电子科技大学

University of Electronic Science and Technology of China

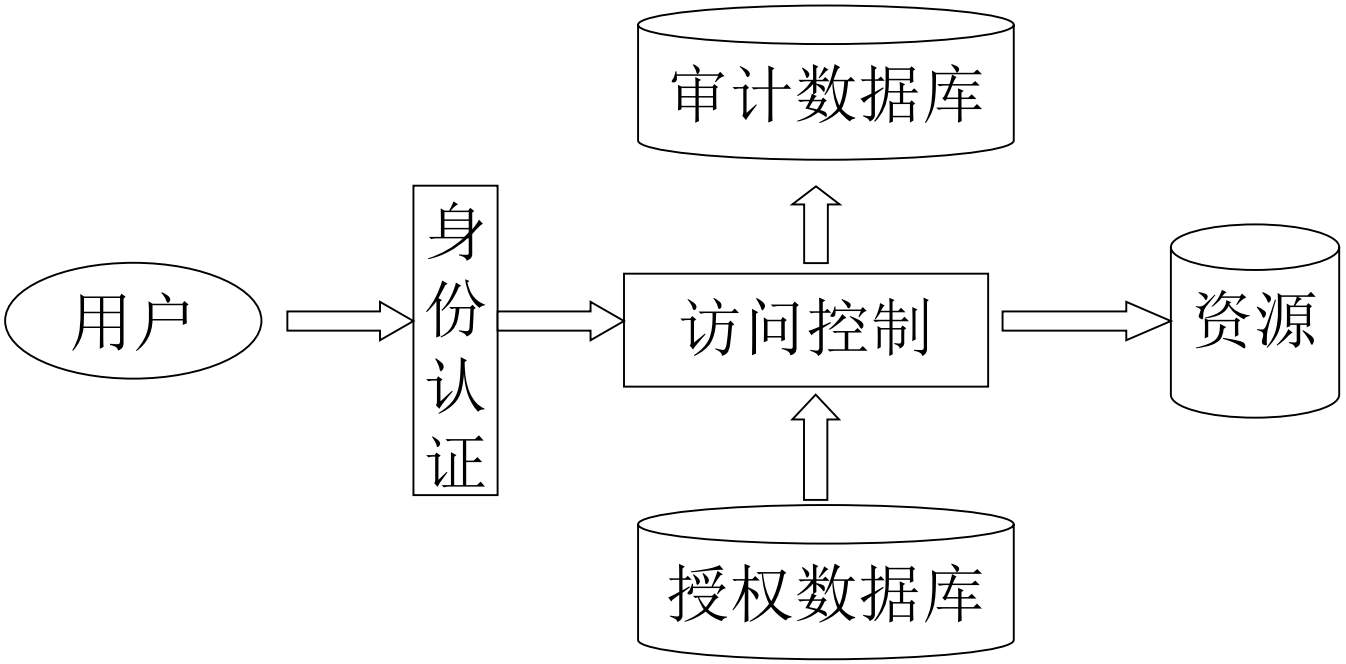
第六章 身份认证

身份认证概述

- 身份认证(authentication ）：
  - 证实主体的真实身份与其所声称的身份是否相符的过程。
- 现实生活中，主要通过各种证件来验证身份，比如：身份证、户口本等。

用户对资源的访问过程

- 获得系统服务所必须的第一道关卡。
- 访问控制和审计的前提。



身份认证攻击：

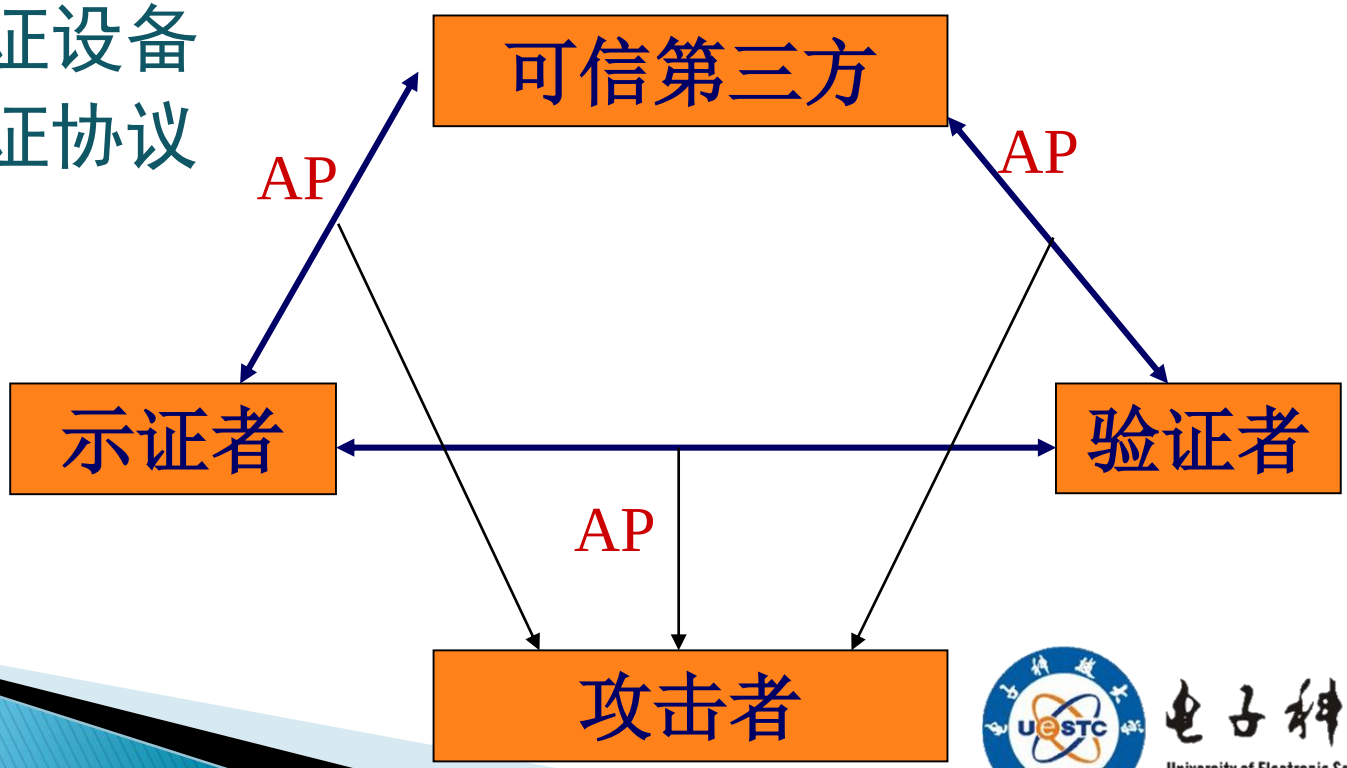
- 数据流窃听(Sniffer)：
  - 攻击者窃听网络数据，辨析认证数据，提取用户名和口令。
- 拷贝/重传：
  - 非法用户截获信息，然后再传送给接收者。
- 修改或伪造：
  - 非法用户截获信息，替换或修改信息后再传送给接收者，
  - 非法用户冒充合法用户发送信息。

身份认证分类

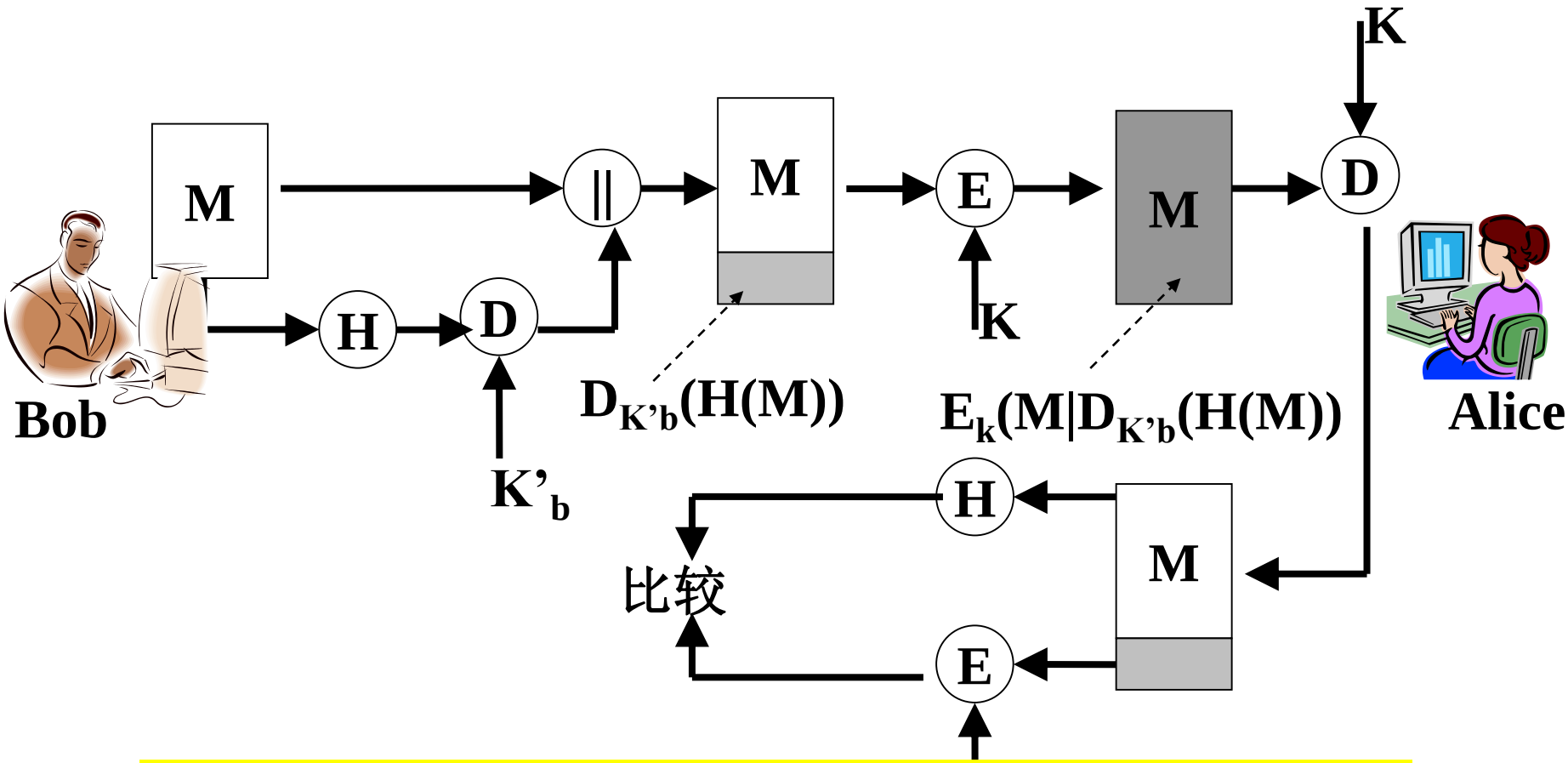
- 根据地域
  - 本地：本地环境的初始化认证
  - 远程：连接远程设备、实体和环境的实体认证。
- 根据方向
  - 单向认证：指通信双方中只有一方向另一方进行鉴别。
  - 双向认证：指通信双方相互进行鉴别。

身份认证组成及模型

- 身份认证系统组成：
  - 认证服务器
  - 认证系统用户端软件
  - 认证设备
  - 认证协议



温故而知新——消息认证完整模型



认证+签名+保密

身份认证依据

- 用户所知Something the user know
  - 密码、口令等
  - 简单，开销小，容易泄密，最不安全；
- 用户所有Something the user possesses
  - 身份证、护照、密钥盘等
  - 泄密可能性较小，安全性高于第一类，系统相对复杂；
- 用户特征Something the user is (or How he behaves)
  - 指纹、笔迹、声音、虹膜、DNA等
  - 安全性最高，如窃取指纹很困难，涉及更复杂算法和实现技术。

