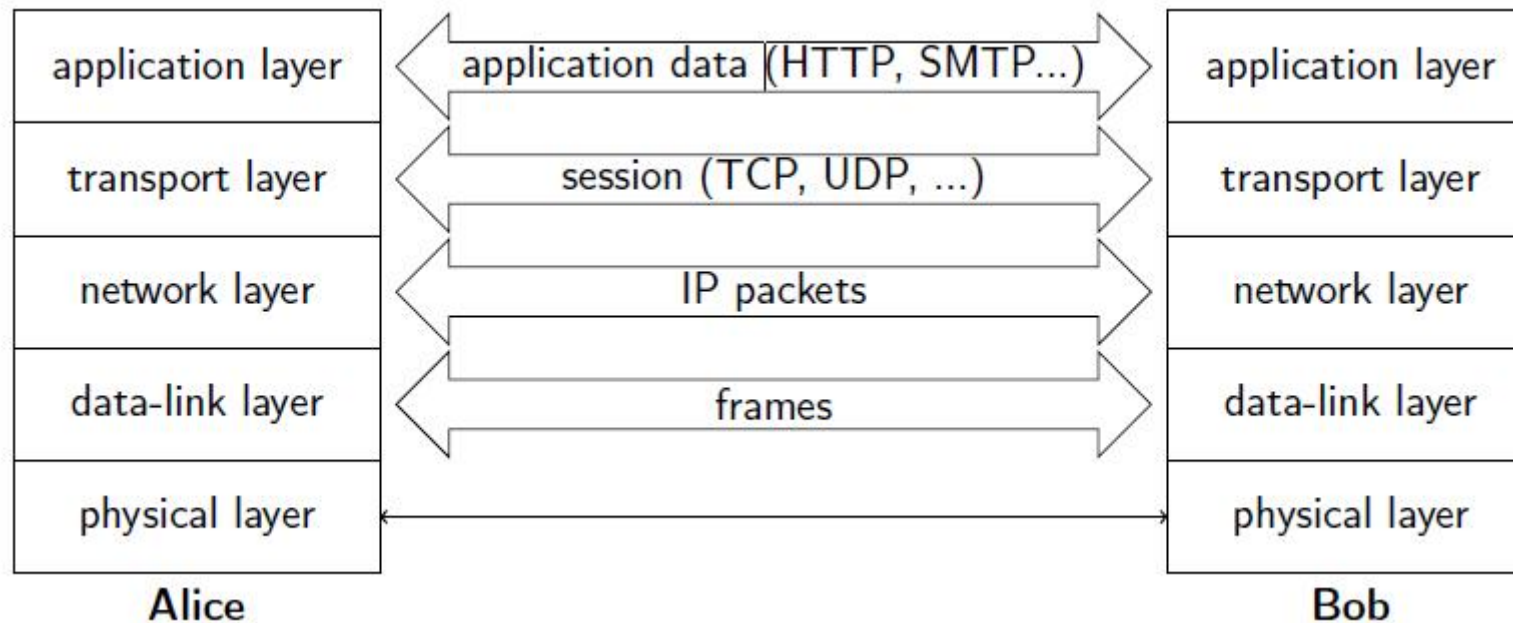


第2章 PGP



- application layer security (SSH, S-MIME, PGP,)
- transport layer security (TLS/SSL,)
- network layer security (IPsec,)
- data-link layer security (WEP, WPA, WPA2,)

内容提要

- PGP概述
- PGP加密与解密
- PGP生成和验证数字签名
- PGP加密+数字签名——解密+签名验证
- PGP信任网 (web of trust)
- PGP实例: GPG4Win

PGP概述

□ Philip Zimmermann设计

- ◆ 目前在瑞士运营公司Silent Cycle, 提供移动/桌面加密通信软件及服务, 目的是保护用户隐私。

□ OpenPGP→RFC4880



PGP概述：安全属性

- 保密性

 - ◆ 对称加密

- 完整性

 - ◆ 对消息摘要进行数字签名

- 身份认证

 - ◆ 数字签名

Building blocks of PGP

- 加解密
- 数字签名
- 压缩
- 电子邮件兼容性

PGP: 压缩

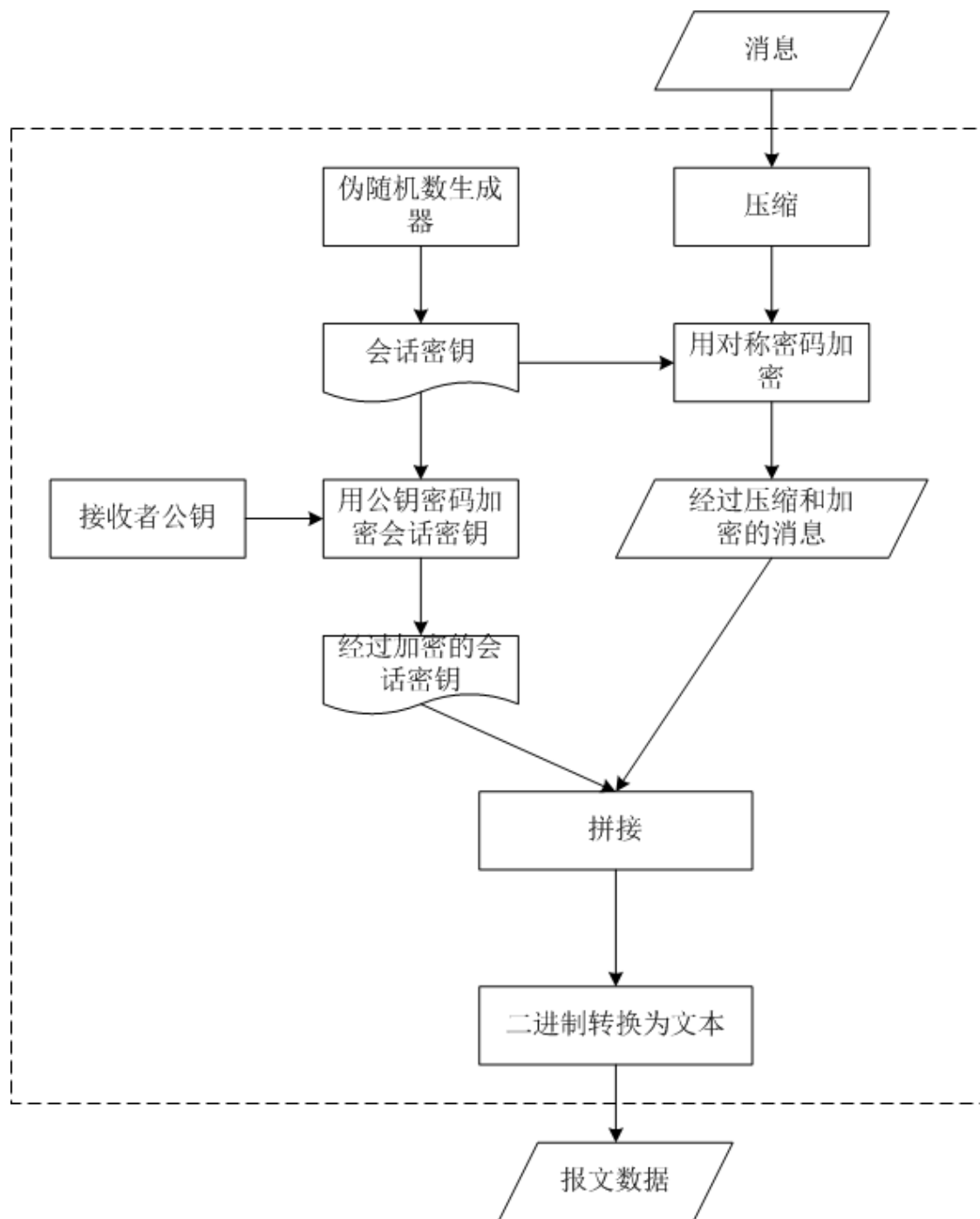
- PGP支持数据的压缩和解压，目的是提高数据存储和传输的效率，支持ZIP、ZLIB、BZIP2等格式

PGP: 电子邮件兼容性

- 电子邮件系统通常支持ASCII文本格式，而加解密通常是对二进制进行操作，因此需要进行兼容性处理，二进制与Radix-64互相转换
- Radix-64编码
 - ◆ 基于base64，增加了检测数据错误的校验和。
 - ◆ Base64编码是一种可以将任何二进制数据都用A~Z、a~z、0~9、+、/共64个字符外加=（用于末尾填充）来表示的编码方法

PGP加密和解密

PGP 的加密流程



PGP: 加密

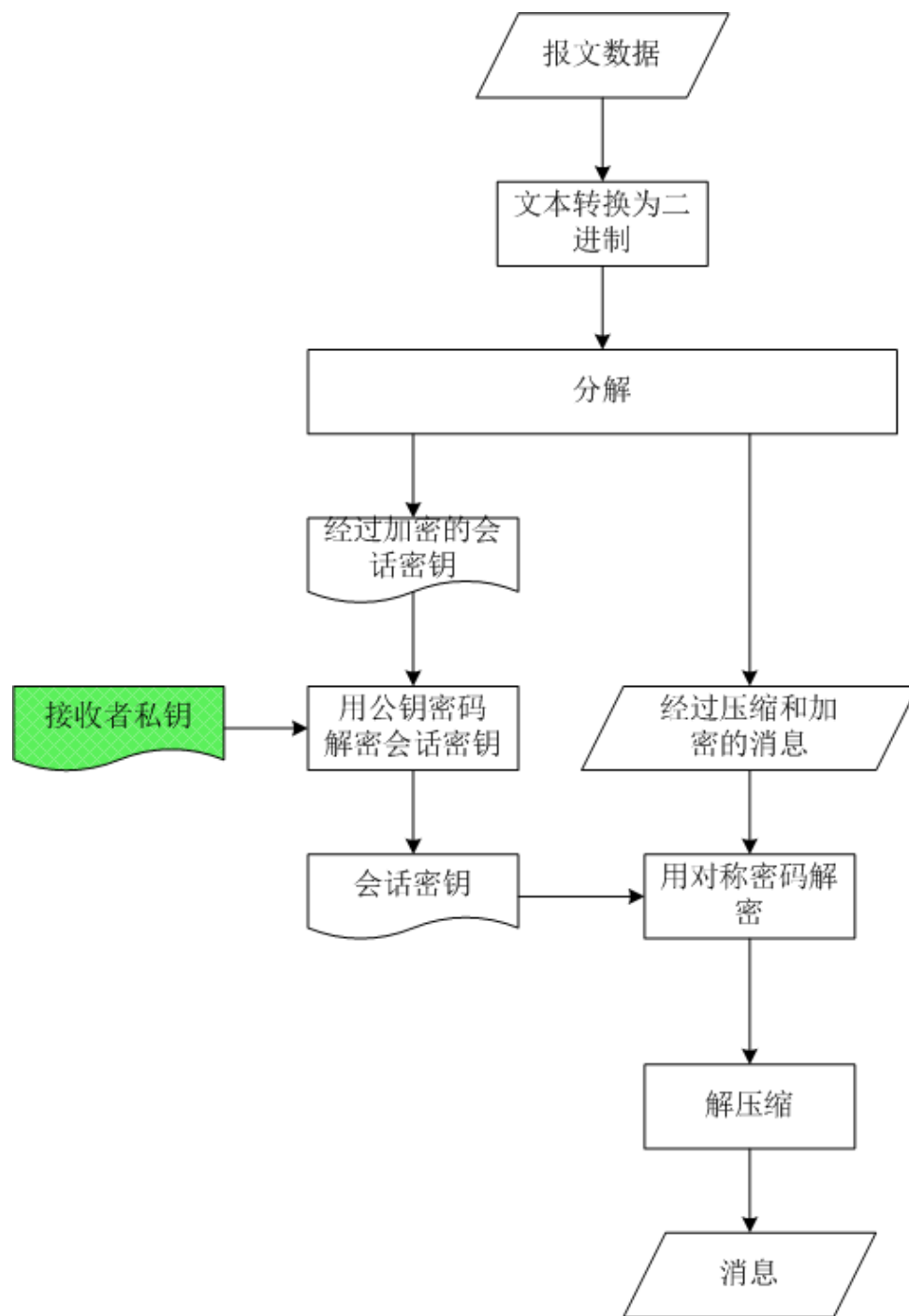
□ 生成和加密会话密钥

- ◆ (1) 用伪随机数生成器生成**会话密钥**
- ◆ (2) 采用公钥加密算法，用接收者的公钥加密会话密钥

□ 压缩和加密消息

- ◆ (3) 压缩消息
- ◆ (4) 使用对称密码对压缩后的消息进行加密，密钥为步骤 (1) 中的**会话密钥**
- ◆ (5) 将加密的会话密钥 (步骤 (2) 中) 与加密的消息 (步骤 (4) 中) 拼接起来
- ◆ (6) 将步骤 (5) 中结果转换为文本数据，得到报文数据。

