什么是密码系统

- 密码系统中的基本概念

明文（Plain text）：作为加密输入的原始信息

密文（Cipher text）:明文经加密变换后得到的结果

密钥（Key）：参与密码变换的参数

加密算法（Encryption）：将明文变为密文的变换函数

解密算法（Decryption）：将密文恢复为明文的变换函数

什么是密码系统

❖ 密码系统中的基本概念

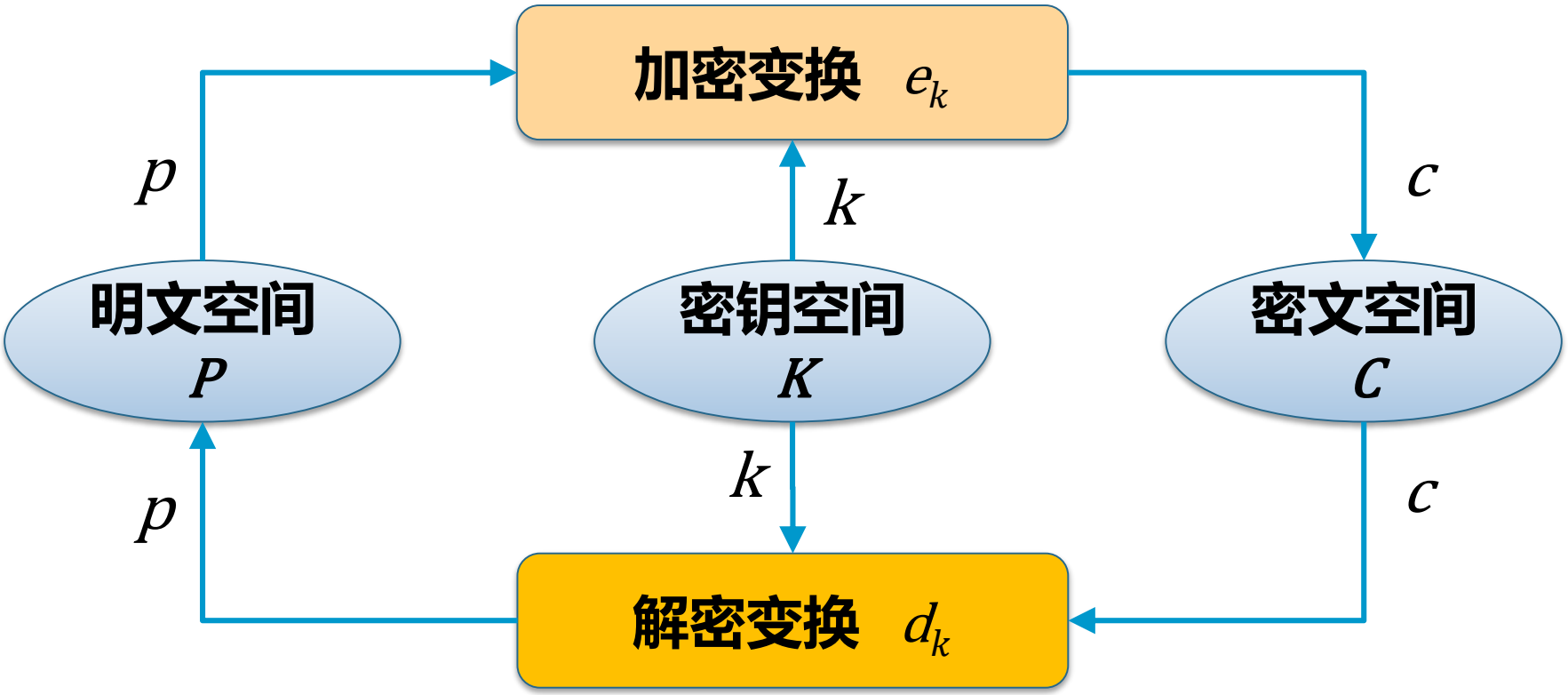
	表示符号	有限集表示符号
明文	p	P
密文	c	C
密钥	k	K
加密算法	$c=e_k(p)$	
解密算法	$p=d_k(c)$	

对于有实用意义的密码算法来说，总是有

$$p = d_k\left(e_k(p)\right)$$

什么是密码系统

❖ 密码系统的基本架构



什么是密码系统

❖对称密码体制的优缺点

- 优点
  - 加解密处理速度快
- 缺点
  - 系统的安全性依赖于密钥的安全分配方案
  - 多人通信时密钥数量急剧增加
  - 通信双方在通信前必须统一密钥
  - 存在抵赖和身份认证等问题

什么是密码系统

- **非对称密码体制的优缺点**
  - 优点
    - 公私密钥对易于管理
    - 密钥分配无需安全信道
    - 可以实现签名和认证
  - 缺点
    - 加解密速度很慢

密码系统的安全性

❖密码系统的安全性

对手知道加密/解密算法!

-荷兰密码学家Kerckhoof《军事密码学》 1883年

在考察密码算法的安全性时，可以将破译算法分为不同级别:

- 全部破译（Total break）：找出密钥
- 全部推导（Global deduction）：找出替代算法
- 实例推导（Instance deduction）：找出明文
- 信息推导（Information deduction）:获取有关密钥或明文的信息

密码系统的安全性

❖ 密码系统的攻击方法

- 唯密文攻击（Ciphertext-only Attack)

密码分析者仅仅通过密文找出所使用的密钥和对应的明文；

- 已知明文攻击（Known-plaintext Attack)

密码分析者获取了一定数量的明文-密文对，试图找到对应的密钥；

- 选择明文攻击（Chosen-plaintext Attack)

密码分析者能够根据自己的要求选择明文，并加密得到对应的密文，从而获得明文-密文对，在此基础上试图找到对应的密钥；

- 选择文本攻击（Chosen-text Attack)

密码分析者已知加密算法，并随意选择明文和密文，进行加解密分析，在此基础上试图找到密钥；

密码系统的安全性

❖ 密码系统的安全性

密码系统安全性评价指标：

- **计算安全性**：使用最好破译算法攻破一个密码系统所需的操作次数；
- **可证明安全性**：将密码算法的安全性归结于一个NP难数学难题，证明两者的求解是等价的。
- **无条件安全性**：在不限制计算资源和时间的条件下依然无法破解密码算法

NP难题：需要在多项式时间内求解的问题

几种简单的古典密码体制

❖移位密码

模运算：假设a和b均为整数，m是一正整数。若m整除b-a，则可将其表示为a≡b(mod m)。式a≡b(mod m)读作“a与b模m同余”，正整数m称为模数。

移位密码的数学描述：

令$\mathcal{P} = \mathcal{C} = \mathcal{K} = \mathbb{Z}_{26}$。对$0 \le K \le 25$，任意$x, y \in \mathbb{Z}_{26}$，定义

$$e_K(x) = (x + K) mod 26$$

和

$$d_K(y) = (y - K) mod 26$$

注：当K取3时，移位密码被称之为凯撒密码，因其首先被儒勒 凯撒使用。

几种简单的古典密码体制

❖代换密码的数学描述

令$\mathcal{P} = \mathcal{C} = \mathbb{Z}_{26}$，$\mathcal{K}$是由26个数字0,1,⋯,25的所有可能的置换组成，对任意的置换$\pi \in \mathcal{K}$，定义

$$e_\pi(x) = \pi(x)$$

和

$$d_\pi(y) = \pi^{-1}(y)$$

这里π^{-1}表示置换π的逆置换。

几种简单的古典密码体制

❖仿射密码的数学描述

令$\mathcal{P} = \mathcal{C} = \mathbb{Z}_{26}$，且

$$\mathcal{K} = \{(a, b) \in \mathbb{Z}^{26} \times \mathbb{Z}^{26} : \gcd(a, 26) = 1\}$$

对任意的$K = (a, b) \in \mathcal{K}$，$x, y \in \mathbb{Z}_{26}$，定义

$$e_K(x) = (ax + b) mod 26$$

和

$$d_K(y) = a^{-1}(y - b) mod 26$$

几种简单的古典密码体制

❖ 维吉尼亚密码

设m是一个正整数，令$\mathcal{P} = \mathcal{C} = \mathcal{K} = (\mathbb{Z}_{26})^m$，对任意的密钥$K = (k_1, k_2, \ldots, k_m)$，定义

$$e_K(x_1, x_2, \ldots, x_m) = (x_1 + k_1, x_2 + k_2, \ldots, x_m + k_m)$$

和

$$d_K(y_1, y_2, \ldots, y_m) = (y_1 - k_1, y_2 - k_2, \ldots, y_m - k_m)$$

几种简单的古典密码体制

❖希尔密码

设$m \geq 2$为正整数，$\mathcal{P} = \mathcal{C} = (\mathbb{Z}_{26})^m$ ，且

$$\mathcal{K} = \{\text{定义在}\mathbb{Z}_{26}\text{上的}m \times m\text{可逆矩阵}\}$$

对任意的密钥$\boldsymbol{K}$，定义：

$$e_K(x) = x\boldsymbol{K}$$

和

$$d_K(y) = y\boldsymbol{K}^{-1}$$

以上运算都是在$\mathbb{Z}_{26}$上进行。

希尔密码也是一种多表代换密码

几种简单的古典密码体制

❖置换密码的数学描述

令m为一正整数，$\mathcal{P}=\mathcal{C}=(\mathbb{Z}_{26})^m$，$\mathcal{K}$是由所有定义在集合$\{1,2,\ldots,m\}$上的置换组成。对任意的密钥（即**置换**）$\pi$，定义：

$$e_\pi(x_1,x_2,\ldots,x_m)=\left(x_{\pi(1)},x_{\pi(2)},\ldots,x_{\pi(m)}\right)$$

和

$$d_\pi(y_1,y_2,\ldots,y_m)=\left(y_{\pi^{-1}(1)},y_{\pi^{-1}(2)},\ldots,y_{\pi^{-1}(m)}\right)$$

其中，π^{-1}为置换π的逆置换。

分组密码的基本概念

分组密码实质上是字长为m的数字序列的**代换密码**。

分组密码的设计问题在于找到一种算法，能在密钥控制下从一个足够大且足够好的置换子集中，简单而迅速地选出一个置换，用来对当前输入的明文的数字组进行加密变换。

分组密码的基本概念

为了保证分组密码算法的安全性，对密码算法有如下要求：

- 分组长度n要足够大：

  防止明文穷举攻击法奏效。

- 密钥量要足够大：

  尽可能消除弱密钥并使所有密钥同等地好，以防止密钥穷举攻击奏效。

- 由密钥确定置换的算法要足够复杂：

  充分实现明文与密钥的扩散和混淆，没有简单的关系可循，要能抗击各种已知的攻击。

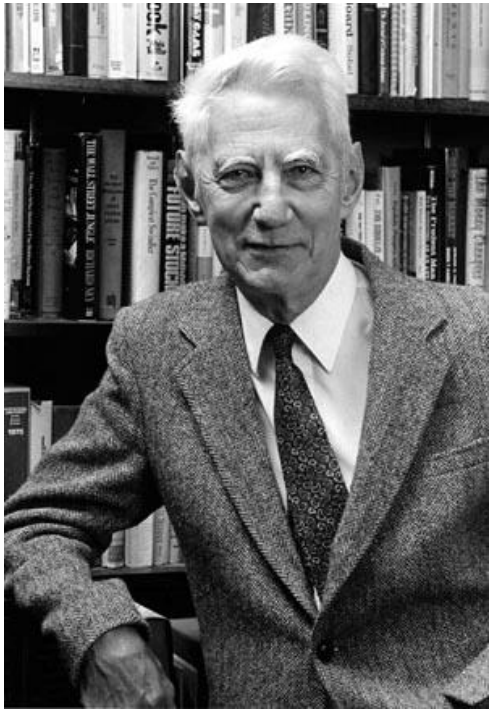
分组密码的基本概念

❖分组密码的设计基本原则

扩散（Diffusion）与混乱（Confusion）原则

扩散：重新排列消息中的每一个比特，以使明文中任何冗余度能扩散到整个密文，将每一比特明文的影响尽可能作用到较多的输出密文位中，或密文的每一比特都要取决于部分或者全部明文位。

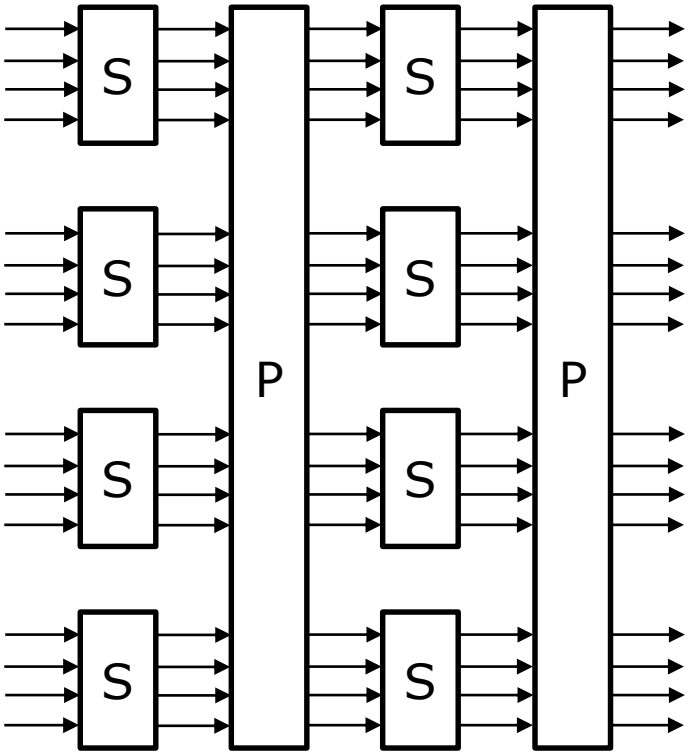
混乱：使密文和密钥之间的统计特征关系尽可能地复杂化，其目的在于隐藏密文和密钥之间的关系，以利于组织攻击者利用密文来找出密钥



Claude Shannon

分组密码的基本概念

- 通过交替使用S盒和P盒，可以构成分组密码的一种基本结构-SP网络



雪崩效应：输入（明文或者密钥）的微小变化会导致输出发生巨大变化的现象！（输入中任意一位的变化，导致任意输出位变化的概率是1/2)

分组密码的基本概念

❖ 轮函数F

- 轮函数F的作用是是对消息序列进行**混乱操作**。为了保证混淆操作的安全性，F必须是非线性的，非线性越强，安全性越高。

- 因此，轮函数F的核心是**S盒**。

分组密码的工作模式

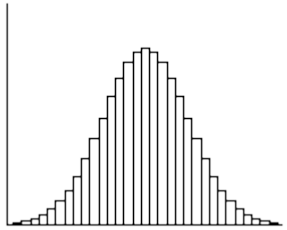
- 1980年12月，美国国家标准局发布了分组密码的的四种工作模式标准（FIPS Publication 81），这四种工作模式可以应用于任何分组密码。
  - 电码本模式（ECB模式）
  - 密码分组链接模式（CBC模式）
  - 输出反馈模式（OFB模式）
  - 密码反馈模式（CFB模式）

分组密码的工作模式

- 电码本模式（ECB模式）
  - 这种模式就是直接使用分组密码：给定一个明文分组序列$x_1 x_2$ …，每一个x_i都用同一个密钥K来加密，产生密文$y_1 y_2$ …
  - ECB模式的一个缺点是：相同的明文分组会产生相同的密文分组。如果消息分组选自一个“低熵”的明文空间，安全性则显著降低

$$H(x) = -\sum_i p(x_i) \log_2 p(x_i)$$

变量的不确定性越大，熵也就越大，反正则越小。

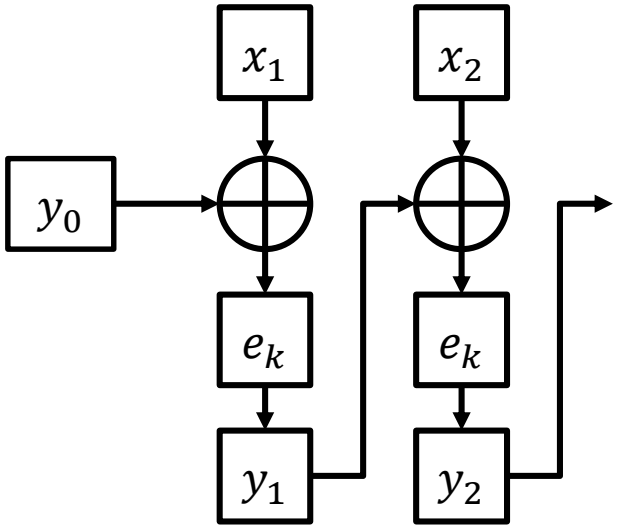


分组密码的工作模式

- 密码分组链接模式（CBC模式）
  - 这种模式下，每一个密文分组在y_i在使用密钥K加密前，都要先跟下一个明文分组x_{i+1}相异或。

这种模式下，如果改变一个明文分组x_i，那么y_i**及其后面的所有密文都会受到影响。**

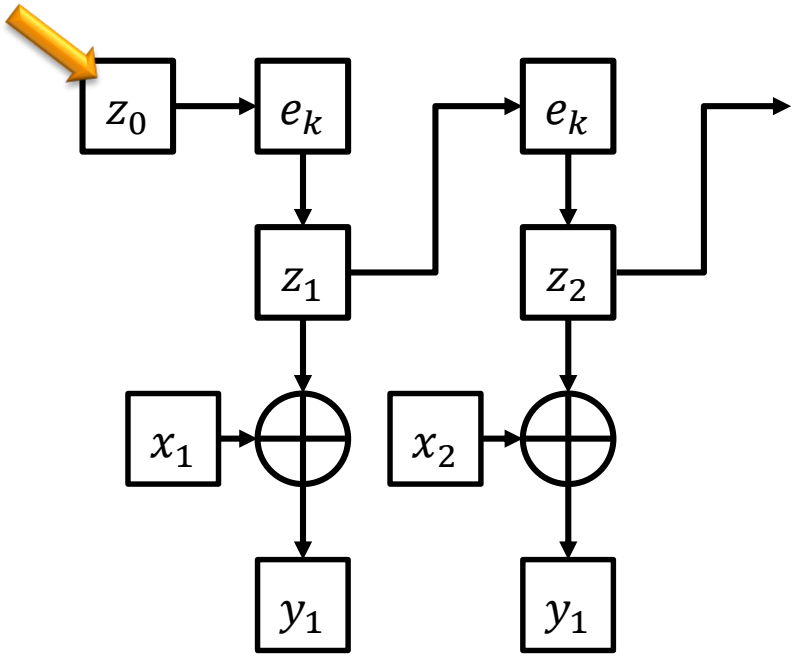
因此，这种模式适合于认证目的。



分组密码的工作模式

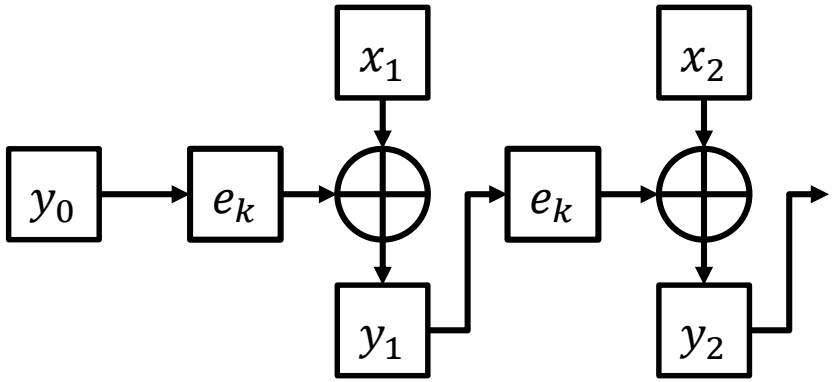
- 输出反馈模式（OFB模式）
  - 这种模式下。利用分组密码算法反复加密一个初始向量，从而生成一个密钥流，将明文与该密钥流进行异或运算实现加解密。

OFB模式实际上就是一个同步流密码。



分组密码的工作模式

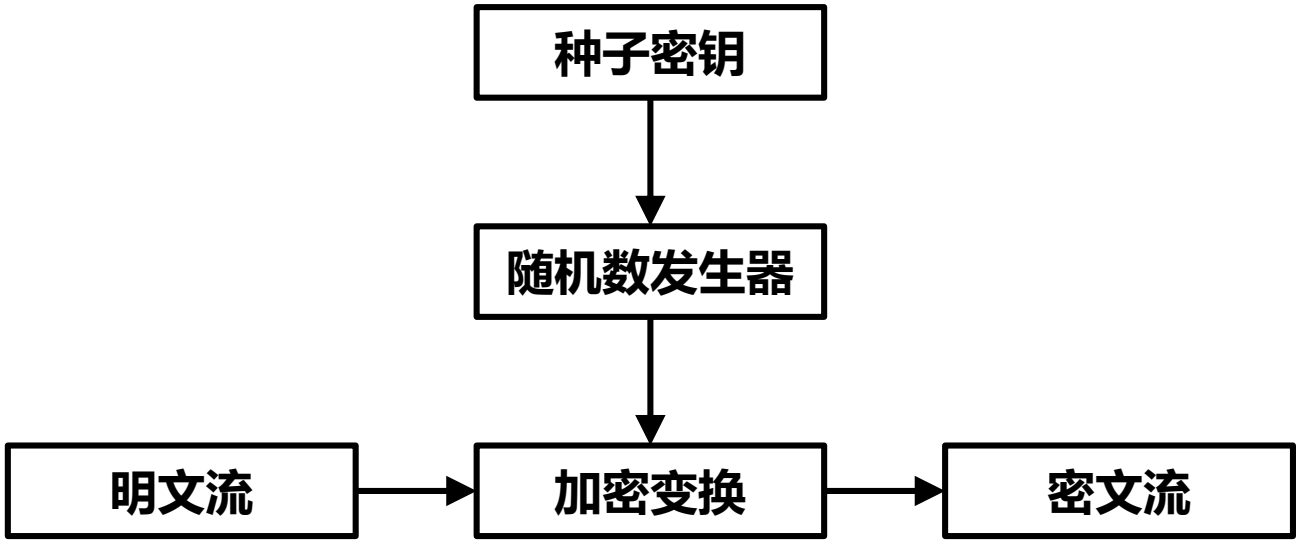
- 密码反馈模式（CFB模式）
  - 在CFB模式中，也产生一个密钥流，与OFB模式的差别在于使用前一步加密生成的密文产生密钥流元素。



序列密码的基本概念

❖ 序列密码的设计思想

序列密码的主要原理是通过随机数发生器产生性能优良的伪随机序列（密钥流），使用该序列加密信息流（逐比特加密），得到密文序列。

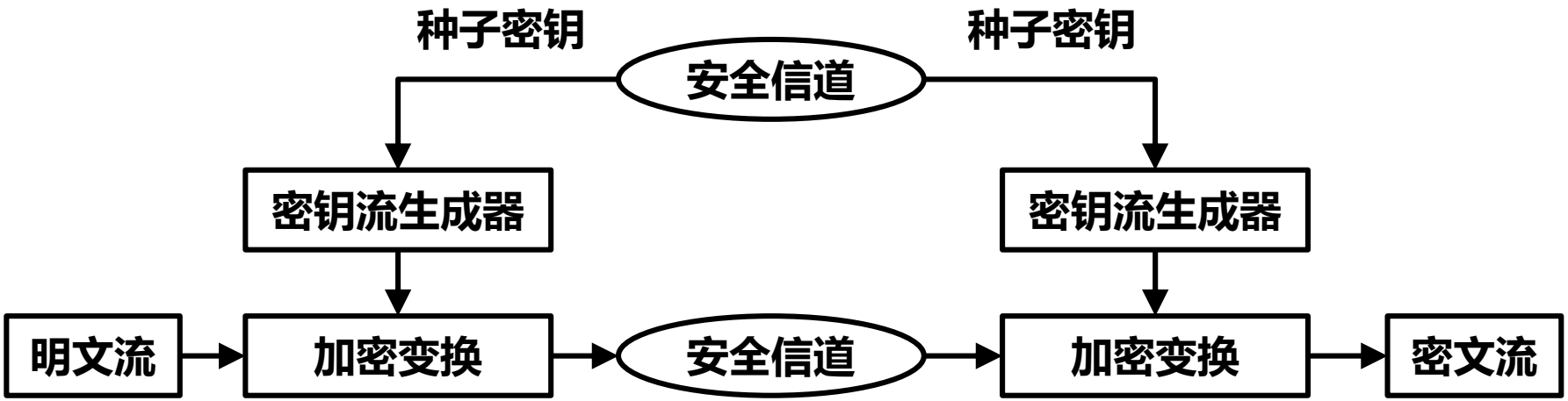


序列密码的基本概念

- **按照加解密过程中密钥序列工作方式的不同，序列密码可以分为以下两种类型：**
  - 同步序列密码SSC（Synchronous Stream Cipher）
  - 自同步序列密码SSSC（Self-Synchronous Stream Cipher）

