《现代密码学》第九章

密钥管理（一）

上章内容回顾

- 数字签名的基本概念
- 一般数字签名算法
  - RSA数字签名技术
  - DSA数字签名技术
  - 基于离散对数的数字签名
  - 椭圆曲线数字签名
- 加密认证方式与签密

本章主要内容

- 密钥管理简介
- 密钥分配
- 密钥协商
- PKI及数字证书简介
- 秘密共享
- 密钥托管

本章主要内容

- 密钥管理简介
- 密钥分配
- 密钥协商
- PKI及数字证书简介
- 秘密共享
- 密钥托管

密钥管理简介

- 柯克霍夫斯原则(Kerckhoffs' Principle)

即使密码系统的任何细节已为人悉知，只要密钥未泄漏，它也应是安全的(19世纪).

- 密钥安全：三分技术，七分管理
- 密钥管理就是在授权各方之间实现密钥关系的建立和维护的一整套技术和程序。
- 在一定的安全策略指导下完成密钥从产生到最终销毁的整个过程，包括密钥的生成、建立（分配和协商）、存储、使用、备份/恢复、撤销、更新、存档和销毁等.

密钥管理简介

- **密钥生成**

密钥生成是密钥生命周期的基础阶段：

1) 密钥的生成一般首先通过密钥生成器借助于某种噪声源产生具有较好统计分析特性的序列，以保障生成密钥的随机性和不可预测性，然后再对这些序列进行各种随机性检验以确保其具有较好的密码特性。

2) 用户可以自己生成所需的密钥，也可以从可信中心或密钥管理中心申请，密钥长度要适中。

3) 不同的密码体制，其密钥的具体生成方法一般是不相同的，与相应的密码体制或标准相联系。

密钥管理简介

- **密钥建立**

密钥的建立就是使密钥安全（完整、保密）到达密钥使用的各实体对象，通常分为密钥分配和密钥协商。

密钥管理简介

• 密钥存储

密钥的安全存储是针对**静态密钥**的保护，通常有两种方法：

- 基于口令的软保护；

  文件形式或利用确定算法来生成密钥。

- 基于硬件的物理保护；

  存入专门密码装置中（存储型、智能型）。

密钥管理简介

- **密钥使用**

利用密钥进行正常的密码操作，如加密、解密、签名等，通常情况下，密钥在有效期之内都可以使用。

应注意使用环境对密钥的安全性的影响。

密钥管理简介

- **密钥备份：** 指密钥处于使用状态时的短期存储，为密钥的恢复提供密钥源，要求安全方式存储密钥，防止密钥泄露。
- **密钥恢复：** 从备份或存档中获取密钥的过程称为密钥恢复。若密钥丧失但未被泄露，就可以用安全方式从密钥备份中恢复。

密钥管理简介

- **密钥撤销：** 若密钥丢失或在密钥过期之前，需要将它从正常使用的集合中删除。
- **密钥更新：** 在密钥有效期快结束时，如果需要继续使用相应密码体制，为保证密钥的安全性，该密钥需要由一个新的密钥来取代，这就是密钥更新。密钥更新可以通过再生密钥取代原有密钥的方式来实现。

密钥管理简介

- **密钥存档：** 当密钥不再正常时，需要对其进行存档，以便在某种情况下特别需要时（如解决争议）能够对其进行检索。存档是指对过了有效期的密钥进行长期的离线保存，密钥的后运行阶段工作.

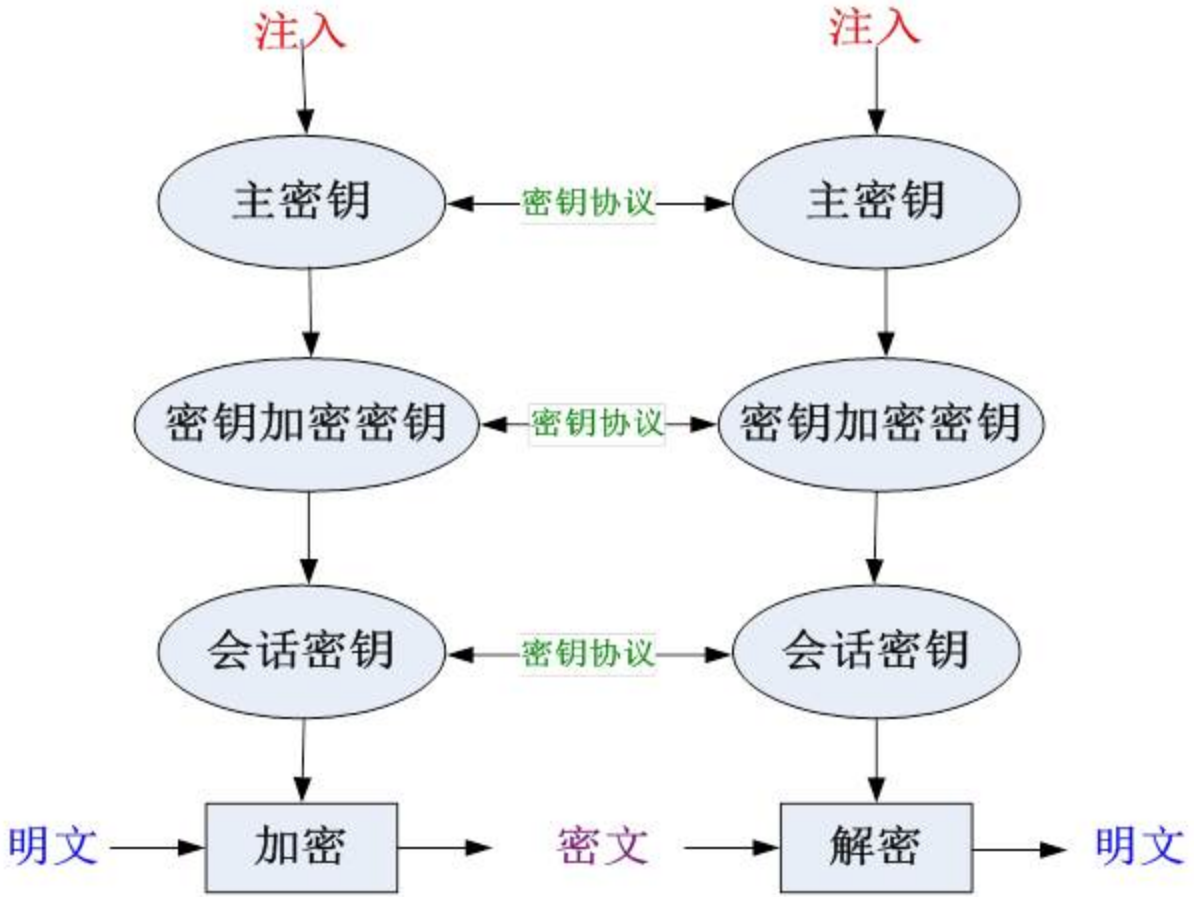
- **密钥销毁：** 对于不再需要使用的密钥，要将其所有复本销毁，而不能再出现。