公钥证书？？

身份认证机制

- 非密码
  - 口令等
- 基于密码算法
  - 对称密码算法
  - 公开密码算法
  - 密码校验函数
- 零知识证明协议

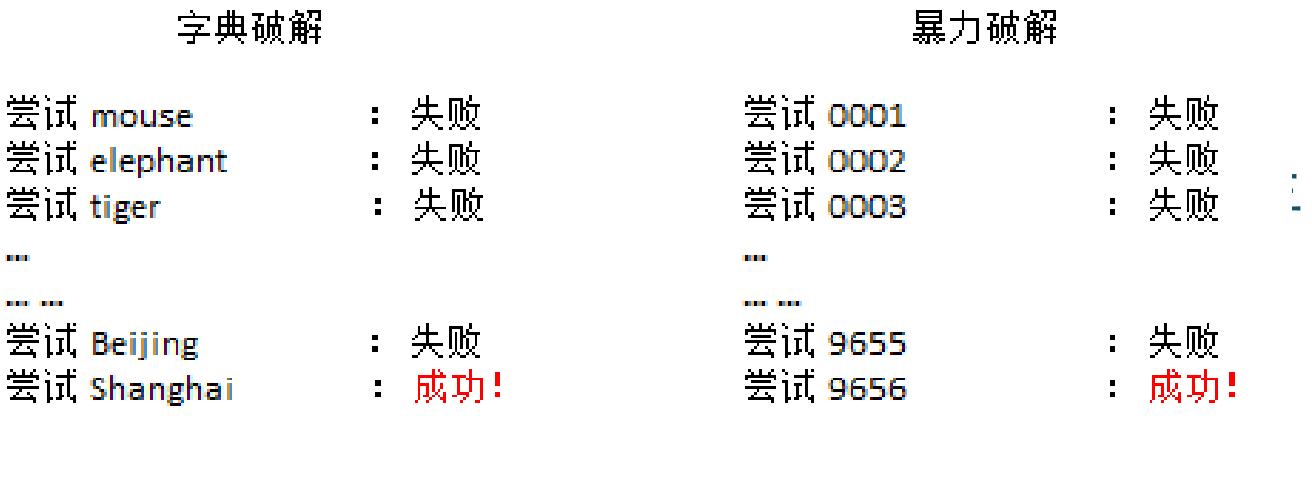
6.1 非密码认证机制

非密码身份认证

- 口令机制
- 基于地址的认证机制
- 基于生物特征的认证机制
- 个人令牌认证机制

口令攻击

- 字典攻击
  - 为方便记日、车牌
  - 形成字典
- 穷举攻击
  - 特殊字典



- 窥探：
  - 接近被攻击系统，安装监视器或亲自窥探用户输入口令。
- 社交工程：
  - 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。
- 垃圾搜索：
  - 搜索被攻击者的废弃物，得到与攻击系统有关的信息，如用户将口令写在纸上又随便丢弃。

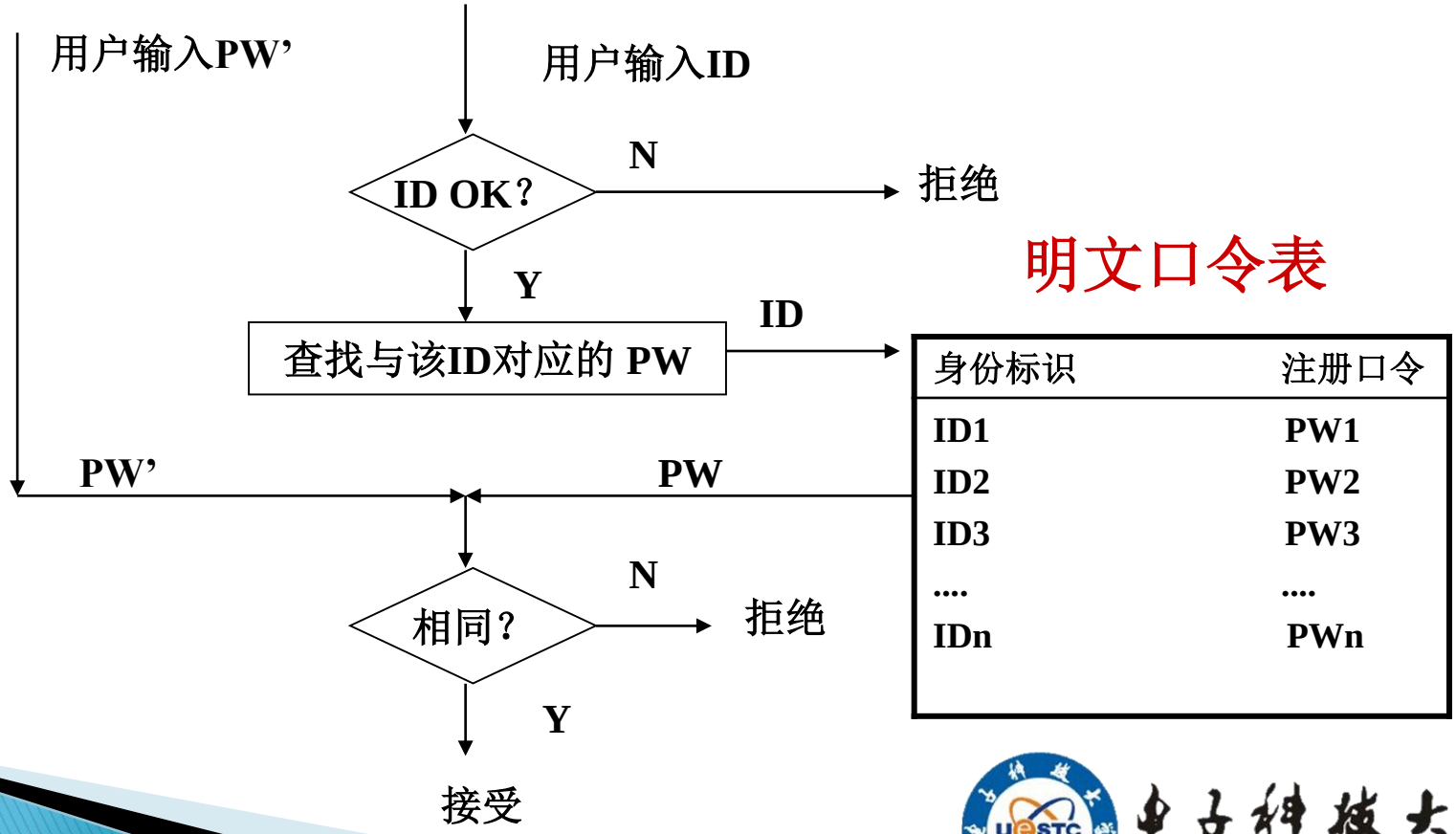
安全口令

- 采用较长的长度
  - 口令破解的难度随口令长度指数增长，如至少包含8个字符
- 采用多种字符的组合
  - 如大小写、数字和各种符号的组合
- 避免使用单词、术语及用户相关信息
  - 如单词、术语以及用户名、姓名、电话、生日、车牌等用户相关信息

口令安全增强策略和机制

- 1、限制猜测次数。
- 2、降低猜测口令速度。
- 3、增加口令长度，增加攻击者搜索空间。
- 4、要求用户选择安全的口令。
- 5、定期更换口令。

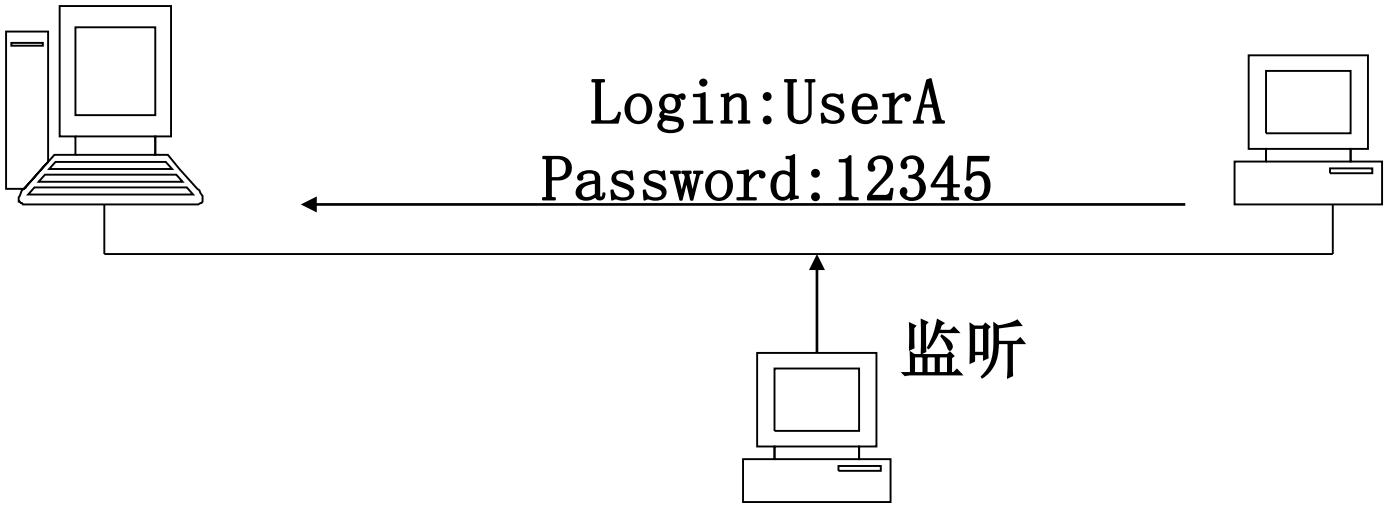
口令机制：明文



明文口令表

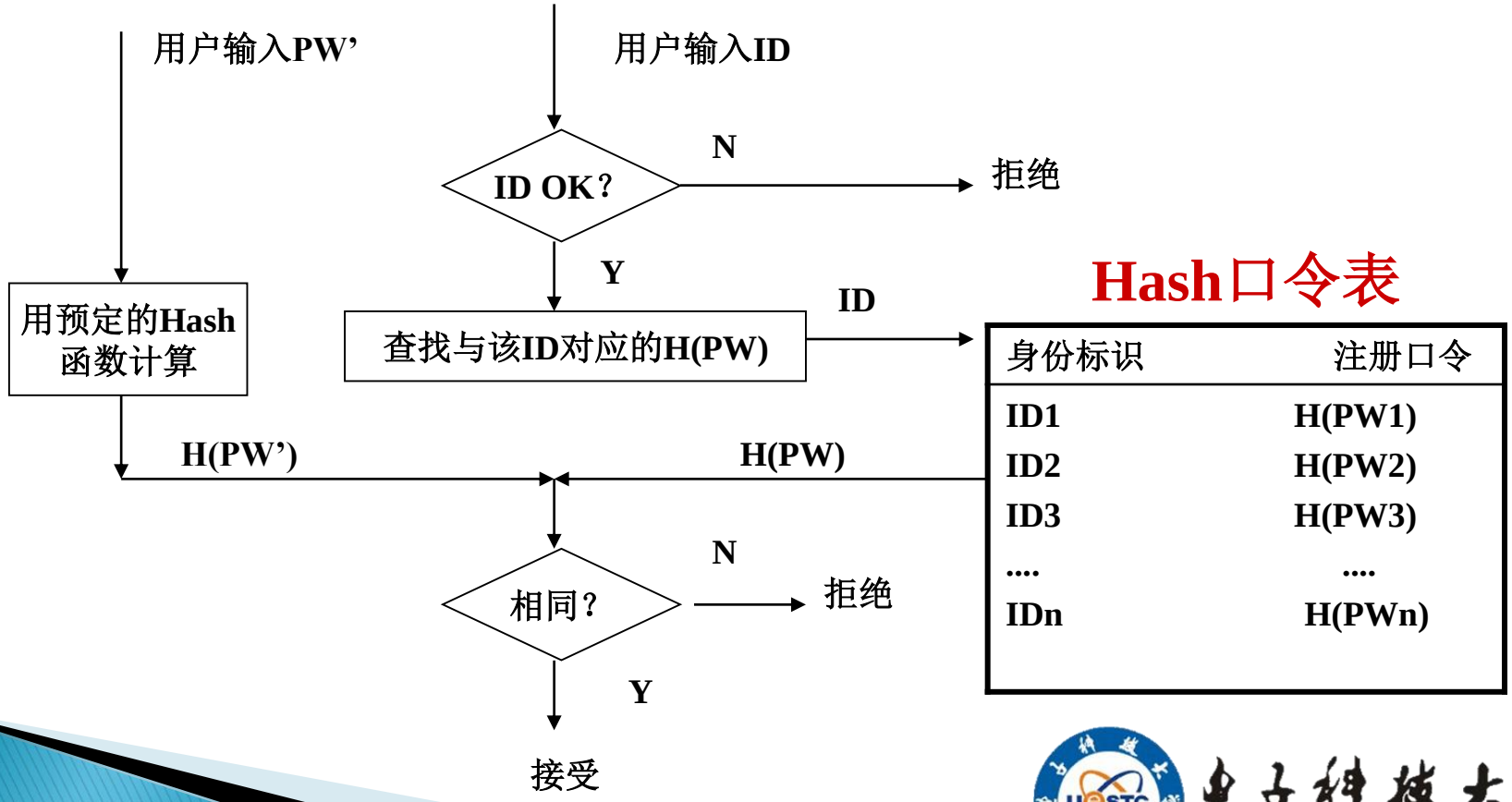
身份标识	注册口令
ID1	PW1
ID2	PW2
ID3	PW3
....	....
IDn	PWn