序列密码的基本概念

- 同步序列密码SSC
  - 在同步序列密码中，如果密钥流的产生完全独立于消息流。



序列密码的基本概念

- **同步序列密码SSC的特点**
  - **（1）同步性**

    在加解密的过程中，消息发送方和接收方必须同步才能做到正确的加解密。如果密文字符在传播的过程中被插入或者删除破坏了同步性，那么解密的过程将失败。
  - **（2）无错误传播性**

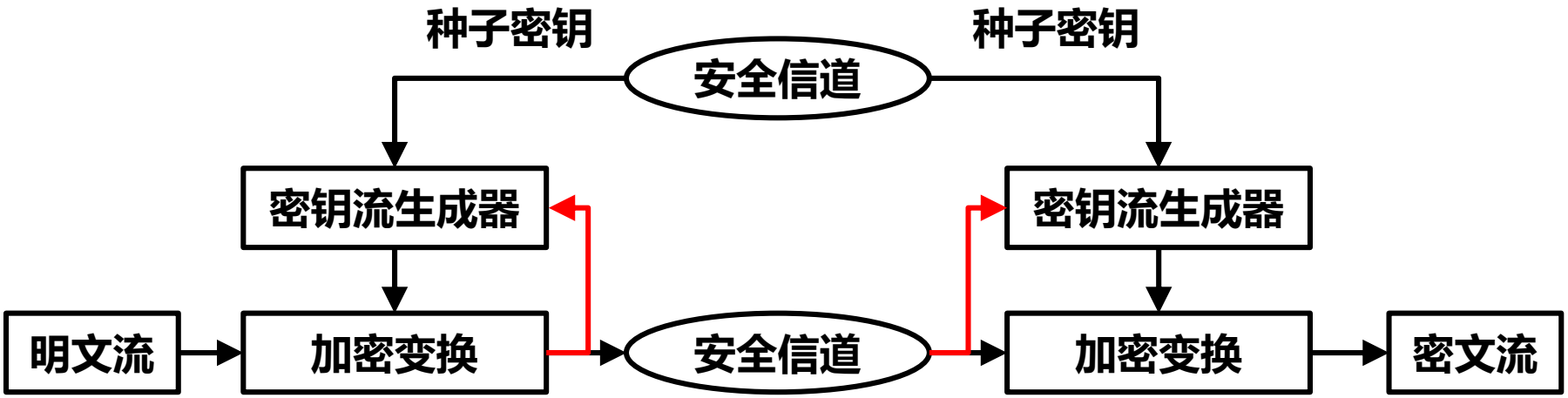
    密文字符在传输的过程中被修改（未被删除）并不影响其他密文字符的解密。
  - **（3）抗主动攻击性**

    一旦主动攻击者对密文字符进行插入、删除或重放操作都会立即破坏系统的同步性

序列密码的基本概念

- 自同步序列密码SSSC

  与同步序列密码相比，自同步序列密码是一种有记忆变换的密码，每一个密钥字符是由前面n个密文字符参与运算推导出来的。



序列密码的基本概念

- 自同步序列密码SSSC的特点
  - **(1) 自同步性**

    由于解密过程仅仅依赖于前面固定个数的密文字符，当密码的同步性遭到破坏时，可以自动重建解密的过程，仅有固定数量的明文字符不可恢复。

  - **(2) 错误传播有限性**

    在密文序列传输的过程中，当一个比特的密文字符被改动，那么至多会有n个比特随后的密文字符解密出错，然后会恢复正确的解密过称。

序列密码的基本概念

- 自同步序列密码SSSC的特点
  - **(3) 抗主动攻击性**

    一旦主动攻击者对密文字符进行插入、删除或重放操作都会立即破坏系统的同步性。但是与同步序列密码相比，检测的能力有所下降。

  - **(4) 明文统计扩散性**

    每个明文字符都会影响其后的整个密文，即明文的统计学特征被扩散到密文中。因此，自同步序列密码对利用明文冗余度而发起的攻击有较强的抗攻击能力。

序列密码的基本概念

❖ 序列随机性能的评价

序列密码产生的密钥序列的随机性是评价序列密码安全性的重要指标。

Golomb随机性假设：

（1）在序列的一个周期内，0和1的个数至多相差1；

（2）在序列的一个周期内，长为i的游程个数占总游程数的2^i，且在等长的游程中0游程和1游程各占一半。

（3）自相关函数是二值的，即对某个整数K，有

$$N \cdot C(t) = \sum_{i=1}^{N-1} (2s_i - 1) \cdot (2s_{i+t} - 1) = \begin{cases} N, t = 0 \\ K, 1 \le t \le N-1 \end{cases}$$

序列密码的基本概念

- 序列随机性能的评价

Golomb随机性假设的意义：

条件1：说明序列中0和1出现的概率是相等的；

条件2：说明在已知位置n前若干位置值的前提下，在第n个位置上出现0和1的概率是相等的；

条件3：如果将序列中任意两个值进行比较，无法得到关于序列本身的相关信息。

Golomb是最早的随机性检测标准，但是这个标准依然是比较弱的。

反馈移位寄存器算法

- 序列密码的设计要求：

（1）密钥序列具有良好的随机性；

（2）用于生成密钥流的根密钥k应该易于分配和管理；

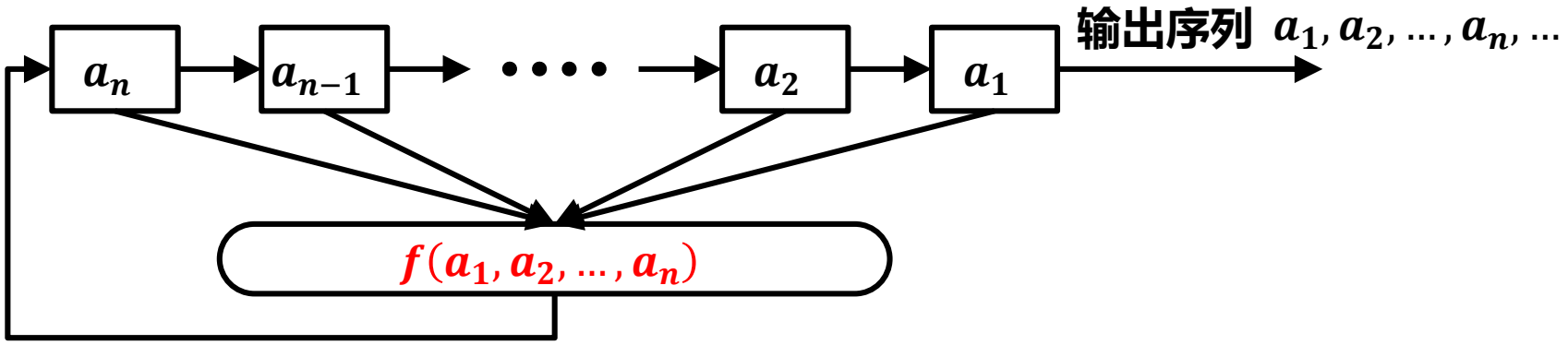
（3）密钥流生成方法应该易于快速实现

反馈移位寄存器算法

一种用于设计密钥流生成器的基本部件——反馈移位寄存器（Feedback Shift Register, FSR）：

（1）线性反馈移位寄存器（Linear Feedback Shift Register, LFSR）；

（2）非线性反馈移位寄存器（Nonlinear Feedback Shift Register, NLFSR）



$$a_{n+1} = f(a_1, a_2, \ldots, a_n)$$

反馈移位寄存器算法

- 线性反馈移位寄存器（Linear Feedback Shift Register, LFSR）

  反馈函数$f(a_1, a_2, \ldots, a_n)$是$a_1, a_2, \ldots, a_n$的线性函数：

  $$f(a_1, a_2, \ldots, a_n) = c_n a_1 \oplus c_{n-1} a_2 \oplus \cdots \oplus c_1 a_n$$

  其中，c_i表示反馈系数，$c_i = \{0, 1\}$。

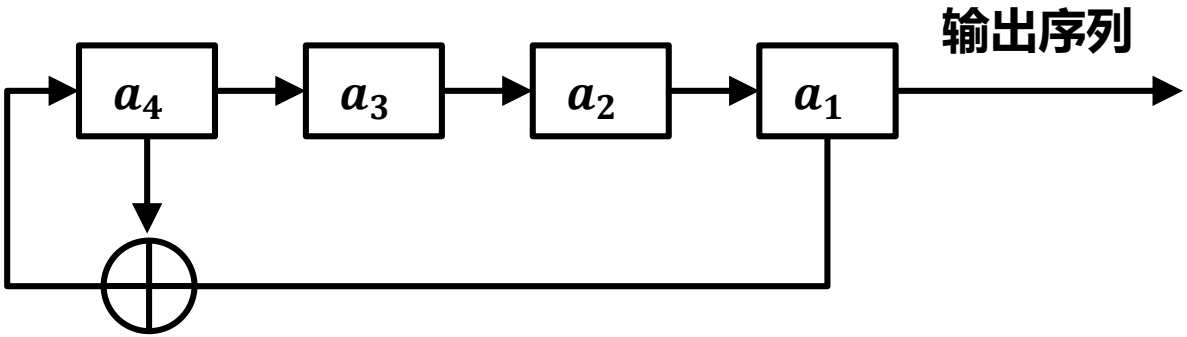
  这种递推关系可以用$GF(2^n)$上的一个特征多项式表示：

  $$f(x) = \sum_{i=1}^{n} c_i x^i$$

  可见，c_i的作用类似于一个开关。这样的线性函数总共有2^n个。

反馈移位寄存器算法

- 例子：如下图所示线性移位寄存器，初始值为 $[0,1,1,0]$



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
a_1	0	1	1	0	0	1	0	0	0	1	1	1	1	0	1	0
a_2	1	1	0	0	1	0	0	0	1	1	1	1	0	1	0	1
a_3	1	0	0	1	0	0	0	1	1	1	1	0	1	0	1	1
a_4	0	0	1	0	0	0	1	1	1	1	0	1	0	1	1	0

$a_4 + a_1$

RC4序列密码算法

- RC4加密算法是大名鼎鼎的RSA三人组中的头号人物Ronald　Rivest在1987年设计的密钥长度可变的流加密算法簇。之所以称其为簇，是由于其核心部分的S-box长度可为任意，但一般为256字节。该算法的速度可以达到DES加密的10倍左右，且具有很高级别的非线性。

- RC4是一种得到广泛应用的序列密码体制，特别是在使用安全套接字层SSL协议的Internet通信和无线通信邻域的信息安全方面，例如它被作为无线局域网标准IEEE 802.11中WEB协议的一部分。

非对称密码的基本概念

一个可逆函数$f: A \to B$，若它满足：

1 对所有$x \in A$，易于计算$f(x)$。

2 对“几乎所有$x \in A$”由$f(x)$求x“极为困难”，以至于实际上不可能做到，则称f为一单向(One-way)函数。

定义中的“易于计算”是指函数值能在其输入长度的多项式时间内求出，即若输入长度为n，计算函数的时间是n^a的倍数，a为一固定的常数。

若计算函数时间是a^n的倍数，则为不可能做到的。

非对称密码的基本概念

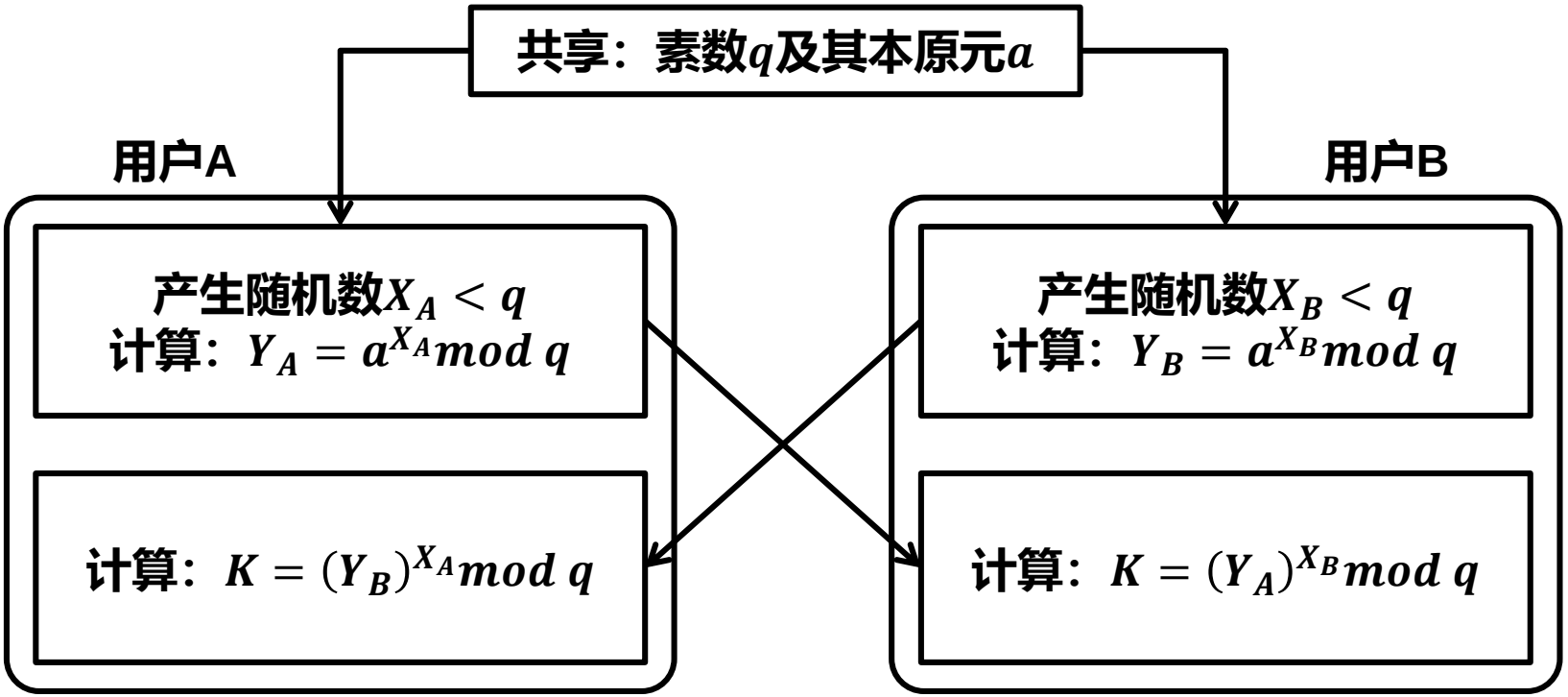
- 单向函数是求逆困难的函数，而单向陷门函数(Trapdoor one-way function)，是在不知陷门信息下求逆困难的函数，当知道陷门信息后，求逆是易于实现的。

- 限门单向函数是一族可逆函数f_k，满足
  - $Y = f_k(X)$ 易于计算（当k和X已知）
  - $X = f_k^{-1}(Y)$ 易于计算（当k和Y已知）
  - $X = f_k^{-1}(Y)$计算上不可行（Y已知但k未知）

研究公钥密码算法就是找出合适的单向限门函数

非对称密码的基本概念

- **Diffie-Hellman密钥交换算法：**



RSA密码体制

- MIT三位年青数学家R.L.Rivest， A.Shamir和L.Adleman[Rivest等1978，1979]发现了一种用数论构造双钥的方法，称作MIT体制，后来被广泛称之为RSA体制。
- 它既可用于密钥分配、又可用于数字签字。
- RSA算法的安全性基于数论中大整数分解的困难性。

RSA密码体制

RSA密码算法

设$n = pq$，其中p和q是素数。设$\mathcal{P} = \mathcal{C} = \mathbb{Z}_n$，且定义：

$$\mathcal{K} = \{(n, p, q, a, b): ab \equiv 1 mod \phi(n)\}$$

其中$\phi(n) = (p-1)(q-1)$，对于$\mathcal{K} = (n, p, q, a, b)$，定义：

$$e_K(x) = x^b mod\, n$$

和

$$d_K(y) = y^b mod\, n$$

其中，$(x, y \in \mathbb{Z}_n)$。值n和b组成了公钥，且值p, q和a组成了私钥。

基于离散对数的公钥密码体制

❖ 离散对数问题

在有限乘法群$(G, \cdot)$中，对于一个n阶元素$\alpha \in G$，定义：

$$\langle \alpha \rangle = \{\alpha^i : 0 \leq i \leq n-1\}$$

显然，$\langle \alpha \rangle$是G的一个子群，$\langle \alpha \rangle$是一个n阶循环群。

取G为有限域$\mathbb{Z}_p$（p为素数）的乘法群，α为模p的本原元。这时有$n = |\langle \alpha \rangle| = p - 1$。

离散对数：对于乘法群$(G, \cdot)$，一个n阶元素$\alpha \in G$和元素$\beta \in \langle \alpha \rangle$，找到唯一的整数$a$，$0 \leq a \leq n-1$，满足

$$\alpha^a = \beta$$

我们将这个整数a记为$\log_\alpha \beta$，称之为β的离散对数。

ElGamal密码体制

❖ $\mathbb{Z}_p^*$上的ElGamal公钥密码体制：

设p是一个素数，使得$(\mathbb{Z}_p^*,\cdot)$上的离散对数问题是难处理的，令$\alpha \in \mathbb{Z}_p^*$是一个本原元。令$\mathcal{P} \in \mathbb{Z}_p^*$，$\mathcal{C} \in \mathbb{Z}_p^* \times \mathbb{Z}_p^*$，定义

$$\mathcal{K} = \{(p, \alpha, a, \beta): \beta \equiv \alpha^a mod\ p\}$$

其中，p, α, β是公钥，a是私钥。

对$K = (p, \alpha, a, \beta)$，以及一个（秘密）随机数$k \in \mathbb{Z}_{p-1}$，定义：

$$e_k(x, k) = (y_1, y_2)$$

其中：

$$y_1 = \alpha^k mod\ p \text{ 和 } y_2 = x\beta^k mod\ p$$

对$y_1, y_2 \in \mathbb{Z}_p^*$，定义：

$$d_k(y_1, y_2) = y_2(y_1^a)^{-1} mod\ p$$

椭圆曲线(有限域)

- 椭圆曲线上的Diffie-Hellman密钥交换
  - 取一素数$p \approx 2^{180}$,两个参数a, b,得到$E_p(a, b)$.
  - 取$E_p(a, b)$的一个生成元$G(x_1, y_1)$，要求G的阶是一个非常大的素数。(阶是满足$nG = O$ 的最小正整数n). $E_p(a, b)$和G公开，A和B密钥交换过程如下
    - A选小于n的整数n_A作为秘密钥，并有$P_A = n_A G$作为公钥
    - B类似选取自己的秘密钥n_B和公开钥P_B
    - A和B分别由$K = n_A P_B$和$K = n_B P_A$ 产生共享的秘密钥
  - 攻击者如想获得K，必须由P_A和G求出n_A或P_B和G求出n_B

椭圆曲线(有限域)

- 椭圆曲线上的Elgamal密码体制
  - 密钥产生过程
    - 选择一个素数p，以及$g(g<p), x<p$,计算公钥$y \equiv g^x mod\ p$,(y,g,p)为公钥，x为秘密钥
  - 加密过程
    - M是发送明文组，选择随机数k，且$(k,p-1)=1$，计算：
      $C_1 = g^k mod\ p$ (随机数k被加密)
      $C_2 = My^k mod\ p$(明文被随机数k和公钥β加密)
    - 密文由C_1、C_2级连构成，即密文$C=C_1||C_2$。
  - 解密过程
    - $M = \frac{C_2}{C_1^x} = \frac{My^k}{g^{kx}} = \frac{Mg^{xk}}{g^{kx}} = M\ mod\ p$

HASH函数的性质及其应用

❖带密钥HASH族的定义

一个HASH族是满足下列条件的四元组$(\mathcal{X}, \mathcal{Y}, \mathcal{K}, \mathcal{H})$：

（1）$\mathcal{X}$是所有可能的消息的集合

（2）$\mathcal{Y}$是由所有可能的消息摘要或认证标签构成的有限集

（3）$\mathcal{K}$是密钥空间，是所有可能的密钥构成的有限集

（4）对每个$K \in \mathcal{K}$，存在一个HASH函数$h_K \in \mathcal{H}$，$h_K: \mathcal{X} \to \mathcal{Y}$

$\mathcal{X}$可以是有限集也可以是无限集；$\mathcal{Y}$总是有限集，如果$\mathcal{X}$是有限集，HASH函数也被称之为压缩函数。$|\mathcal{X}| \geq |\mathcal{Y}|$。

不带密钥的HASH函数是一个函数$h: \mathcal{X} \to \mathcal{Y}$，可以讲不带密钥的HASH函数看做仅仅只有一个密钥的HASH族，即$|\mathcal{K}| = 1$。

HASH函数的性质及其应用

❖ HASH函数的性质

- 从应用需求上来说，HASH函数必须满足以下性质：
  - ① 能够应用到任何大小的数据块上；
  - ② 能够生成大小固定的输出；
  - ③ 对任意给定的消息，计算相对简单，软硬件实现可行；
- 从安全需求上来说，HASH函数必须满足以下性质：
  - ① 对任意给定的散列值h，找到满足$H(x) = h$的x在计算上是不可行的；
  - ② 对任意给定的x，找到满足$H(y) = H(x)$，且$y \neq x$的y在计算是不可行的；
  - ③ 要发现满足$H(y) = H(x)$且$y \neq x$的对(x, y)是计算上不可行的

HASH函数的性质及其应用

- 条件1是单向性要求；
- 条件2是弱无碰撞性要求；
- 条件3是强无碰撞性要求；

- 满足条件1和2的HASH函数称之为弱HASH函数，或者称这个HASH函数是弱无碰撞的。

- 同时满足条件1,2和3的HASH函数称之为强HASH函数，或者称这个HASH函数是强无碰撞的。

消息认证码

- 认证（Authentication）是防止网络系统遭受主动攻击的重要技术
- 认证的主要目的有两个
  - 第一，验证消息的发送者是真的，而不是冒充的，称为实体认证，包括信源、信宿等的认证和识别。
  - 第二，验证信息的完整性，即验证数据在传送或存储过程中未被篡改、重放或延迟，称为消息认证。

消息认证码

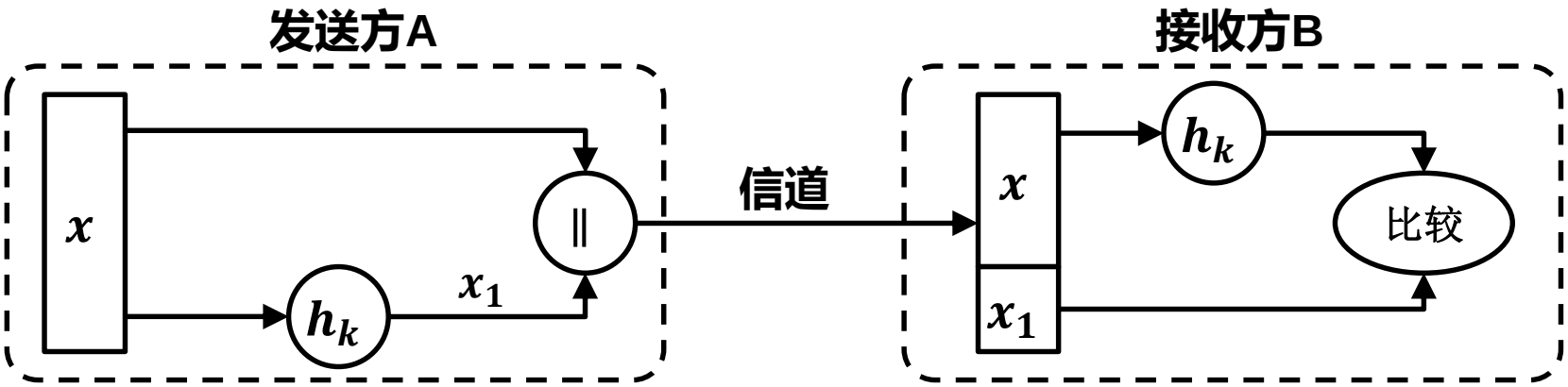
- 带密钥的Hash函数称为消息认证码(MAC：Message Authentication Code).
  - 消息认证码是实现消息认证的重要工具.
  - MAC有两个不同的输入，一个是消息x，另一个是密钥K.
  - MAC产生定长的输出.