密钥管理简介

• 生命周期

- **使用前状态：**密钥不能用于正常的密码操作。
- **使用状态：**密钥是可用的，并处于正常使用中。
- **使用后状态：**密钥不再正常使用，但为了某种目的对其进行离线访问是可行。
- **过期状态：**密钥不再使用，所有密钥记录被删除。

密钥管理简介



密钥分层管理示例

密钥管理简介

- **会话密钥（Session Key）**

  在一次通信或数据交换中，用户之间所使用的密钥，是由通信用户之间进行协商得到的。它一般是动态地、仅在需要进行会话数据加密时产生，并在使用完毕后立即进行清除掉的，也称为数据加密密钥(Data Encrypting Key)。

- **密钥加密密钥（Key Encrypting Key）**

  一般是用来对传输的会话密钥进行加密时采用的密钥，又称为二级密钥(Secondary Key)。密钥加密密钥所保护的对象是实际用来保护通信或文件数据的会话密钥。

- **主密钥（Master Key）**

  对应于层次化密钥结构中的最高层次，它是对密钥加密密钥进行加密的密钥，主密钥应受到严格的保护。

本章主要内容

- 密钥管理简介
- **密钥分配**
- 密钥协商
- PKI及数字证书简介
- 秘密共享
- 密钥托管

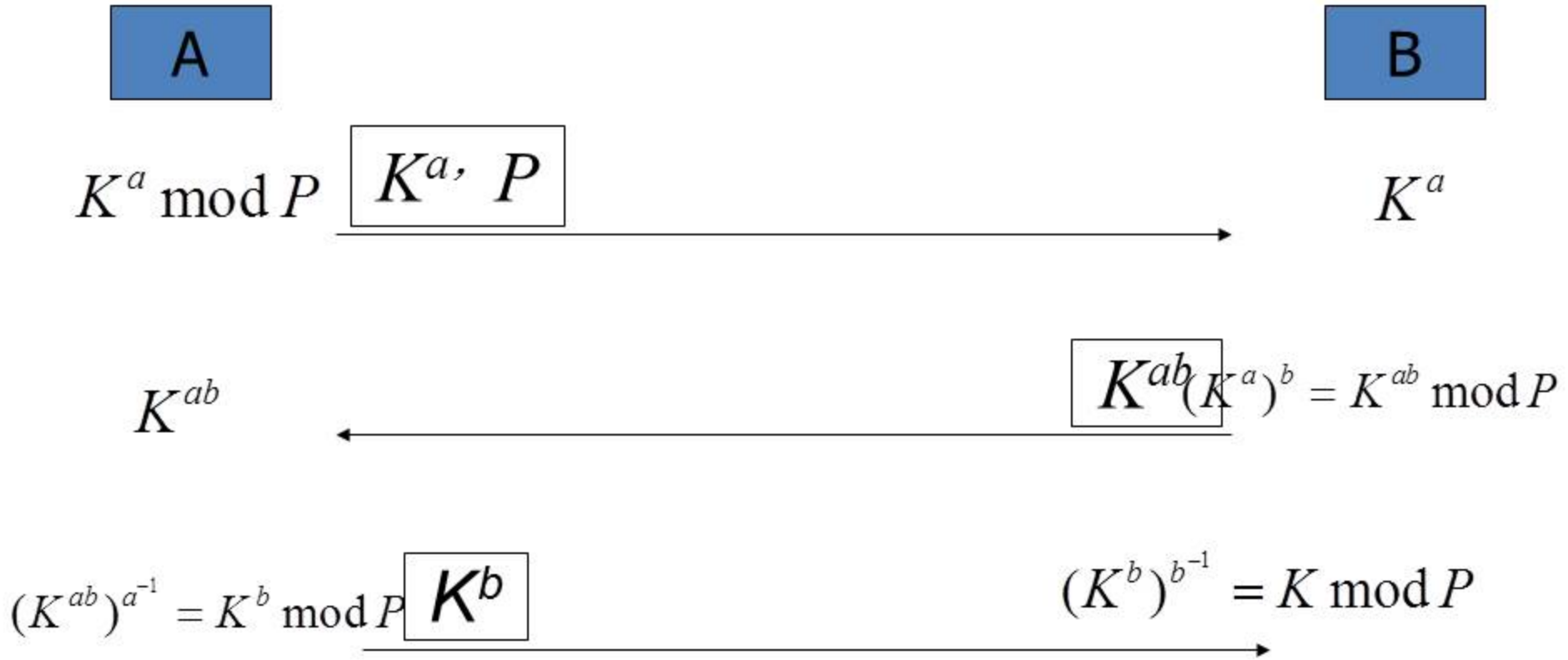
密钥分配

1）无密钥分配

假设A希望传递密钥K给B

- 随机选取素数$P>K$，选取Z_P^*中随机数a，计算K^a，将(K^a, P)发送给B；