PGP解密流程



PGP: 解密

□ 解密私钥

- ◆ (1) 接收者输入解密的口令
- ◆ (2) 求口令的散列值, 生成用于解密私钥的密钥
- ◆ (3) 对钥匙串中经过加密的私钥进行解密

□ 解密会话密钥

- ◆ (4) 将报文数据 (文本形式) 转换为二进制数据
- ◆ (5) 将二进制数据分解为两部分: 会话密钥的密文和消息的密文
- ◆ (6) 用步骤 (3) 中生成的接收者的私钥解密会话密钥

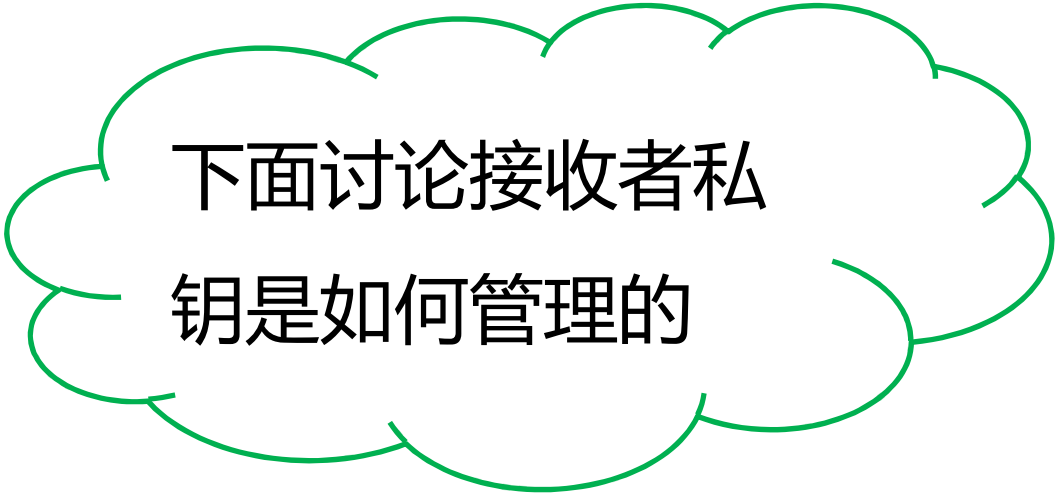
PGP: 解密

□ 解密和解压消息

- ◆ (7) 使用步骤 (6) 中生成会话密钥解密消息密文，得到压缩过的消息
- ◆ (8) 对步骤 (7) 中的输出进行解压缩
- ◆ (9) 得到原始消息

接收者私钥管理

- PGP解密中，我们用到了接收者的私钥，但是私钥是如何管理的并没有提及。

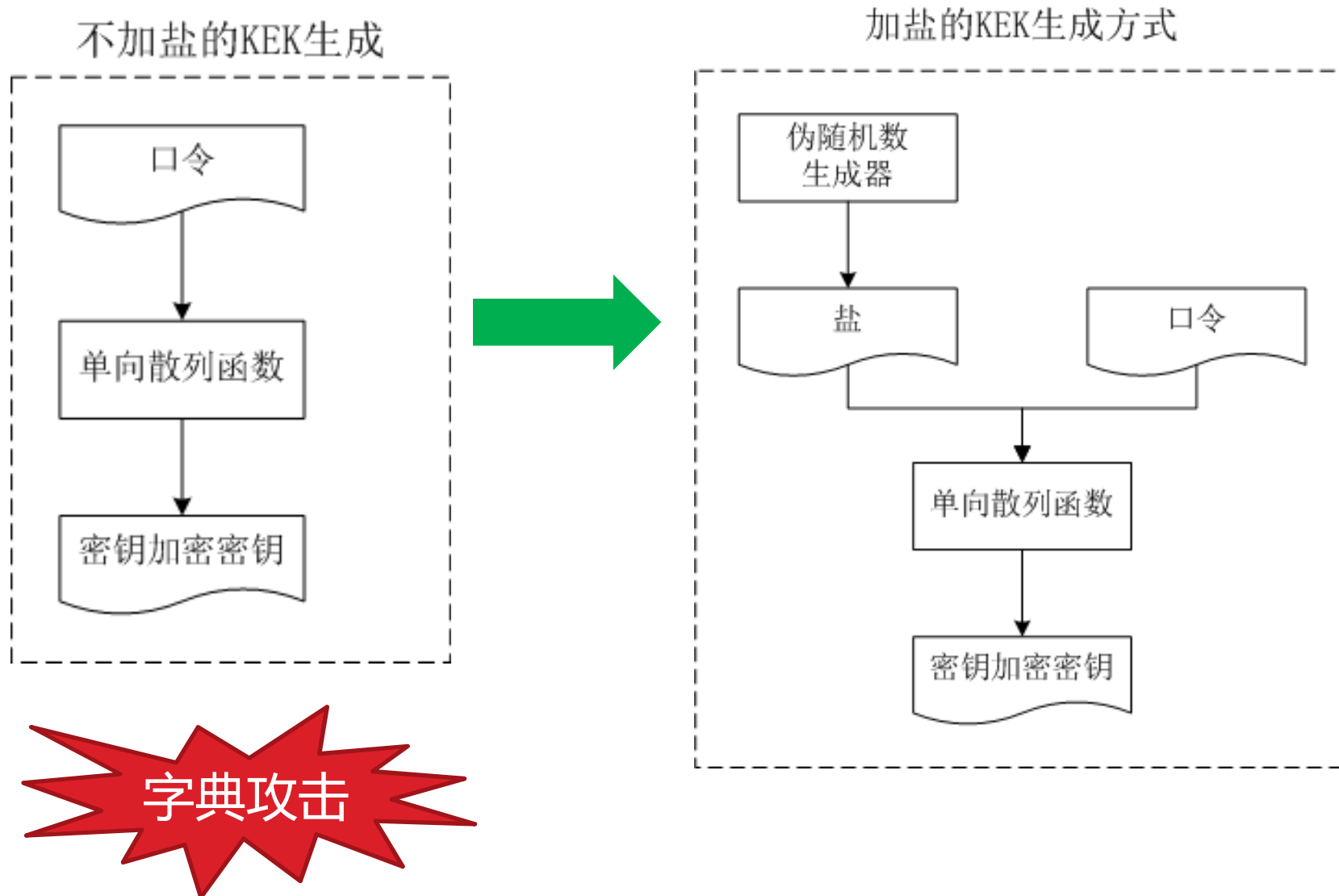


下面讨论接收者私
钥是如何管理的

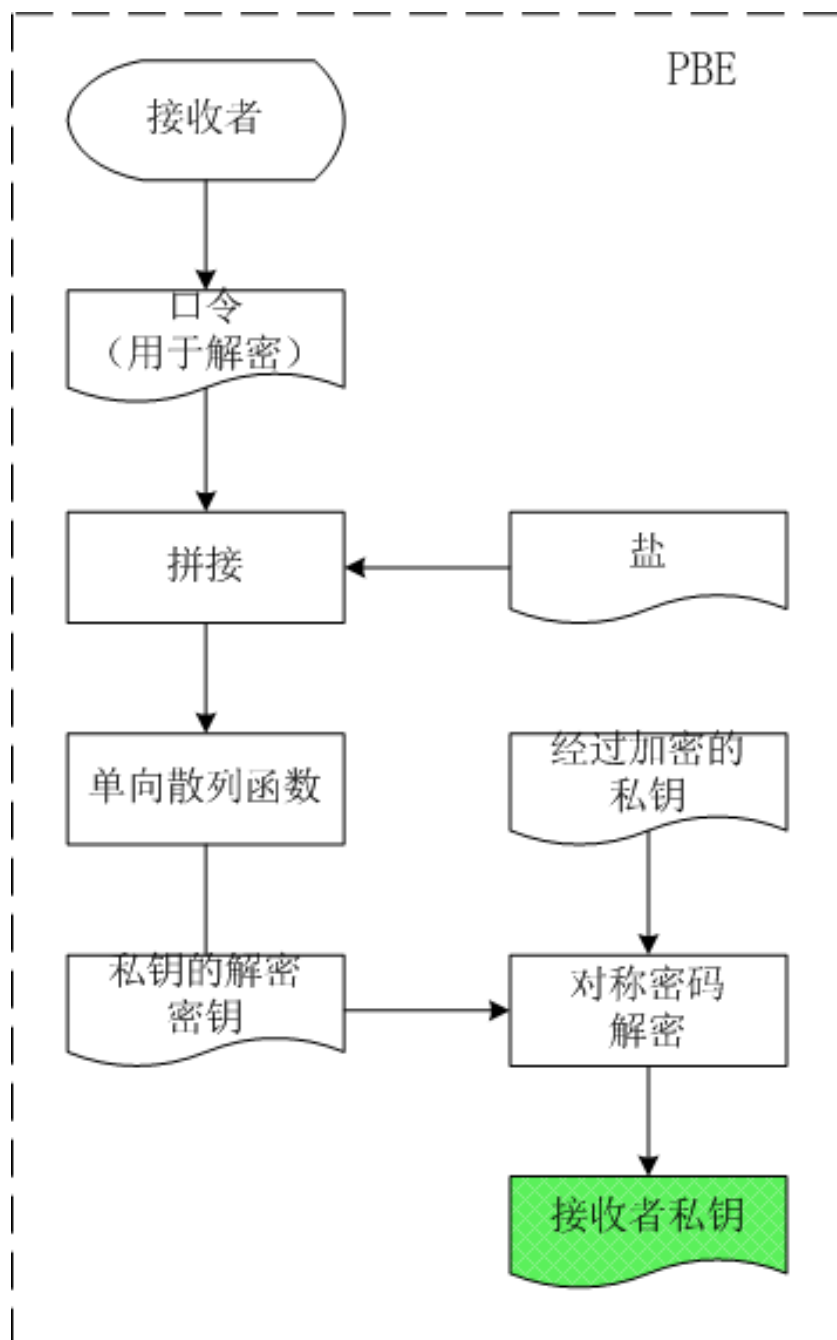
接收者私钥管理

- 私钥——记在脑袋里
 - ◆ 记不住：长串随机数
- 私钥——明文存放计算机上
 - ◆ 不安全
- 加密存放在计算机上
 - ◆ 合理：加密私钥的密钥怎么管理？
- PBE to rescue

