗取用户口令等。比如，在终端上可能发现如下信息：
 - Please enter your user name to logon:
 - Your password:
 - 这很可能是一个模仿登录信息的特洛伊木马程序，他会记录口令，然后传给入侵者。

A decorative blue horizontal bar with white horizontal stripes is positioned to the left of the section header.

口令认证的攻击类型

- (7) 垃圾搜索：攻击者通过搜索被攻击者的废弃物，得到与攻击系统有关的信息，如果用户将口令写在纸上又随便丢弃，则很容易成为垃圾搜索的攻击对象。

安全口令的特点

- 在口令的设置过程中，有许多个人因素在起作用，攻击者可以利用这些因素来解密。由于口令安全性的考虑，人们会被禁止把口令写在纸上，因此很多人都设法使自己的口令容易记忆，而这就给攻击者提供了可乘之机。为防止攻击猜中口令。安全口令具有以下特点：
- (1) 位数>6位。
- (2) 大小写字母混合。如果用一个大写字母，既不要放在开头，也不要放在结尾。
- (3) 可以把数字无序的加在字母中。
- (4) 系统用户一定用8位口令，而且包括~!@# \$%^&*<>?:"{}等特殊符号。

A decorative blue horizontal bar with white horizontal stripes is positioned to the left of the title.

不安全口令的类型

- 不安全的口令则有如下几种情况：
 - （1）使用用户名（帐号）作为口令。这种方法便于记忆，可是在安全上几乎是不堪一击。几乎所有以破解口令为手段的黑客软件，都首先会将用户名作为口令的突破口。
 - （2）用用户名（帐号）的变换形式作为口令。将用户名颠倒或者加前后缀作为口令，比如说著名的黑客软件John，如果用户名是fool，那么它在尝试使用fool作为口令之后，还会试着使用诸如fool123、fool1、loof、loof123、lofo等作为口令。
-

不安全口令的类型

- (3) 使用自己或者亲友的生日作为口令。这种口令有着很大的欺骗性，因为这样往往可以得到一个6位或者8位的口令，但实际上可能的表达方式只有 $100 \times 12 \times 31 = 37200$ 种，即使再考虑到年月日三者共有六种排列顺序，一共也只有 $37200 \times 6 = 223200 (< 2^{19})$ 种。
- (4) 使用常用的英文单词作为口令。这种方法比前几种方法要安全一些。如果选用的单词是十分偏僻的，那么黑客软件就可能无能为力了。

加强口令安全性的措施

- 应采取两个步骤以消除口令漏洞。第一步，所有弱口令应被加强。但是当用户被要求改变或加强他们的弱口令时，他们经常又选择一个容易猜测的。这就导致了第二步，用户的口令在被修改后，应加以确认。可以用程序来拒绝任何不符合安全策略的口令。
- 可以采取以下措施来加强口令的安全性：
 - (1) 在创建口令时执行检查功能。如检查口令的长度。
 - (2) 强制使口令周期性过期。也就是定期更换口令。
 - (3) 保持口令历史记录，使用户不能循环使用旧口令。

口令认证协议

- 系统登录时的口令认证协议
 - 可以用口令，也可以替换成指纹等生物特征
 - 最简单的是明文发送口令或生物特征，系统存储该特征的hash值，系统收到口令或特征后计算hash值进行比对
 - 复杂的安全的方式是不明文发送口令，而是与系统基于该口令进行交互，甚至协商出密钥
- 一个经典的方法是Needham口令认证协议
 - 主机存储用户名 ID_u 及其口令 Pwd 的hash值(或单向函数值)
 - 登陆时，首先用户提交身份 ID_u ，然后主机提示输入口令，用户根据提示输入口令 Pwd ，主机验证其hash值是否等于事先存储的hash值

口令认证协议

- 在Unix系统中的口令认证协议即采用此种方法，单向函数使用的是DES算法，即用用户的口令加密64bit的连0串，其中DES算法也做了适当修改
- 一次口令认证协议S/KEY
 - 口令登陆时的明文发送无法防止在线口令攻击，所以希望每次发送的口令最好不一样，此即一次口令认证。同时为了多次登陆，这些口令应该能由一个口令导出，即它们是相关的
 - Lamport的一次口令方案
 - 主机H存储初始口令为 $(ID_u, hash^n(Pwd), n)$ ，U记住Pwd，H中当前口令记录是 $(ID_u, hash^c(Pwd), c)$ ，其中c大于等于1且小于等于n
 - 登陆时首先提交身份 ID_u ，H提示U输入口令；U计算 $hash^{c-1}(Pwd)$ 发给H，h计算一次hash并检查当前口令是否满足 $hash^c$ ；H更新U的当前口令记录为 $(ID_u, hash^{c-1}(Pwd), c-1)$
 - 其中n和c用于同步，以免受到攻击