- B收到A发送的信息后，选取Z_P^*中随机数b，计算$(K^a)^b$，将$((K^a)^b)$发送给A；
- A收到B发送的信息后，计算$((K^a)^b)^{a\wedge\{-1\}}=K^b$，将$K^b$发送给B；
- B收到A发送的信息后，计算$(K^b)^{b\wedge\{-1\}}=K$；

密钥分配


A
B
$K^a \bmod P$
$K^a,\ P$
K^a
K^{ab}
K^{ab}
$(K^a)^b = K^{ab} \bmod P$
$(K^{ab})^{a^{-1}} = K^b \bmod P$
K^b
$(K^b)^{b^{-1}} = K \bmod P$

密钥分配

2）对称密码系统密钥分配

两个用户（主机、进程、应用程序）在用对称密码体制进行会话密钥K_s分配，需有共享的密钥加密密钥。

用户A和B共享密钥加密密钥的方式：

(1) A、B事先共享一个对称密钥（无中心）.

(2) A和B分别与第三方C共享对称密钥（有中心）

.

密钥分配

（1）无中心的密钥分配(简化版)

K_{AB}为两个用户A和B的共享密钥加密密钥,A与B建立会话密钥K_s:

① A选取随机会话密钥，用K_{AB}加密后发送给B.

密钥分配

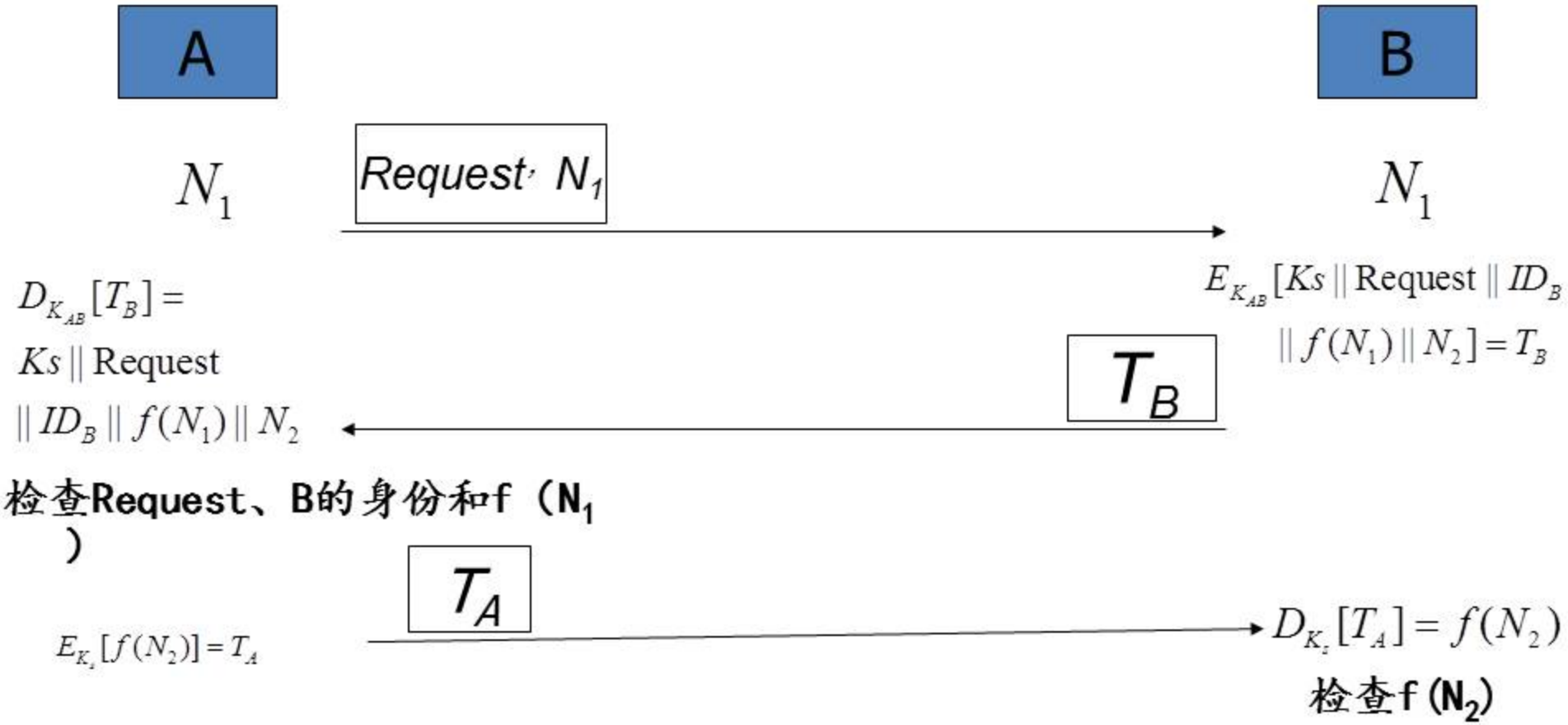
（1）无中心的密钥分配（改进版）

① A向B发出建立会话密钥的请求和一个随机数 N_1.