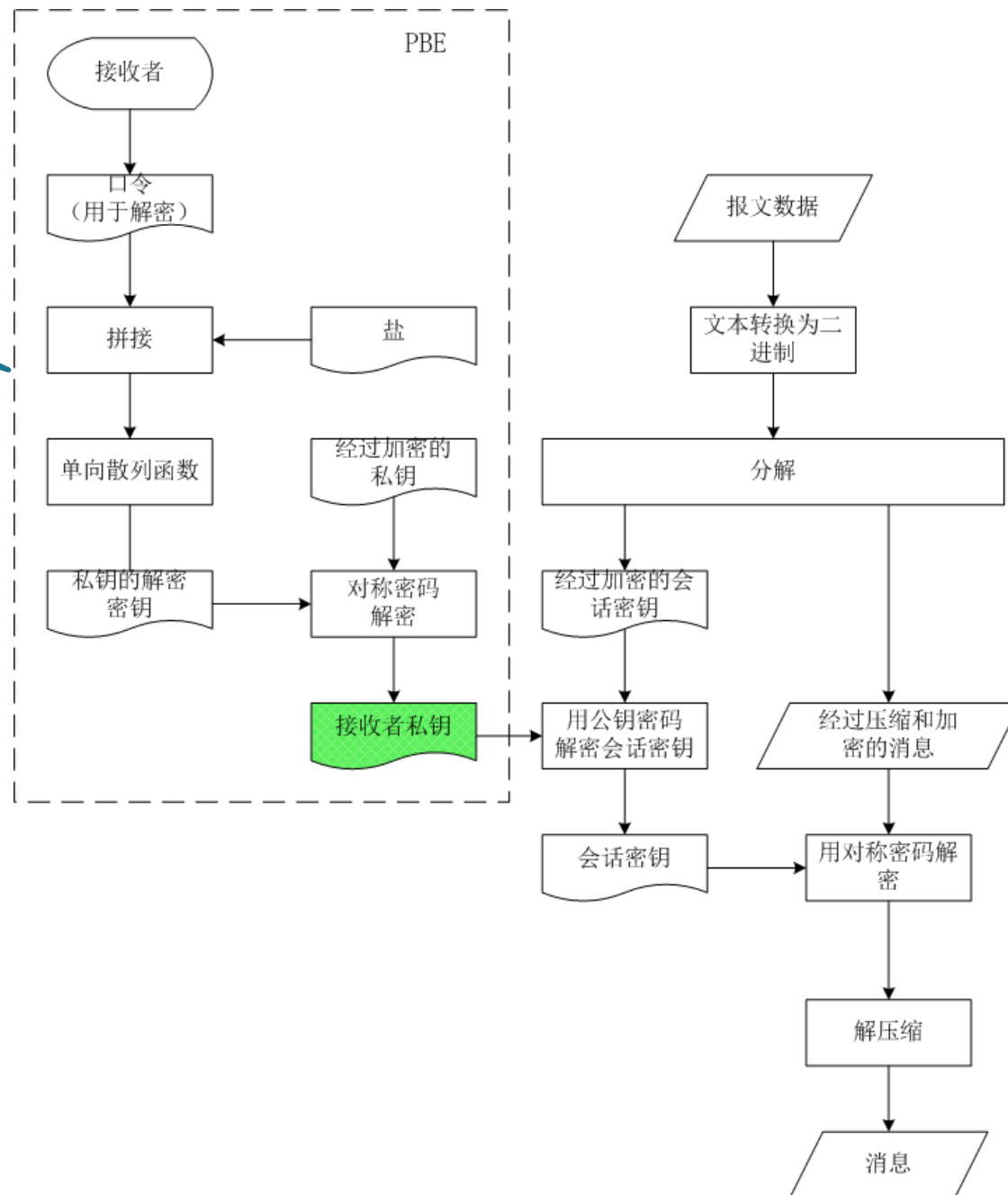
Password Based Encryption



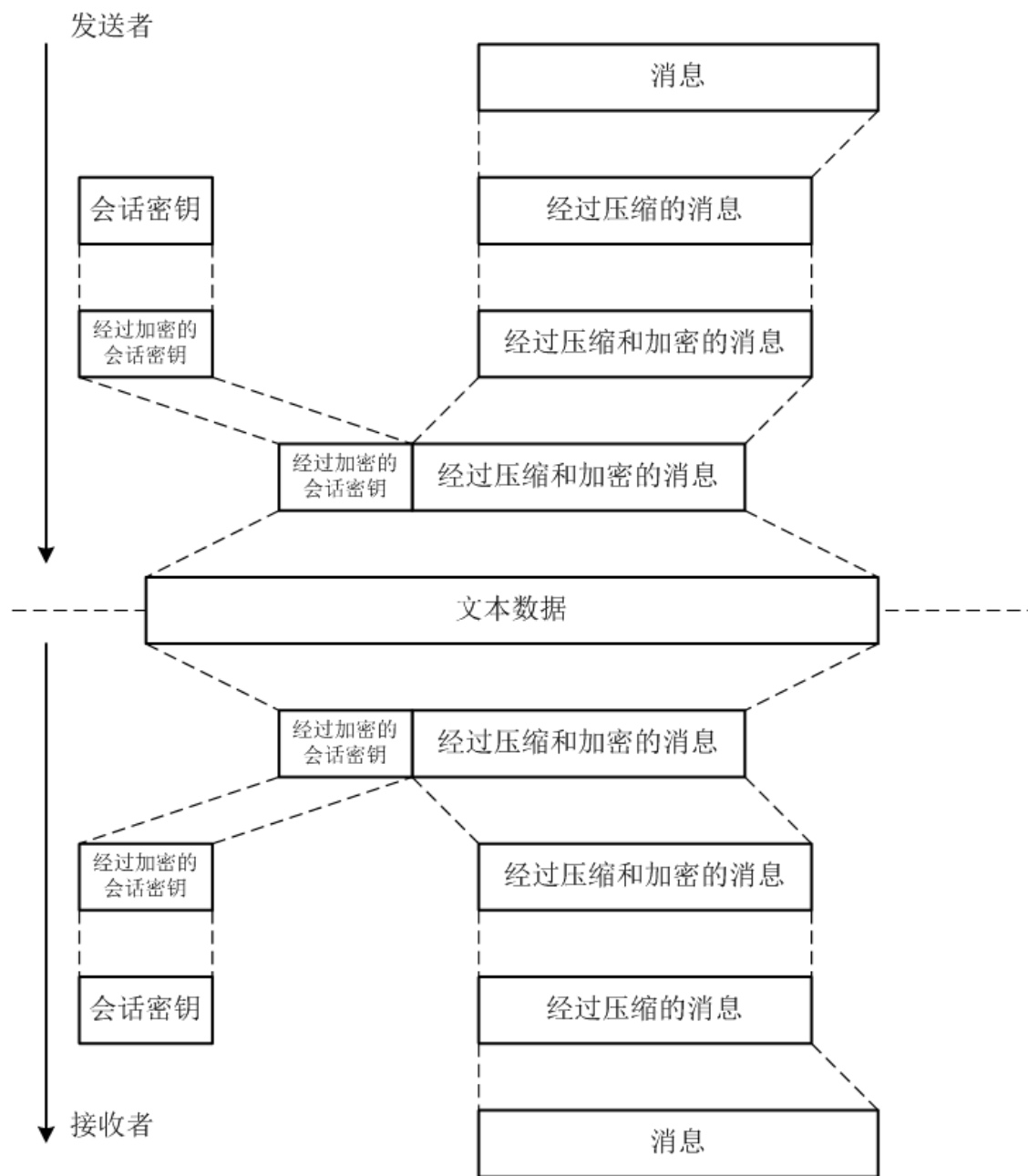
接收者私钥管理: PBE



合并PBE和 解密部分



完整的加解密过程



只采用加解密功能，具有消息的保密性，也确保了接收者是正确的，但是无法保证发送者的身份。因此，在只需要消息的保密性的前提下，只用PGP的加解密功能是可以的。

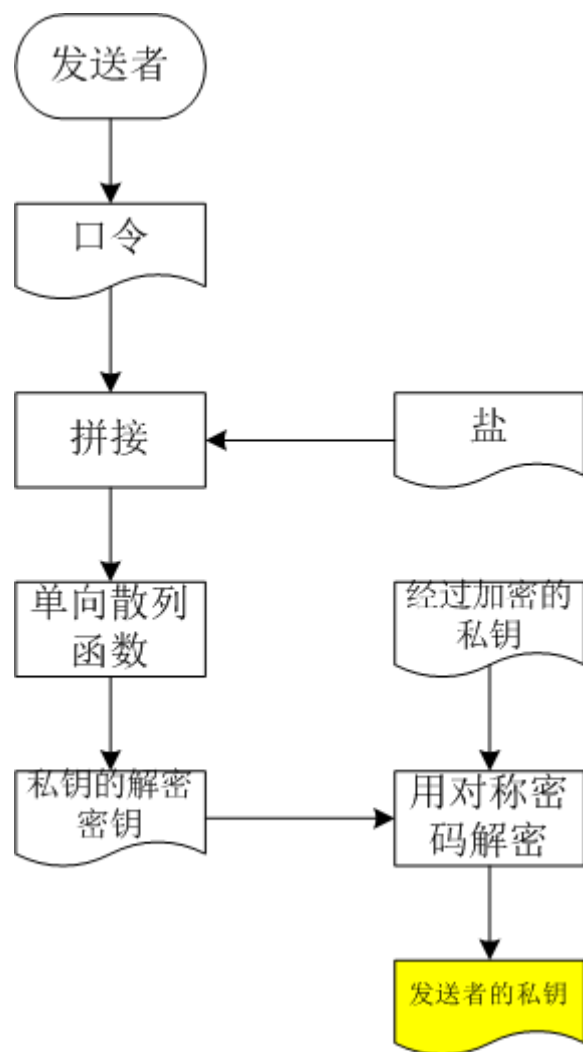
生成和验证数字签名

生成数字签名

□ 解密私钥

- ◆ (1) 发送者输入用于解密私钥的口令
- ◆ (2) $\text{hash}(\text{口令} || \text{盐}) \rightarrow \text{解密私钥的密钥(KEK)}$
- ◆ (3) 解密密匙串中经过加密的私钥