明文口令机制攻击

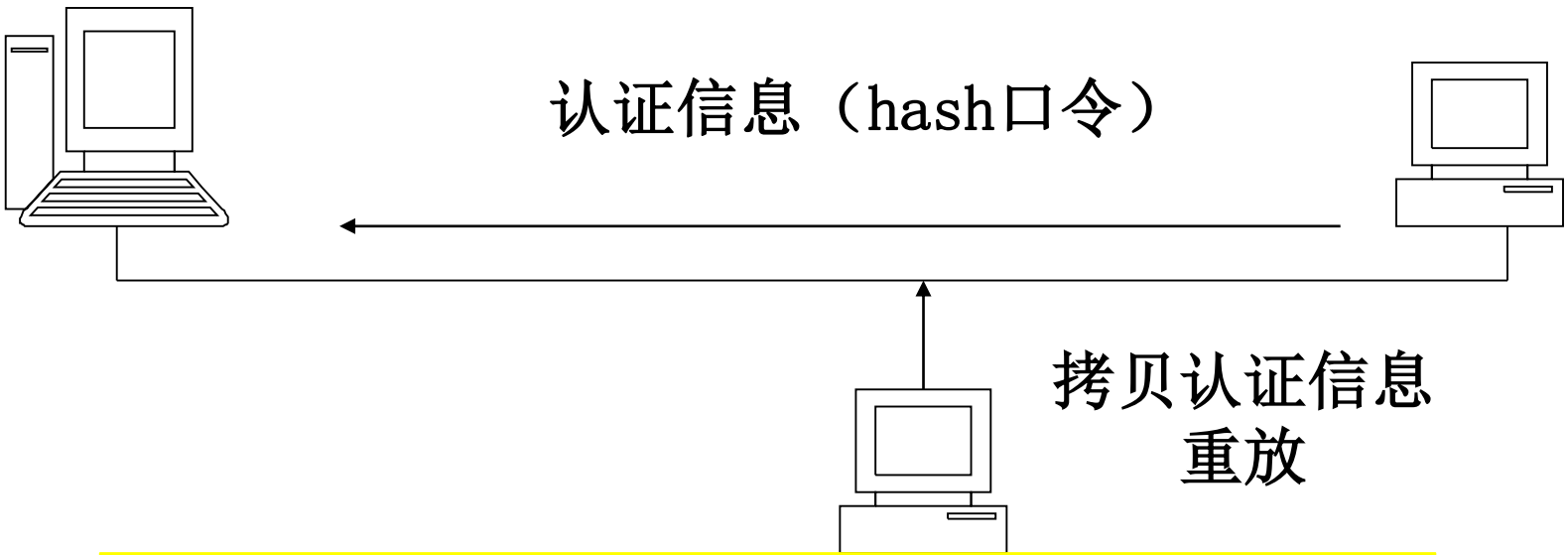


- 面临威胁：
  - ✓ 获取口令文件
  - ✓ 监听解析口令
  - ✓ 重放攻击

口令机制：hash口令表

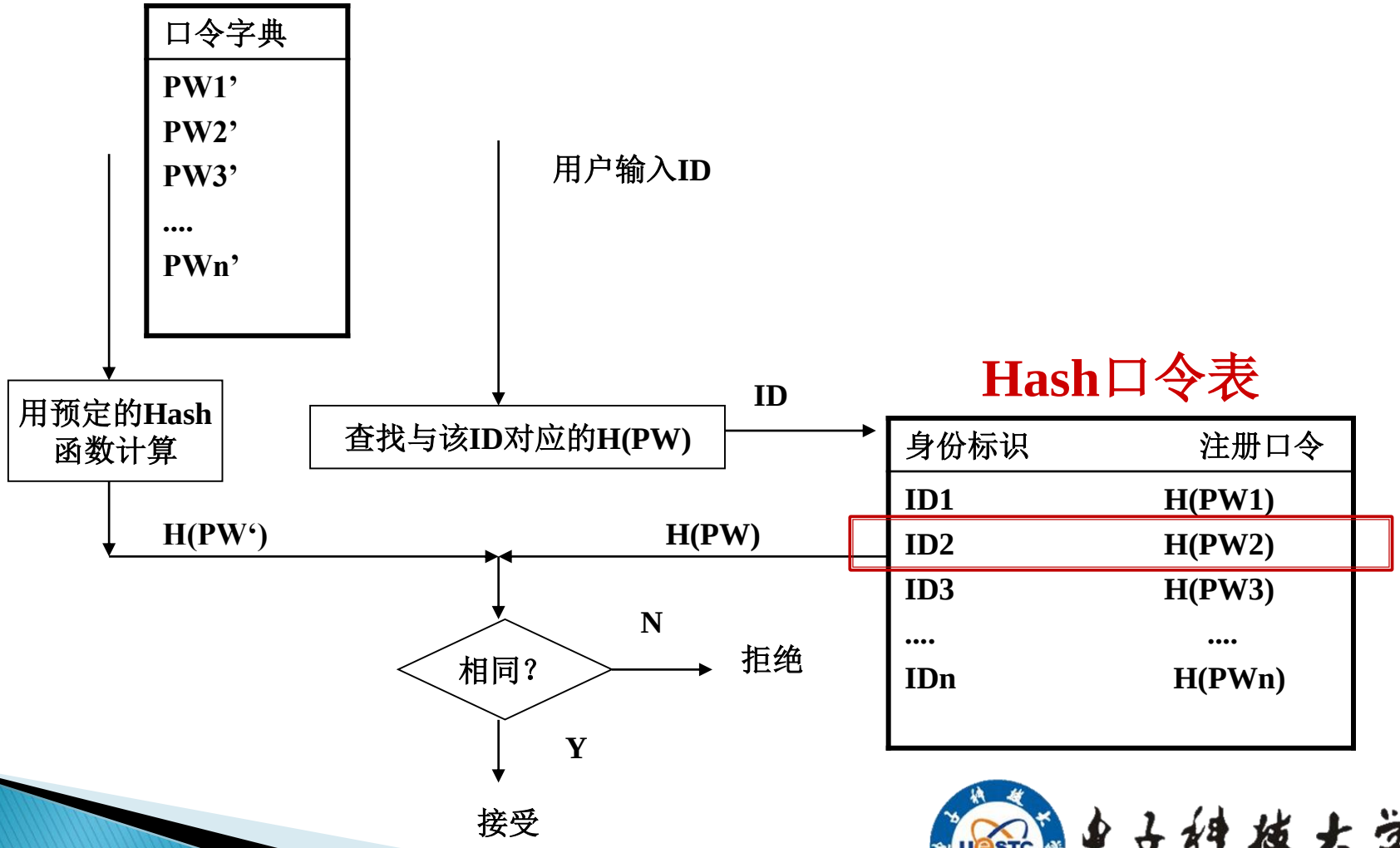


hash口令机制攻击



- 面临威胁：
  - ✓ 重放攻击
  - ✓ 字典攻击

hash口令机制——字典攻击



字典攻击——查表法获取口令

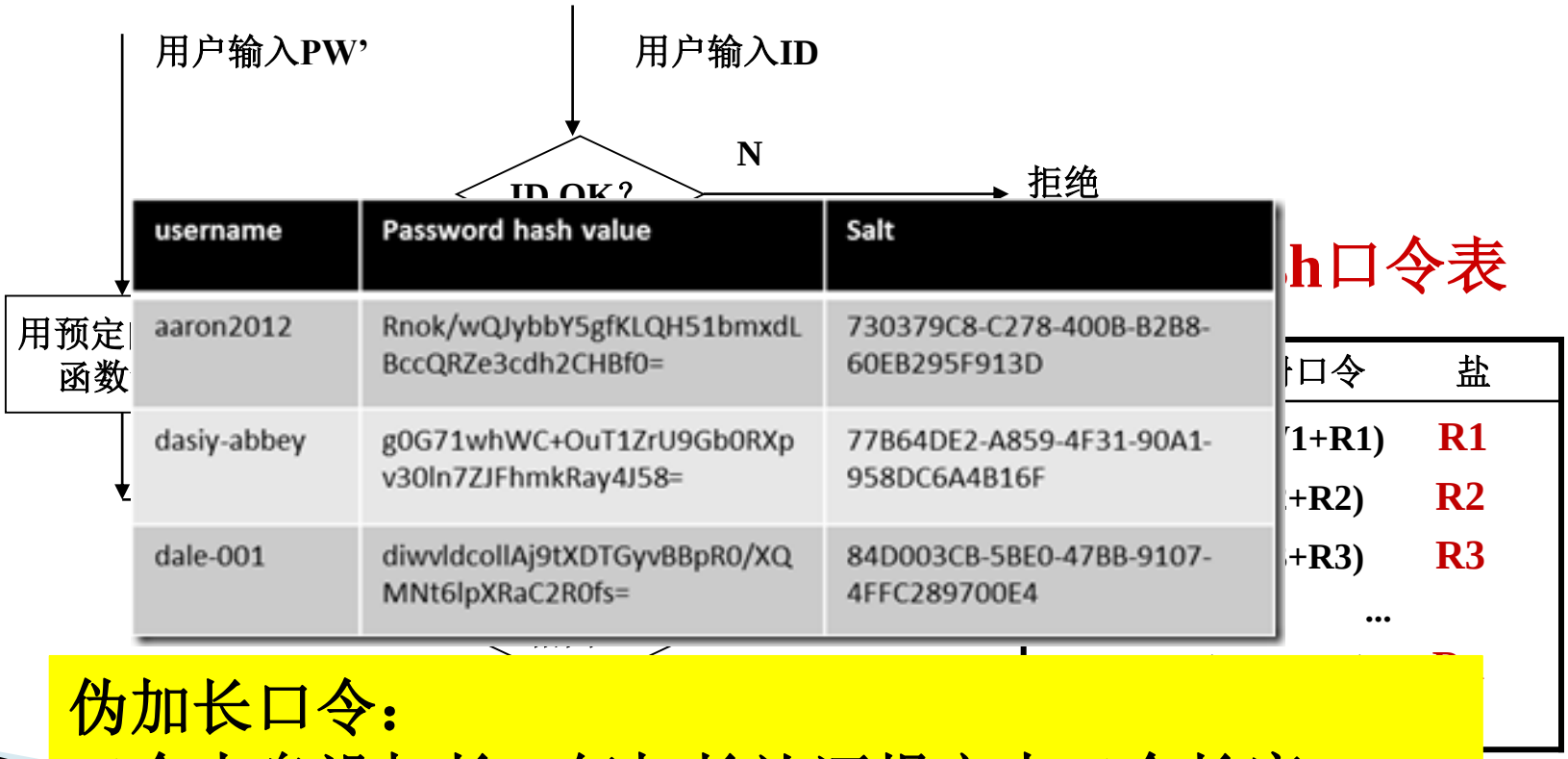
- 计算潜在口令(口令字典)的哈希，形成表；
  - 彩虹表：庞大的、针对各种可能的字母组合预先计算好的哈希值的集合，主流的彩虹表都是100G以上。
- 用获取（嗅探窃取）的口令哈希查表

password	Hash value
mouse	R8XCjK4ldM31oZT+dxfeaPgnb0v4PmU4MJJQVq6zKkg=
elephant	zQjExDFt8g2cMEUP53bc3kgQAp5kHN5SbFu//sH3cKM=
tiger	8VwWuZ+C2CAXZ9OoQf9AhJyKG4Ev+/0uOT0raqZoKm4=
Welcome11	KSrSJmclTjYPWxbPEafFNM8b4XqbNB+py75CRt9jMvY=
P@ssword	KO+2jculB+zRgr6tMeTi0Vmw+