② B用共享密钥K_{AB}对应答的消息加密，并发送给A；应答的消息包括：B选取的会话密钥K_S、B的身份、f(N_1)和另一个随机数N_2.

③ A使用新建立的会话密钥K_S对f(N_2)加密后发回给B.

密钥分配


A
B
N_1
Request, N_1
N_1
$E_{K_{AB}}[Ks \| \text{Request} \| ID_B \| f(N_1) \| N_2] = T_B$
$D_{K_{AB}}[T_B] =$
$Ks \| \text{Request} \| ID_B \| f(N_1) \| N_2$
T_B
检查Request、B的身份和f（N_1）
T_A
$E_{K_s}[f(N_2)] = T_A$
$D_{K_s}[T_A] = f(N_2)$
检查f（N_2）

密钥分配

（2）有中心的密钥分配

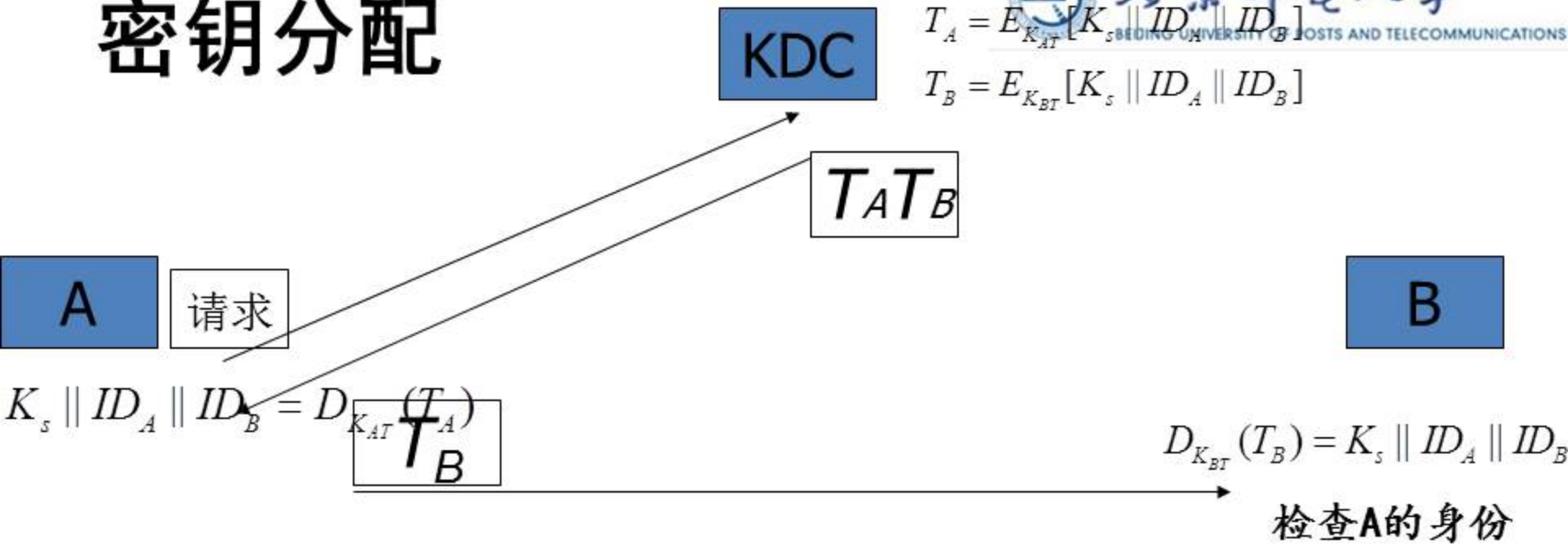
密钥分配中心（KDC）与所有用户预设共享密钥加密密钥：

用户A与KDC有共享的密钥K_{AT}，

用户B与KDC有共享的密钥K_{BT}；

会话密钥K_S可以由通信双方选取，亦可由KDC选取。

密钥分配



(简化版)会话密钥K_S由KDC选取

① A向KDC发出和B会话的请求

② KDC选取会话密钥，分别用和A和B的共享密钥加密的会话密钥$E_{K_{AT}}(K_s \| ID_A \| ID_B)$、$E_{K_{BT}}(K_s \| ID_A \| ID_B)$传回给A;