生成数字签名：解密私钥

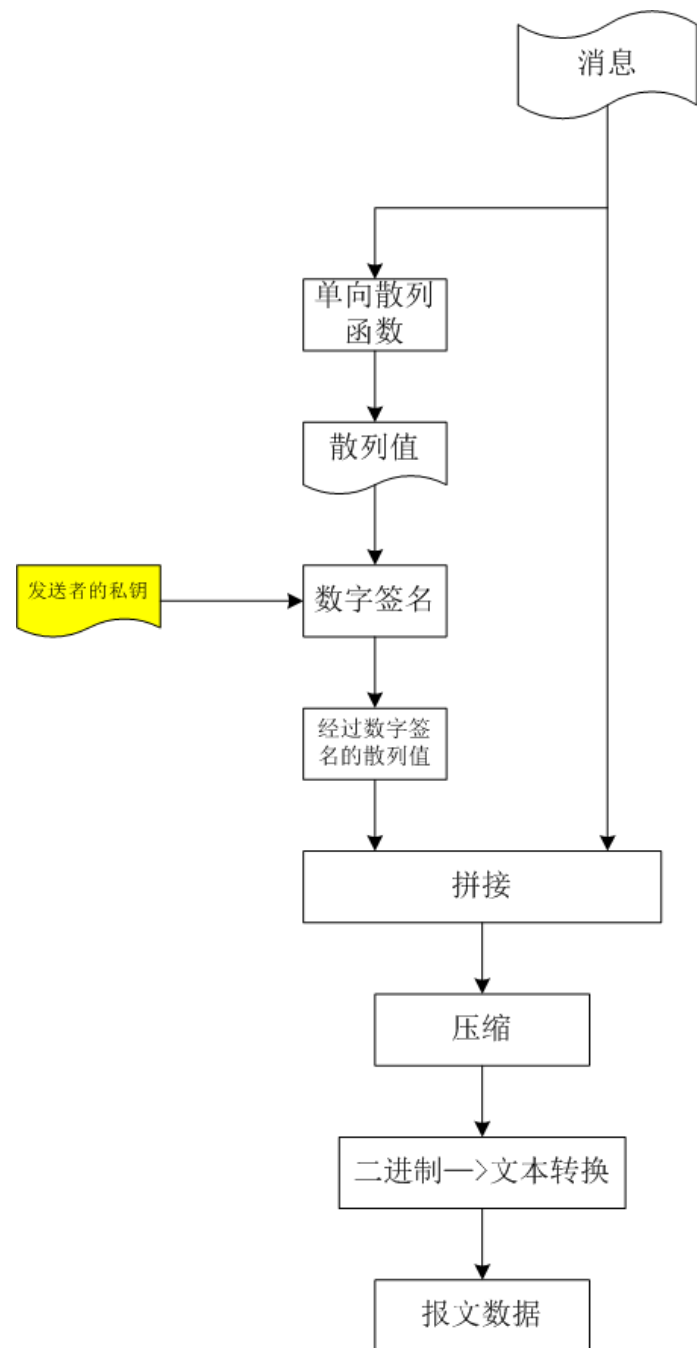


生成数字签名

□ 生成数字签名

- ◆ (4) 用单向散列函数计算消息的散列值
- ◆ (5) 用步骤 (3) 中私钥对上述散列值进行签名
- ◆ (6) 将步骤 (5) 中生成的数字签名与消息进行拼接
- ◆ (7) 将步骤 (6) 中的结果进行压缩
- ◆ (8) 将步骤 (7) 中的结果转换为文本数据, 即: 报文数据

生成数字签名



验证数字签名

□ 提取发送者发送的散列值

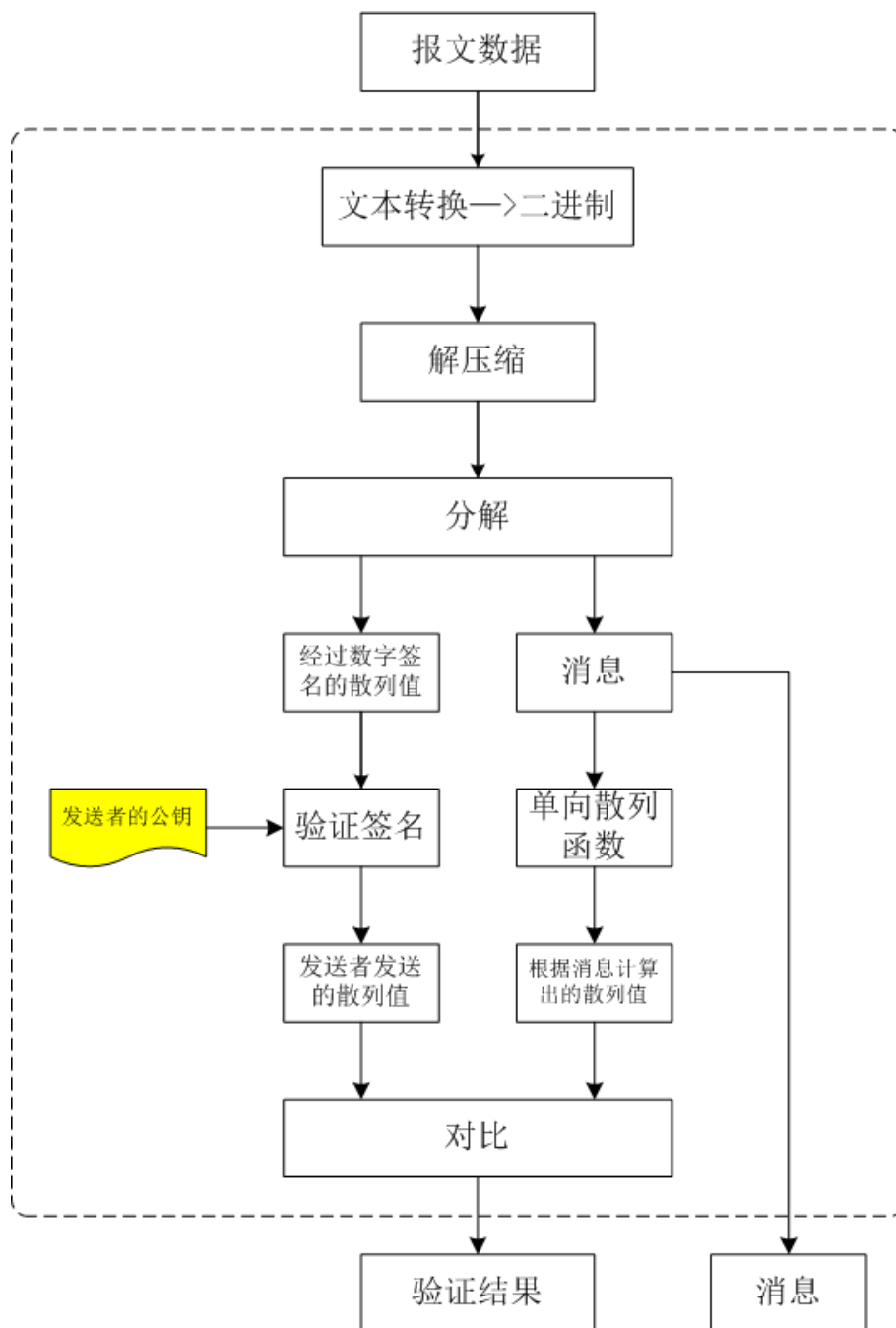
- ◆ (1) 将报文数据（文本形式）转换为二进制数据
- ◆ (2) 解压缩二进制数据
- ◆ (3) 将解压后的数据分离为经过签名的散列值和消息两部分
- ◆ (4) 将经过签名的散列值用发送者的公钥进行解密，提取出发送者对原始消息生成的散列值

验证数字签名（续）

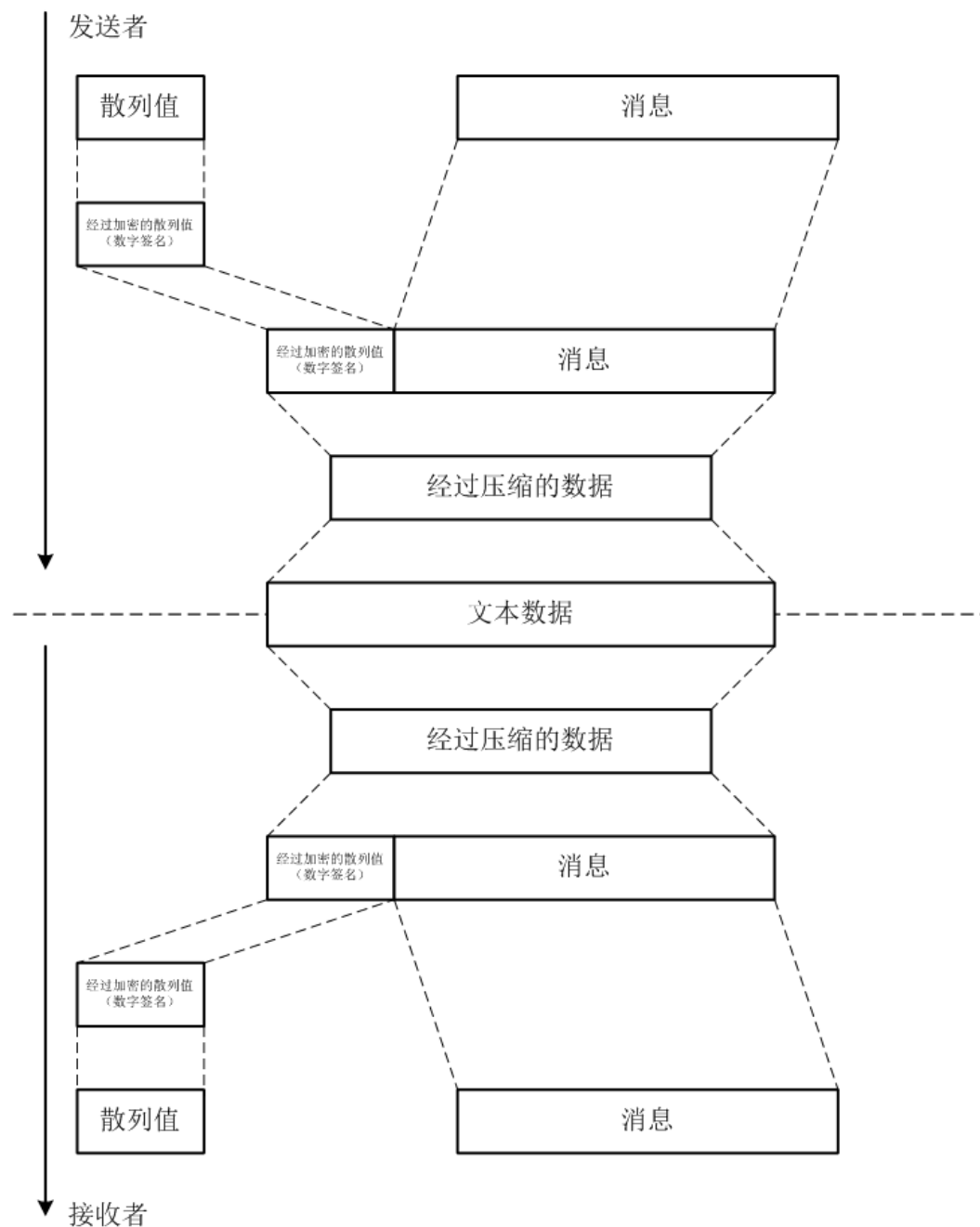
□ 对比散列值

- ◆（5） 将步骤（3）中的消息单向散列函数计算散列值
- ◆（6） 对比步骤（4）的散列值和步骤（5）的散列值，如果一致，则数字签名验证成功，步骤（3）发送的消息就是发送者发送的消息。