- 设计MAC算法的要求

  在不知道密钥的情况下，难以找到两个不同的消息具有相同的输出。

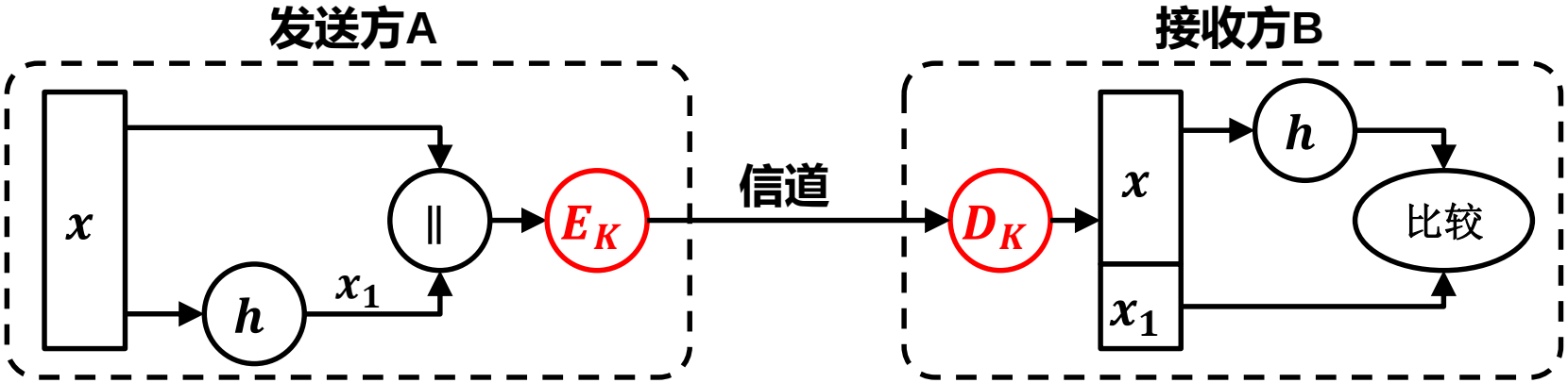
消息认证码的应用

- 使用MAC校验数据完整性
  - 设用户A将消息x发送给接收者B，A与B共享秘密的MAC密钥K，h_K是MAC。
  - 用户A计算x的MAC $h_k(x)$，将数据$C = x \parallel h_K(x)$发送给B
  - 用户B通过其它方法确定用户A的身份，分开接收到的数据x'，使用共享密钥K计算$h_k(x')$，并与接收到的$h_k(x)$相比较。



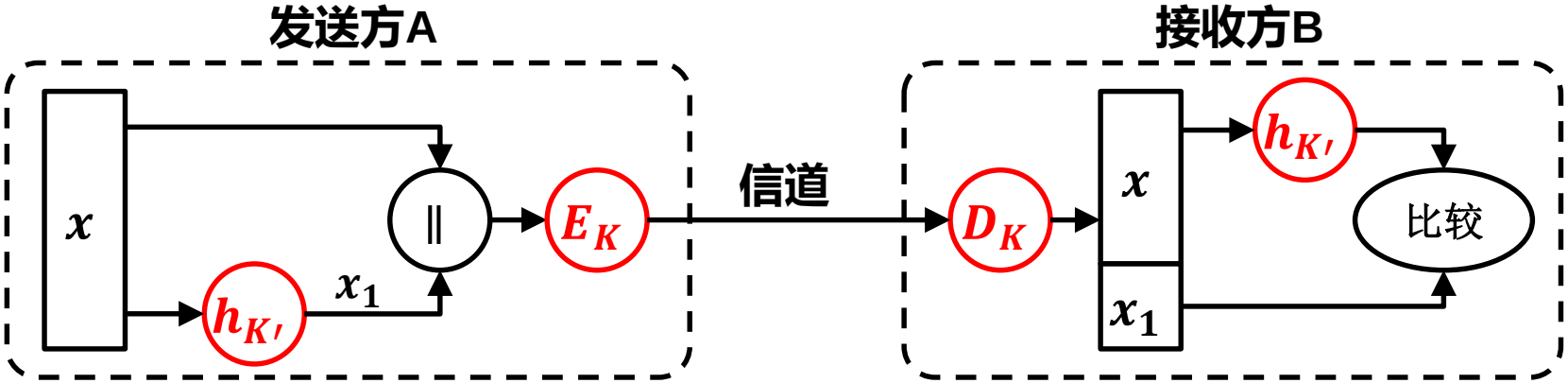
消息认证码的应用

- **使用Hash函数和加密校验数据完整性**
  - 设A与B共享分组密码的密钥K, E_K是加密算法,h是公开的Hash函数.
  - 用户A计算$C = E_K(x \| h(x))$,并将数据C发送给B
  - 用户B利用密钥K进行解密,得到x'和$h(x)$，然后计算$h(x')$并与接收到的$h(x)$相比较.
  - 由于这里对$h(x)$进行了加密,所以对Hash函数h的要求可以适当降低.



消息认证码的应用

- 使用MAC和加密校验数据完整性
  - 设A与B共享分组密码的密钥K和MAC的密钥K', E_K是加密算法, $h_{K'}$是MAC。
  - 用户A计算$C = E_K\left(x \parallel h_{K'}(x)\right)$并将数据$C$发送给B
  - 用户B利用密钥K进行解密,得到x'和$h_{K'}(x)$，然后计算$h_{K'}(x')$并与接收到的$h_{K'}(x)$相比较.
  - 该技术的优点是即使加密算法被攻破, MAC仍然能提供完整性保护作用. 其缺点是需要管理K和K'两个密钥.



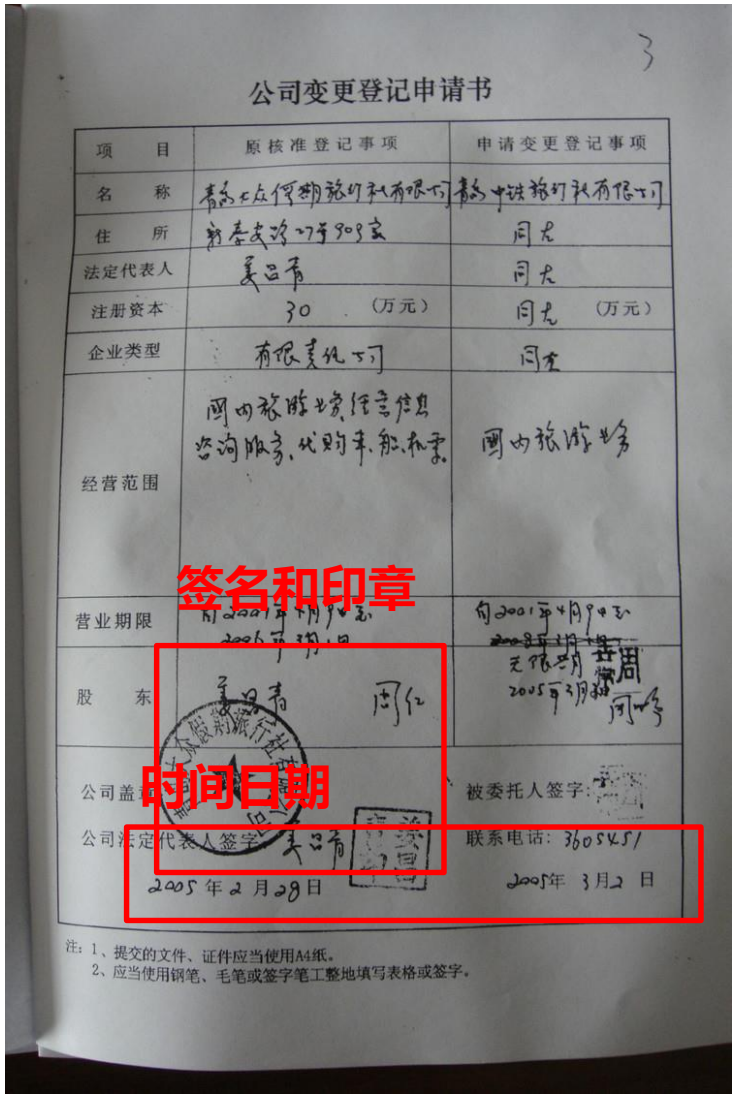
消息认证码的应用

❖ 实现数据源认证

- 数据源认证也称为消息认证,它是要求证实一个实体在过去某个时刻建立的数据源.数据源认证也包括数据完整性.提供数据源认证的方法有:
  - (1) 使用消息认证码MAC;
  - (2) 对附加上散列值的消息进行加密.
- 设用户A将消息x发送给接收者B,A与B共享密钥K,则用户A向B发送以下数据可实现数据源的认证.
  - (1) $E_K(x||h(x))$: 提供保密(仅双方共享K)和认证(加密保护散列值);
  - (2) $x||E_K(h(x))$: 提供认证（加密保护散列值）;
  - (3) $x||h(x||S)$：提供认证（仅双方共享随机数S）;
  - (4)$E_K(x||h(x||S))$：提供保密和认证（仅双方共享K、S）

数字签名的基本概念

- 数字签名应具备的性质：
  - 精确性：签名与文档具有一一对应关系，不同的文档签名得到结果应该是不同的。（不可分割性）
  - 唯一性：签名应基于签名者唯一性特征，从而确定签名的不可伪造性和不可否认性。（难以模仿性）
  - 时效性：签名应具有时间特征，防止签名的重复使用。



数字签名的分类

❖ 数字签名的分类

- 直接数字签名的两种主要形式

（1）发送方A使用自己私钥对消息直接进行签名，接收方B用发送方的公钥对签名进行鉴别。

$$A \rightarrow B: e_{A_{Private}}(M)$$

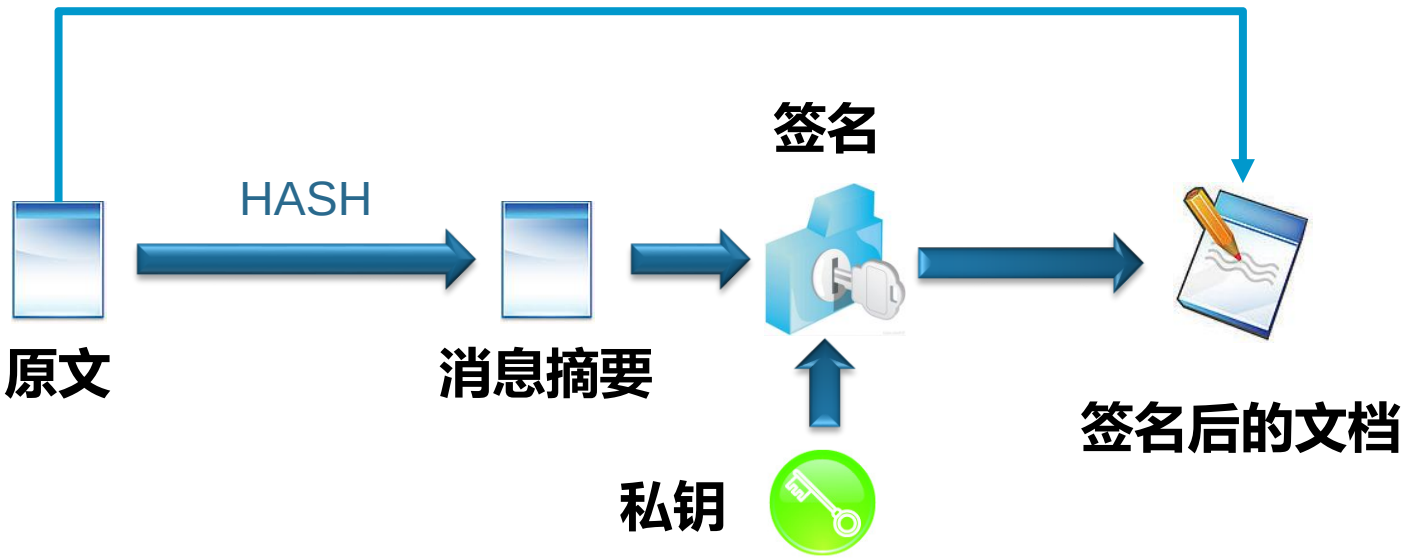
数字签名的分类

❖ 数字签名的分类

- 直接数字签名的两种主要形式

（2）发送方A首先生成签名文档的消息摘要，然后对消息摘要进行数字签名：

$$A \rightarrow B: M|e_{A_{Pri}}(H(M))$$



数字签名的分类

❖ 数字签名的分类

- 可仲裁数字签名的三种主要形式

（1）单密钥加密方式，仲裁者可以获知消息

发送方A，接收方B，仲裁者X；

A和X间共享密钥k_{AX}， B和X间共享密钥k_{BX}

步骤1： A发送给X

$$M \parallel e_{k_{AX}}\left(ID_A \parallel H(M)\right)$$

步骤2：X发送给B

$$e_{k_{BX}}\left(ID_A \parallel M \parallel e_{k_{AX}}\left(ID_A \parallel H(M)\right) \parallel T\right)$$

数字签名的分类

❖ 数字签名的分类

- 可仲裁数字签名的三种主要形式

（1）单密钥加密方式，仲裁者可以获知消息

解决纠纷机制：

B发送给X

$$e_{k_{BX}}\left(ID_A \parallel M \parallel e_{k_{AX}}\left(ID_A \parallel H(M)\right)\right)$$

仲裁者X可以恢复出

数字签名的分类

❖ 数字签名的分类

- 可仲裁数字签名的三种主要形式

（2）单密钥加密方式，仲裁者不能获知消息

发送方A，接收方B，仲裁者X；

A和X间共享密钥k_{AX}， B和X间共享密钥k_{BX}， A和B间共享密钥k_{AB}

步骤1： A发送给X

$$e_{k_{AB}}(M) \| e_{k_{AX}}\left(ID_A \| H\left(e_{k_{AB}}(M)\right)\right)$$

步骤2：X发送给B

$$ID_A \| e_{k_{BX}}\left(ID_A \| e_{k_{AB}}(M) \| e_{k_{AX}}\left(ID_A \| H\left(e_{k_{AB}}(M)\right)\right) \| T\right)$$

数字签名的分类

❖ 数字签名的分类

- 可仲裁数字签名的三种主要形式

（3）双密钥加密方式，仲裁者不能获知消息

A的公私密钥对为$(k_{A_{Pri}}, k_{A_{Pub}})$， B的公私密钥对为$(k_{B_{Pri}}, k_{B_{Pub}})$， X的公私密钥对为$(k_{X_{Pri}}, k_{X_{Pub}})$

步骤1： A发送给X

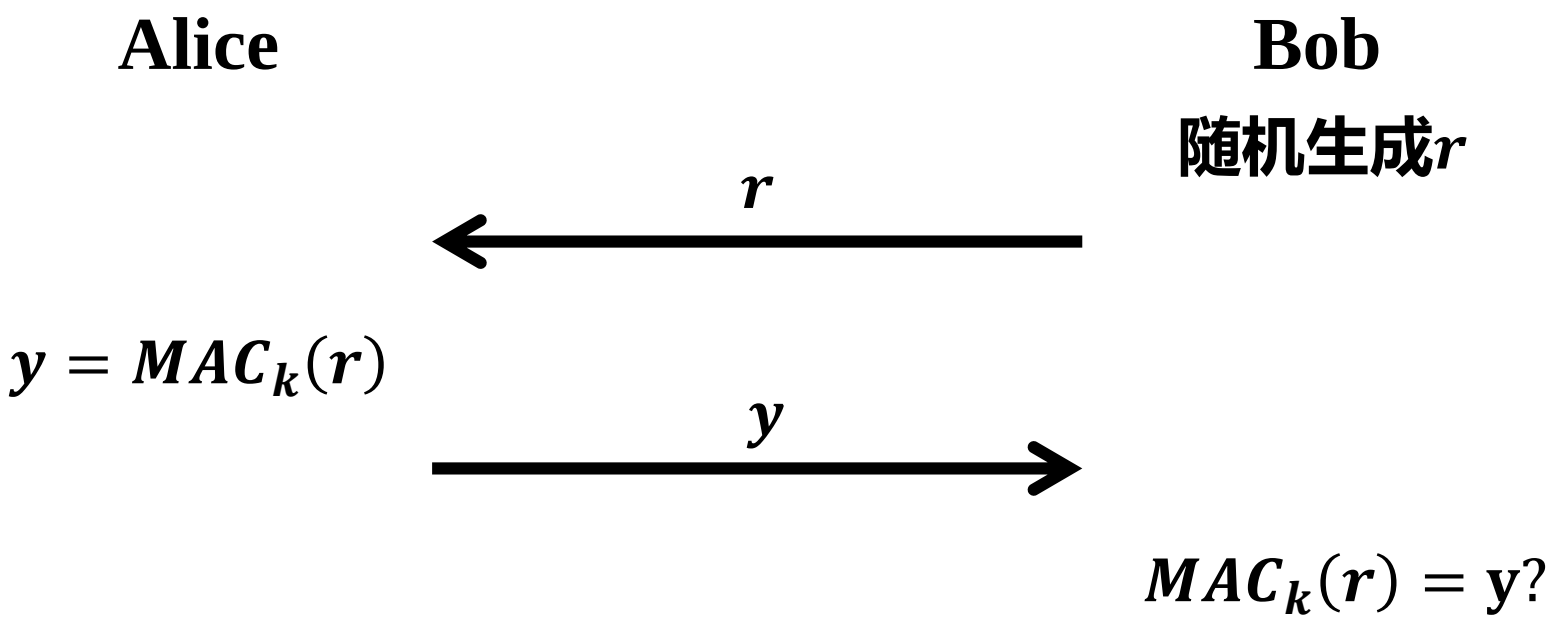
$$ID_A \parallel e_{k_{A_{Pri}}}\left(ID_A \parallel e_{k_{B_{Pub}}}\left(e_{k_{A_{Pri}}}(M)\right)\right)$$

步骤2：X发送给B

$$e_{k_{X_{Pri}}}\left(ID_A \parallel e_{k_{B_{Pub}}}\left(e_{k_{A_{Pri}}}(M)\right) \parallel T\right)$$