③ A解密得到会话密钥K_s，并将消息转发给B，B解密得到与A的会话密钥K_s

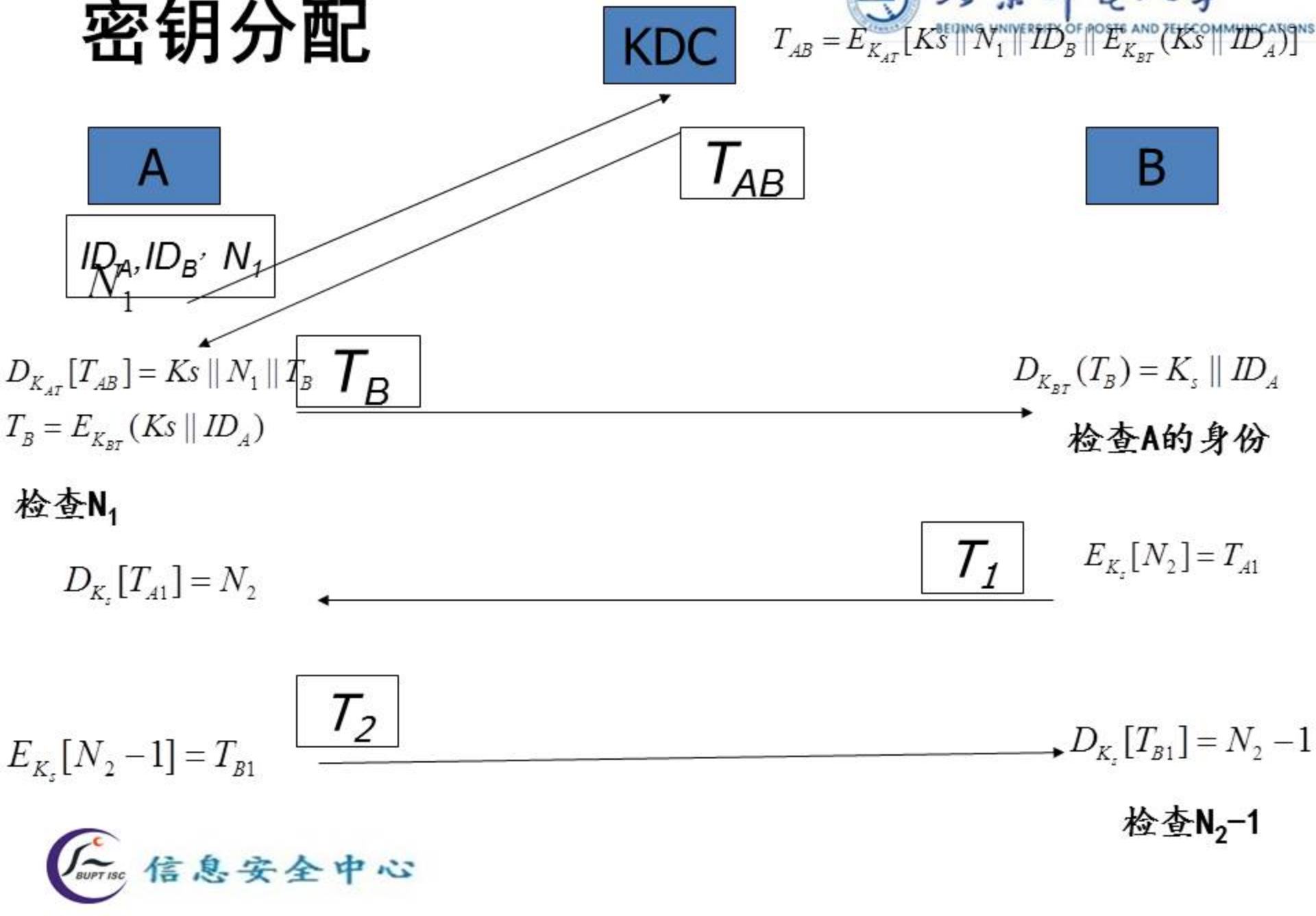
密钥分配

- **Needham-Schroeder密钥分发协议**

1978年由R. Needham和M. Schroeder设计，该协议是密钥分发技术的里程碑，许多密钥分发协议都是在其基础上发展而来的。

用户A与KDC有共享的密钥K_{AT}，用户B与KDC有共享的密钥K_{BT}；A与B建立会话密钥K_s，需要经过下列5个步骤：

密钥分配



密钥分配

① A向KDC发出会话密钥请求：Request$||N_1$

第1项是A和B的身份，第2项是这次业务的惟一识别符N_1，N_1为一次性随机数，可以是时戳、计数器或随机数。

② KDC为A的请求发出应答

$$E_{K_{AT}}(K_s||\ ID_B\ ||N_1||E_{K_{BT}}(K_s||ID_A\ ||N_1))$$

消息中包括A希望得到的两项内容：

一次性会话密钥K_s；A要通话的B的身份ID_B；随机数N_1消息中还有B希望得到的两项内容： 一次性会话密钥K_s；A的身份ID_A。

密钥分配

③ A存储会话密钥，并向B转发$E_{K_{BT}}(K_s \| ID_A \| N_1)$.

④ B用会话密钥K_s加密另一个随机数N_2，并将加密结果$E_{K_s}(N_2)$发送给A.

⑤ A以N_2-1作为对B的应答，并将应答用会话密钥加密后$E_{K_s}(N_2\text{-}1)$发送给B.

密钥分配

NS协议的缺点：

- B无法判断收到的K_s是否是新鲜的。一旦K_s暴露，敌手可以重放消息③并成功完成后续挑战应答协议。
- 即使敌手没有得到泄露的密钥K_s，A也无法通过消息②,④,⑤判断B已经知道会话密钥K_s. 这是因为敌手可以任意构造消息④.

① A→KDC: $\mathrm{ID_A}, \mathrm{ID_B}, N_1$
② KDC→ A: $\mathrm{E}_{K_{AT}}(K_s \| \mathrm{ID_B} \| N_1 \| \mathrm{E}_{K_{BT}}(K_s \| \mathrm{ID_A} \| N_1))$
③ A→M(B): $\mathrm{E}_{K_{BT}}(K_s \| \mathrm{ID_A} \| N_1)$
④ M(B)→A: N_M
⑤ A→P(B): $\mathbf{E}_{K_S}[\mathrm{D}_{K_S}(N_M) - 1]$

密钥分配

- 简化Kerberos协议

  Kerberos是MIT作为Athena计划的一部分开发的认证服务系统，而Kerberos密钥分配协议是Kerberos认证系统的一部分，认证协议共分六步，这里做若干简化。