验证数字签名

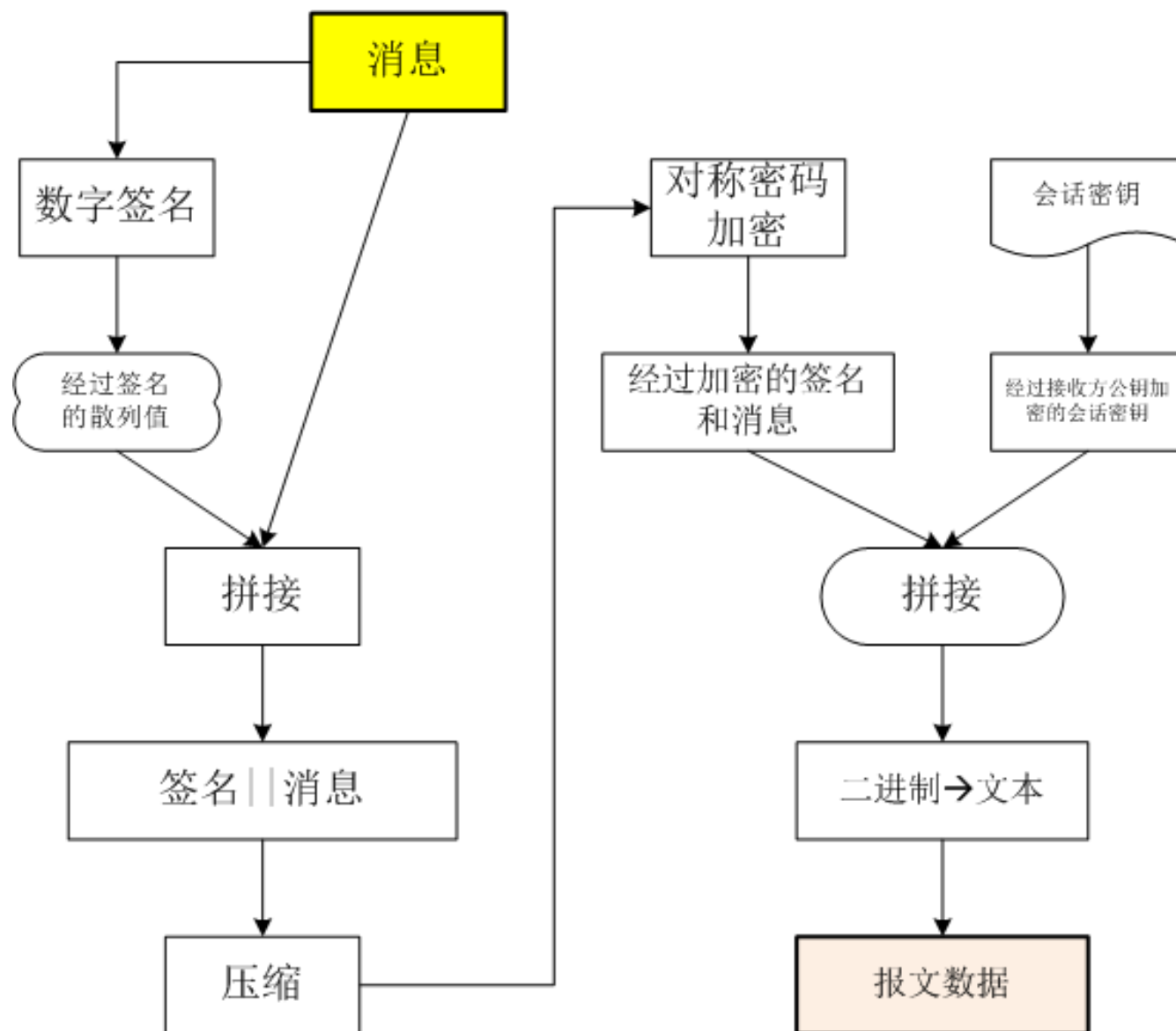


生成和验证数字签名

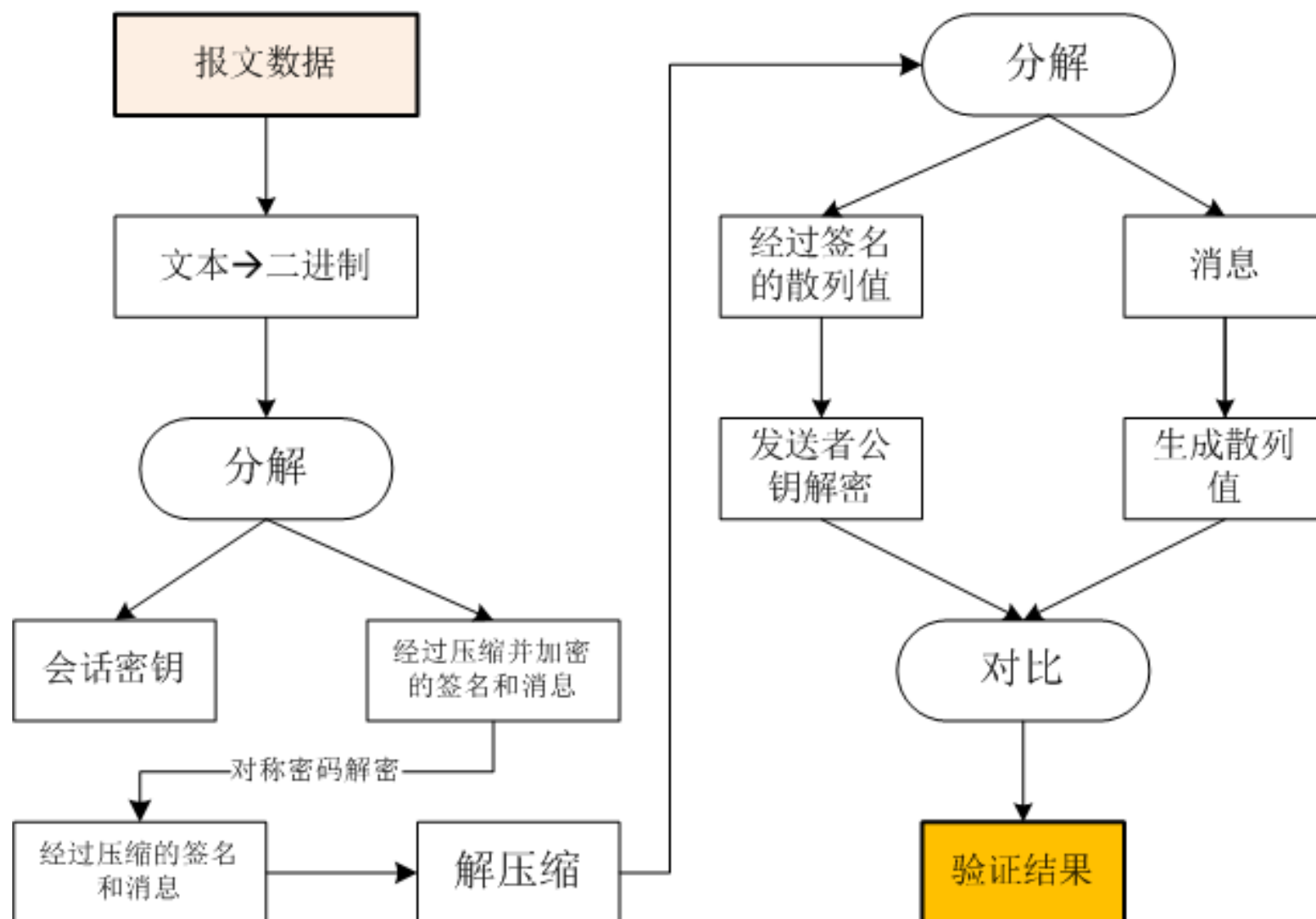


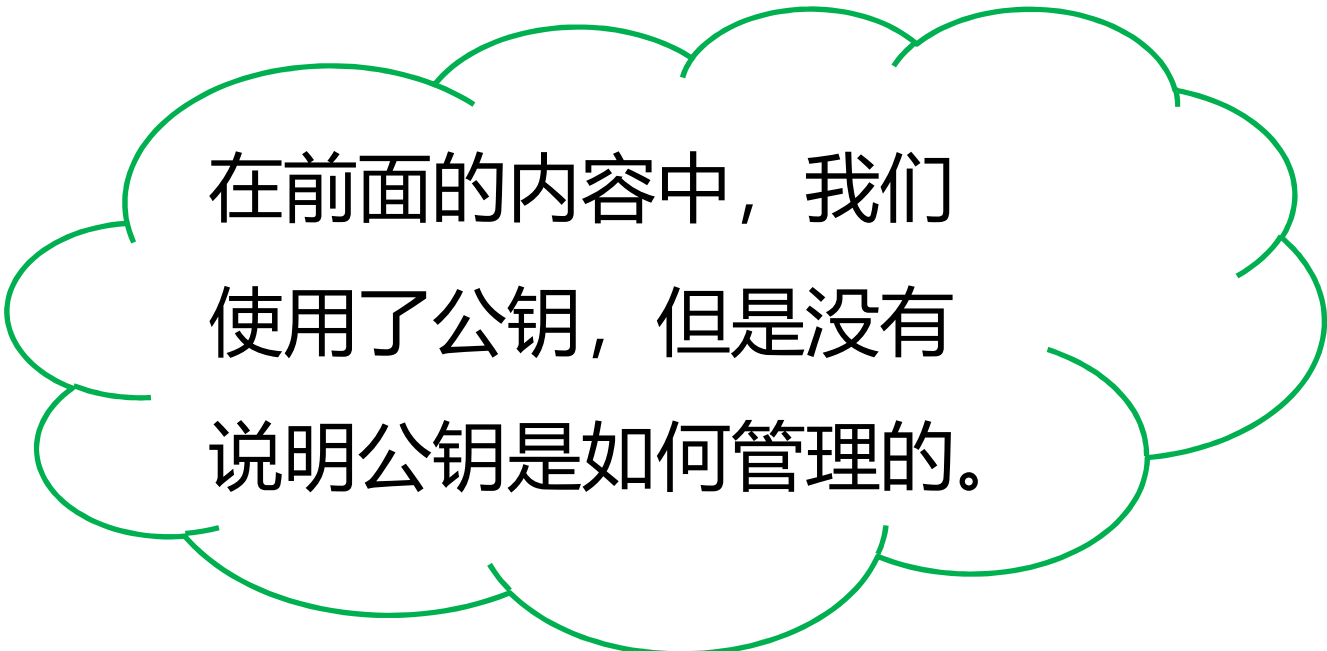
PGP生成数字签名并加密以及解密并
验证数字签名

生成数字签名并加密



解密并验证数字签名





在前面的内容中，我们
使用了公钥，但是没有
说明公钥是如何管理的。

PGP的公钥管理

PGP的公钥管理

- PGP的正确运行依赖于公钥的合法性
- PGP没有假定信任任何机构，即使是国家
 - ◆ 不依赖于PKI
- PGP采用信任网来确保公钥的合法性

信任网：基本原则

- 通过自己的数字签名进行确认
- 通过自己完全信任的人的数字签名进行确认
- 通过自己有限信任的多个人的数字签名进行确认

通过自己的数字签名进行确认

- Alice和Bob在一次meeting上认识，分别时，Bob交给Alice一个U盘，并说“以后email联系，这是我的公钥”。
- Alice回家后，将Bob的公钥从U盘中导入到自己的PGP公钥串中。Alice确信U盘中的公钥就是Bob本人的，因此，Alice对这个公钥加上自己的数字签名，以证明这个公钥是合法的。