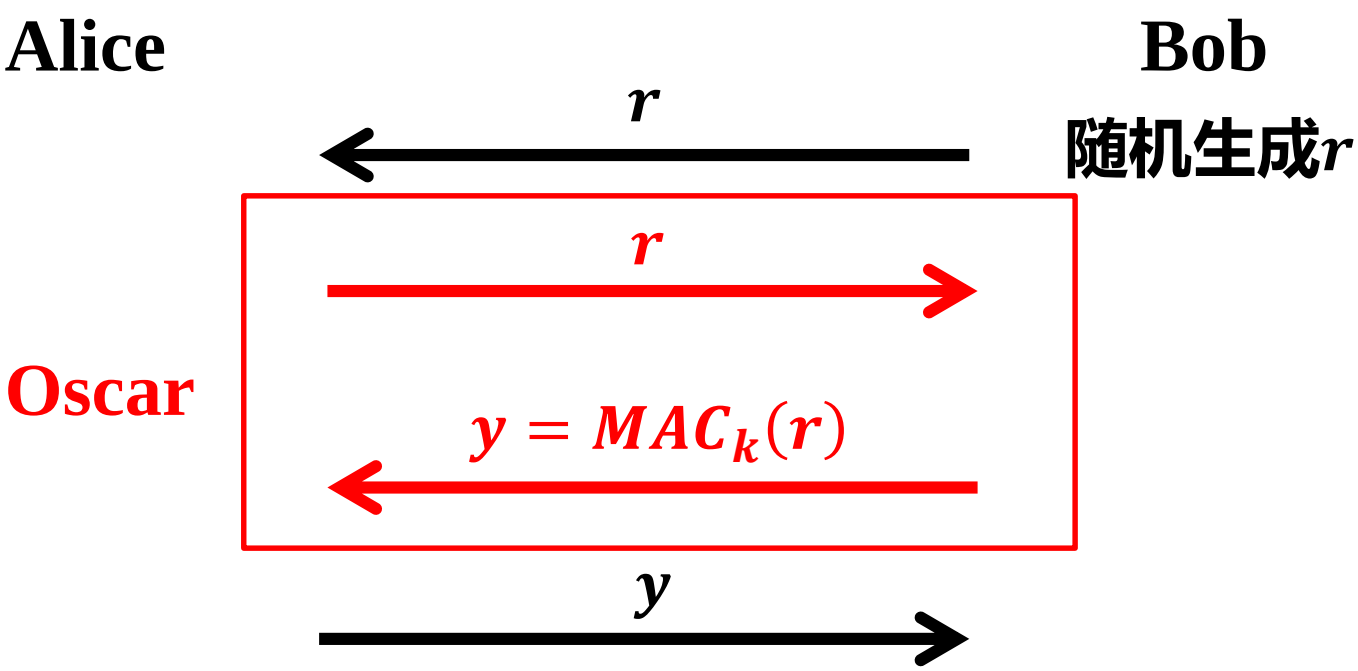
对称密钥环境下的识别与认证

- 案例1：



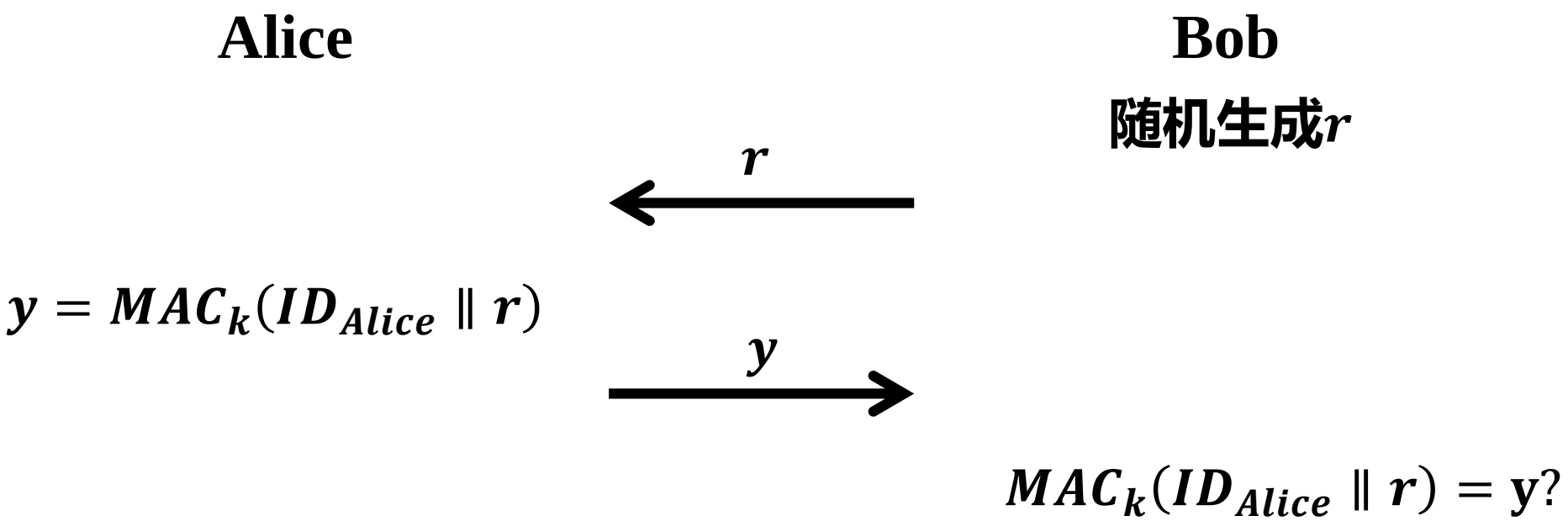
对称密钥环境下的识别与认证

❖ 案例1：



对称密钥环境下的识别与认证

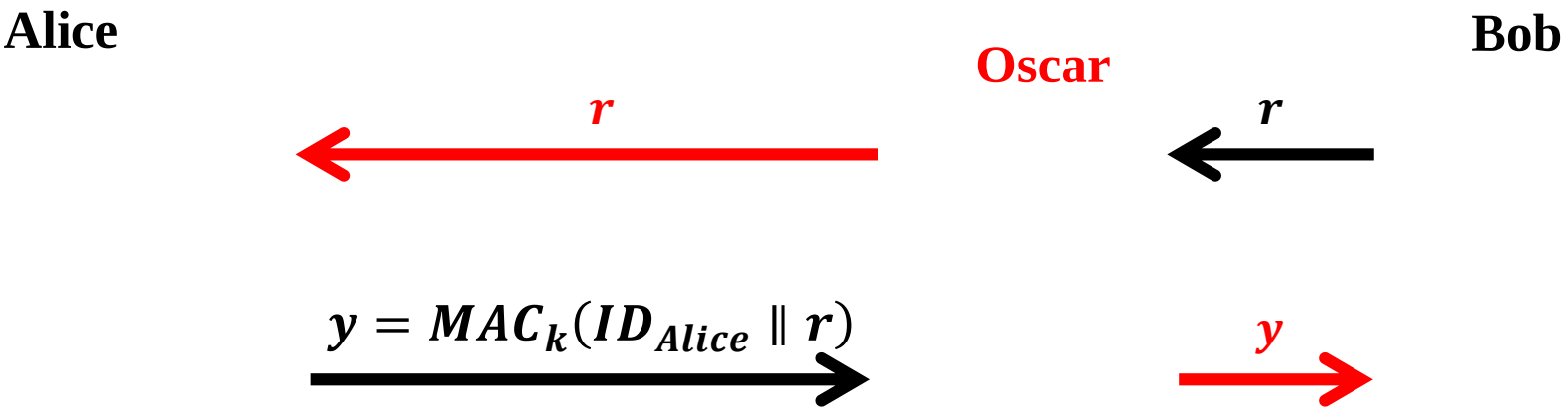
❖改进的模型



对称密钥环境下的识别与认证

❖ 攻击模型和敌手目标

- 中间入侵攻击者



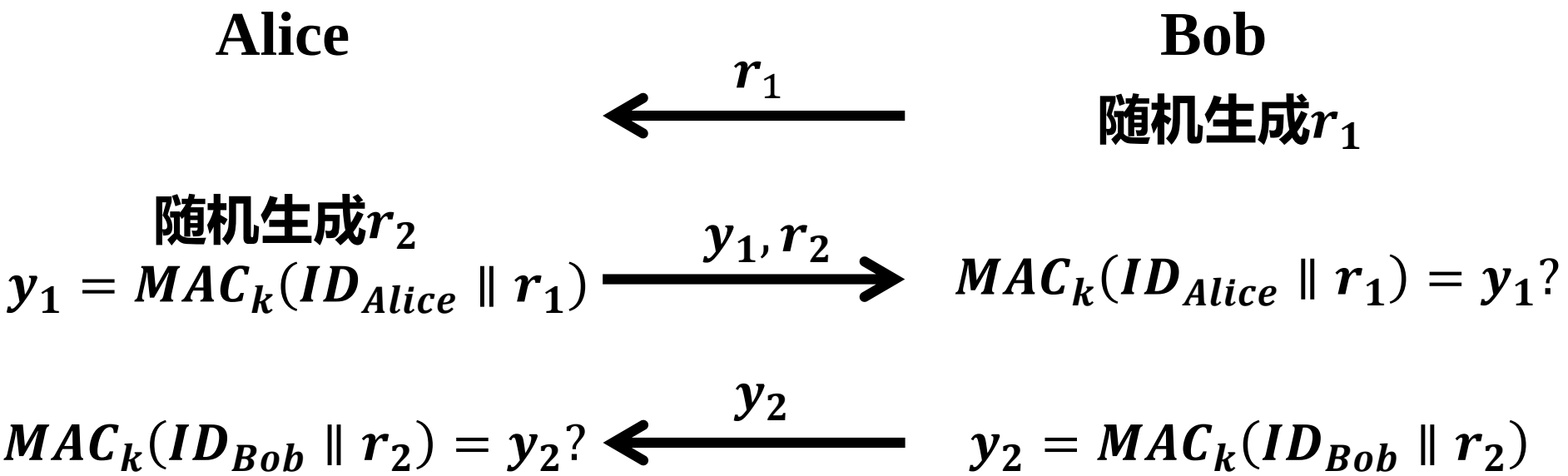
Oscar不是主动参与者，因此不是一个真正的攻击

对称密钥环境下的识别与认证

- 一个安全的交互身份识别方案应具有的输出结果：
  - 假定Alice和Bob是回话中的两个参与方，他们都是诚实的。假定敌手是被动的，那么Alice和Bob都将“接受”。
  - 如果敌手是主动的，则回话完成后，诚实的参与方都不会接受。

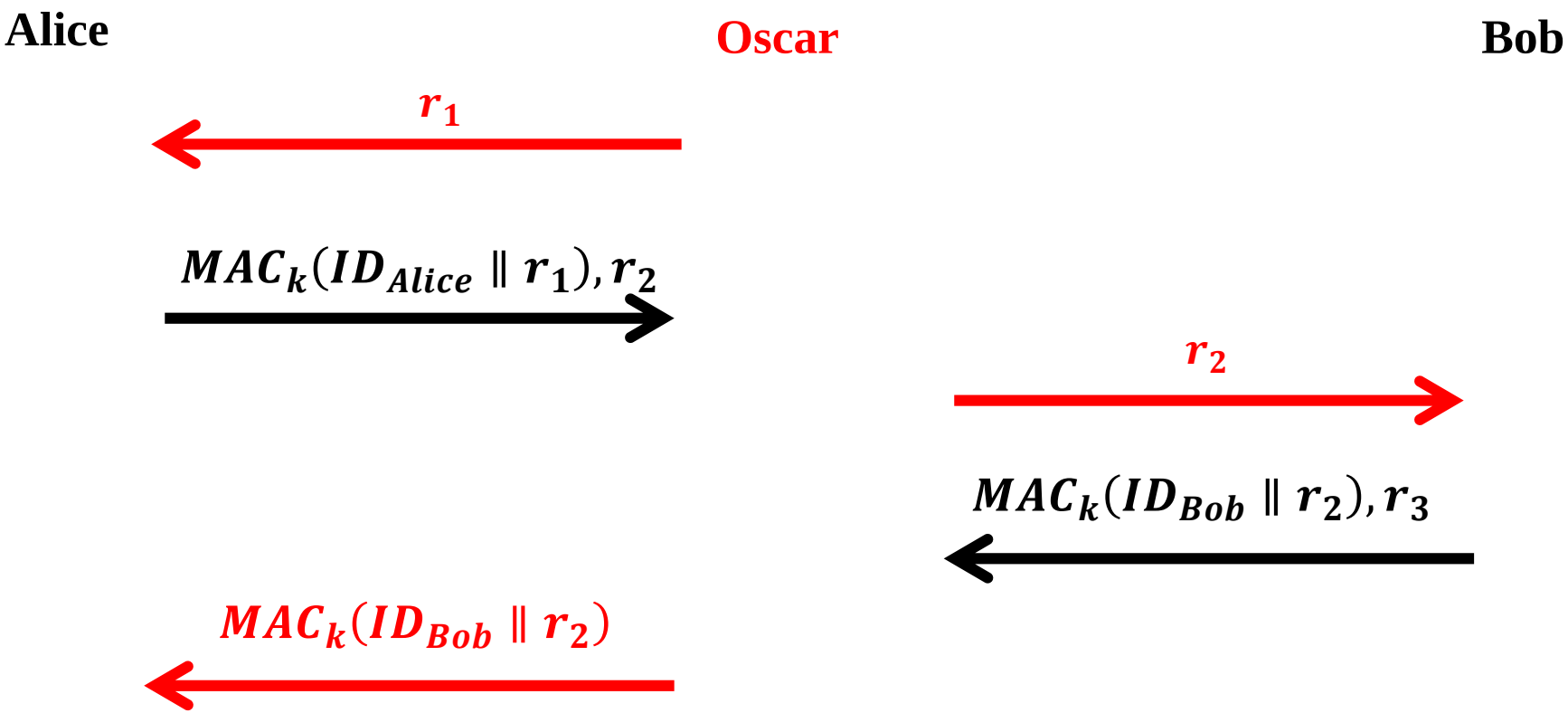
对称密钥环境下的识别与认证

- 基于改进模型的交互认证：



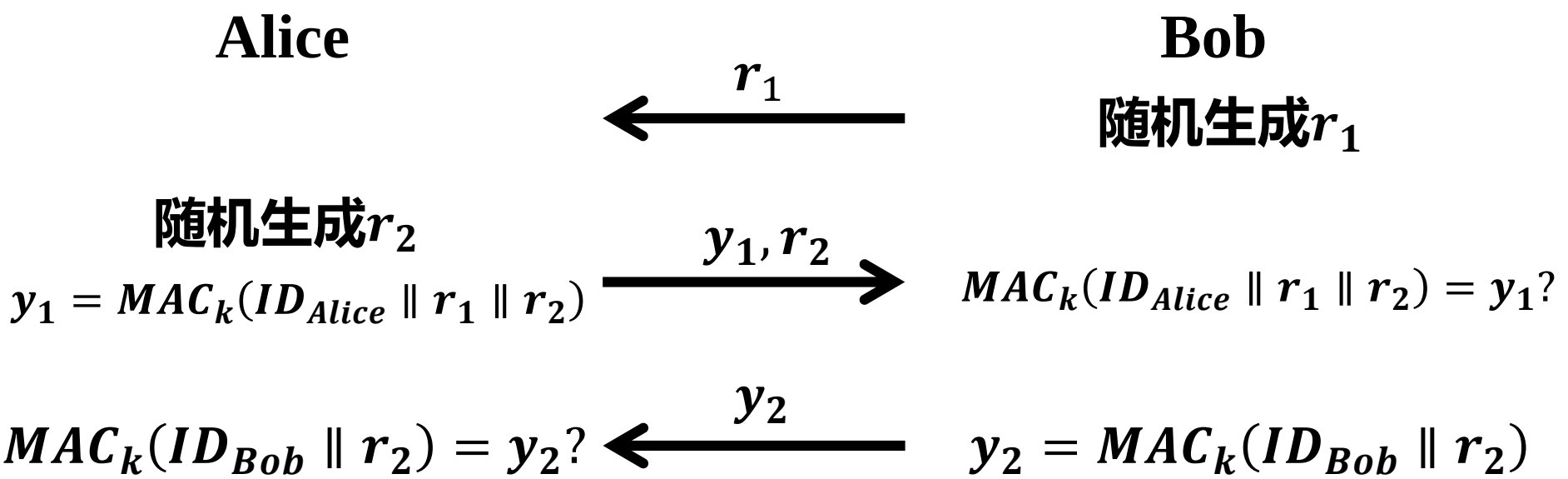
对称密钥环境下的识别与认证

- 基于改进模型的交互认证安全吗?



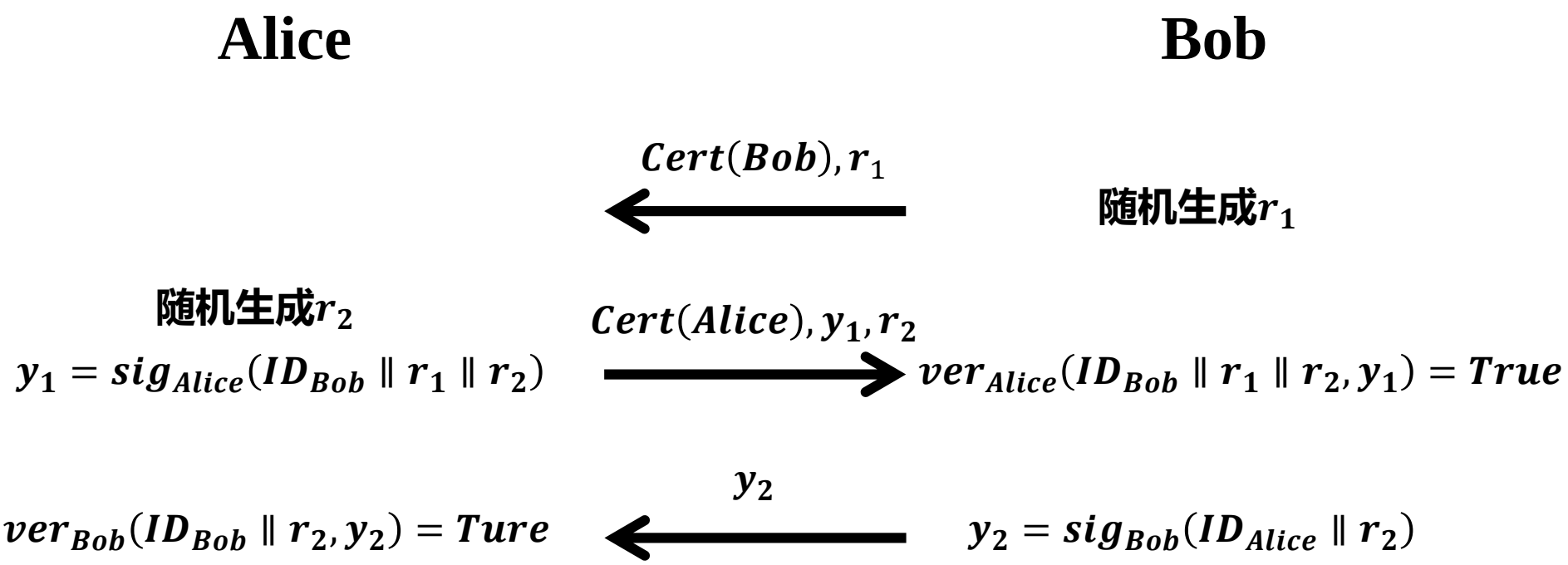
对称密钥环境下的识别与认证

❖ 再改进：



非对称密钥环境下的识别与认证

❖ 公钥环境下的交互身份识别方案



非对称密钥环境下的识别与认证

❖一种不安全的公钥交互认证方案

Alice **Bob**

$Cert(Bob), r_1$ ←

随机生成r_1

随机生成r_2

$y_1 = sig_{Alice}(ID_{Bob} \| r_1 \| r_2)$

$Cert(Alice), y_1, r_2$ →

$ver_{Alice}(ID_{Bob} \| r_1 \| r_2, y_1) = True$

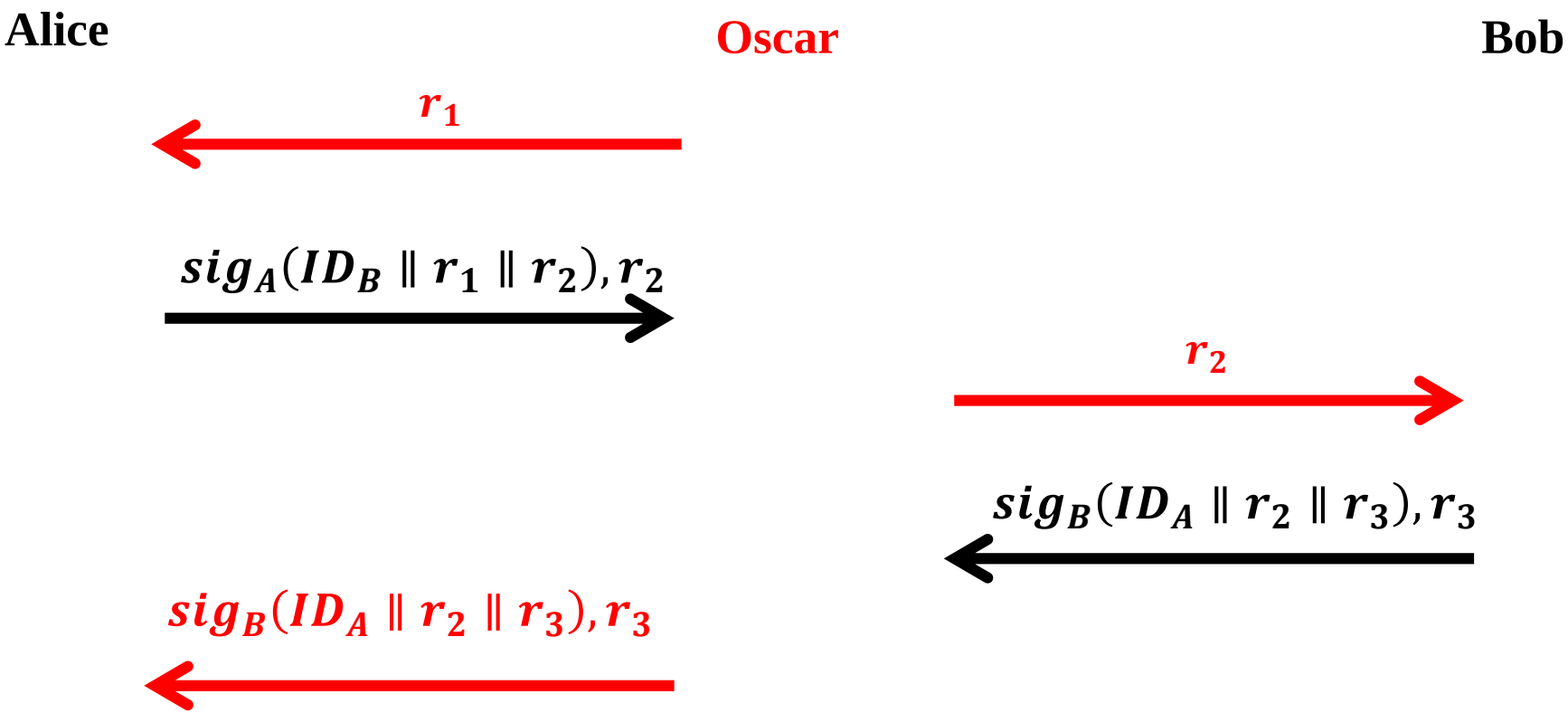
$ver_{Bob}(ID_{Bob} \| r_2 \| r_3, y_2) = Ture$

y_2 ←

$y_2 = sig_{Bob}(ID_{Alice} \| r_2 \| r_3)$

非对称密钥环境下的识别与认证

- 一种不安全的公钥交互认证方案



密钥分配概述

- 长期密钥和会话密钥
  - **长期密钥**是预先计算（非交互），并长期安全存储。长期密钥可以是一对用户拥有的秘密密钥，也可以是TA和一个用户共同拥有的秘密密钥，可以是对称密钥，也可以是非对称密钥中的私钥。
  - **会话密钥**是短期会话中使用的密钥，当会话结束之时就会把会话密钥丢弃。长期密钥通常用于协议中来传输加密会话密钥

密钥分配概述

- **会话密钥的优势**
  - **会话密钥限制了攻击者可以得到的密文数量；**
  - **只要方案设计的好，会话密钥可以限制密钥泄露事件带来的暴露问题；**

因此，会话密钥适用于泄露可能性比较高的“危险”环境中。

密钥分配概述

- **密钥分配和密钥协商**
  - **在密钥分配机制中，一方（通常是TA）选择一个或者多个密钥，并以加密的形式传送给另外的一方或者多方；**
  - **在密钥协商协议中，两方（或者多方）通过在公共网络上通信共同建立一个密钥；**

密钥分配概述

❖ Diffie-Hellman 密钥预分配方案

- 公开的域参数包括：群$(G,\cdot)$，一个阶为n的元素$\alpha \in G$;
- V利用U的证书中的公钥b_U和他自己的私钥a_V计算

$$K_{U,V} = \alpha^{a_U a_V} = b_U^{a_V}$$

- U利用V的证书中的公钥b_V和他自己的私钥a_U计算

$$K_{U,V} = \alpha^{a_U a_V} = b_V^{a_V}$$

无条件安全的密钥预分配

- 如果对于网络中任意一对用户（共有 n 个用户），TA选择一个随机密钥，并通过“离线”的安全信道发送给用户，那么每个用户必须存储$n-1$个密钥，TA需要安全的传送C_2^n个密钥。代价太高！

- 因此，需要尽可能减少传输和存储信息的数量，并仍然使得每对用户能够独立的计算一个秘密密钥。

无条件安全的密钥预分配

- **无条件安全密钥预分配方案的安全模型：**

  假定TA向n个网络用户安全地分发秘密信息。敌手可以收买至多包括k（预先指定的安全参数）个用户的用户子集合。敌手的目标是确定一对未被收买的用户的秘密长期密钥。

- **无条件安全性的定义**

  任何与$\{U, V\}$不相交的至多包括k个用户的集合都不能确定有关$K_{U,V}$的任何信息。

无条件安全的密钥预分配

❖ Blom密钥预分配方案

- 密钥取自有限域$\mathbb{Z}_p$，其中$p \geq n$是素数。TA通过一个安全信道向每个用户发送$k+1$（与n无关）个$\mathbb{Z}_p$的元素。
- 考虑$k=1$的特殊情况，则TA通过安全信道向每个用户发送2个$\mathbb{Z}_p$中的元素 。

无条件安全的密钥预分配

Blom密钥预分配方案（$k = 1$）

- 素数p是公开参数，用户U公布一个元素$r_U \in \mathbb{Z}_p$。元素r_U必须是不同的。
- TA选择3个随机元素 $a, b, c \in \mathbb{Z}_p$ （不必要求不同）并构造多项式：

$$f(x, y) = a + b(x + y) + cxy \bmod p$$

- TA为用户U计算多项式

$$g_U(x) = f(x, r_U) \bmod p$$

并通过安全信道把$g_U(x)$传送给U。注意$g_U(x)$是x的线性多项式，所以可以写成

$$g_U(x) = a_U + b_U x$$

其中

$$a_U = a + br_U \bmod p \qquad b_U = b + cr_U \bmod p$$

无条件安全的密钥预分配

Blom密钥预分配方案（$k = 1$）

- 如果U和V想要进行通信，他们就使用公共的密钥

$$K_{U,V} = K_{V,U} = f(r_U, r_V) = a + b(r_U, r_V) + cr_U r_V \bmod p$$

其中U计算

$$K_{U,V} = g_U(r_V)$$

而V计算

$$K_{V,U} = g_V(r_U)$$

例子：假定有用户U，V，W，$p = 17$，用户公开信息$r_U = 12$，$r_V = 7$以及$r_W = 1$，假定TA选择$a = 8$，$b = 7$和$c = 2$。

无条件安全的密钥预分配

例子：假定有用户U，V，W，$p = 17$，用户公开信息$r_U = 12$，$r_V = 7$以及$r_W = 1$，假定TA选择$a = 8$，$b = 7$和$c = 2$。

$$f(x, y) = 8 + 7(x + y) + 2xy$$

多项式g表示为：

$$g_U(x) = 7 + 14x \qquad g_V(x) = 6 + 4x \qquad g_W(x) = 15 + 9x$$

用户U和V之间的密钥：

$$K_{U,V}(x) = g_U(r_V) = 7 + 14 \times 7 \; mod \; 17 = 3$$

$$K_{V,U}(x) = g_V(r_U) = 6 + 4 \times 12 \; mod \; 17 = 3$$

其他密钥：

$$K_{U,W}(x) = 4 \qquad K_{V,W}(x) = 10$$

无条件安全的密钥预分配

- **Blom密钥预分配方案（任意k）**
  - **多项式$f(x,y)$的次数要等于k：**

$$f(x,y)=\sum_{i=0}^{k}\sum_{j=0}^{k}a_{i,j}x^{i}\,y^{j}\ mod\ p$$

其中$a_{i,j}\in\mathbb{Z}_p,(0\le i\le k,0\le j\le k)$，并且对于所有的$i$和$j$有$a_{i,j}=a_{j,i}$

无条件安全的密钥预分配

- **Blom密钥预分配方案（任意k）**
  - 素数p是公开参数，用户U公布一个元素$r_U \in \mathbb{Z}_p$。元素r_U必须是不同的。
  - 对于$0 \le i, j \le k$，TA选择随机元素$a_{i,j} \in \mathbb{Z}_p$使得对所有的i和j有$a_{i,j} = a_{j,i}$。TA构造多项式

$$f(x,y) = \sum_{i=0}^{k} \sum_{j=0}^{k} a_{i,j} x^i y^j \bmod p$$

  - TA为用户U计算多项式

$$g_U(x) = f(x, r_U) \bmod p = \sum_{i=0}^{k} a_{U,i} x^i$$

并通过安全信道把系数向量$(a_{U,0}, \ldots, a_{U,k})$传送给U

无条件安全的密钥预分配

- **Blom密钥预分配方案（任意k）**
  - 对于任何两个用户U和V，其密钥为$K_{U,V} = f(r_U, r_V)$，其中U计算

$$K_{U,V} = g_U(r_V)$$

而V计算

$$K_{V,U} = g_V(r_U)$$

无条件安全的密钥预分配

- **Blom密钥预分配方案的一个缺点是存在一个必须事先制定的苛刻的安全门限，一旦多余k个用户决定联合，整个方案将被攻破。**

- **Blom密钥预分配方案在存储需求方面是最优的，可以证明，在任何可以抵抗k个用户联合的无条件安全在密钥预分配方案中，每个用户的存储至少是密钥长度的k+1倍。**