用户A与KDC有共享的密钥K_{AT}，用户B与KDC有共享的密钥K_{BT}；A与B建立会话密钥K_s，，需要经过下列4个步骤：

密钥分配

① A向KDC发出会话密钥请求： A和B的身份

② KDC为A的请求发出应答

$$E_{K_{AT}}(K_s \| T_{KDC} \| L \| ID_B \| E_{K_{BT}}(K_s \| ID_A \| T_{KDC} \| L))$$

消息中包括A希望得到的两项内容：

- 一次性会话密钥K_s、A要通话的B的身份ID_B，及KDC的时间戳和生存时间$T_{KDC} \| L$；
- 消息中还有B希望得到的两项内容： 一次性会话密钥K_s；A的身份ID_A及KDC的时间戳和生存时间$T_{KDC} \| L$。

密钥分配

③ A解密判断时间戳和L的正确性，存储会话密钥K_s；向B转发$\mathrm{E}_{K_{BT}}(K_s \| \mathrm{ID_A} \| T_{KDC} \| L))$，同时用$K_s$加密自己的时戳$\mathrm{E}_{K_s}(\mathrm{ID_A} \| T_{\mathrm{A}})$;

④ B先用K_{BT}解密第一块消息，得到K_s. 再用K_s解密第二块消息，检查两次的时戳和生存期。

最后，B以T_A+1作为对A的应答，并将应答用会话密钥加密后$\mathrm{E}_{K_s}(T_{\mathrm{A}}+1)$发送给A。

密钥分配

- **有中心的密钥分配问题**

网络中如果用户数目非常多而且分布的地域非常广，一个KDC就无法承担为用户分配密钥的重任。问题的解决方法是使用多个KDC的分层结构。

- 同一范围的用户在进行保密通信时，由本地KDC为他们分配密钥。
- 两个不同范围的用户想获得共享密钥，则可通过各自的本地KDC，而两个本地KDC的沟通又需经过一个全局KDC。

密钥分配

3）基于公钥系统的密钥分配

网络中用户（主机、进程、应用程序）在用非对称密码体制进行会话密钥K_s分配，则每个用户需要事先选定自己的公钥/私钥(PK,SK)对作为密钥加密密钥。

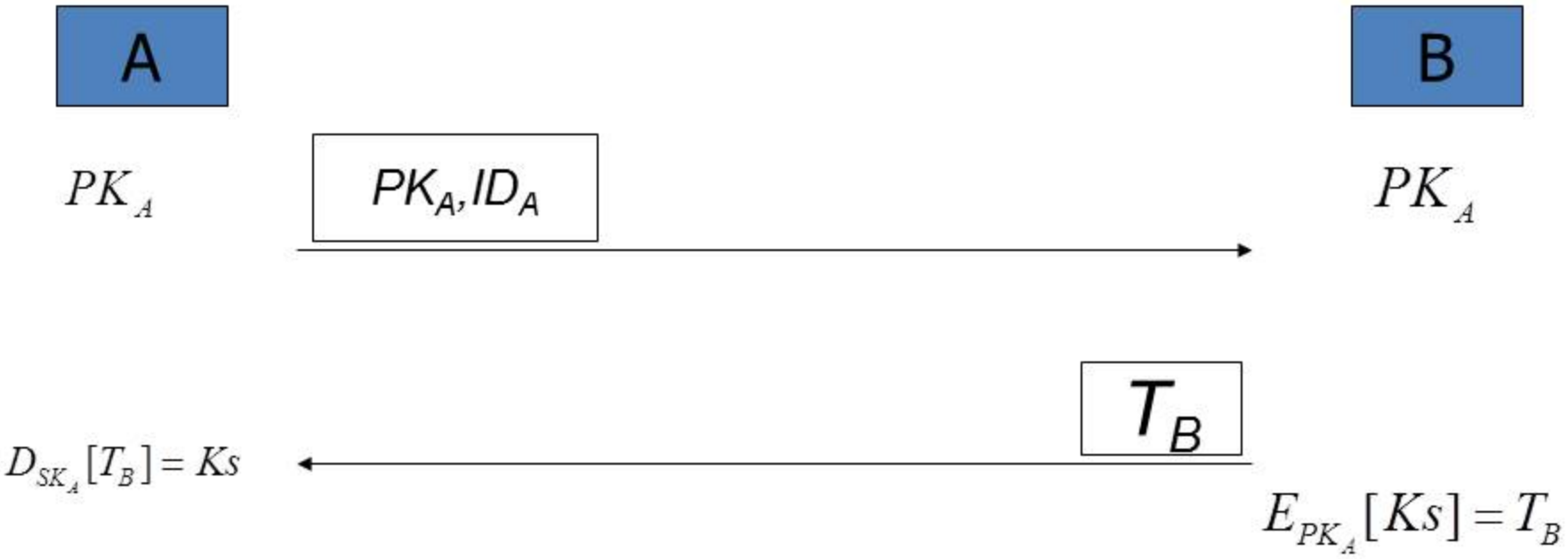
简化版协议：

① A把自己的身份$\mathrm{ID_A}$和公钥PK_A发送给B。

② B用A的公开钥PK_A加密随机选取的会话密钥K_s，发送$\mathrm{E}_{PK_A}(K_s \| \mathrm{ID_B})$给A。

只有A能解读②中的加密消息，得到K_s

密钥分配



本章主要内容

- 密钥管理简介
- 密钥分配
- **密钥协商**
- PKI及数字证书简介
- 秘密共享
- 密钥托管

密钥协商

- 密钥协商是保密通信双方（或更多方）通过公开信道的通信来共同形成秘密密钥的过程。一个密钥协商方案中，密钥的值是某个函数值，其输入量由两个成员（或更多方）提供。
- 密钥协商的结果是：参与协商的双方（或更多方）都将得到相同的密钥，同时，所得到的密钥对于其他任何方都是不可知的。

密钥协商

- **D-H密钥协商**

设p是一个大素数，$g \in Z_p$是模p本原元，p和g公开，所有用户均可获取，并可为所有用户所共有。

①用户A随机选取一个大数a，$0 \le a \le p-2$.