通过自己的数字签名进行确认

- 随后，Alice收到来自Bob的email，这封邮件带有Bob的数字签名，为了验证Bob的数字签名，PGP需要执行以下步骤。
- (1) PGP从Alice的公钥串中寻找Bob的公钥。
(根据数字签名的原理，验证某个数字签名，需要对应的公钥)
- (2) Alice的公钥串中有Bob的公钥。（因为，Alice在之前导入过Bob的公钥）

通过自己的数字签名进行确认

- (3) (但是, 这个公钥合法吗?) PGP发现公钥串中Bob的公钥带有Alice的数字签名。
(Alice在导入Bob的公钥的时候, 对它做了数字签名)
- (4) 为了验证Alice的数字签名, PGP需要从Alice的公钥串中寻找Alice自己的公钥。
(Alice的公钥串中也包含了Alice自己的公钥)

通过自己的数字签名进行确认

- (5) PGP使用Alice自己的公钥对Bob的公钥上的数字签名进行验证。如果成功，则表明这的确是Bob的公钥。
- (6) PGP使用合法的Bob的公钥对邮件上附带的Bob的数字签名进行验证。如果验证通过，则表明该email确实来自于前段时间开会时遇到的Bob所发。

通过自己**完全信任**的人的数字签名进行确认

- Alice有个男朋友叫Trent。在Alice的公钥串中，包含带有Alice签名的Trent的公钥。Alice完全信任Trent，可以认为“Trent介绍的人也是可信的”。因此，Alice设置Trent的公钥为“我完全信任Trent的数字签名”这一状态，并加上自己的数字签名。也就是一旦验证是Trent签名的公钥，就是合法的公钥。

通过自己**完全信任**的人的数字签名进行确认

- 在PGP中，用户可以设置对每个公钥所有者的“所有者信任级别”（owner trust），因为Alice完全信任Trent，因此把Trent的“所有者信任级别”设置为“完全信任（Fully Trusted）”。

公钥所有者的信任级别

Ultimately trusted	绝对信任（是持有私钥的本人）
Fully trusted	完全信任
Marginally trusted	有限信任
Never trust this key	不信任
Not enough information	未知密钥
No owner trust assigned	未设置

通过自己完全信任的人的数字签名进行确认

- ▣ 假设某一天，Alice收到了自称来自Carrol的邮件，邮件中附带有Carrol的公钥，而且这个公钥带有Trent的数字签名。Alice把Carrol的公钥导入自己的公钥串时，Alice的PGP通过Trent的公钥验证了Carrol的公钥的合法性。因为Alice完全信任Trent，因此Alice的PGP会认为Carrol的公钥是合法的。在这一场景中，Alice信任Trent，Trent通过给Carrol的公钥数字签名而起到了“介绍人”的角色，从而Alice可以认为Carrol的公钥是合法的。
- ▣ 注意：因为Alice的公钥串中没有Carrol的公钥，因此无法通过Alice的公钥串来直接确认Carrol的邮件。

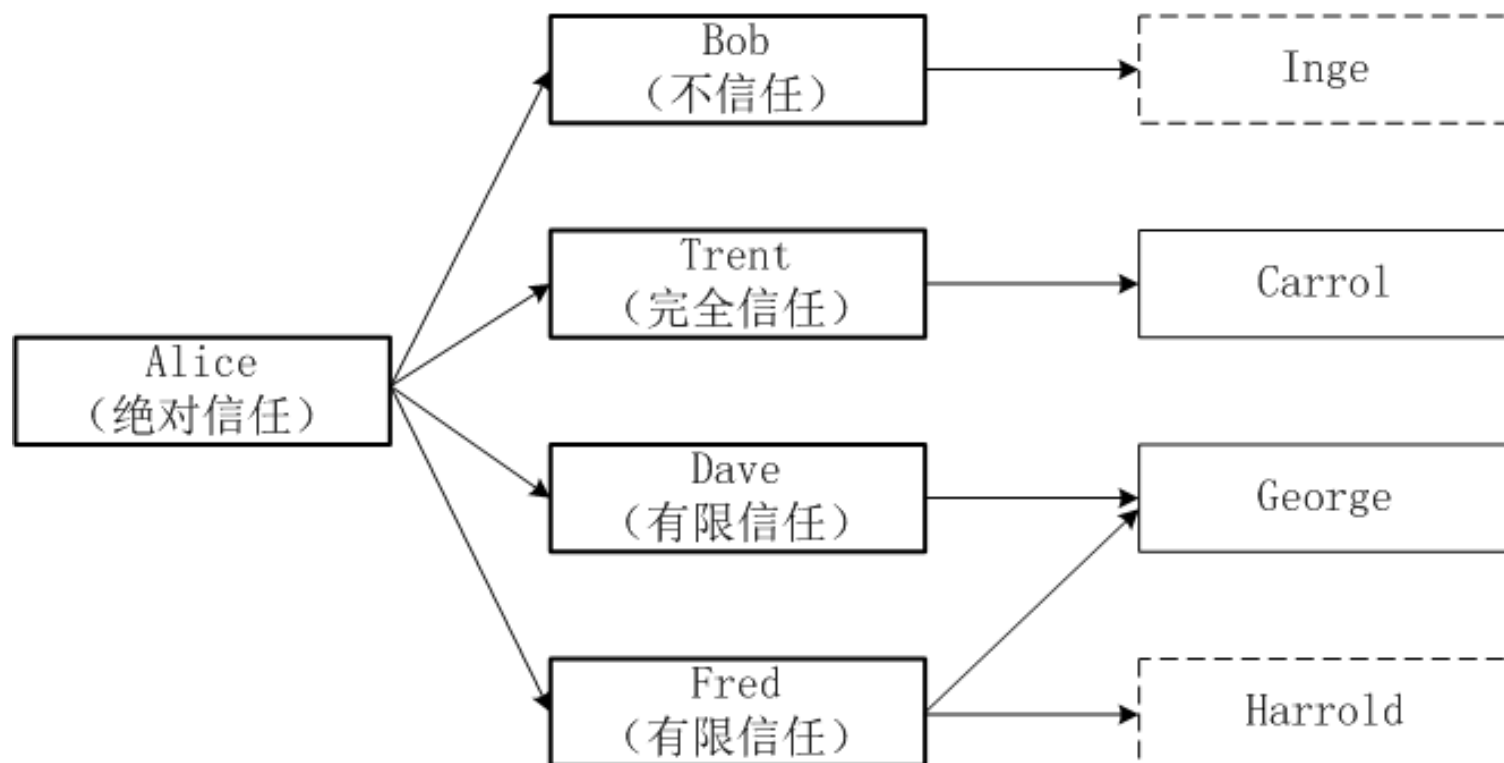
通过自己有限信任的多个人的数字签名进行确认

- 假设Alice有两个男朋友，分别叫Dave和Fred。
在Alice的公钥串中，包含有带Alice签名的上述两人的公钥，而且Alice把Dave和Fred的所有者信任级别都设置为“有限信任”。

通过自己有限信任的多个人的数字签名进行确认

- 某一天，Alice收到了来自George的公钥，该公钥带有Dave和Fred的数字签名，Alice的PGP会确认该公钥是合法，确实属于George。如果仅有Dave或者Fred之一对该公钥进行数字签名，因为Alice对Dave和Fred只是有限信任，因此不能确认该公钥是合法的。只有当2个或以上的有限信任的人对某个公钥签名时，才能确认该公钥是合法的。