查找 KSrSJmclTjYPWxbPEafFNM8b4XqbNB+py75CRt9jMvY= ： Welcome11
查找 SdFikdclTjYPWxbPEafFNM8b4Xq62597p+75CRt9jMv= ： 失败
查找 KO+2jculB+zRgr6tMeTi0Vmw+RhYYdHr/mChLfsxAwA= ： P@ssword
…
… …

查表有效在于：
口令长度有限，口令字典及表开销有限（计算可行）
相同口令对应相同hash

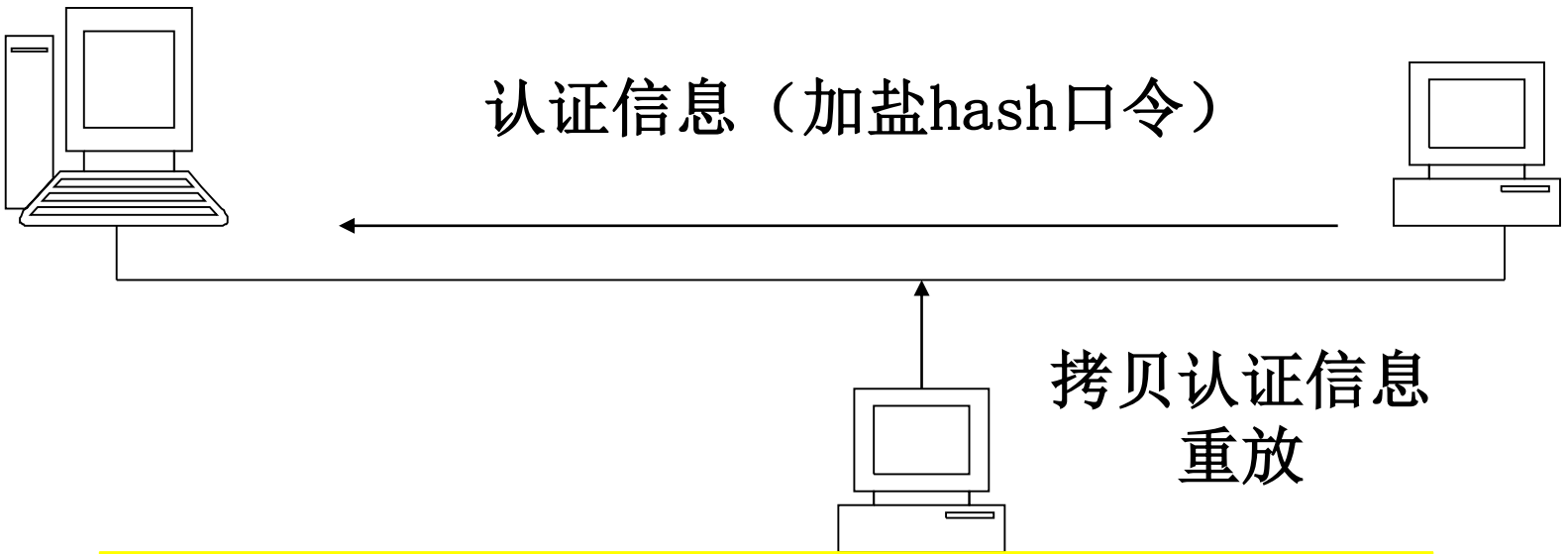
口令机制：加盐Hash口令表



username	Password hash value	Salt
aaron2012	Rnok/wQJybbY5gfKLQH51bmxdLBccQRZe3cdh2CHBf0=	730379C8-C278-400B-B2B8-60EB295F913D
dasiy-abbey	g0G71whWC+OuT1ZrU9Gb0RXpv30ln7ZJFhmkRay4J58=	77B64DE2-A859-4F31-90A1-958DC6A4B16F
dale-001	diwvldcollAj9tXDTGyvBBpR0/XQMNt6lpXRaC2R0fs=	84D003CB-5BE0-47BB-9107-4FFC289700E4

伪加长口令：
口令本身没加长，仅加长认证报文中口令长度
构造PW+R哈希表困难（表大，R随机）

加盐hash口令机制攻击

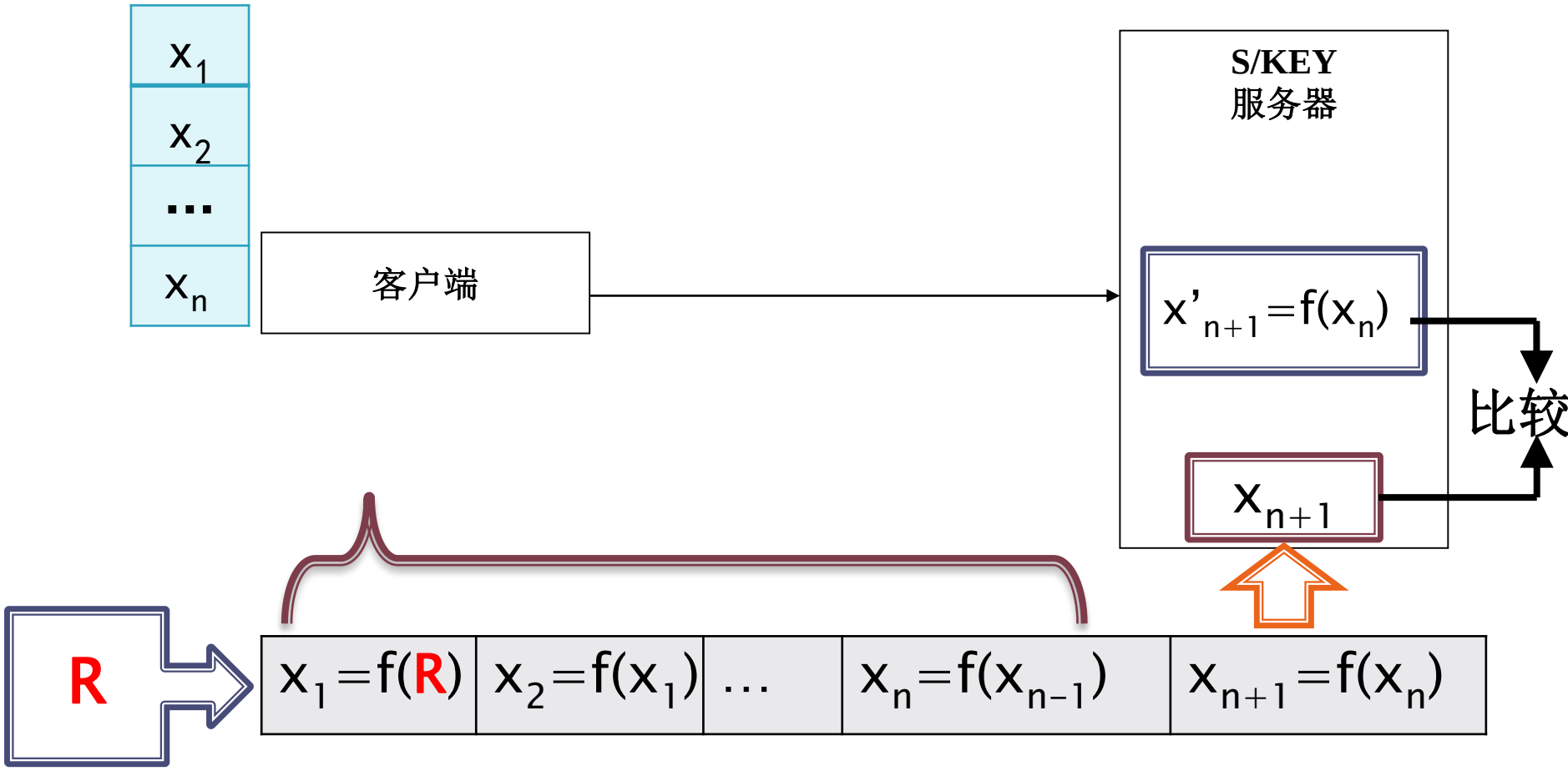


- 面临威胁：
  - ✓ 重放攻击

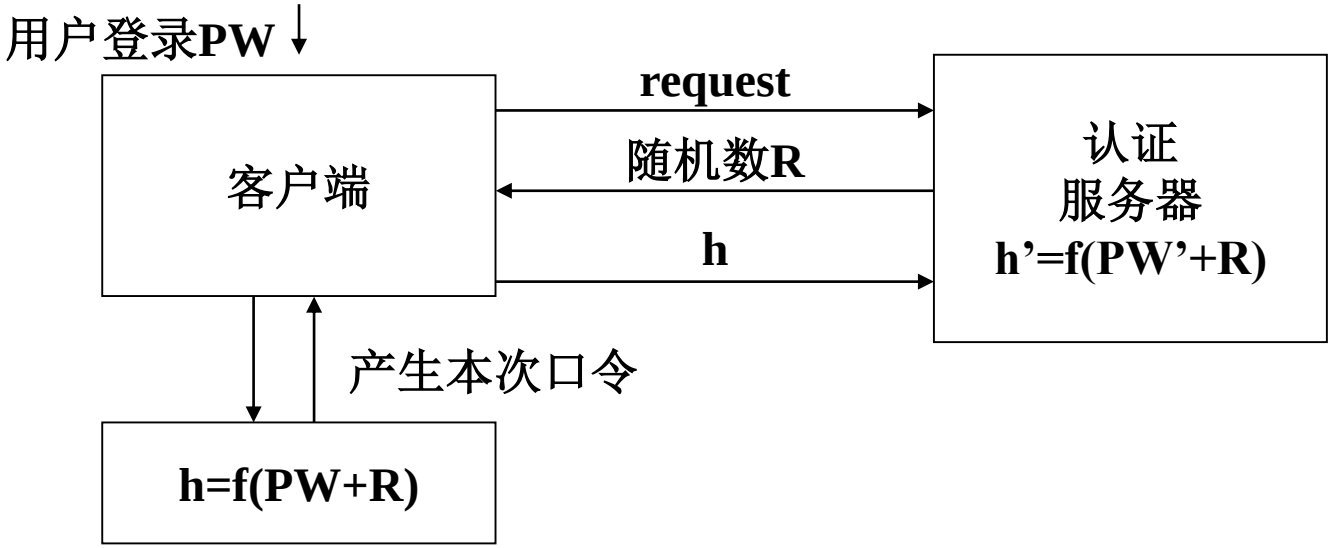
对抗重放攻击——一次性口令

- 在登录过程中加入不确定因素，使每次登录过程中传送的信息都不相同
- 不确定口令的方法：
  - 口令序列
  - 挑战/回答
  - 时间戳

口令序列S/KEY



挑战/回答



- 类似加盐，但每次认证盐不同

CAPTCHA

- Completely Automated Public Turing Test to Tell Computers and Humans Apart (全自动区分计算机和人类的图灵测试)
- 防止计算机自动化口令猜测

一次性随机数
CAPTCHA

时间戳

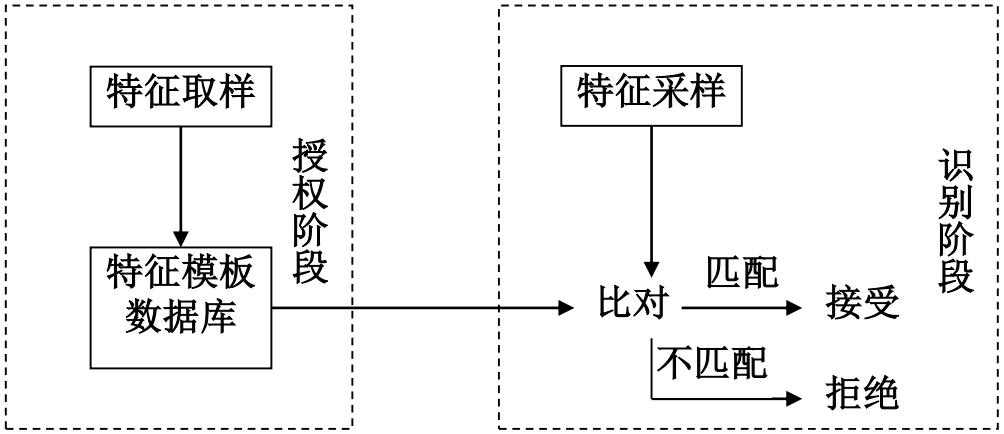
- 以用户登录时间作为随机因素，如：
  - 用户计算，登录口令=hash(用户名＋口令 ＋时间）
  - 系统验证， hash(用户名＋口令 ＋时间）
- 要求双方较高时间同步准确度，一般采取以分钟为时间单位的折中办法。