Diffie-Hellman密钥协商

Diffie-Hellman密钥协商方案

- 公开域参数包括群$(G,\cdot)$和阶为n的元素$\alpha \in G$。
- U选取一个随机数a_U，$0 \le a_U \le n-1$，然后计算

$$b_U = \alpha^{a_U}$$

并将b_U发送给V

- V选取一个随机数a_V，$0 \le a_V \le n-1$，然后计算

$$b_V = \alpha^{a_V}$$

并将b_V发送给U

- U计算

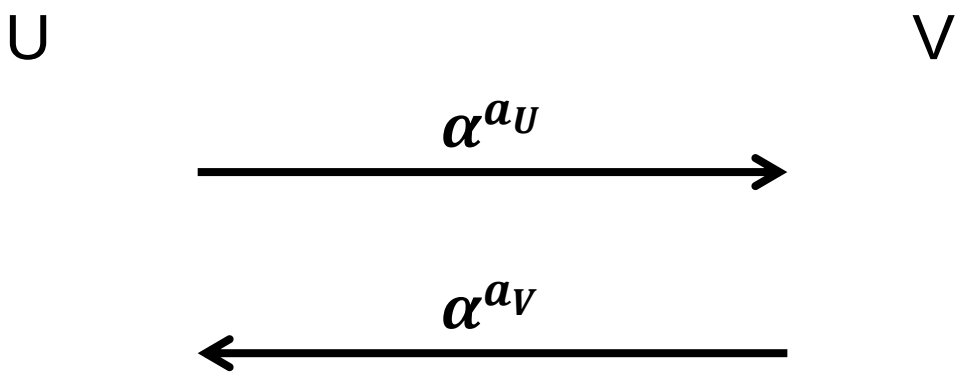
$$K = (b_V)^{a_U}$$

- V计算

$$K = (b_U)^{a_V}$$

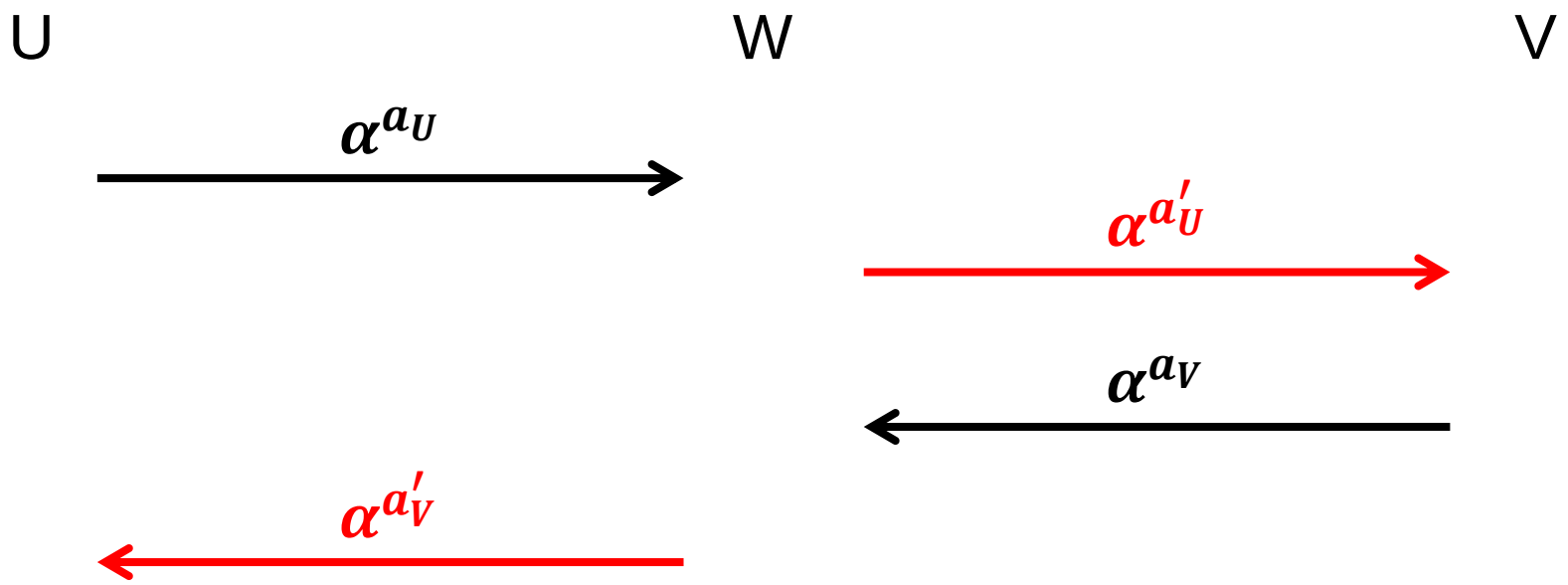
Diffie-Hellman密钥协商

- **Diffie-Hellman密钥协商方案的缺陷：**



Diffie-Hellman密钥协商

- **Diffie-Hellman密钥协商方案的缺陷：**



Diffie-Hellman密钥协商

- **认证密钥协商方案需要满足的性质：**
  - 交互识别：

方案是一个安全的交互识别方案，即在主动敌手实施任何攻击流程后，没有一个诚实的参与者会“接受”。

  - 密钥协商：

如果不存在主动敌手，则双方参与者计算出相同的新会话密钥K。除此之外，一个被动敌手将计算不出关于K的任何信息。

Diffie-Hellman密钥协商

Diffie-Hellman密钥协商方案的改进版

- 该方案使用了通常由TA签名过的证书。每个用户U有一个签名方案，签名算法记为sig_U，验证算法记为ver_U。
- TA也有一个签名方案，其公开验证算法记为ver_{TA}。每一个用户有一个证书

$$Cert(U) = \left(ID_U, ver_U, sig_{TA}(ID(U), ver_U)\right)$$

这里ID_U是U的识别信息。

Diffie-Hellman密钥协商

- 认证密钥协商方案（端-端密钥协商方案）
  - 公开域参数包括群$(G,\cdot)$和一个阶为n的元素$\alpha \in G$。
  - U选取一个随机数a_U，$0 \le a_U \le n-1$，然后计算

$$b_U = \alpha^{a_U}$$

并将$Cert(U)$，b_U发送给V。

  - V选取一个随机数a_V，$0 \le a_V \le n-1$，然后计算

$$b_V = \alpha^{a_V}$$

$$K = (b_U)^{a_V}$$

$$y_V = sig_V(ID_U \| b_V \| b_U)$$

并将$Cert(V)$，b_V，y_V发送给U

Diffie-Hellman密钥协商

- 认证密钥协商方案（端-端密钥协商方案）
  - U使用ver_V来验证y_V。如果签名y_V无效，则她会“拒绝”并退出。否则她会“接受”，计算

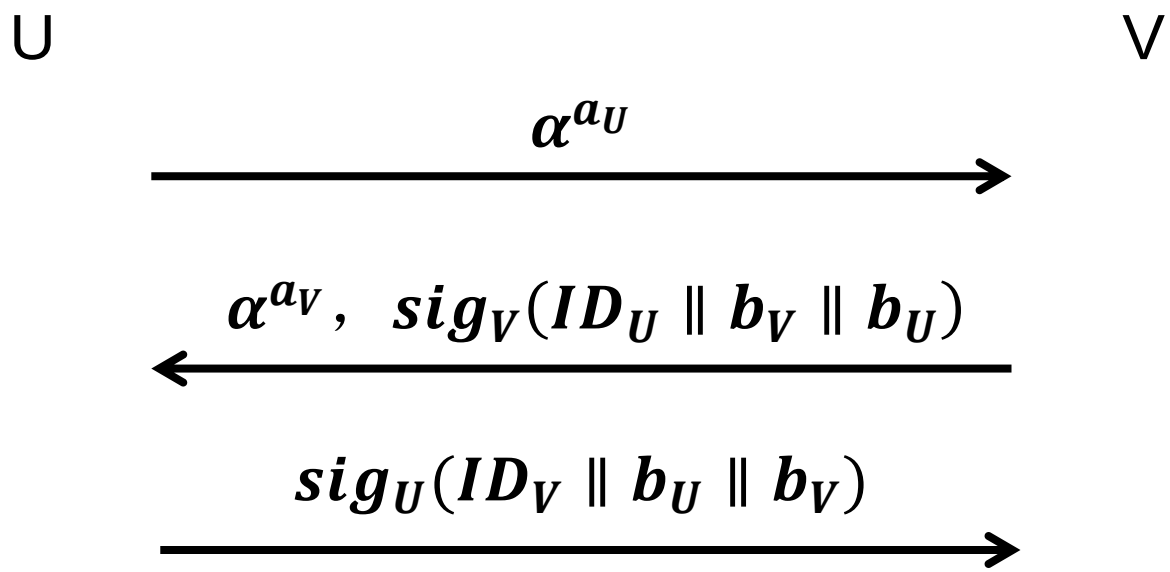
$$K = (b_V)^{a_U}$$

$$y_U = sig_U(ID_V \| b_U \| b_V)$$

  并将y_U发送给V。
  - V使用ver_U来验证y_U。如果签名y_U无效，则他会“拒绝”；否则，他会“接受”。

Diffie-Hellman密钥协商

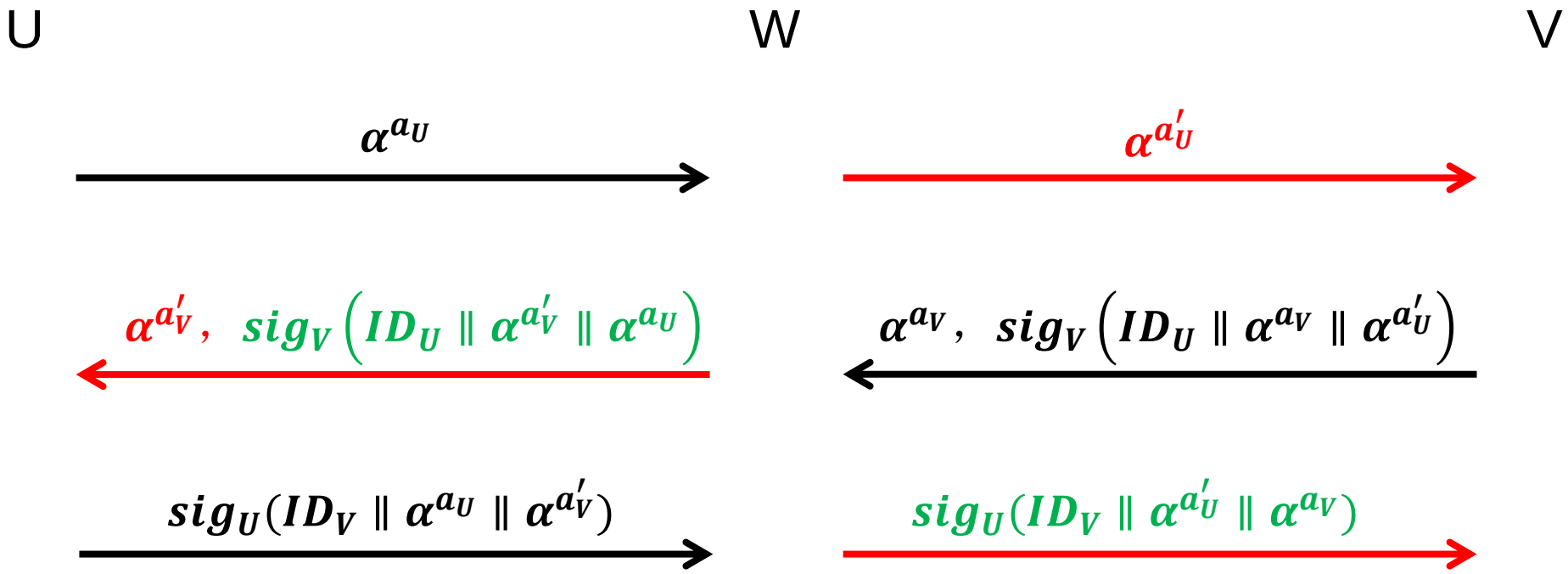
- 认证密钥协商方案（端-端密钥协商方案）



U和V之间能够相互识别对方身份；
U知道V可以计算出密钥K，反正亦然；

Diffie-Hellman密钥协商

- 认证密钥协商方案（端-端密钥协商方案）



Diffie-Hellman密钥协商

❖ 密钥协商方案中的三个层次的保证机制

- 隐式密钥认证

如果U可以被确保除V之外，没有人能计算出K（特别的，敌手不能计算K），则我们说该方案提供了隐式密钥认证。

- 隐式密钥确认

如果U可以确保V可以计算出K（假设V是按照规定执行了该方案），并且除了V之外，没有人能计算出K，则我们说该方案提供了隐式密钥确认。

- 显示密钥确认

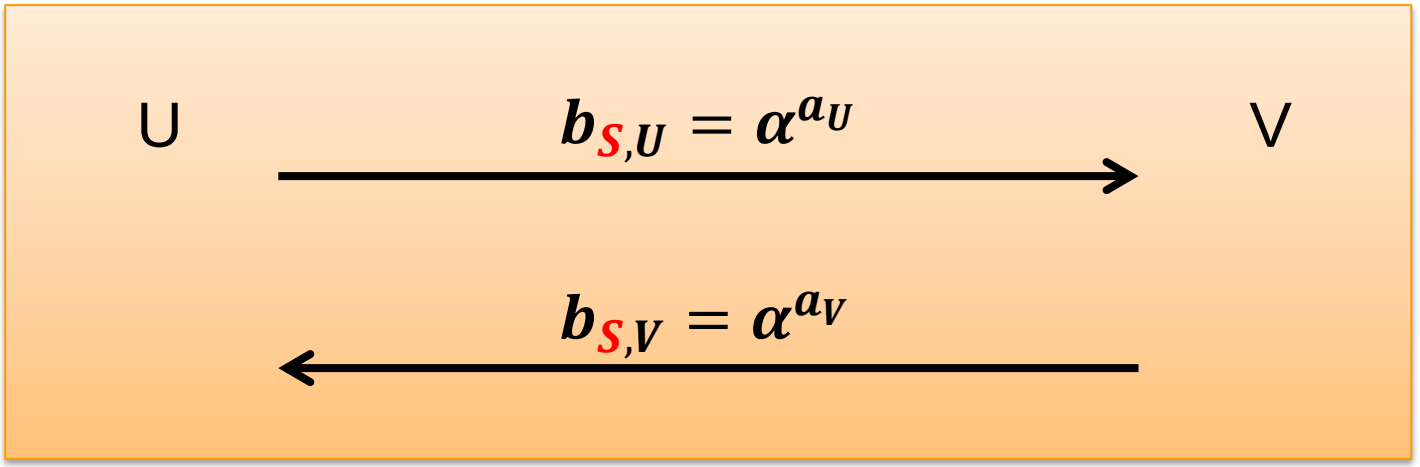
如果U可以确保V已经计算出了K，并且除了V之外，没有人能计算出K，则我们说该方案提供了显示密钥确认。

Diffie-Hellman密钥协商

❖ 已知会话密钥攻击

- Oscar观察到密钥协商方案中的几次会话，称为$S_1, S_2, \ldots, S_t$，Oscar的目标是得出其他目标会话的密钥，并且Oscar不是目标会话的参与者。

会话S



Oscar期望获得K_S

Diffie-Hellman密钥协商

已知会话密钥攻击

- 因为端-端方案是一个安全的识别方案，所以Oscar不可能在会话中活动，然后向一个“不接受”的用户请求密钥。因此我们仅允许Oscar向在一个会话中“接受”的用户请求密钥。
- 考虑Oscar有一个同伴为W，W协助Oscar获取关于K_S的信息。

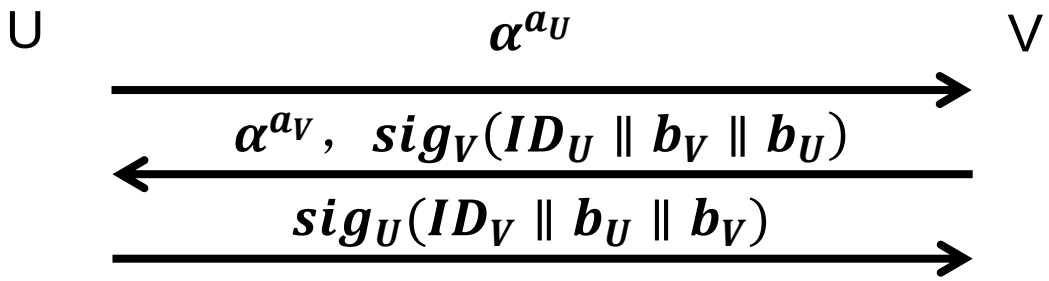
Diffie-Hellman密钥协商

已知会话密钥攻击

- Oscar任意选取一个值$b_{S',Oscar} = \alpha^{a_{S',Oscar}}$;
- 然后W在由α产生的子群中选取$b_{S',W} = \alpha^{a_{S',W}}$
- Oscar发送请求并得到$K_{S'} = CDH\left(b_{S',Oscar}, b_{S',W}\right) = \left(\alpha^{a_{S',Oscar}}\right)^{a_{S',W}}$，那么Oscar获取了一个三元组
$$\left(b_{S',Oscar}, b_{S',W}, CDH\left(b_{S',Oscar}, b_{S',W}\right)\right)$$
- Oscar通过若干次会话，累计了一个三元组列表$\mathcal{J}$。
- Oscar的目标是寻找一个多项式算法A，使得通过$A\left(\mathcal{J}, \left(b_{S,U}, b_{S,V}\right)\right)$计算得到关于$K_S$的某些信息

MTI密钥协商方案

- 端-端密钥协商方案是一个三段式方案：



- MTI方案是一个双流协商方案

MTI密钥协商方案

❖MTI密钥协商方案

- 公开域参数包括群$(G,\cdot)$和一个阶为n的元素$\alpha \in G$。每一个用户T有一个秘密指数a^T，其中$0 \leq a_T \leq n-1$，对应的公开值为

$$b_T = \alpha^{a_T}$$

b_T被包含在T的证书里并被TA签名

- U随机选取r_U，$0 \leq r_U \leq n-1$，计算

$$s_U = \alpha^{r_U}$$

然后U将$Cert(U)$和s_U发送给V

- V随机选取r_V，$0 \leq r_V \leq n-1$，计算

$$s_V = \alpha^{r_V}$$

然后U将$Cert(U)$和s_U发送给V

MTI密钥协商方案

MTI密钥协商方案

- V计算出会话密钥

$$K = s_U^{a_V} b_U^{r_V}$$

其中他从$Cert(U)$中获得了b_U的值。

- U计算出会话密钥

$$K = s_V^{a_U} b_V^{r_U}$$

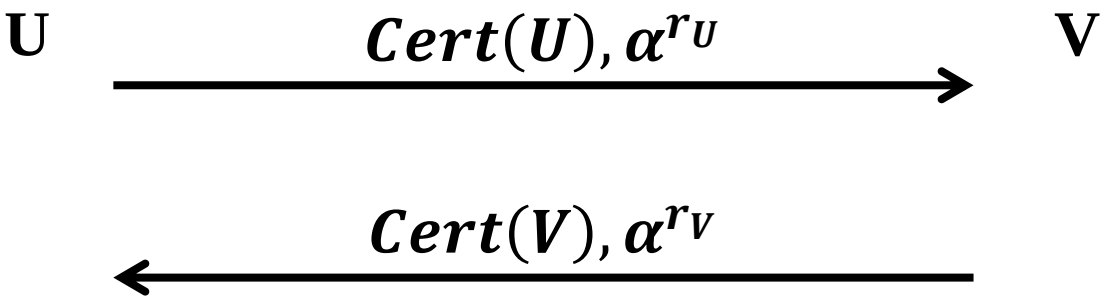
其中他从$Cert(V)$中获得了b_V的值。

会话结束时，U和V都计算出相同的会话密钥

$$K = \alpha^{r_U a_V + r_V a_U}$$

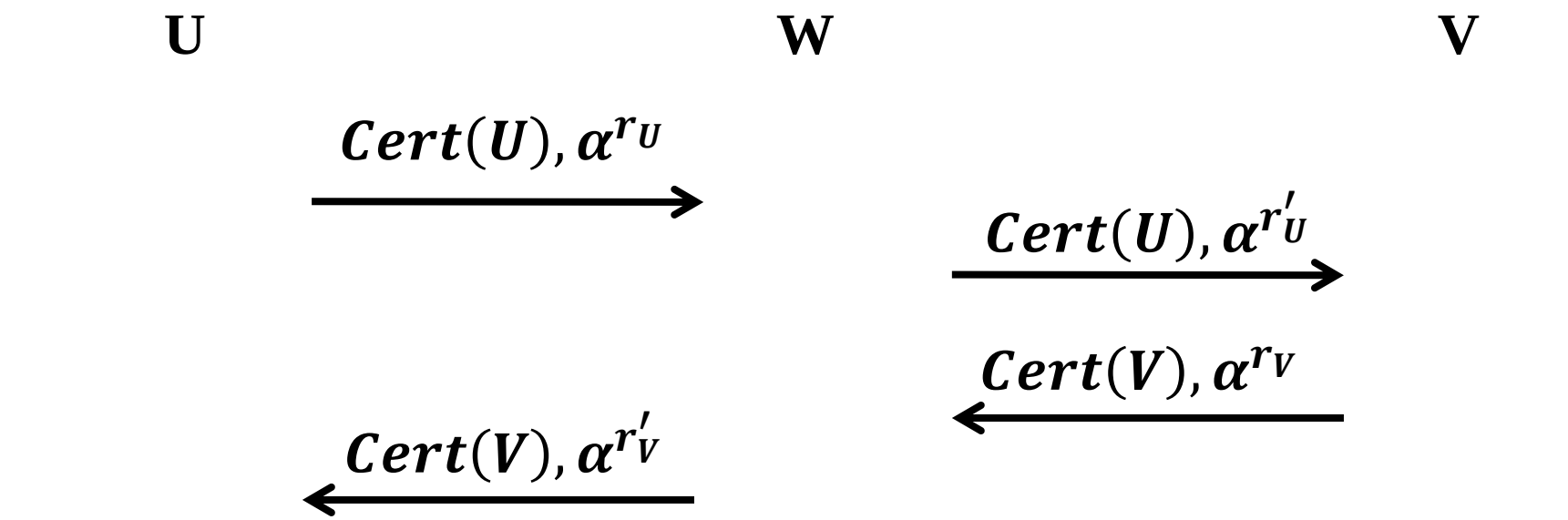
MTI密钥协商方案

- MTI密钥协商方案的信息流（双流协商方案）



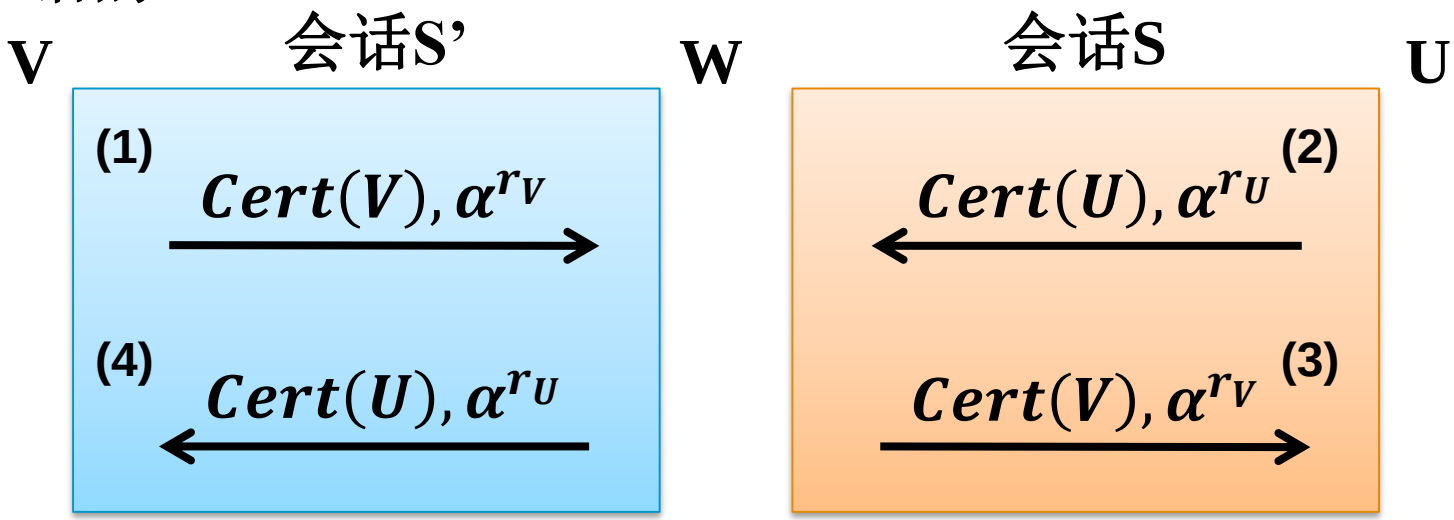
MTI密钥协商方案

❖MTI密钥协商方案



MTI密钥协商方案

- 针对MTI的已知会话密钥攻击
  - 并行会话已知会话密钥攻击：敌手W是一个在两个会话中的主动参与者，在会话S中，W假装是和U进行对话的V，在并行发送的会话S'中，W家中是和V进行对话的U。