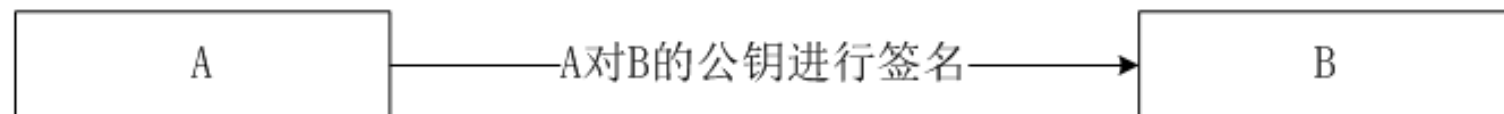
Alice的信任网



Alice直接签名的
合法密钥

可以判断为合法
的密钥

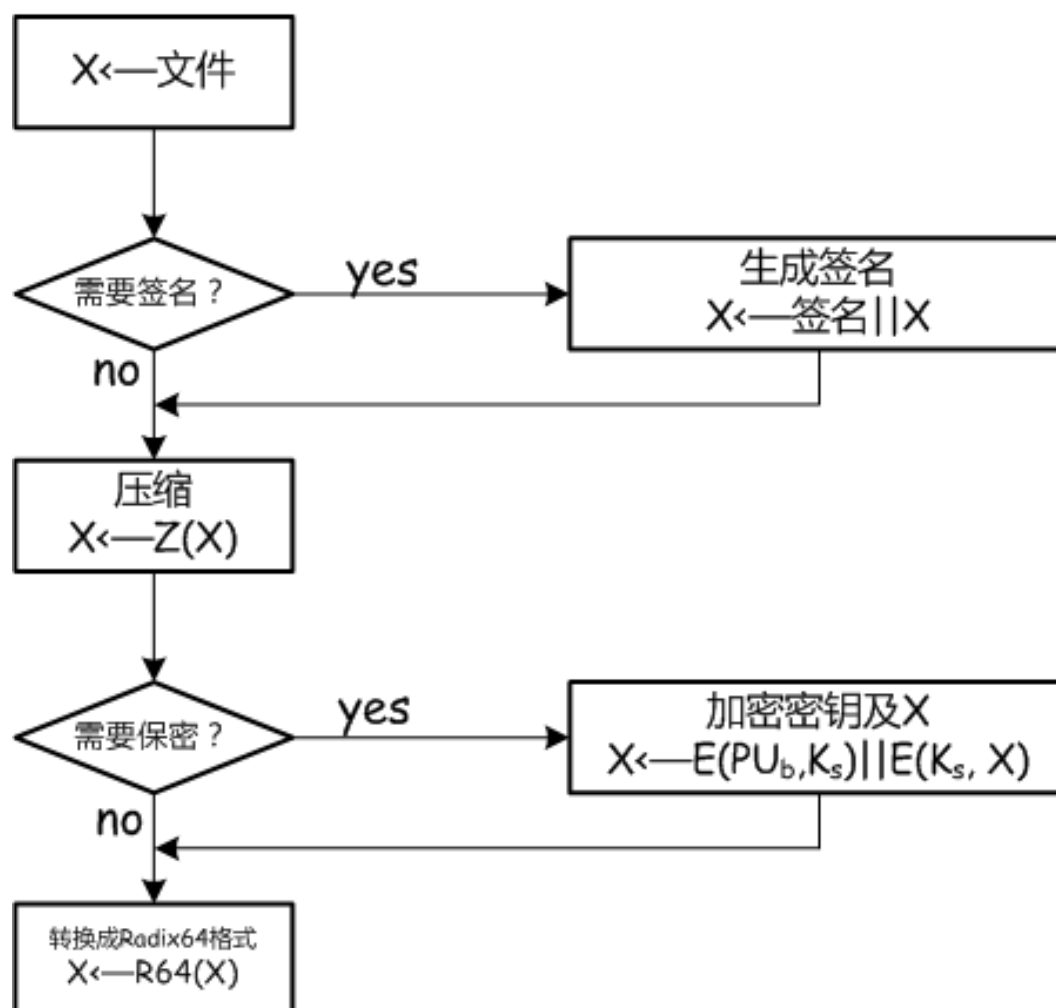
不能判断为合法
的密钥



PGP总结

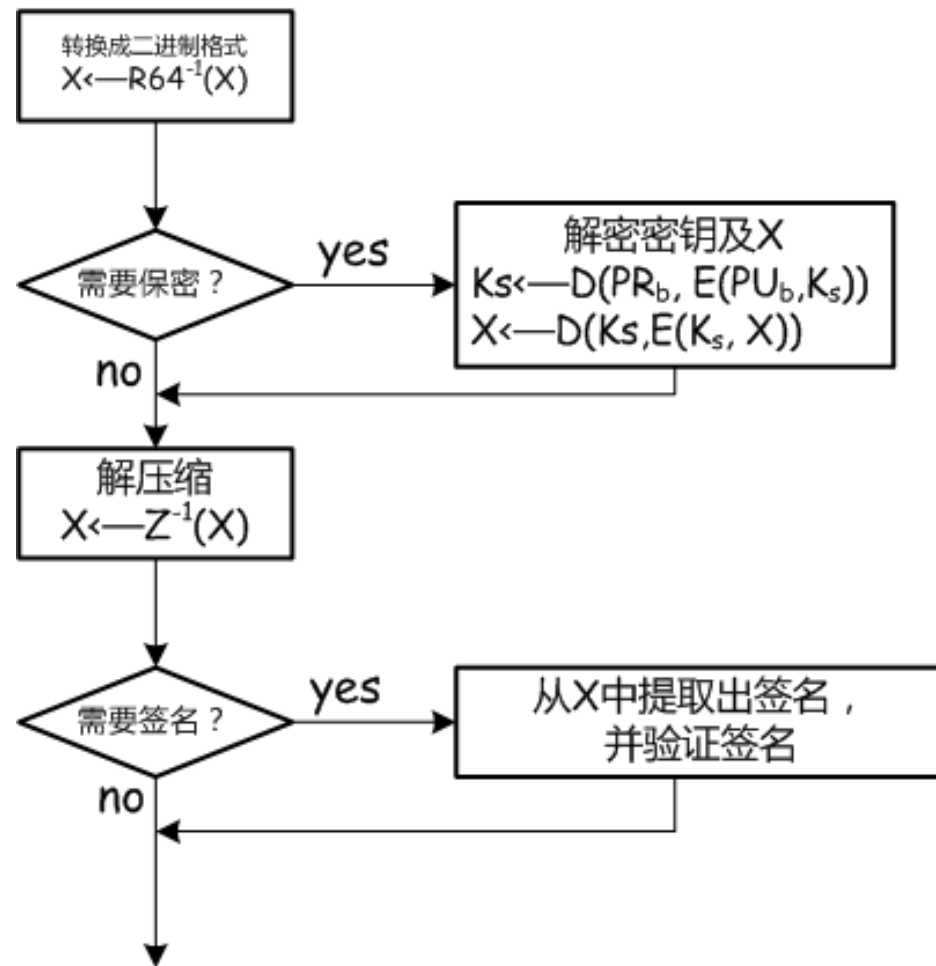
PGP流程总结：发送方

发送方：Alice



PGP流程总结：接收方

接收方：Bob



PGP操作算法总结

操作	使用算法	描述
数字签名	DSS/SHA 或 RSA/SHA	使用SHA-1生成消息的散列值，采用DSS或者RSA，用发送者的私钥加密该散列值并与原消息拼接。
消息加密	CAST 或 IDEA 或RSA	采用CAST-128或IDEA或3DES，使用由发送者生成的一次性会话密钥加密消息。使用RSA算法中接收者的公钥加密该会话密钥并与消息拼接。
压缩	ZIP	采用ZIP算法来实现消息压缩以便存储和传输
电子邮件兼容性	Radix-64转换	与电子邮件系统兼容，在二进制和Radix-64编码之间互相转换。

PGP实例操作

PGP软件: GPG4Win

- GPG (Gnu Private Guard, GnuPG)
 - ◆一款基于OpenPGP标准开发的密码学软件
- GPG4Win组件
 - ◆GnuPG 本身
 - ◆GPA 一个证书管理器
 - ◆GpgOL Outlook 加密扩展
 - ◆GpgEX Explorer 资源管理器加密扩展
 - ◆Claws Mail 一个邮件客户端
 - ◆Kleopatra 证书管理器
 - ◆Compendium 帮助文档集, 仅英语和德语

GPG4Win

- 创建密钥对
- 导入/导出密钥
- 加密文本
- 解密文本

PGP4Win: 创建密钥对

- 菜单操作File→New Certificate
 - ◆ Create a personal OpenPGP key pair
 - ◆ Enter details
 - Name
 - Email address
 - ◆ Create Key
 - Enter passphrase



PGP4Win: 导出公钥

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
```

```
Version: GnuPG v2
```

```
mQENBFmiImEBCACywmP8ehLhPDFyRdfc/Cad3W0tImIxRPKW0hLUPqEZXjavyP5L
gHO/pw+xOk0aqcnTeS8ANN5IcGn66rKDloY4OZSc1X4pUUF/Ej11jkgKcuHf044p
noLhAumconxMs8rYs34ErMDupiPWO+Dyu2T/QZvq/HLVYnzJfNi+epNtZcfVPxyx
IUTfDv3gtQDJiB6GZUqmgSrtYvOtEwWtW3p0GkG8bEhRePVrr2D5SnEOQq8hJWka
lVakoPMkT018W8T9EBfsw+A4YQ5SfSCaMkYcbhJ2VsYXf60OhWmdNw5Ow6E7NvjP
jftUnz5esu4XVhfzTOHnexoW4I8vY/tCz+jhABEBAAG0HXRvbSBjcnVpemUgPHRv
bUB1ZXN0Yy5lZHUuY24+iQE5BBMBCAAjBQJZoiJhAhsDBwsJCAcDAgEGFQgCCQoL
BBYCAwECHgECF4AACgkQ5ealmxV138D8swf+PFekqLLXOgz90N7wgobZzBNkyRGC
6vPs4RLJKWTCJ4ImHZSjghqEZYqpQ+YlAqCpKd1DYXUVUtQESgpo311/cTkfYS3T
NL4plW9C+ITkWQImPbEBbe/8acenZyYBm4I0LDfXPDxUgReQhaB7bigLfjtVJIdI
GHCi7HgDP6AyDiZU/PSwPUUZu74faPa4AEaL2oxFX65oj2K0sKT/Vbljm8H+4CIA
hETc6pyoxpGm9RDlkVHTd2/uCYD18tNzf4le+wrT3kVwaKVc4VpJvcZF1zTm5tvq
Z6LzBN7ov0I5tfr1Jw3XiZYbAGWpVabuXZF1x2kqTsrX9Fu0hxt6uUEj8bkBDQRZ
oiJhAQgAo9CdCb6/Q+7pExrICGuhy8mKf8PACP6kL+BTt/92BUIvVp/3lyb5d1Pr
La9H4qVSmz06LB94G0txqKxUNs7IZnMyI3bfG8eGj4rQcd5e5Zd2mT061Ep15a4B
7OdiG5zA2hioGMBuHX39dreGbk3G/6vujt5cwv5cTcq1ZTJ1NDu88EDGwnHUxsiZ
ORt5CW5NpoNPWtT0H75hYYmq1o1L5CX/ne9EBCp8jJP9Ixuf//iKWxm7vNVd7bMU
```

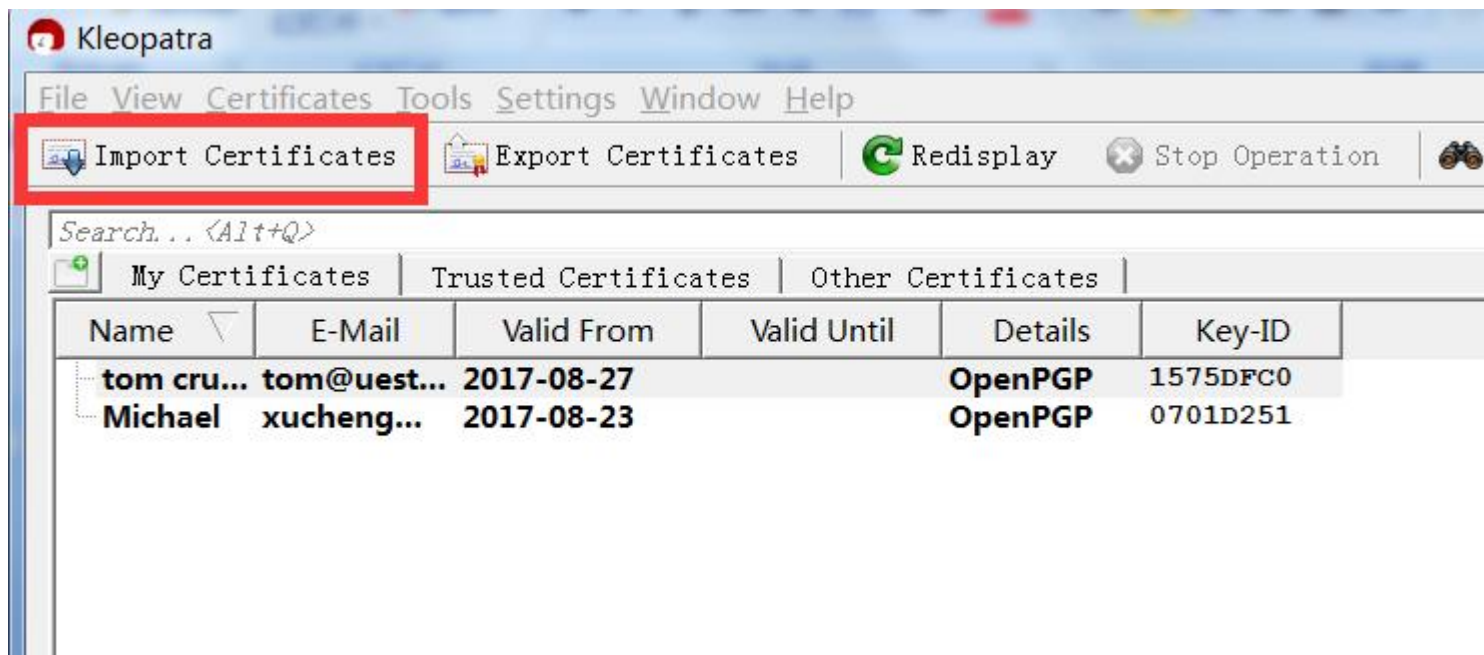

GPG4Win: 导出私钥