基于地址的认证机制

- 以声称者地址作为认证基础
- 验证者对每一个主体都保持一份合法呼叫地址文件。
- 自身不能被作为鉴别机制，但可作为其它机制的有用补充。
- Eg. wifi路由器MAC绑定

基于生物特征的认证机制

- 特征：
  - 指纹、声音、手迹、视网膜、手形。
- 优点：
  - 绝对无法仿冒的使用者认证技术。
- 缺点：
  - 较昂贵。
  - 不够稳定(辩识失败率高)。



个人令牌认证机制

- 通常要与一个口令或PIN结合使用
- 令牌要
  - 通常
    杂的能
    支持
    自己的
    网络
- 通常还

6.2基于密码的认证机制

采用对称密码的认证机制

- 基于对称密码加解/密处理构造认证协议
  - 通信双方共享一个对称密钥，作为认证依据
  - 该密钥在询问—应答协议中处理或加密信息交换。

基于对称密码的认证

- 记号
  - A, B 期望进行身份认证的两个用户
  - R_A，R_B ，是A，B产生的随机数
  - T_A，T_B 是A，B产生的时间戳
  - K_{AB} ： 会话密钥

基于对称密码的认证

- ISO/IEC9798-2协议，基于时间戳
  - 单向认证

$$1.\ A \rightarrow B\text{: } \{T_A, B\}_{K_{AB}}$$

  - 双向认证

$$1.\ A \rightarrow B\text{: } \{T_A, B\}_{K_{AB}}$$

$$2.\ B \rightarrow A\text{: } \{T_B, A\}_{K_{AB}}$$

基于对称密码的认证

- ISO/IEC9798-2协议，基于一次性随机数
  - 单向认证

1. $B \rightarrow A: R_B$
2. $A \rightarrow B: \{R_B, B\}_{K_{AB}}$

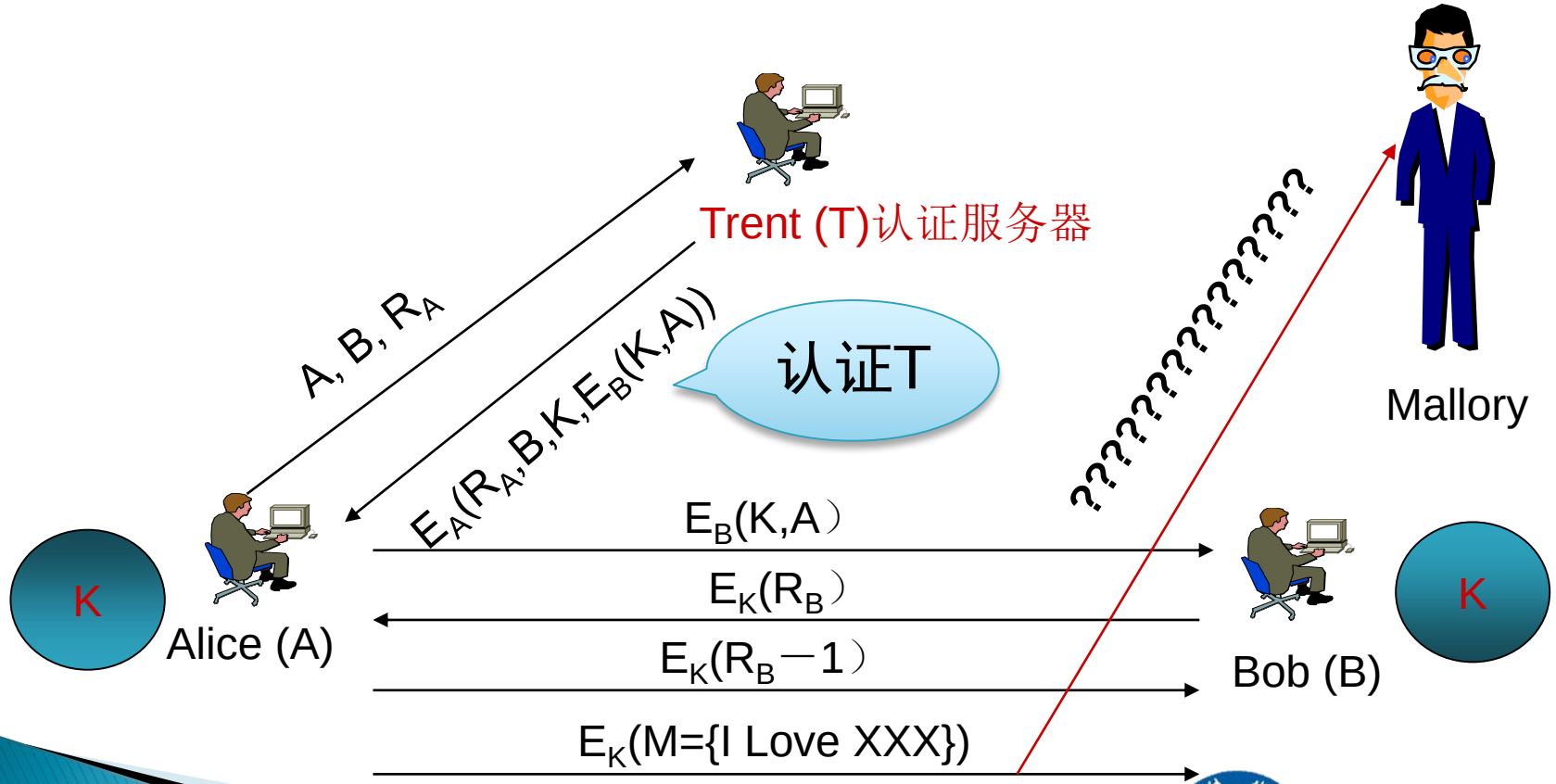
  - 双向认证

1. $B \rightarrow A: R_B$
2. $A \rightarrow B: \{R_A, R_B, B\}_{K_{AB}}$
3. $B \rightarrow A: \{R_A, A\}_{K_{AB}}$

实例——Needham-Schroeder

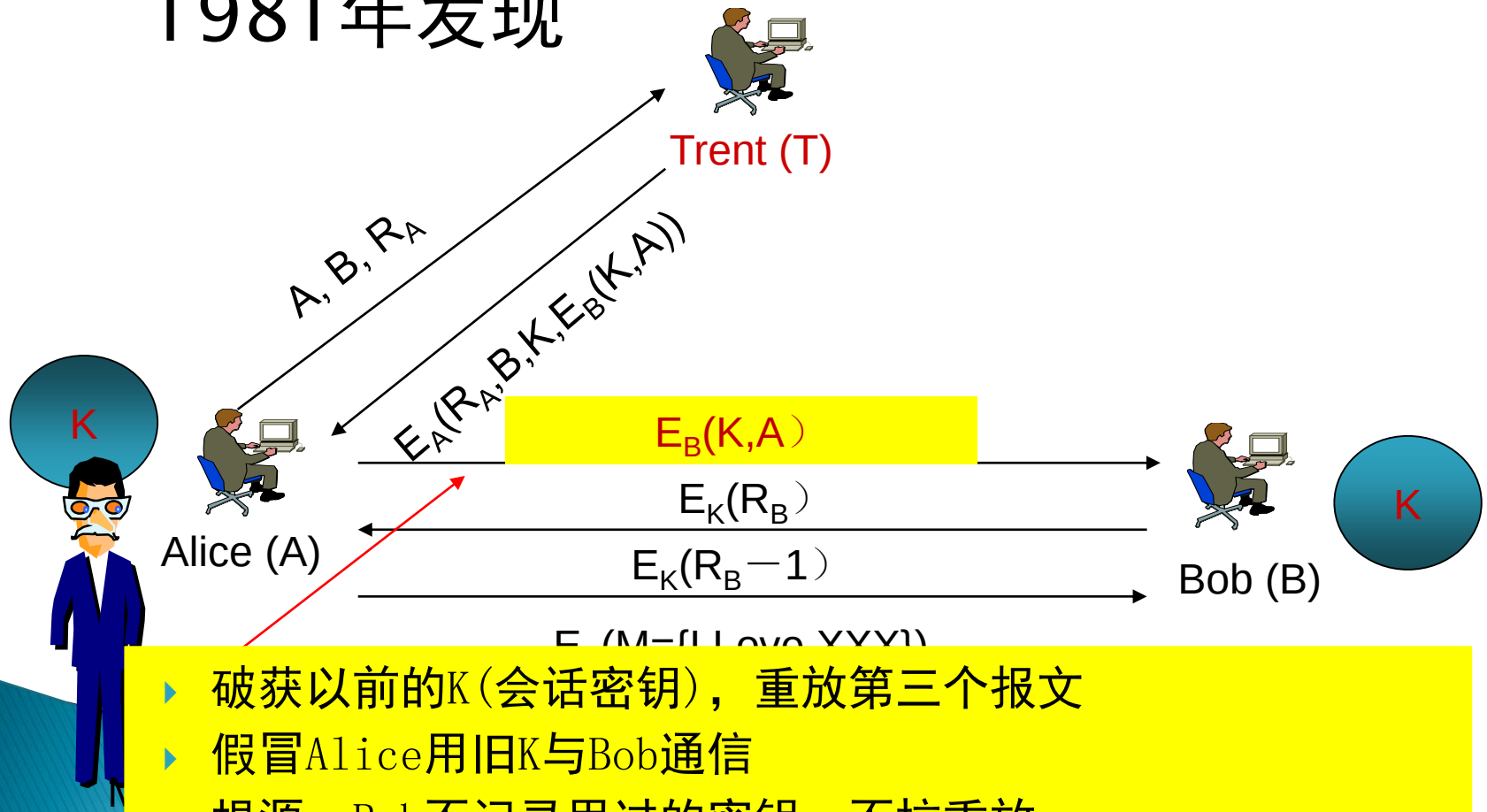
- 早期著名的认证协议，许多广泛使用的认证协议都以其为基础设计。
- 通讯双方互相证实对方身份，并为后续加密通讯建立一个会话密钥。
- 假设：
  - 存在一个可信第三方Trent
  - Trent和Bob共享密钥B
  - Trent和Alice共享密钥A
  - $E_X(M)$表示用密钥X对消息加密
- 记号：
  - K：Trent为Bob和Alice产生的会话密钥