  - 在会话结束后，W请求会话S的密钥K（被允许的），从而获得了S'的密钥

MTI密钥协商方案

- 针对MTI的已知会话密钥攻击
  - **可以进行并行会话攻击的原因是：**

$$K\big((r_U, a_U)(r_V, a_V)\big) = K\big((r_V, a_V)(r_U, a_U)\big)$$

要消除这种攻击，就必须改变这种对称性。

解决这种对称性的方案可以使用HASH函数：

假定实际的会话密钥被定义为：

$$K = h(\alpha^{r_U a_V} \parallel \alpha^{r_V a_U})$$

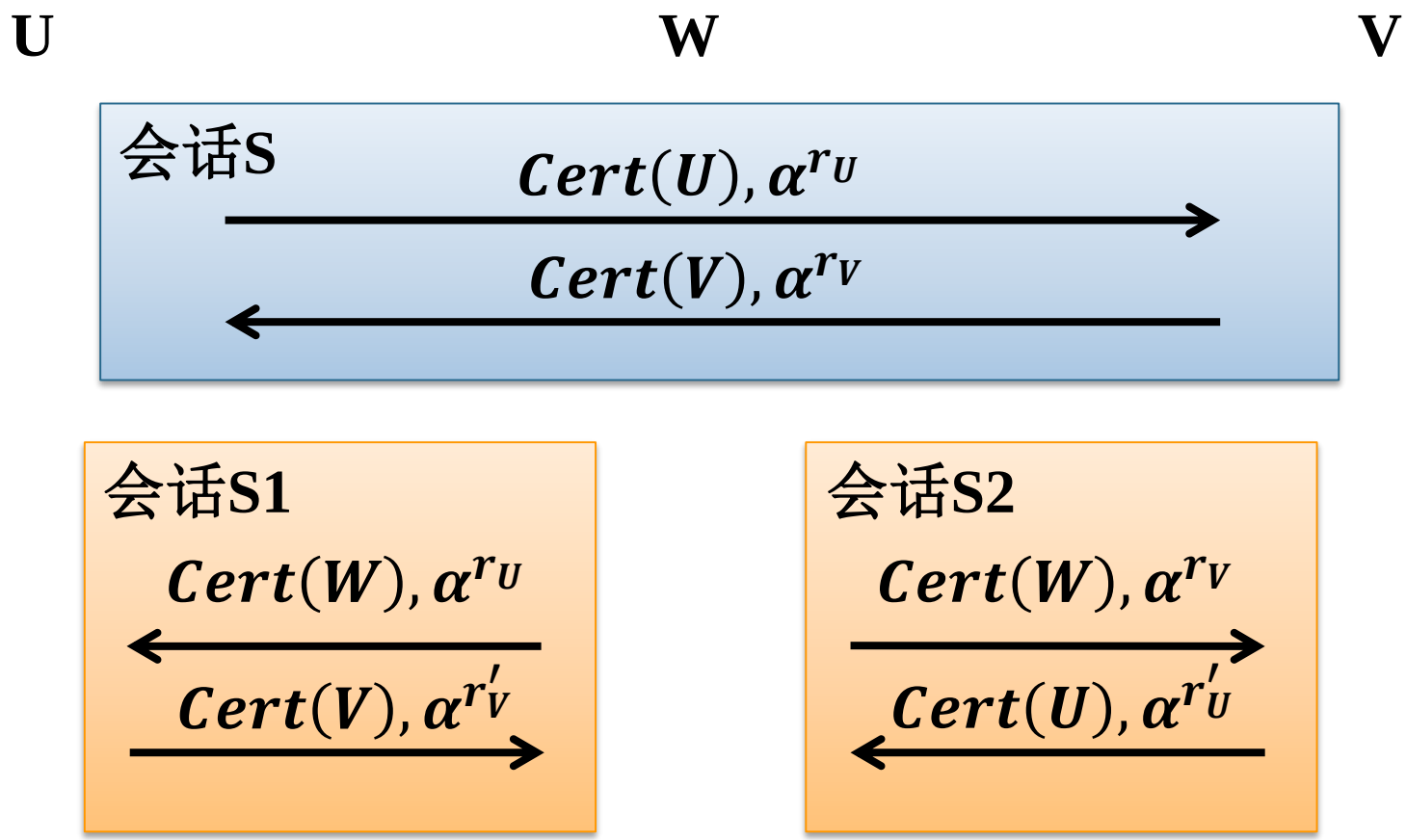
那么会话发起者（U）计算$K = h\left(b_V^{r_U} \parallel s_V^{a_U}\right)$；

那么会话响应者（V）计算$K = h\left(s_U^{a_V} \parallel b_U^{r_V}\right)$；

这种情况下S和S’的密钥是什么?

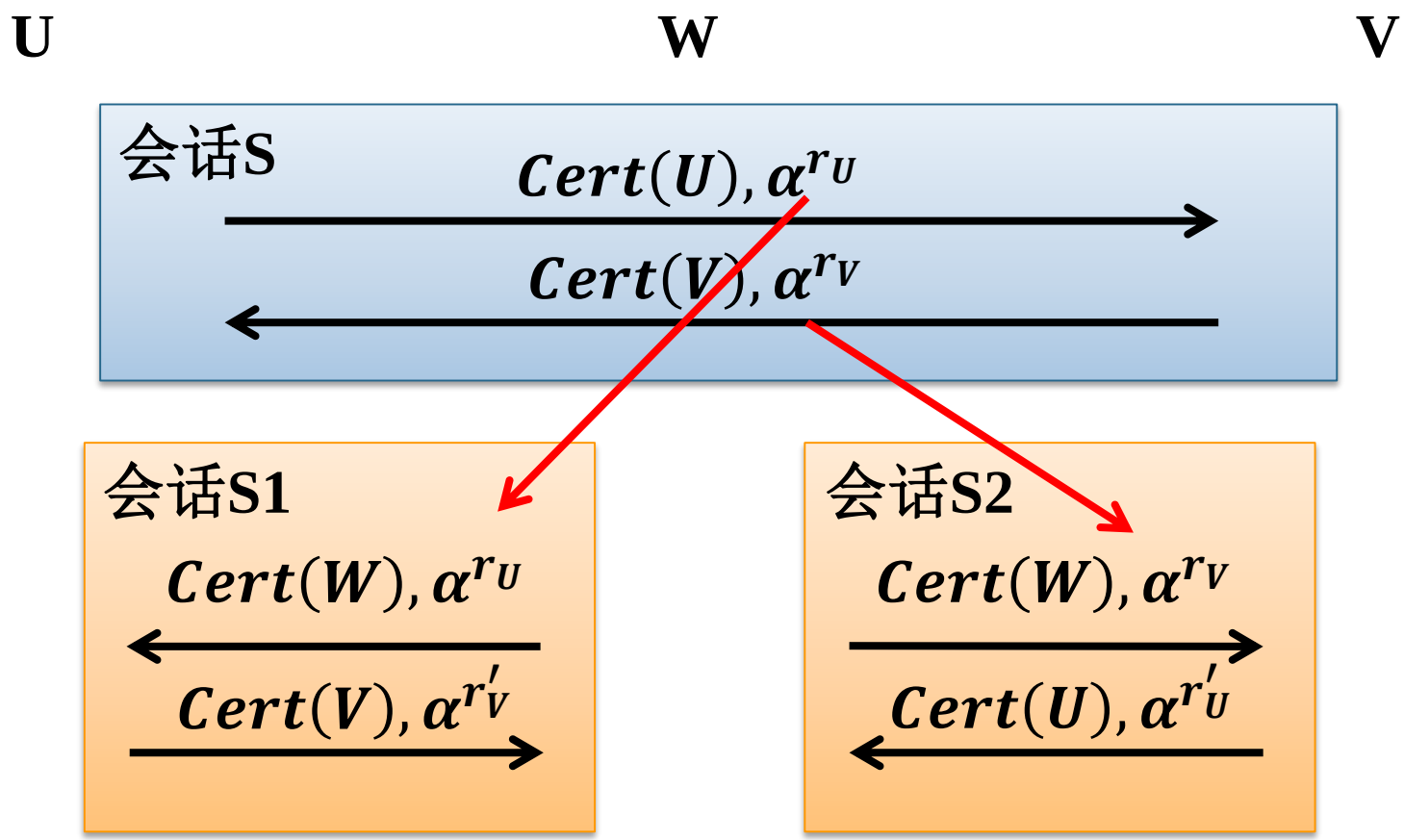
MTI密钥协商方案

- 针对MTI的Burnmester三角攻击



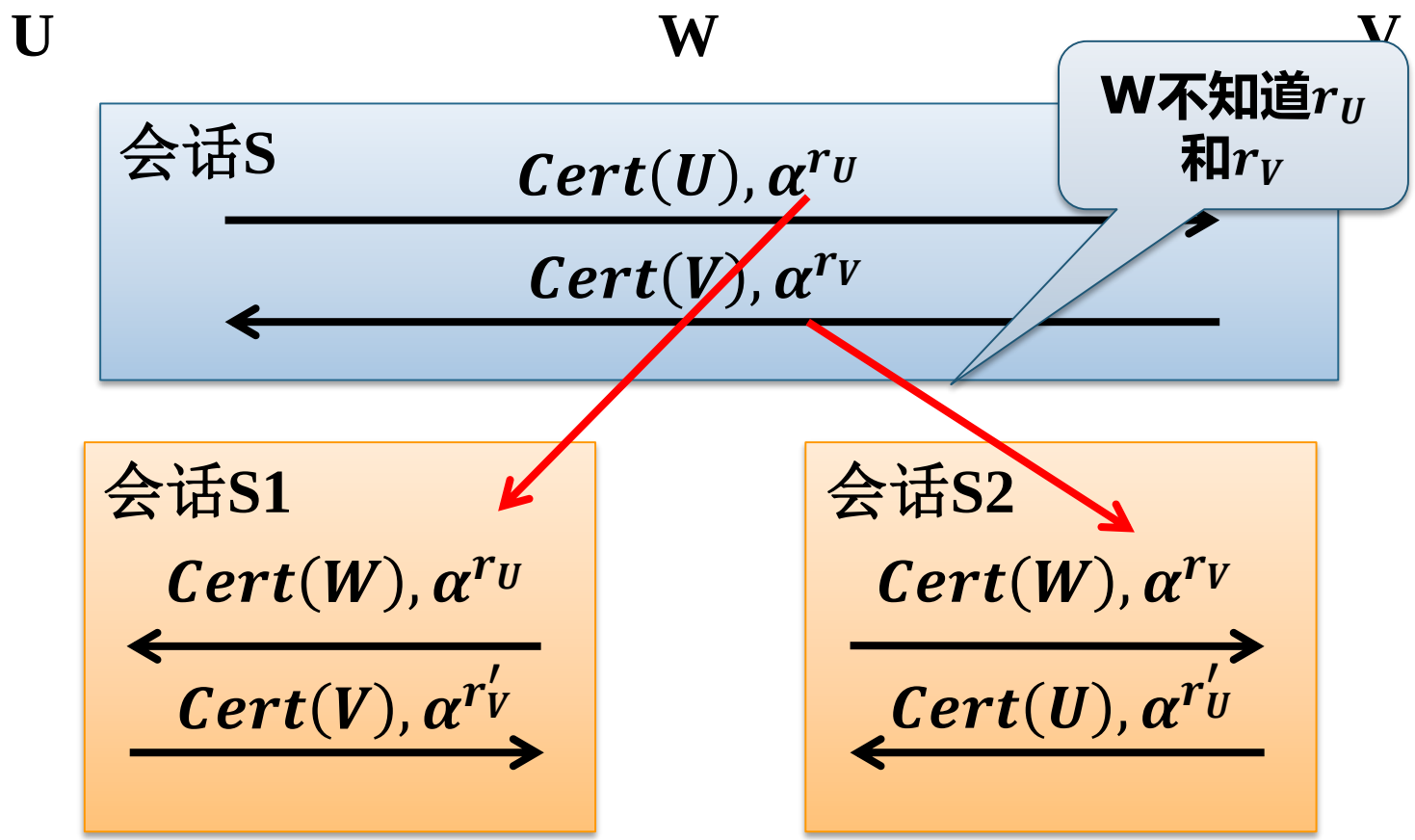
MTI密钥协商方案

- 针对MTI的Burnmester三角攻击



MTI密钥协商方案

- 针对MTI的Burnmester三角攻击



MTI密钥协商方案

- 针对MTI的Burnmester三角攻击
  - 会话S，S1，S2的密钥K，K_1，K_2分别为：

$$K = \alpha^{r_U a_V + r_V a_U}$$

$$K_1 = \alpha^{r_U a_V + r'_V a_W}$$

$$K_2 = \alpha^{r'_U a_W + r_V a_U}$$

给定K_1和K_2，可以计算：

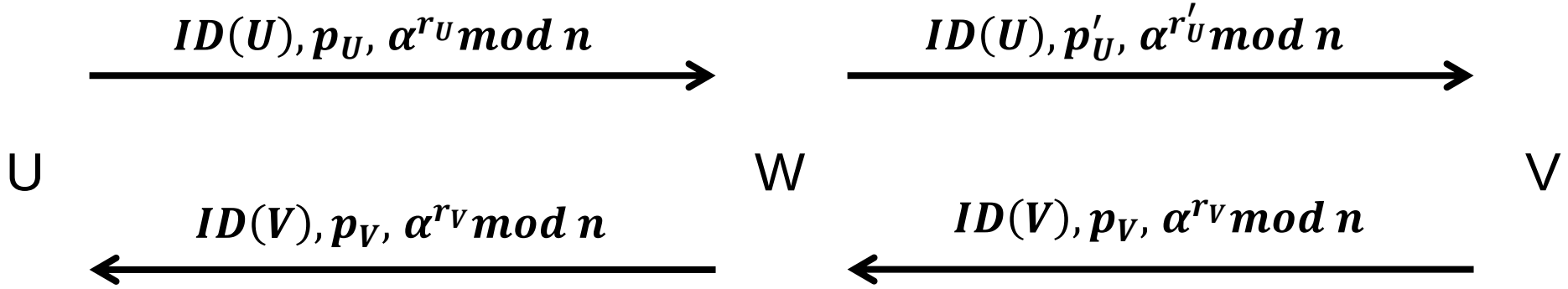
$$K = \frac{K_1 K_2}{(s'_V s'_U)^{a_W}} = \frac{\left(\alpha^{r_U a_V + r'_V a_W}\right)\left(\alpha^{r'_U a_W + r_V a_U}\right)}{\left(\alpha^{r'_V}\alpha^{r'_U}\right)^{a_W}}$$

$$= \alpha^{r_U a_V + r'_V a_W + r'_U a_W + r_V a_U - (r'_V + r'_U) a_W}$$

$$= \alpha^{r_U a_V + r_V a_U}$$

使用自认证密钥的密钥协商

- 假设TA不要求U向TA发送a_U：



V计算密钥：$K' = \alpha^{r'_U a_V + r_V a'_U} mod n$

U计算密钥：$K = \alpha^{r_U a_V + r_V a_U} mod n$

W计算：$K' = s_V^{a'_U} \left(p_V^e + ID(V)\right)^{r'_U} mod n$

加密密钥交换

- 假设相关各方事先拥有一个共享密钥。
- 两个使用者U,V，事先拥有一个共享密钥（称之为口令），表示为$\boldsymbol{pwd_{U,V}}$。“口令”暗示$\boldsymbol{pwd_{U,V}}$的长度非常短（2^{20}~2^{30}），不可以作为密钥使用。
- 客户端C需要与服务器端S通信的时候，需要使用口令，记为$\boldsymbol{pwd_C}$

加密密钥交换

- 一个简陋的使用口令加密密钥的方法：
  - 服务器随机选取一个128比特的会话密钥K
  - 服务器使用客户端C的口令pwd_C作为密钥来加密K，即$y = e_{pwd_C}(K)$， $e(\cdot)$和$d(\cdot)$分别用来表示密码体制的加解密函数。
  - S把y发送给C，C将解密y并获得K
  - C和S在接下来就可以使用K进行加密信息。

加密密钥交换

EKE密钥交换协议

- 公开域参数由群$(G,\cdot)$和具有阶n的$\alpha \in G$构成。U和V拥有共享口令$pwd_{U,V}$， $e(\cdot)$和$d(\cdot)$分别表示加解密函数
- U随机选择a_U，$0 \le a_U \le n-1$。接着他计算

$$b_U = \alpha^{a_U}，\ y_U = e_{pwd_{U,V}}(b_U)$$

然后他把$ID(U)$和y_U发送给V

- V随机选择a_V，$0 \le a_V \le n-1$。接着他计算

$$b_V = \alpha^{a_V}，\ y_V = e_{pwd_{U,V}}(b_V)$$

然后他把$ID(V)$和y_V发送给U

- U计算

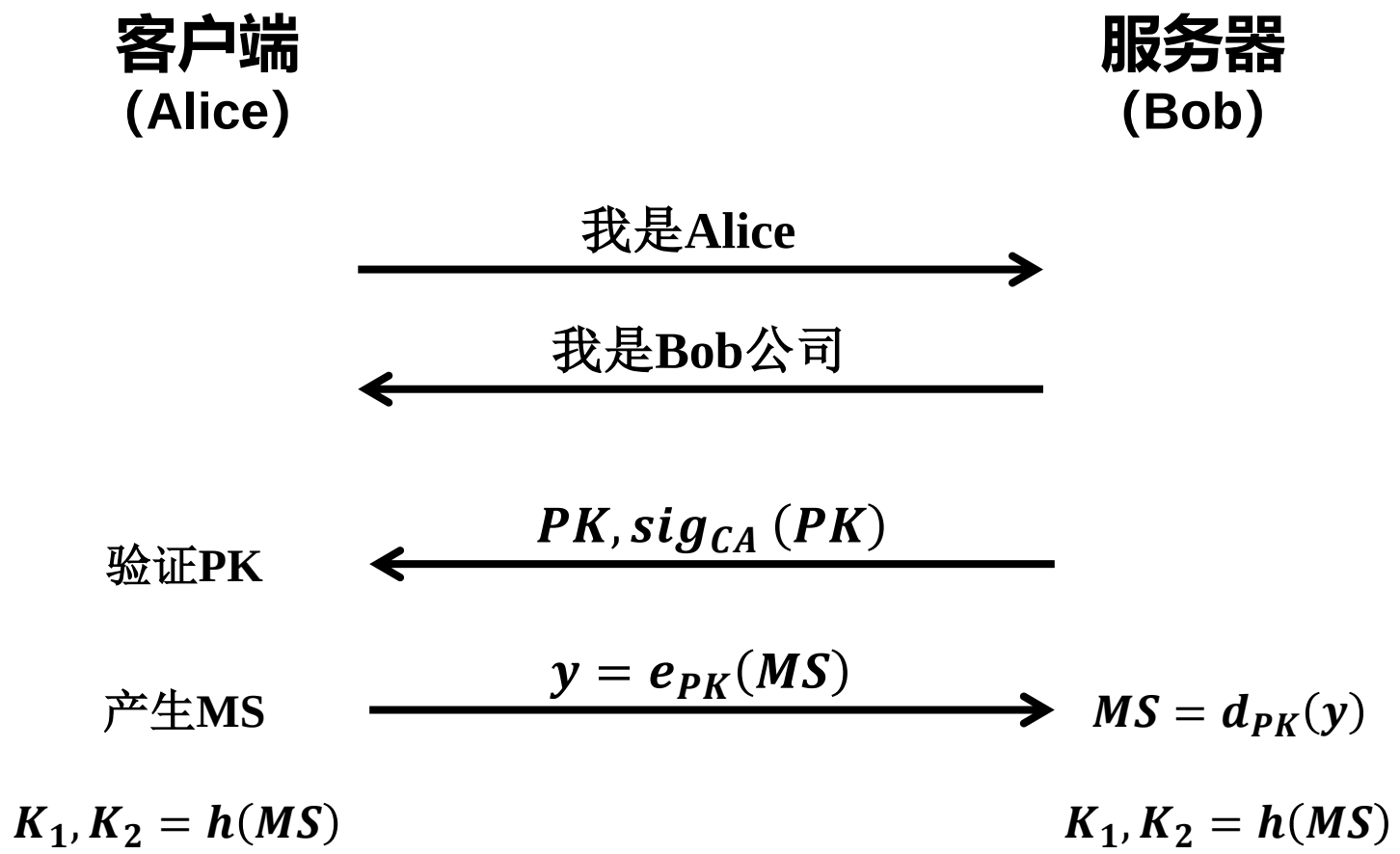
$$b_V = d_{pwd_{U,V}}(y_V)，\ K = (b_V)^{a_U}$$