```
-----BEGIN PGP PRIVATE KEY BLOCK-----
```

```
Version: GnuPG v2
```

```
lQO9BFmiImEBCACywmP8ehLhPDFyRdfc/Cad3W0tImlxRPKW0hLUPqEZxjavyP5L
gHO/pw+xOk0aqcnTeS8ANN5IcGn66rKDloY40ZSclX4pUUF/EjlljkgKcuHfO44p
noLhAumconxMs8rYs34ErMDupiPWO+Dyu2T/QZvq/HLVYnzJfNi+epNtZcfVPyxy
IUTfDv3gtQDJiB6GZUqmgSrtYvOtEwWtW3p0GkG8bEhRePVrr2D5SnEOQq8hJWka
lVakoPMkTO18W8T9EBfsw+A4YQ5SfSCaMkYcbhJ2VsYXf600hWmdNw5Ow6E7NvjP
jftUnz5esu4XVhfzTOHnexoW4I8vY/tCz+jhABEBAAH+AwMCEnT6EwvWzYTD9Bvk
M2qtcPqosJwl4igPVM3xMxI+7lf+wylIF/Cfx3G0e+uJqIXB5mqp0nv15e7ur6uj
S4GzMXDYFRlrgkvPvbe3qTKfT0L9Z+jLecb9x1qT08GXkHHgScpTC685j2BX93fv
wmkApI8RzAr2697TLgWl8Ixx5KH5pBslB4ypQpJrhfXXAslCoG0R4ZtgirzDVlZO
F/4e+VYbIyLihtnUHqhkmCllrPyoY0mq7v1L59hLPcpOqYVcsIveU5Tr5e/v4M3
mKLQ00ECs28sNPwHTK8K/LVPRjvjhIm8ow7Rj0dsYg6+xeMnudjUUwqKG45SVZJU
iPBTg4TXuugafYVt51vVUo+zFJtDAIeXh7xJGPYUcc9GqRkp3XsFQdjMQjUkYbxg
wCXW3kYMkgd8bXBX+K+UTEjbj0rLp9E2jtwENMpeLIk88pC5S4BoCnvSqmQ7sxiz
xpftouI9LcuMvpHnVFt3KDs2Nv5XZCOGem7QM9NOIUBKWM/Tp9WNjlTzgDQCnubf
u9BcINQWrcu5EB1ATloaNjRcriGpNbsYdmthu4CUz6eivVVq+KYwqlsOTlyFn6Ek
H2ZrQ5FZj6W/Uubs+3KVYM/sShllxQouTnrX7RKwNrJ6kZvpvYUvsJFKmG9Piv4q
n2NzkeF8S9d9Q2Ak+vmkOeiOdHmPkU/ag5vdcOw1.TFTvCq+CAMEAsqGRGFPhv7fTn
```

GPG4Win: 导入证书



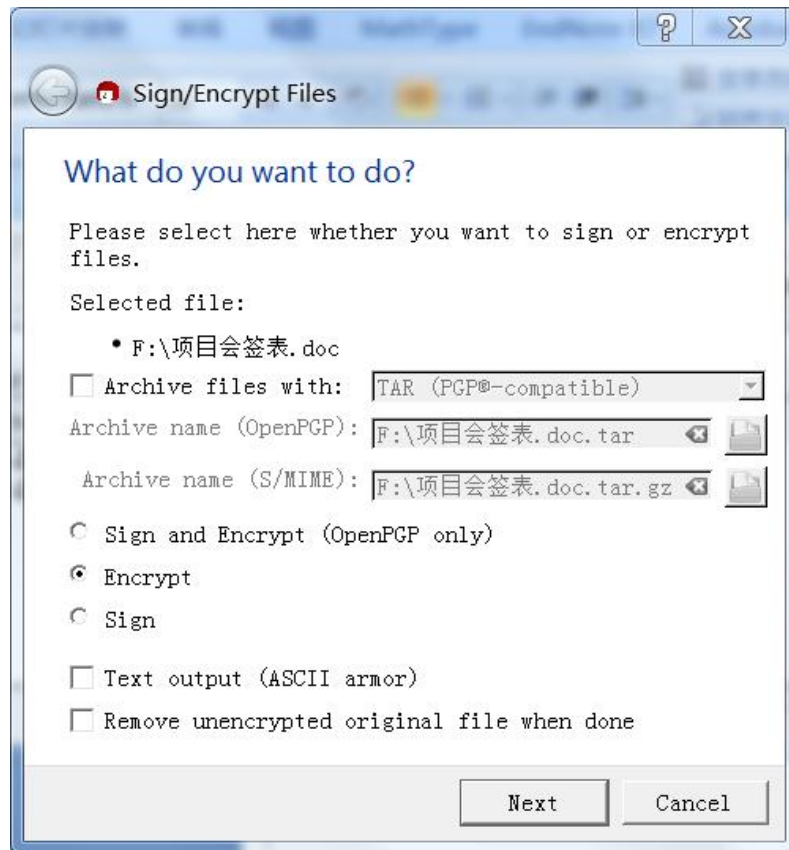
获取公钥证书:

面对面提交

从目录服务获取, 如: <http://pgp.mit.edu>

GPG4Win: 加密文件

□ 菜单操作: File→Sign/Encrypt Files

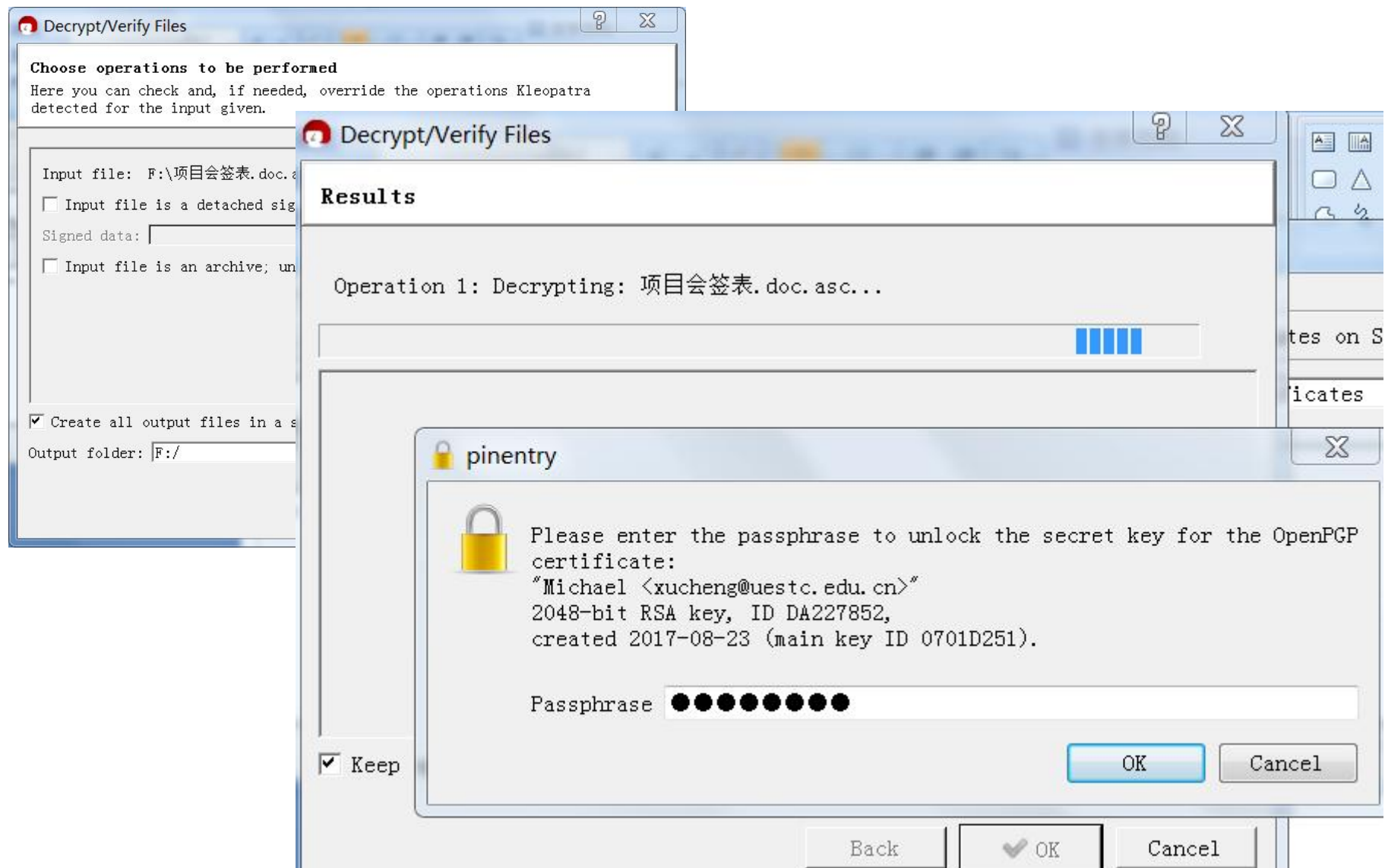


-----BEGIN PGP MESSAGE-----

Version: GnuPG v2

hQEMA7zyLoLaInhSAQf9HxQDFuUMUqeVG95l9p8C/1d2lkTl
El8ESfdZ8xAvy+QvSVnczZTQvuQW69MbmVJ0g3S3/99rORU-
t3lSIz7rFi0gFyn09Q/beloliP8nmRXogn4YwZX0lcXsYVn:
urfVYDGuEjVJjLESHBXgoX6jmgPNUZq3r+Zey/z/SZ0pM2Wl
EinOg5M7dlPzBv9NQbKD9jUKUSNX4/VgKcustU5q5hoL+gy(
PWkTVig3vi008jlXNKc0SXprZ50rijPTITJdfwsc29LsAY1l
3/Y6Ytq9600Dw9lwoGsG6+cK1BCFkHmBffSVxhiakqtW/Ci
2Gl8HACNtDcUvkkwIHJPkj5zSUmuq0587xZBE7r1CO8DohEl
FrccarQbqeqj5wPqNWcPF08NUms6uefqewwmRkX9Ln6liK3:
1QZ/47W/1k08cBfY5R2iKvhukm40/G73/FKU1VuoTrX0b4+l
AbPcCpjbGMkb3lZoSyDtX9Ix8pHiUrcyCjJoGlF0j2G05K2l
QQRsesZ3yBLWoI9zP95Bs+Vx6m60XqqVEGM1+Nlp06dx1jx(
CQLl9KmJ+pu+8HmxJhjGtT2tXTbMgHf34QDeaYgt9vje9jWl

GPG4Win: 解密文件



GPG4Win: 解密文件 (续)

解密后的文件内容:

电子科技大学科研项目申请书审查会签表

填表日期: 年 月 日

项目名称			
项目类型		所在单位	
项目负责人		联系电话	

项目负责人对所有上报材料中关于个人信息、申报内容均真是有效的承诺:

小结

□ PGP用到了哪些密码技术？

- ◆ 哈希函数
- ◆ 对称加密算法
- ◆ 公钥加密算法
- ◆ 数字签名

小结

- 学习PGP的主要目的是学习其设计思想，PGP已经有较长的历史了，随着技术的发展，其中加密模块、散列算法、数字签名模块有些已经过时，但是我们可以用最新的密码算法来替换不安全的密码算法。

课后作业

- 设计并实现一个基于口令的文件加解密软件
 - ◆ 用对称密码加密文件数据
 - ◆ 用PBE方式生成加密/解密密钥
 - ◆ C + openssl、Python3、