Needham－Schroeder协议



针对Needham-Schroeder的攻击

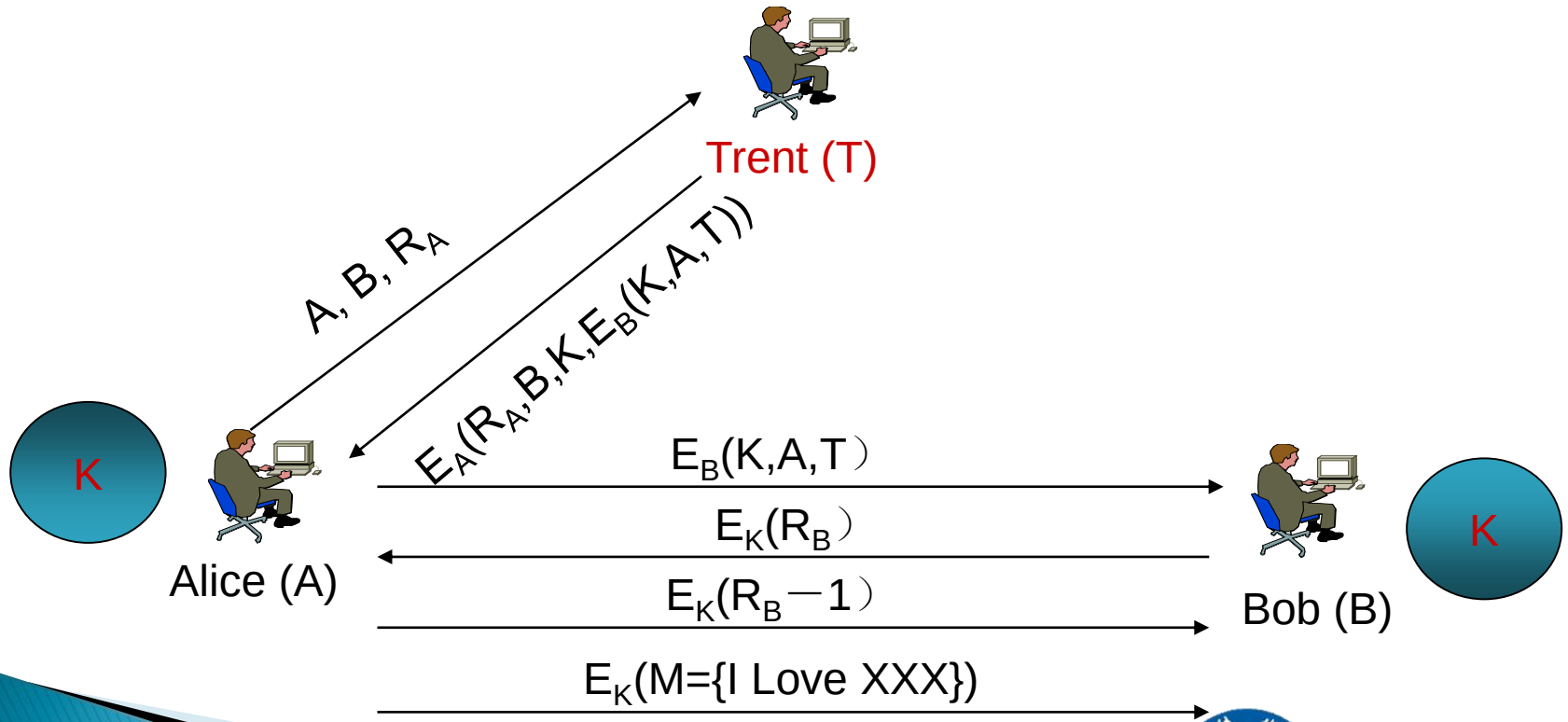
- 旧的会话密钥仍有用-Denning和Sacco在1981年发现



- 破获以前的K(会话密钥)，重放第三个报文
- 假冒Alice用旧K与Bob通信
- 根源：Bob不记录用过的密钥，不抗重放

Needham－Schroeder协议补充方案

- 旧的会话密钥仍有用－解决方案：时戳



采用公开密码算法的机制

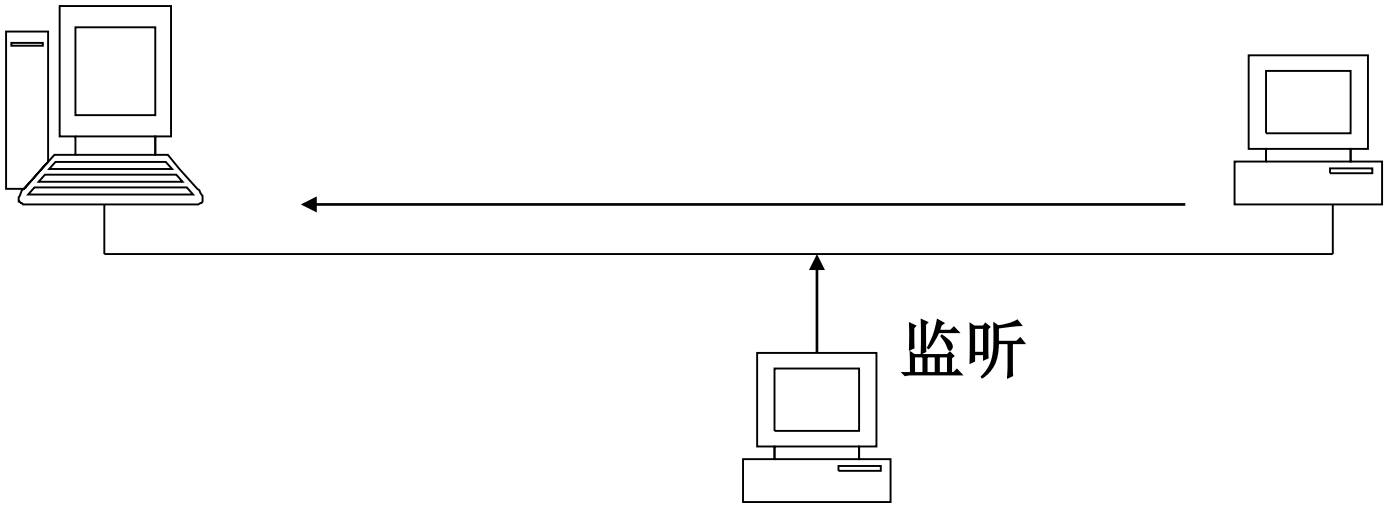
- 1、签名：声称者用声称（他拥有）的签名密钥（私钥）来证实身份。
  - 使用签名密钥签署某消息，签名包含一非重复值以抵抗重放攻击。
  - 验证者用声称者的有效公钥（公钥证书）验证身份。
- 2、加密：声称者用其私钥解密信息来证实身份
  - 验证者用声称者公钥加密信息
  - 声称者用私钥解密信息

基于公钥密码的认证

- 记号

$E_X(M)$	用X 的公钥加密M
$Sig_X(M)$	X 对 M 做的签名
R_X	X 产生的随机数
T_X	X 选择的时间戳
$\{M\}_K$	用对称密钥K 对消息 M 加密

温故而知新——口令机制攻击

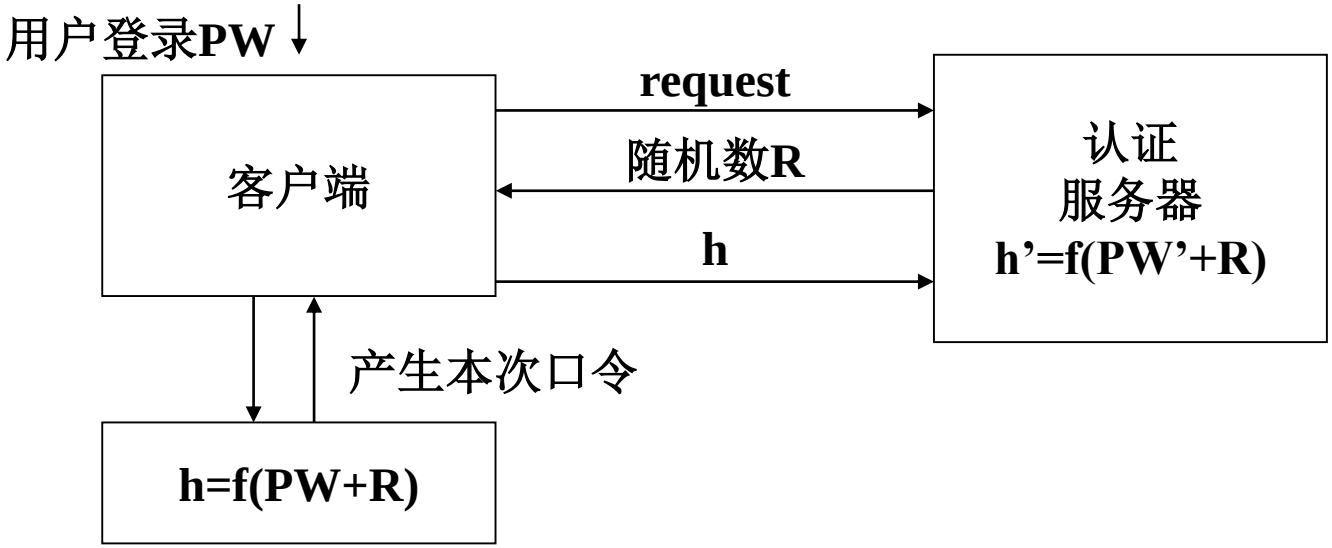


- 监听解析口令
- 获取口令文件
- 重放攻击

加盐

一次性口令

温故而知新——挑战/回答



- 类似加盐，但每次认证盐不同

温故而知新——采用对称密码认证机制

- 基于对称密码加解/密处理构造认证协议
  - 通信双方共享一个对称密钥，作为认证依据
  - 该密钥在询问—应答协议中处理或加密信息交换。

温故而知新——采用公开密码算法机制

- 1、签名：声称者用声称（他拥有）的签名密钥（私钥）来证实身份。
  - 使用签名密钥签署某消息，签名包含一非重复值以抵抗重放攻击。
  - 验证者用声称者的有效公钥（公钥证书）验证身份。
- 2、加密：声称者用其私钥解密信息来证实身份
  - 验证者用声称者公钥加密信息
  - 声称者用私钥解密信息