计算$K_a = g^a \pmod p$,并将结果传送给用户B.

②用户B随机选取一个大数b，$0 \le b \le p-2$.

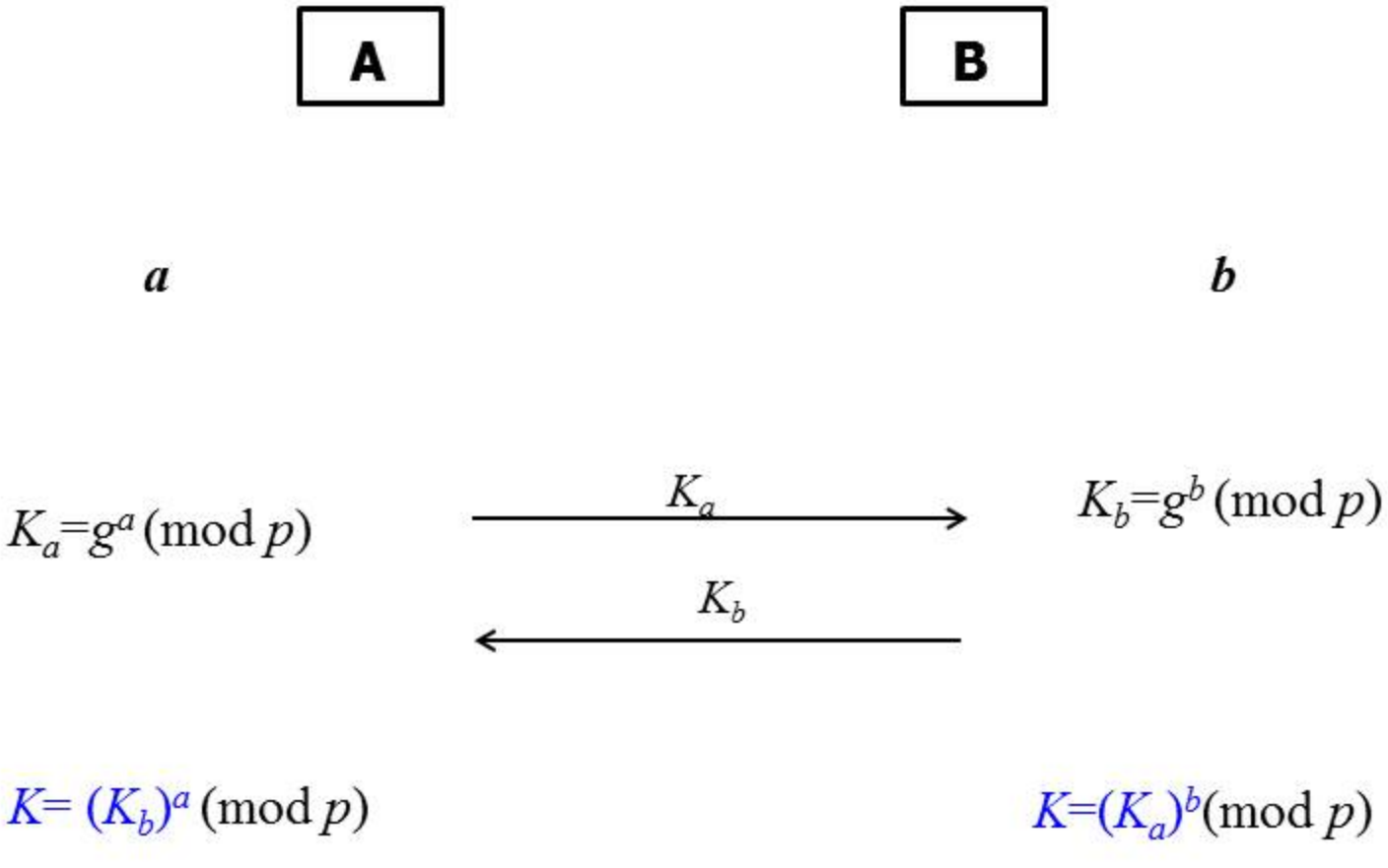
计算$K_b = g^b \pmod p$，并将结果传送给用户A.