- V计算

$$b_U = d_{pwd_{U,V}}(y_U)，\ K = (b_U)^{a_V}$$

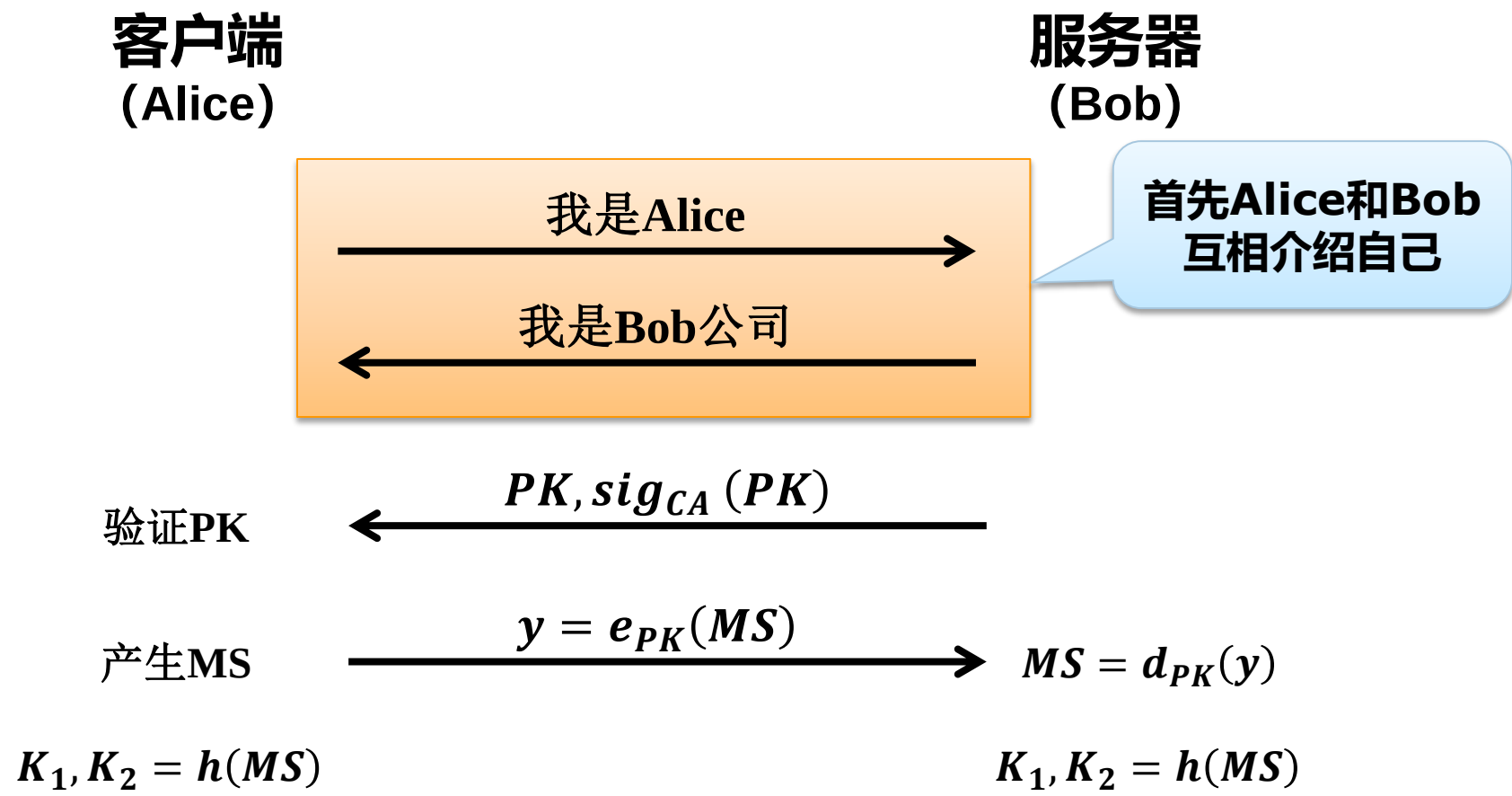
PKI简介

- 安全套阶层SSL



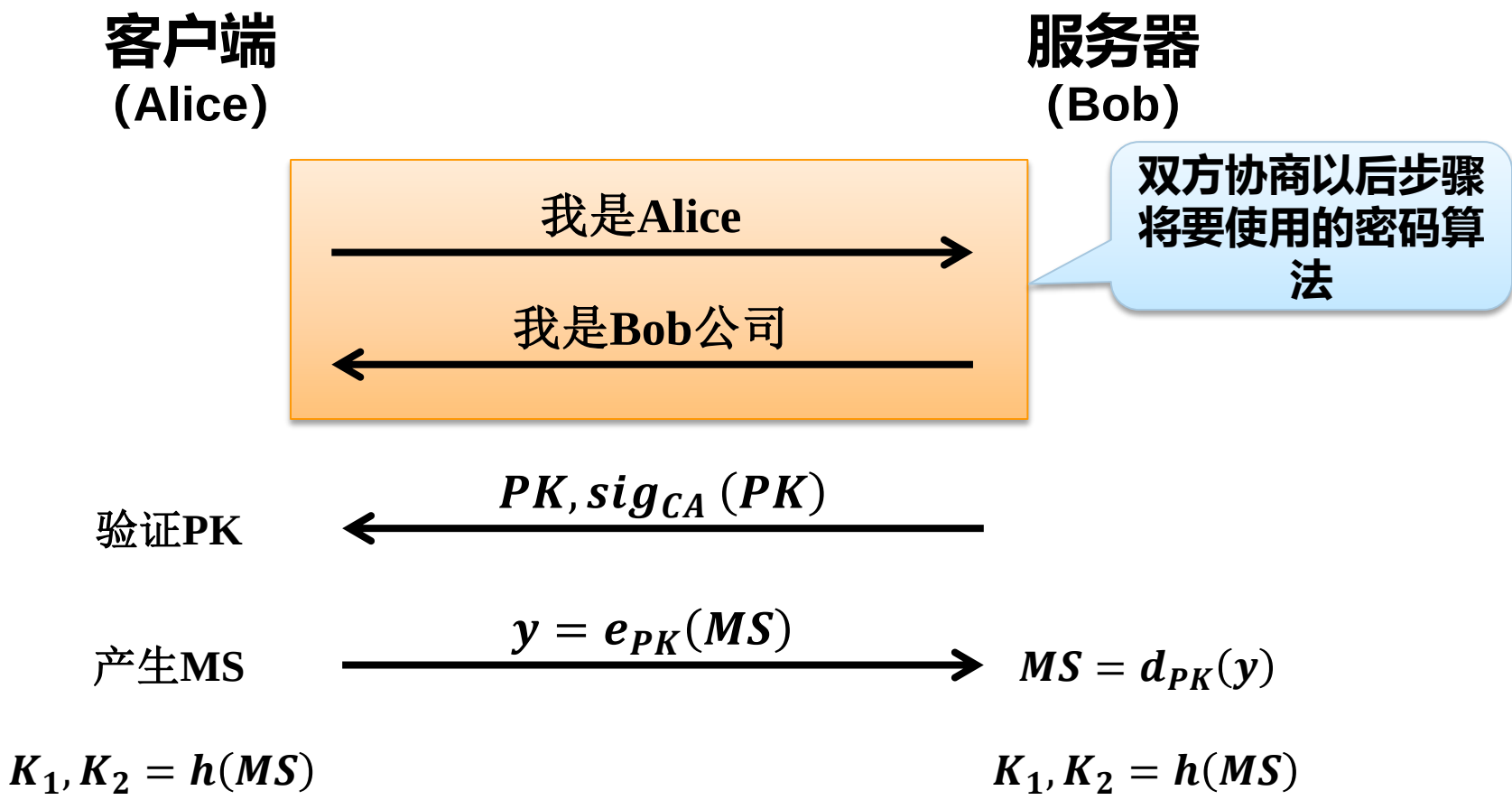
PKI简介

- 安全套阶层SSL



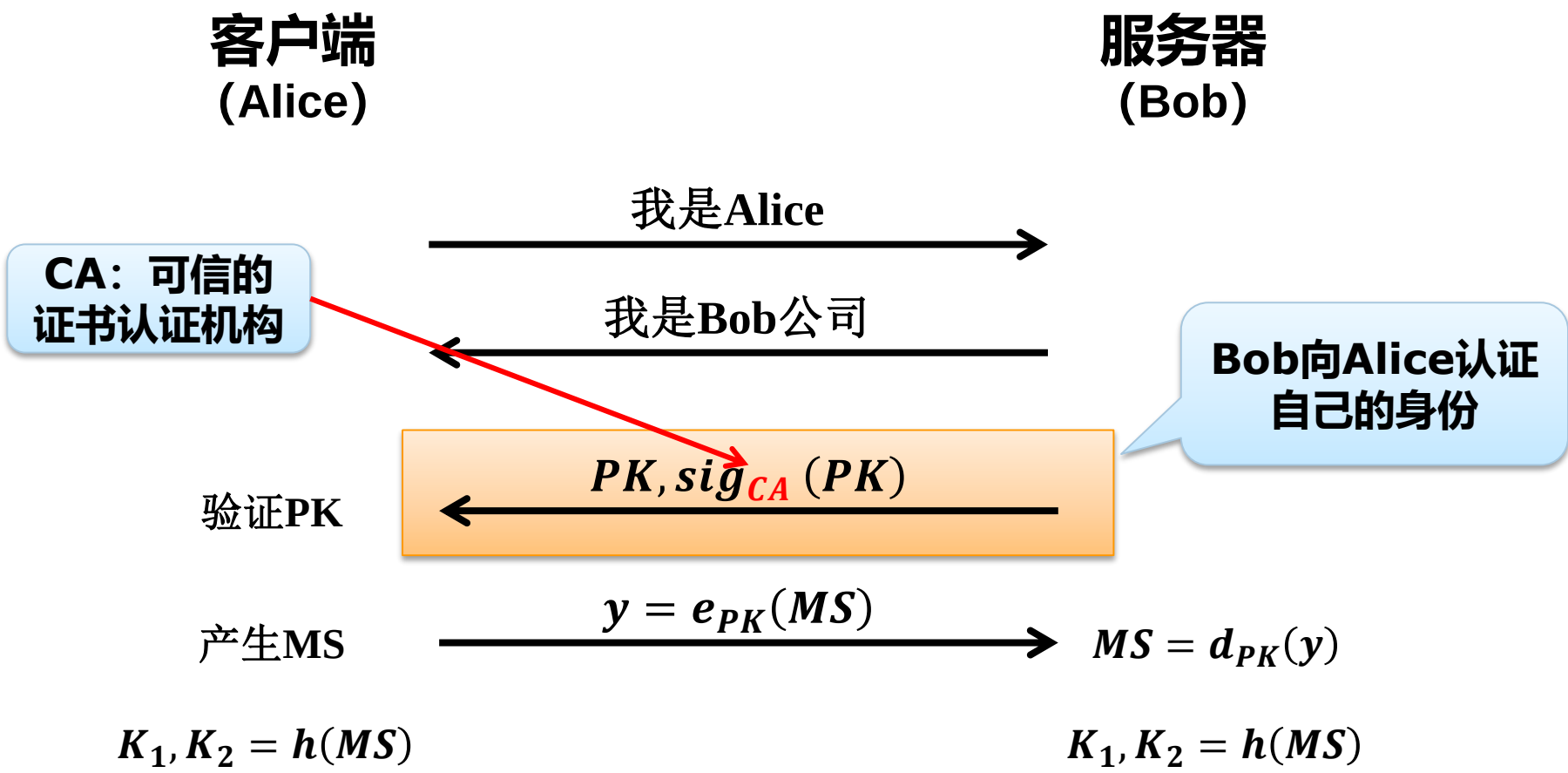
PKI简介

❖ 安全套阶层SSL



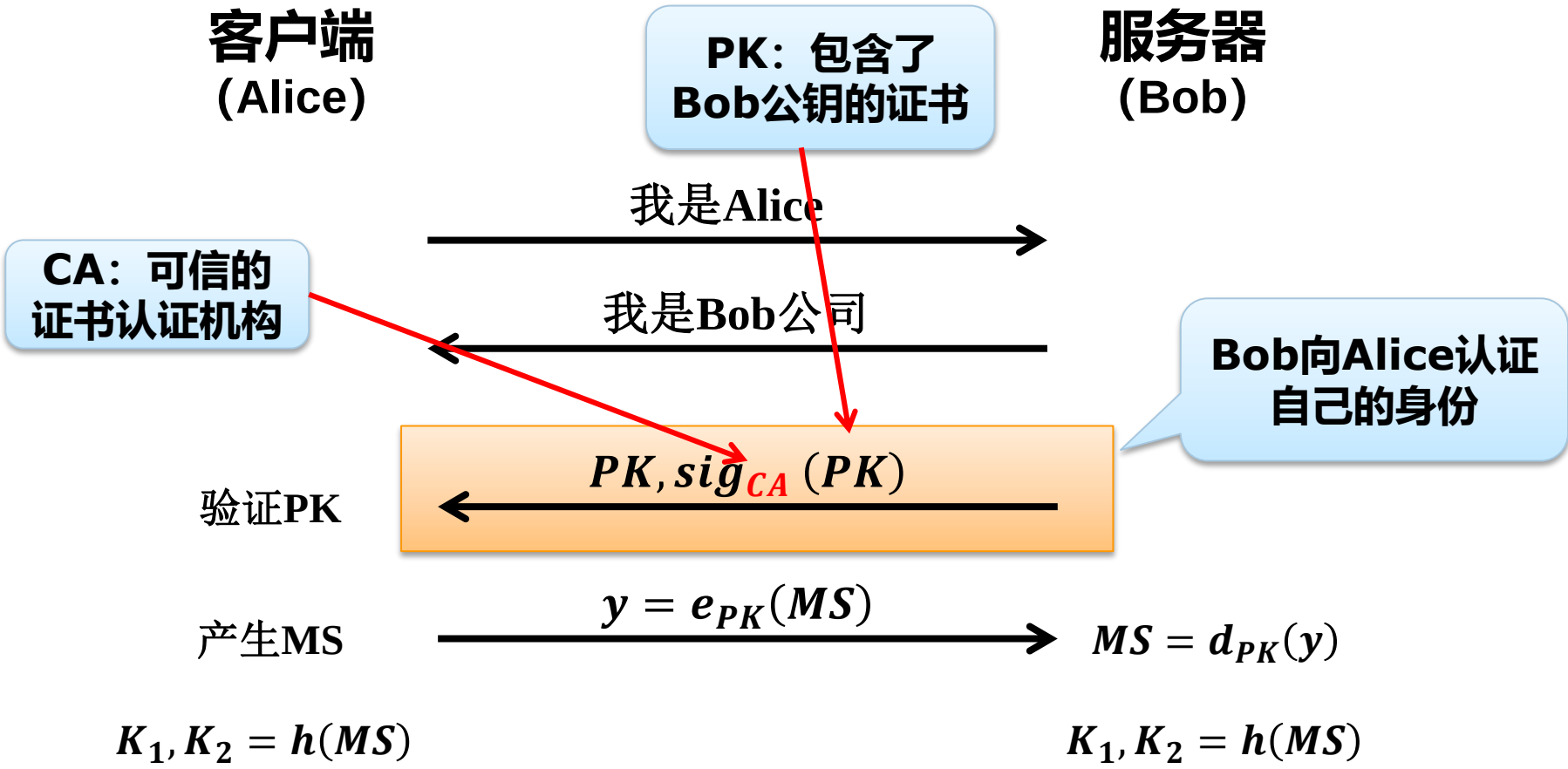
PKI简介

❖ 安全套阶层SSL



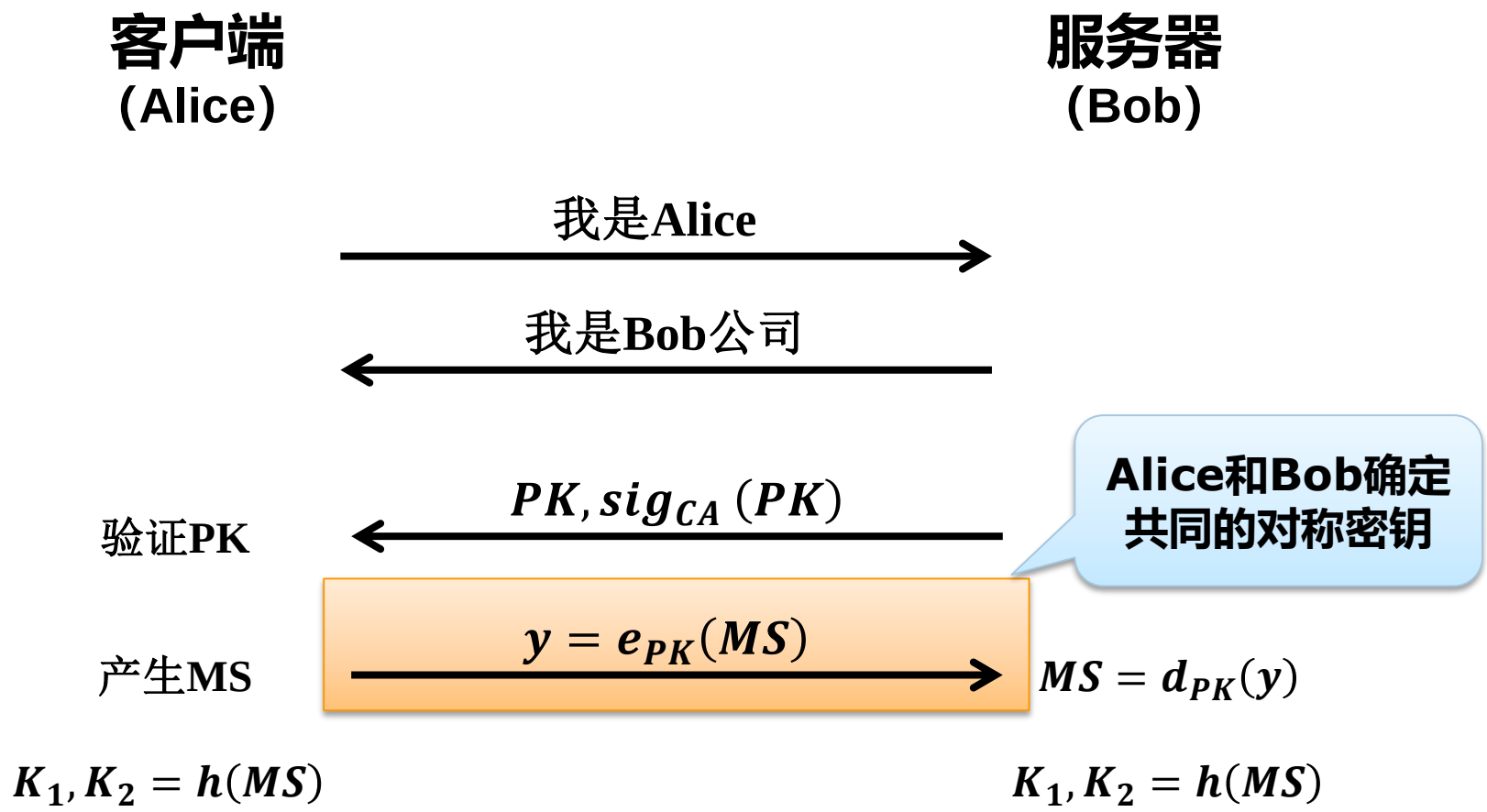
PKI简介

❖ 安全套阶层SSL



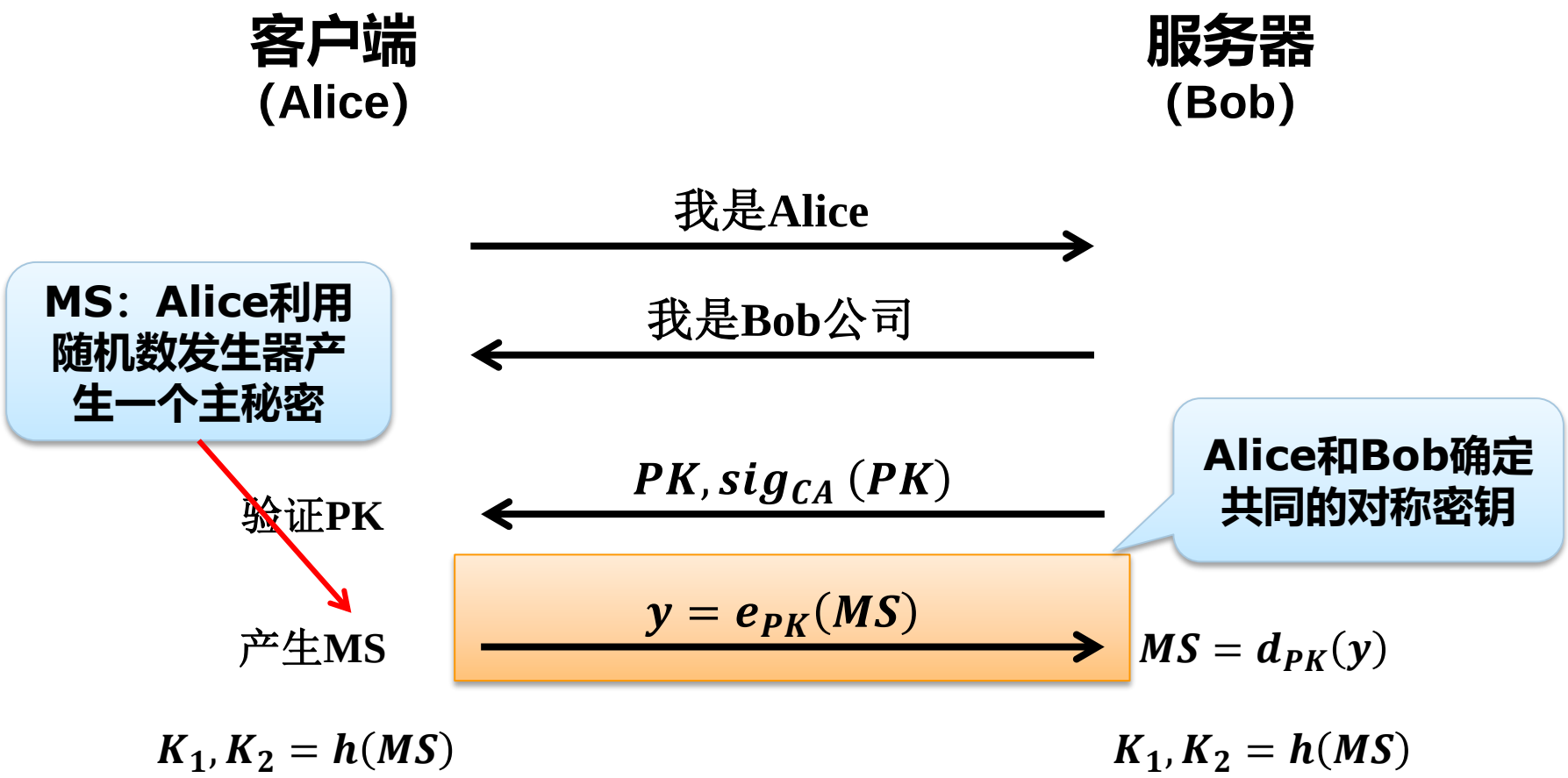
PKI简介

- 安全套阶层SSL



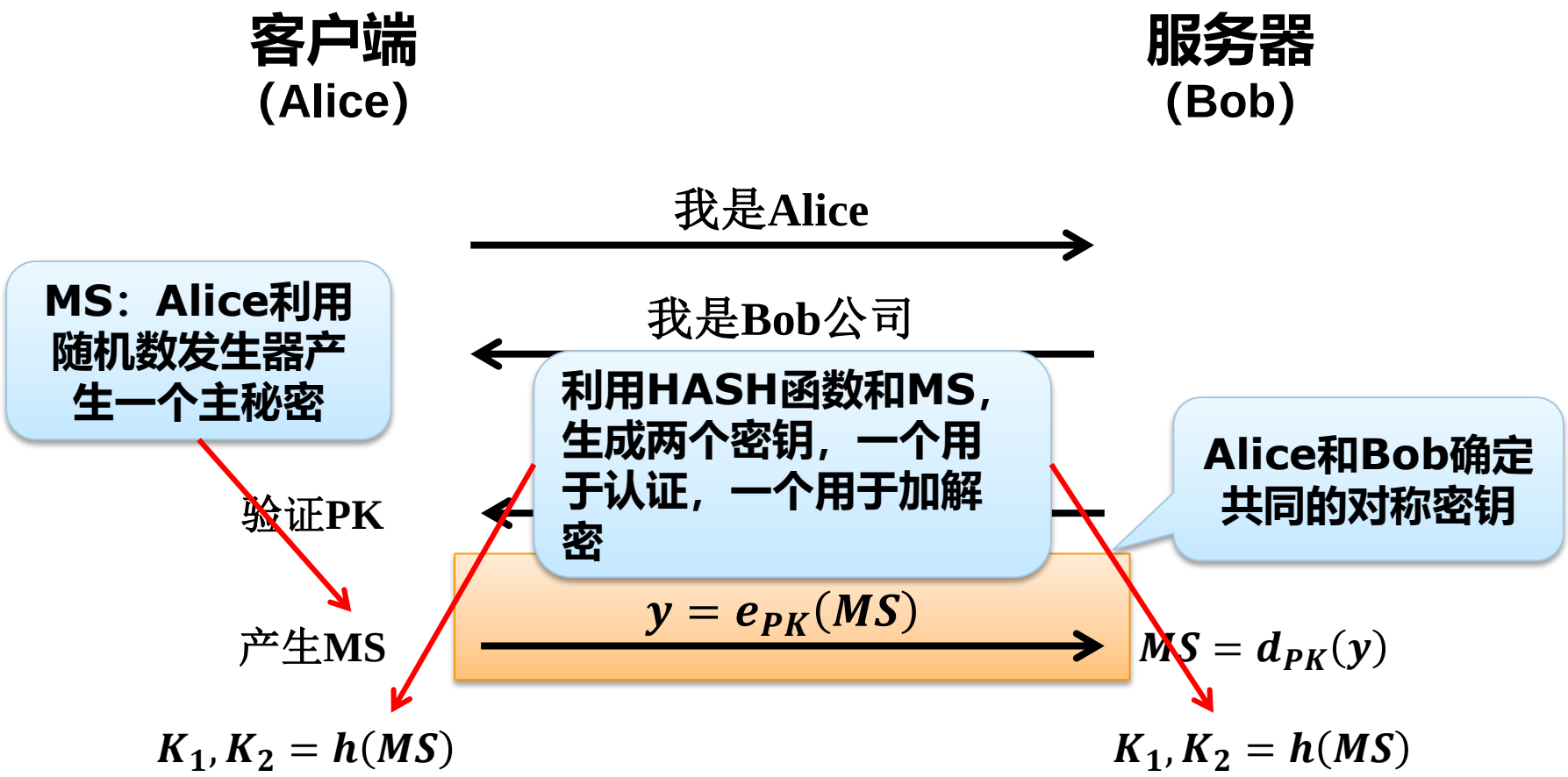
PKI简介

- 安全套阶层SSL



PKI简介

- 安全套阶层SSL



证书

- 证书是PKI的基本构建块。最简单的证书形式是证书把一个身份和一个公钥绑定。

$$Cert(Alice) = \underbrace{ID_{Alice}}_{\text{身份}} \| \underbrace{ver_{Alice}}_{\text{公钥}} \| \underbrace{sig_{CA}(ID_{Alice} \| ver_{Alice})}_{\text{CA的签名}}$$