基于口令的认证与密钥协商

- 本节介绍基于口令的远程认证协议，收发双方除了基于口令认证身份外，还要协商会话密钥。我们以Islam等提出的协议为例给出一个示例方案。口令是一种低熵密钥
- 1. 符号假设
 - ID_A : 用户A的身份标识; S : 远程服务器的身份标识; D : 候选口令字典集; PW_A : 用户 U_A 的口令; $U_A: U_A = PW_A \times P$; d_s : 服务器S的高熵密钥; SK : 用户和服务器协商出的会话密钥; $P_s: P_s = d_s \times P$; G 是素数阶 q 的加法交换群(主要是椭圆曲线群); P : 群 G 的生成元; $h()$: hash函数; \parallel : 字符串的连接;
- 2. 系统初始化协议
 - 所有用户和服务器协商好椭圆曲线密码系统参数，服务器选择的系统私钥和公钥对为 (d_s, P_s) 。然后服务器选择一哈希函数 $h()$ ，如SHA-256，和密钥导出函数 $kdf: G_p \rightarrow \{0,1\}^n$ ，公布参数 $\{E_p(a,b), P, P_s, n, h(), kdf\}$
- 3. 注册协议
 - 1) $U_A \rightarrow S: \langle ID_A, U_A \rangle$ 用户选择自己的身份标识 ID_A 和口令 PW_A ，计算 $U_A = PW_A \times P$ ，通过安全信道发送 ID_A, U_A 给服务器S
 - 2) 服务器S收到上一步消息后把 $\langle ID_A, U_A \rangle$ 保存在数据库内

基于口令的认证与密钥协商

- 4. 用户认证和密钥协商协议

- 用户希望登陆到远程服务器并和服务器协商会话密钥的时候，首先输入身份 ID_A 和口令 PW_A ，并执行如下协议

- 1) $U_A \rightarrow S: \langle ID_A, T_A, R_A, H_A \rangle$ 用户选择一随机数 r ，并计算

$$R_A = rP,$$

$$T_A = r \times PW_A \times P + rP_S,$$

$$H_A = h(r \times PW_A \times P_S, T_A, R_A),$$

然后给服务器发去消息 $\langle ID_A, T_A, R_A, H_A \rangle$

基于口令的认证与密钥协商

4. 用户认证和密钥协商协议

- 用户希望登陆到远程服务器并和服务器协商会话密钥的时候，首先输入身份 ID_A 和口令 PW_A ，并执行如下协议

- 2) $S \rightarrow U_A : \langle T_S, H_S \rangle$ 服务器接到1)步发来的消息后做如下运算：

- (1) 验证 ID_A 的合法性，如果不正确则停止协议，否则继续以下协议
- (2) 计算 $K = d_S \times R_A$
- (3) 提取 $r \times U_A = T_A - K = r \times PW_A \times P$ ，然后计算 $H_A^* = h(d_S \times (T_A - K), T_A, R_A)$
- (4) 验证 H_A^* 是否等于 H_A ，如果不等终止协议，否则继续执行
- (5) 选择一个随机数 s ，并计算 $T_s = r \times U_A + s \times P_S$ 和 $H_s = h(s \times P_S)$
- (6) 服务器发送消息 $\langle T_S, H_S \rangle$ 给用户

$$U_A = PW_A \times P$$

$$P_S = d_S \times P$$

$$R_A = rP,$$

$$T_A = r \times PW_A \times P + rP_S,$$

$$H_A = h(r \times PW_A \times P_S, T_A, R_A),$$

基于口令的认证与密钥协商

- 3) $U_A \rightarrow S: \langle ID_A, H_{AS} \rangle$
 - (1) 用户接到2)中服务器发来的消息后, 执行如下协议
 - (2) 计算 $s \times P_S = T_S - r \times U_A, H_S^* = h(s \times P_S)$
 - (3) 检查 H_S^* 是否等于 H_S , 如果不等则停止协议, 否则继续
 - (4) 计算 $H_{AS} = (r \times U_A, s \times P_S)$ 并发送消息 ID_A, H_{AS} 给服务器
- 4) $S \rightarrow U_A: \text{Access granted/denied}$
 - 服务器S计算 $H_{SA} = (r \times U_A, s \times P_S)$, 并把 H_{SA} 和 H_{AS} 比较, 如果不等则服务器发 **Access denied** 给用户, 否则发送 **Access granted** 给用户表示认证成功
 - 当协议执行完毕, 用户和服务器共享秘密会话密钥 $SK = h(ID_A, T_A, T_S, R_A, H_S, H_{AS}, K)$ 其中 $K = (r \times PW_A) \times s \times P_S = (s \times d_S) \times r \times U_A = r \times s \times d_S \times PW_A \times P$



感谢聆听!

liaoyj@uestc.edu.cn