基于公钥密码的认证

- ISO/IEC 9798-3 协议，基于时间戳
  - 单向认证

1. $A \rightarrow B$: T_A , B, $Sig_A(T_A, B)$

  - 双向认证

1. $A \rightarrow B$: T_A , B, $Sig_A(T_A, B)$

2. $B \rightarrow A$: T_B , A, $Sig_B(T_B, A)$

基于公钥密码的认证

- ISO/IEC 9798-3协议，基于一次性随机数
  - 单向认证

1. $B \rightarrow A$: R_B

2. $A \rightarrow B$: $R_B, B, Sig_A(R_B, B)$

  - 双向认证

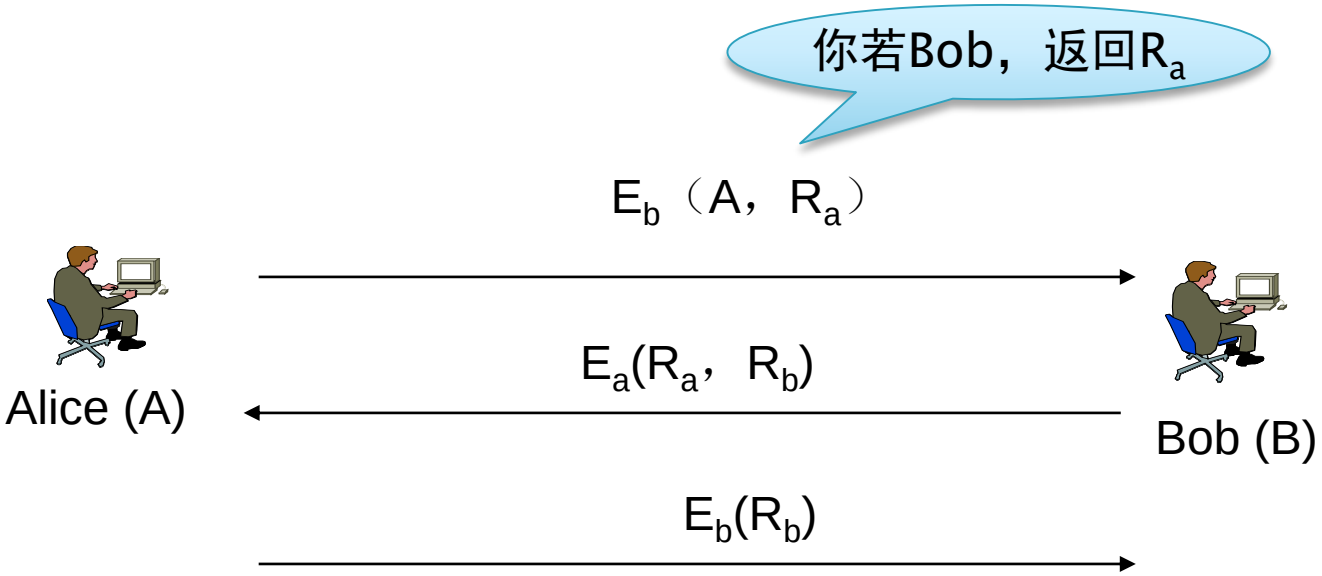
1. $B \rightarrow A$: R_B

2. $A \rightarrow B$: $R_A, R_B, B, Sig_A(R_A, R_B, B)$

3. $B \rightarrow A$: $R_A, A, Sig_B(R_A, A)$

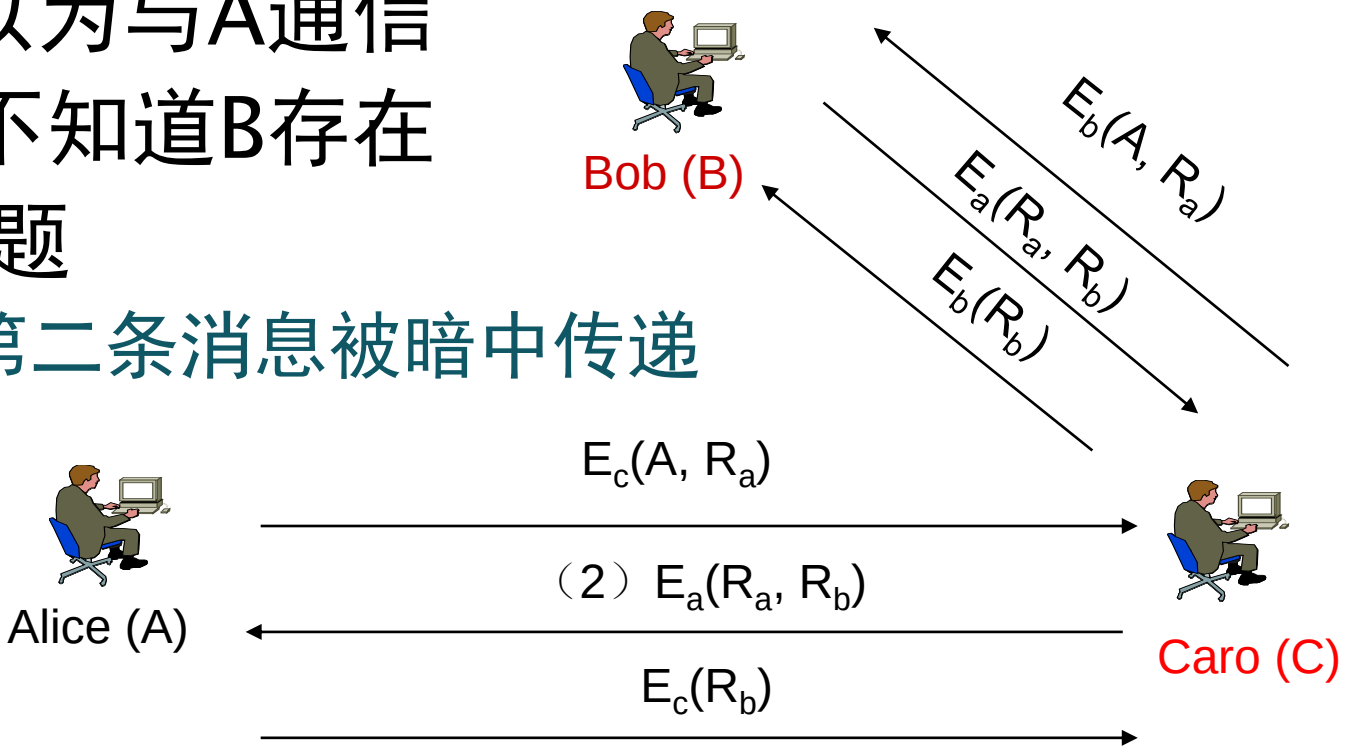
Needham－Schroeder（公钥方案）

- Alice和Bob事先获取对方公钥，通过加解密进行认证。
- 只认证，不交换会话密钥



Needham－Schroeder（公钥方案）

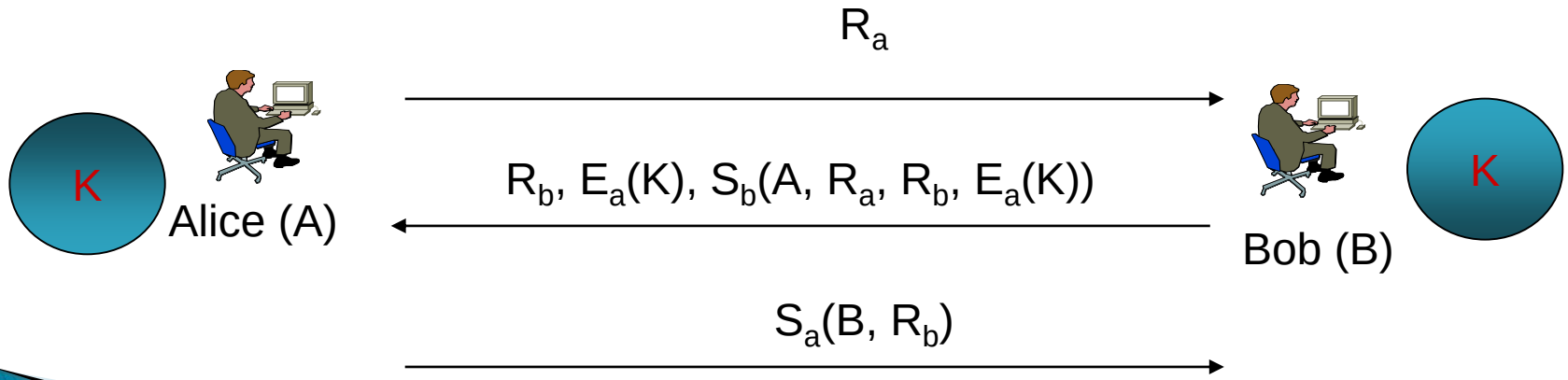
- A与C通信，C假冒A与B通信
- B以为与A通信
- A不知道B存在
- 问题
  - 第二条消息被暗中传递