信任模型

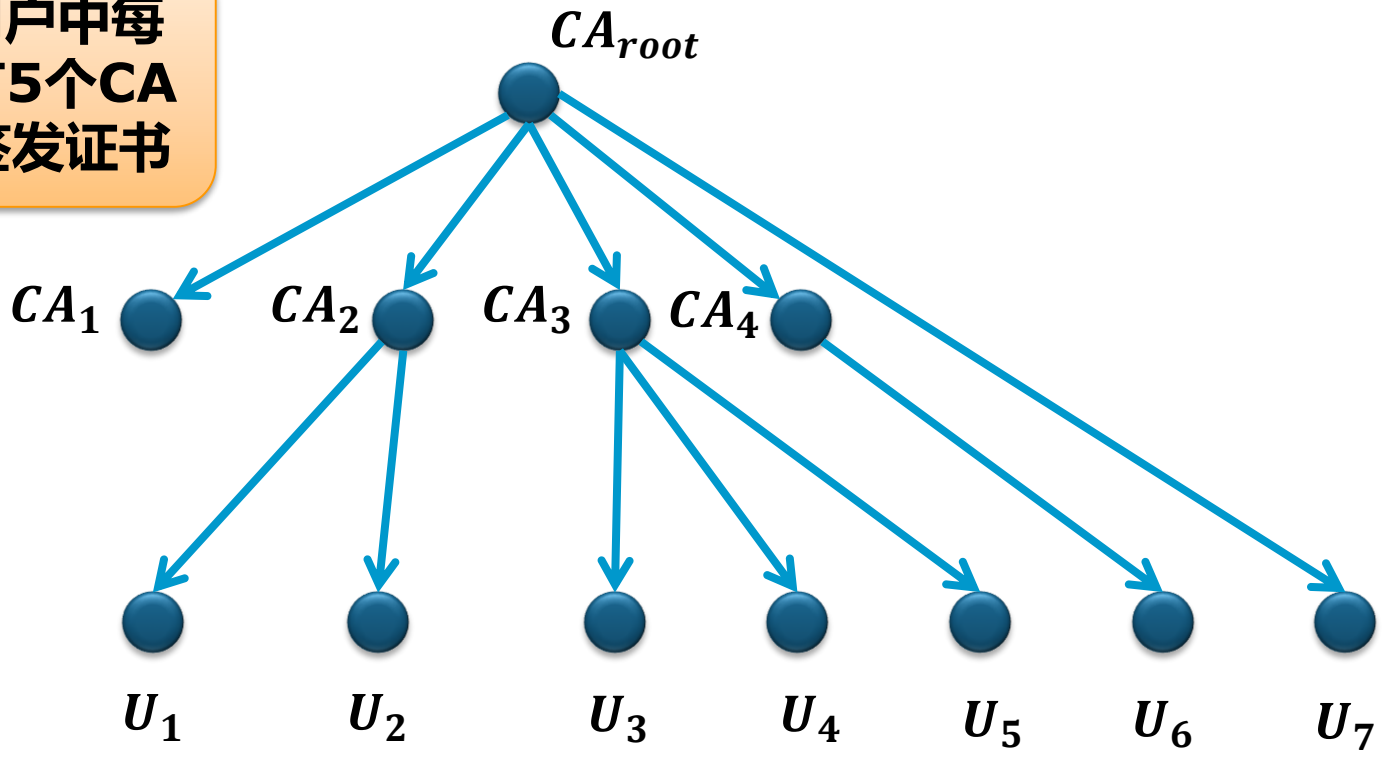
- 很多情况下，证书并不是由被信任的一个CA直接签发，而是从被信任的CA到一个给定的证书有一个证书路径。证书路径上的每一个证书都由前一个证书签名。通过验证证书路径上所有的证书，用户可以相信证书路径上最后一个证书是可信的。

- 证书信任模型的例子：
  - 严格层次模型
  - 网络化PKI模型
  - Web浏览器模型
  - 用户为中心的模型

信任模型

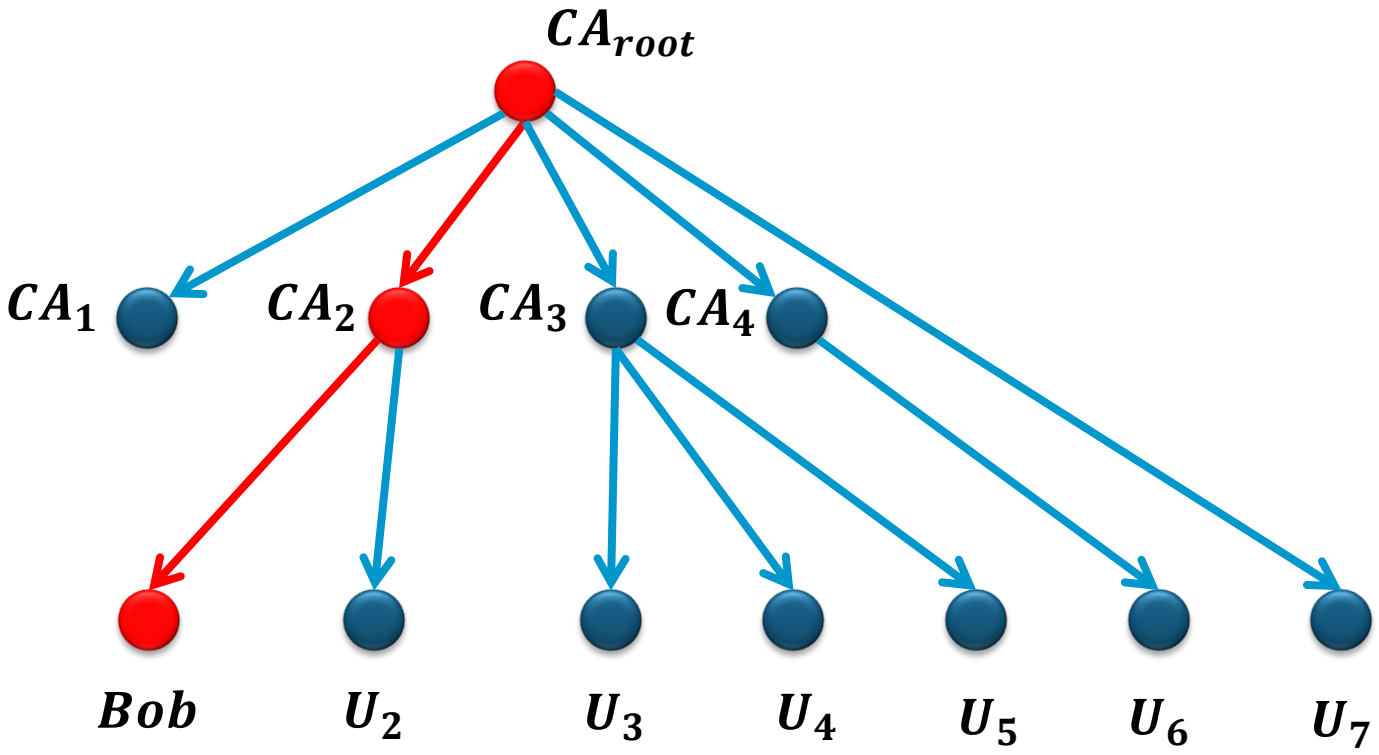
❖ 严格层次模型

7个终端用户中每一个都是有5个CA中的一个签发证书



信任模型

❖严格层次模型



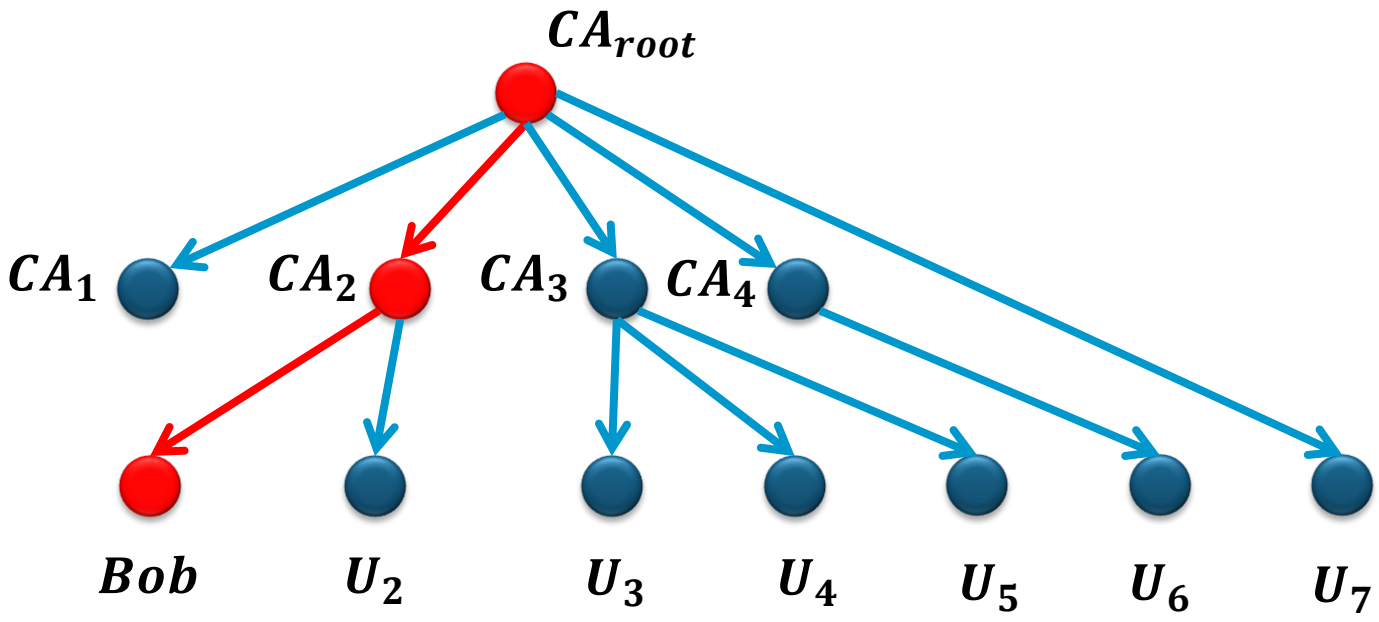
假设：Alice想验证Bob的身份

信任模型

❖严格层次模型

- Bob以如下路径发送所有的证书给Alice

$$CA_{root} \rightarrow CA_1 \rightarrow Bob$$



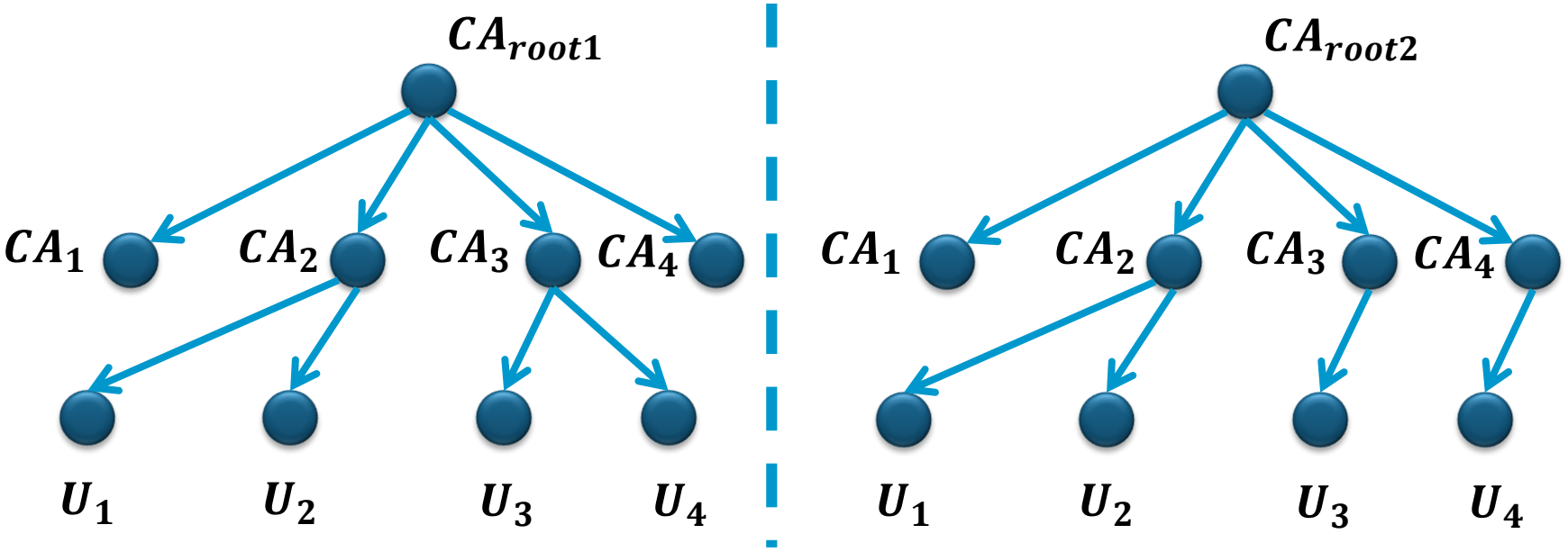
信任模型

❖严格层次模型

- Alice知道CA_{root}的验证密钥$ver_{CA_{root}}$，以及Bob提供的证书路径，Alice按照如下方式进行验证：
  1. Alice用$ver_{CA_{root}}$验证$Cert(CA_{root})$；
  2. Alice用$ver_{CA_{root}}$验证$Cert(CA_1)$；
  3. Alice从$Cert(CA_1)$中提取公钥ver_{CA_1}；
  4. Alice用ver_{CA_1}验证$Cert(Bob)$；
  5. Alice用$Cert(Bob)$中提取Bob的公钥

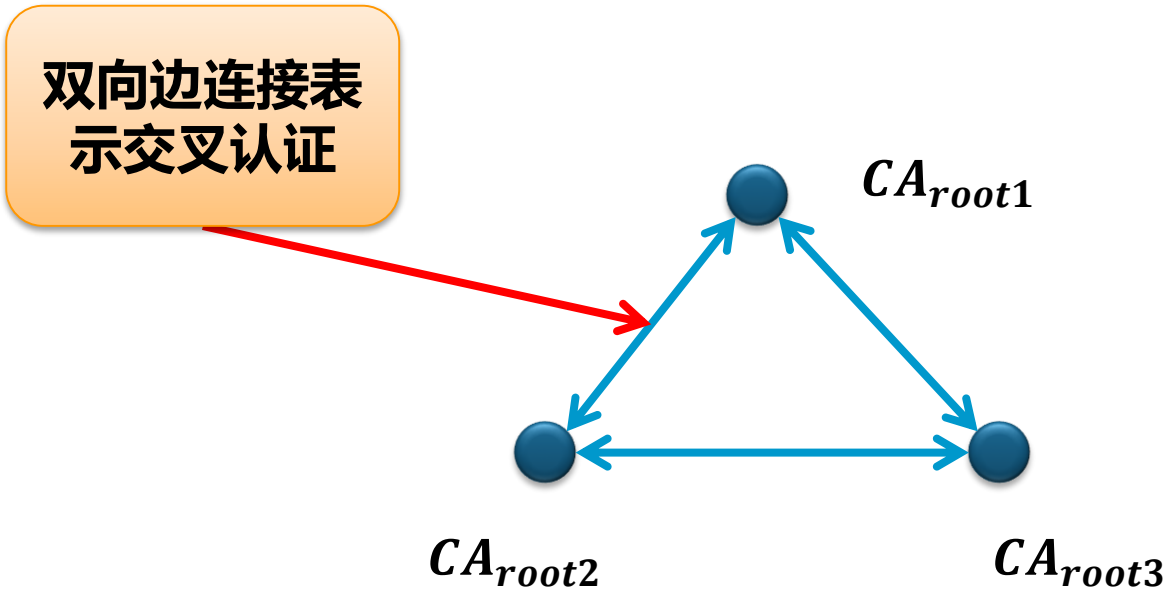
信任模型

- 网络化PKI模型：
  - 严格层次网络对于单一组织机构能够有效运行；
  - 如何将两个不同PKI域的CA连接起来呢？



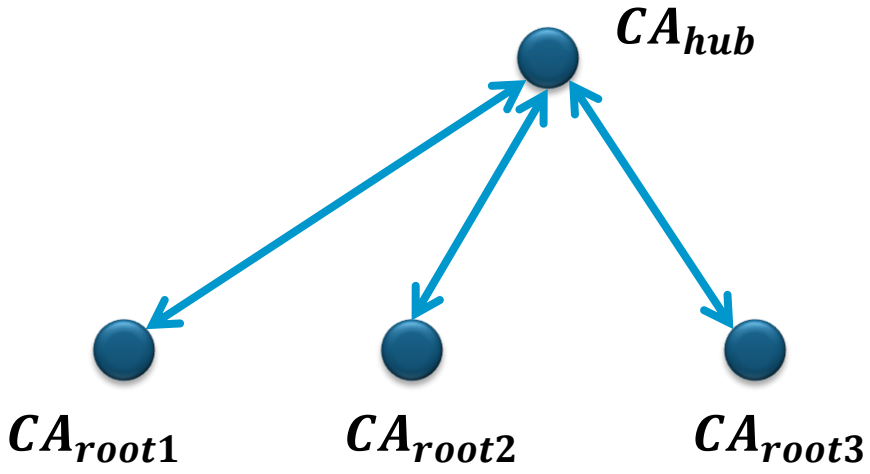
信任模型

- 网络化PKI模型：
  - 在网络模型配置中，所有的根CA彼此相互认证，因此，n个CA需要n(n-1)次交叉认证



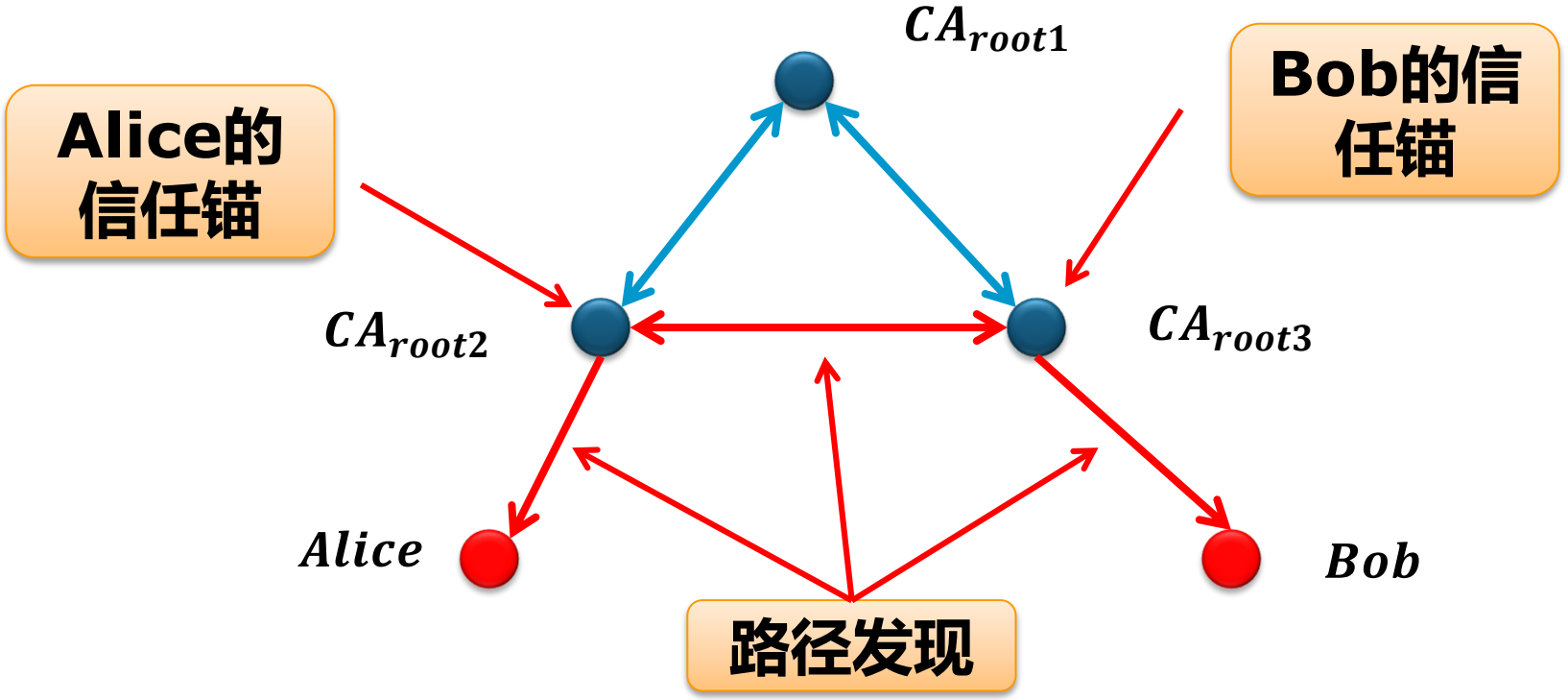
信任模型

- 网络化PKI模型：
  - “桥-代理”模型，这种模式中n个根CA都独立的与一个新的桥CA交叉认证，需要交叉认证的次数为2n。



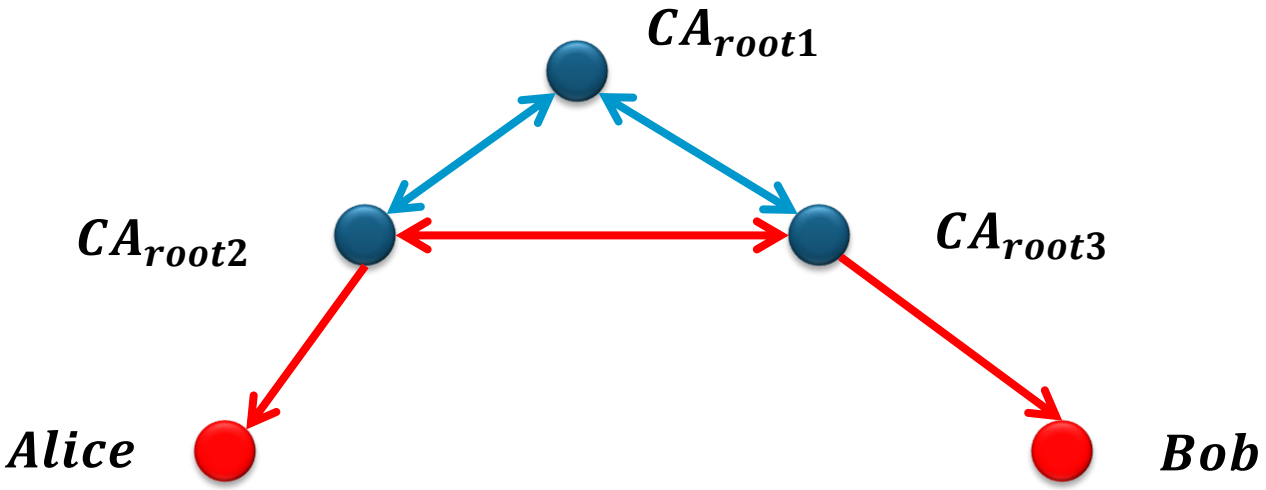
信任模型

- 网络化PKI模型：
  - 在网络化PKI模型中，在两个不同PKI域中的用户如何相互发现对方呢?



信任模型

- 网络化PKI模型：
  - Bob向Alice发送的证书信任路径是$CA_{root3} \rightarrow Bob$；
  - Alice需要找到被CA_{root2}签发的CA_{root3}证书，理想情况下Alice可以从CA_{root2}的目录服务器上找到。



信任模型

- 网络化PKI模型：
  - Bob向Alice发送的证书信任路径是$CA_{root3} \rightarrow Bob$；
  - Alice从CA_{root1}得到CA_{hub}证书的一个副本，这个证书是由CA_{root1}签发的；
  - Alice从CA_{hub}维护的目录服务器上查找由CA_{hub}签发的CA_{root3}的证书