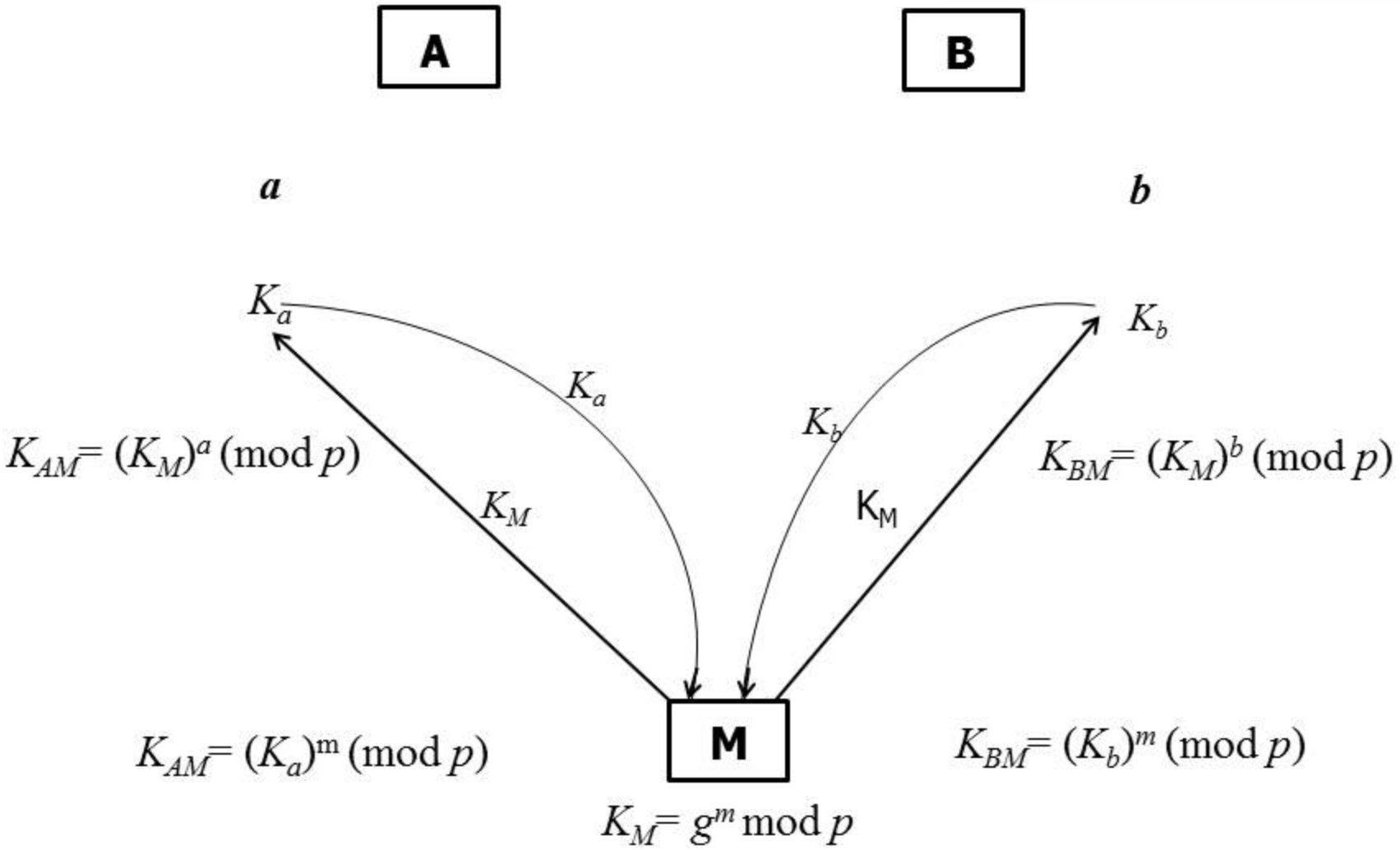
③用户A计算$K = (K_b)^a \pmod p$；

用户B计算$K = (K_a)^b \pmod p$.

用户A和用户B各自计算生成共同的会话密钥K.

正确性：$K = (K_b)^a = (g^b)^a = g^{ab} = (g^a)^b = (K_a)^b$.


A
B
a
b
$K_a=g^a \pmod p$
K_a
K_b
$K_b=g^b \pmod p$
$K=(K_b)^a \pmod p$
$K=(K_a)^b \pmod p$


A
B
a
b
K_a
K_b
K_a
K_b
$K_{AM} = (K_M)^a \pmod p$
$K_{BM} = (K_M)^b \pmod p$
K_M
K_M
M
$K_{AM} = (K_a)^m \pmod p$
$K_{BM} = (K_b)^m \pmod p$
$K_M = g^m \bmod p$

密钥协商

① A选择 $\alpha \in_u [1, p-1)$，计算 $g_\alpha \leftarrow g^\alpha \pmod p$；发送 g_α 给M（B）；

② M对某个 $m \in [1, p-1)$，计算 $g_m \leftarrow g^m \pmod p$；发送 g_m 给B；

③ B选择 $b \in_u [1, p-1)$，计算 $g_b \leftarrow g^b \pmod p$；发送 g_b 给M（A）；

④ M向A发送 g_m ；

⑤ A计算 $k_1 \leftarrow g_m^\alpha \pmod p$；

B计算 $k_2 \leftarrow g_m^b \pmod p$；

密钥协商

DH协议初步改进

A的签名算法为$\mathrm{Sig_A}$，签名验证算法为$\mathrm{Ver_A}$；
B的签名算法为$\mathrm{Sig_B}$，签名验证算法为$\mathrm{Ver_B}$.

设p是一个大素数，$g \in Z_p$是模p本原元，p和g公开，所有用户均可获取，并可为所有用户所共有。

①用户A随机选取一个大数a，$0 \le a \le p-2$.
计算$K_a = g^a \pmod p$,并将结果传送给用户B.

密钥协商

② 用户B随机选取一个大数b，$0 \le b \le p-2$.
计算$K_b = g^b \pmod p$，然后计算$K = (K_a)^b \pmod p$和

$$E_B = E_K[Sig_B(g^a \pmod p, g^b \pmod p))]$$

用户B将$(PK_B, g^b \pmod p, E_B)$传送给用户A。

③ 用户A先计算$K = (K_a)^b \pmod p$，解密E_B，再验证B签名的Ver_B有效性。确认有效后，计算

$$E_A = E_K[Sig_A(g^a \pmod p, g^b \pmod p))]$$

最后，把(PK_A, E_A)发给用户B。

④ 用户B解密后，验证A的签名Ver_A的有效性。

本节要点回顾

- 密钥管理简介
- 密钥分配
- 密钥协商

THE END!