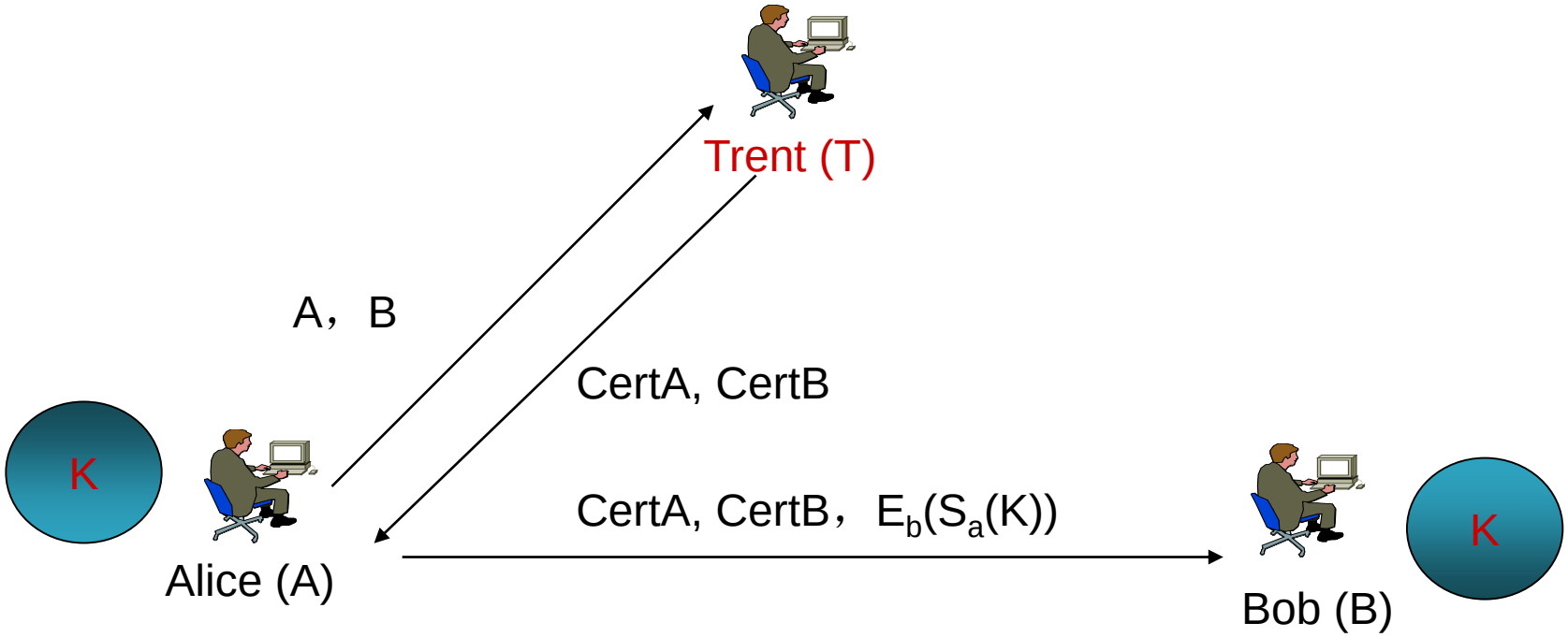
在第二条消息中增加发送方标识阻止这种攻击

$E_a(R_a, C/B, R_b)$

Needham－Schroeder（签名方案＋密钥交换）



Denning-Sacco方案（公钥体制＋可信第三方）

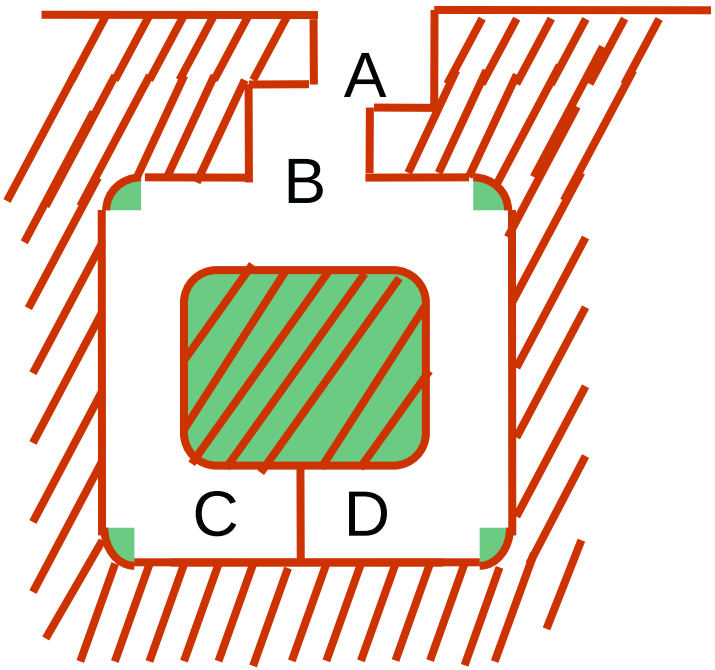


零知识身份认证

- 身份认证通常要求口令或身份信息。
- 零知识证明
  - 信息拥有者不泄露任何信息能向验证者证明它拥有该信息。

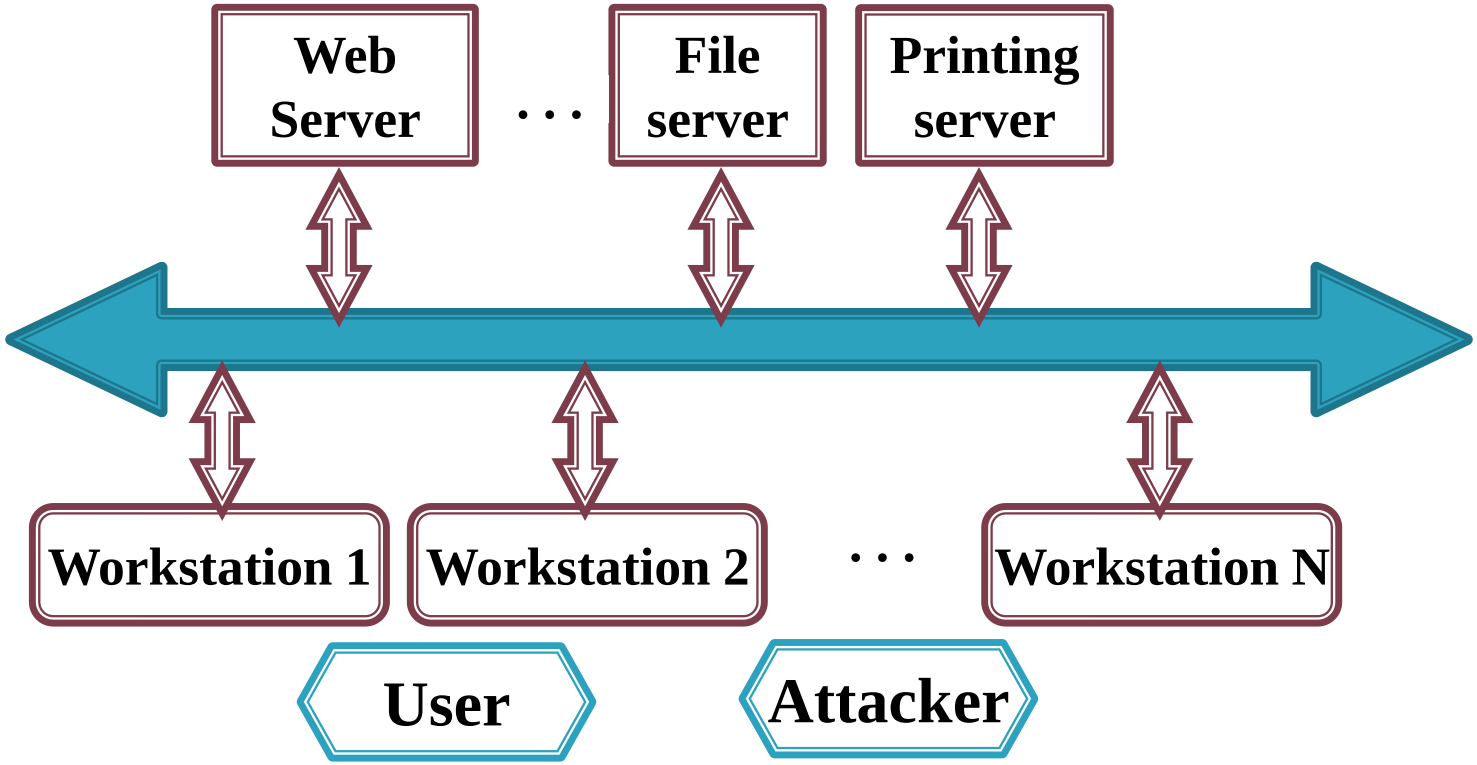
零知识身份认证——洞穴例子

- C和D间有道密门，需咒语打开；
- Peggy向Victor证明她知道咒语，但不想泄露咒语。



6.3 Kerberos 认证协议

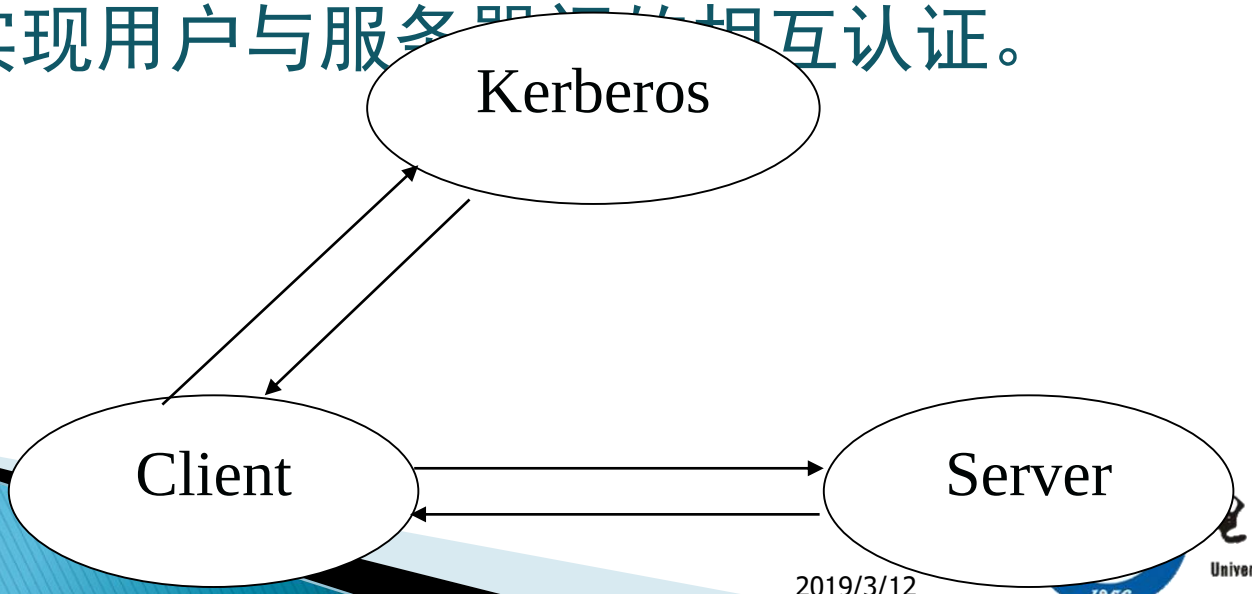
Kerberos的产生背景



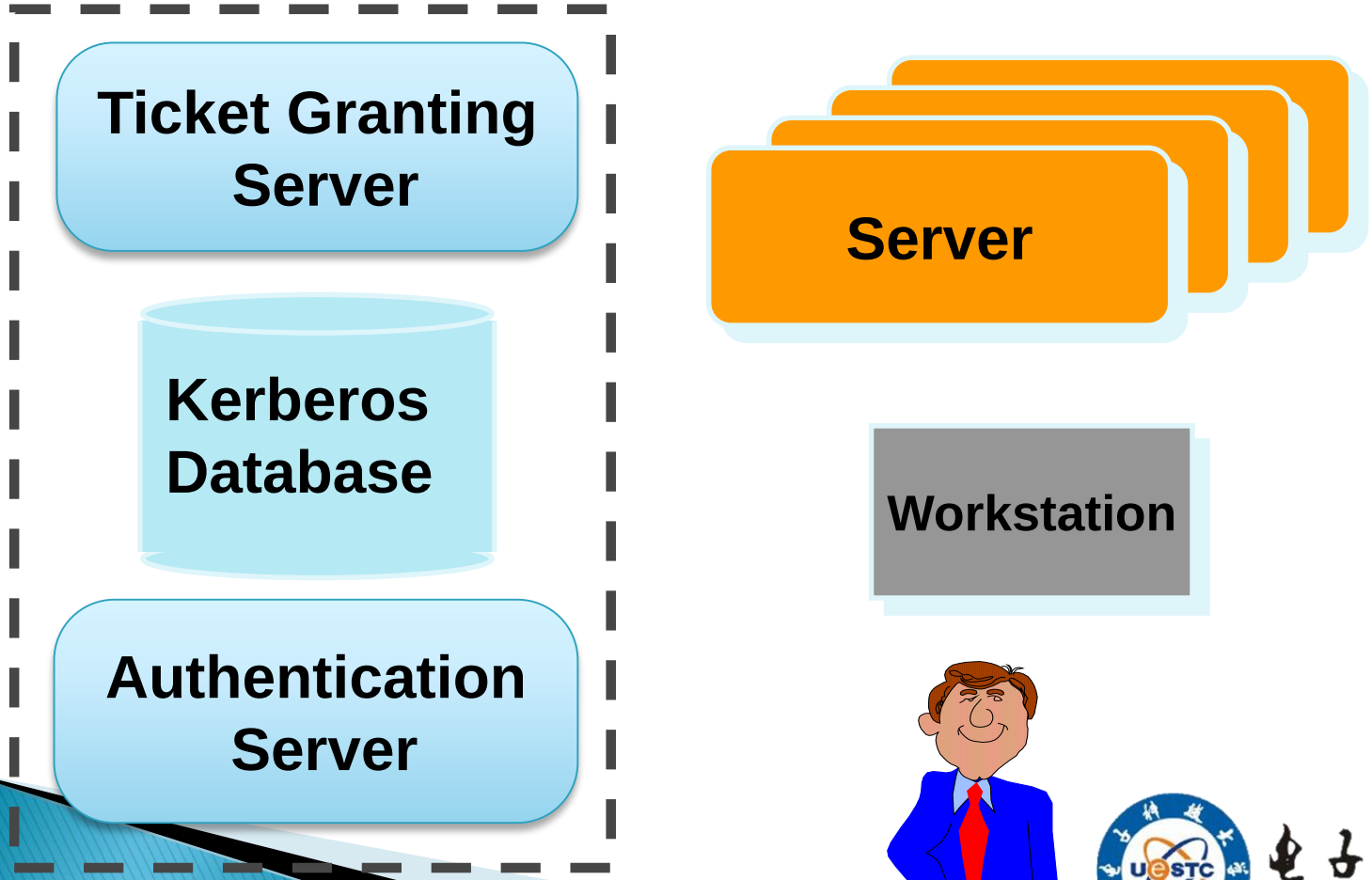
- 用户需要从多台服务器得到服务，访问安全方法有三种：
  - 1) 登录工作站认证用户；
  - 2) 服务器认证请求工作站；
  - 3) 服务认证用户。

Kerberos简介

- 基于对称密码技术**集中式**的身份认证框架结构。
  - 使用DES加密算法
  - 引入可信第三方
  - 采用基于Needham-Schroeder协议
  - 实现用户与服务器间的相互认证。



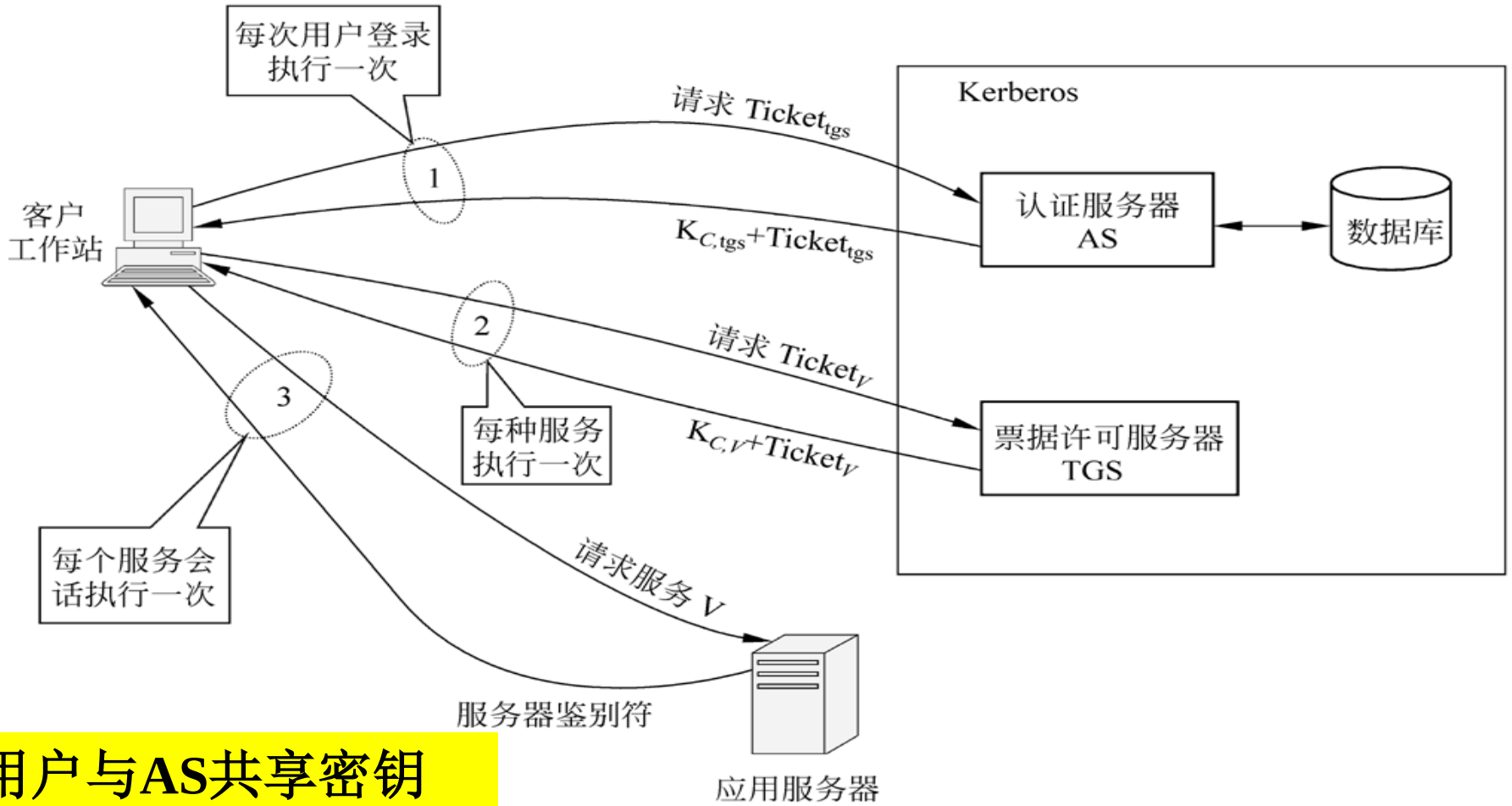
Kerberos认证系统模型



Kerberos设计思路（续）

- 解决办法
  - 使用票据，介绍信
  - 引入票据许可服务器TGS(ticket-granting server)
  - AS不直接向客户发放访问应用服务器的票据，而由TGS向客户发放
- 票据重用（两种票据）
  - 票据许可票据$Ticket_{tgs}$（Ticket granting ticket）
    - 客户访问 TGS的票据
    - 由 AS 发放，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TicketV （Service granting ticket）
    - 客户访问应用时的票据；
    - 由TGS发放

Kerberos（V4）协议交互过程



用户与AS共享密钥

$Ticket_{tgs} = E_{Ktgs}\{K_{C,tgs}, ID_C, AD_C, ID_{tgs}, TS_2, LT_2\}$

$Ticket_V = E_{KV}\{K_{C,V}, ID_C, AD_C, ID_V, TS_4, LT_4\}$