Lec 05. 保密通信技术

传统上，对称加密被用于实现消息保密性，面临**局域网内监听**和**搭线窃听**两种攻击方式；传输媒介主要有线缆、微波链路、卫星通道。

通信加密的两种基本选择：链路加密、端到端加密。

链路加密：在通信链路两端加上加密设备，链路上信息传输安全。

- **缺点**
  - 所有潜在通信链路两端都需要安装加密设备，要求许多加密设备；
  - 共享链路的每对节点都共享唯一密钥，密钥数量很大；
  - 每次分组交换都需要解密消息，报文在传输中多次加解密；
  - 报文在每个交换机处以明文形式存在，易受攻击，对用户透明；
  - 用户对报文的安全无法控制。
- **优点**
  - 能够鉴别主机；
  - 通讯模式是安全的。

端到端加密：在两端系统中进行，由源主机或终端加密数据，密文通过网络传给目的主机或终端。目的主机与源主机共享一个密钥。一般端到端仅加密数据，而不加密信息头，以实现包的路由。用户数据安全而传输过程不安全。

- **缺点**：不能隐蔽通信量；
- **优点**
  - 加密过程由两个端系统完成，报文在传输过程中仅进行了一次加解密；
  - 报文在交换机处仍然是安全（加密）的；
  - 能够鉴别用户。

两种方式的共用：主机用端到端加密密钥来加密用户数据；整个分组使用链路加密。分组在网络中传输时，各个节点用链路加密密钥来解密信息头。然后再加密，发送到下一条链路；仅在分组交换节点的存储器内可以看到信息头的明文。

现实的问题：WAP Security，较早的手机由于计算能力限制，需要经过外部网关进行通信，如果使用端到端加密，有**性能鸿沟**；有学者提出手机至 WAP 网关中使用链路加密，而 WAP 网关与外部使用端到端加密。

加密方式与层次的关系：链路加密一般处于较低层次，如 OSI 的物理层或链路层；端到端加密可以位于网络层、传输层、表示层或应用层，甚至可以用于两个不同体系结构的网络之间，也可以用于相同体系但相互隔离的网络之间，有着较好的灵活性。

端到端加密中不同加密策略的实现

- 只加密用户数据（加密位于应用层），TCP 头不需要加密——数据在任何时候不会以明文形式存在；
- 同时加密 TCP 头与数据（加密位于 TCP 层）——数据在过网关时（TCP 连接被终止并开始重新连接）会以明文形式存在；
- 同时加密 IP 头、TCP 头与数据（加密位于 link 层）——数据在路由器和网关中会以明文形式存。

- **结论**：加密所在的逻辑层越高，所需要加密的数据越少，而且数据的安全性越高（优点）；但是随着实体个数的上升，会更加复杂且密码更多（缺点）。
- 对网络层的加密，可以使用带有加密函数的前端处理器 FEP（通常是末端系统的网卡）来实现整个网络传输的安全性；对具有类似存储转发功能的电子邮件等应用，获得端到端加密只能在应用层上进行（否则将会在网关被打开）。

侧信道攻击：指监控成员间的通信量、消息数目、消息长度等，在军事和通信领域作用很大；

隐信道：允许进程以危害系统安全策略的方式用来传输信息的通信信道（以通信设备设计者所不知道的方式进行通信），如使用消息大小、消息频率来表示消息进行通信。

攻击者可从数据传输分析中得到的信息：通信双方的身份、通信双方通信的频率、报文模式、报文长度、报文书里那个、特定的通信之间交谈所关联的时间。主要防范手段有：

- 链路加密方式：
  - 分组首部（报文头）加密，但攻击者仍可以看到网络中的总通信量（由于分组加密长度不变）；
  - 通信量填充（浪费带宽）。
- 端到端加密方式：
  - 若为应用层，则攻击者可以判断出通信双方；若为传输层，则攻击者可以得到网络层的地址和传输格式；
  - 填充数据单元（浪费带宽）；
  - 发送空白报文（浪费带宽）。

对称加密中的密钥分配及管理

对称加密中的密钥分配及管理：对称密码要求消息双方共享密钥，且不为他人所知，那么如何分配和更新密钥成为了新的问题，处理不当会造成很多麻烦。对于参与者 A 与 B，有以下几种密钥分配方式：

1. 由 A 选定，物理地传送给 B；
2. 第三方选定，物理地传送给 A 和 B；
3. 使用 A 和 B 共有的密钥加密后传送给另一方；
4. A 和 B 都有到第三方的加密连接，则由第三方用加密连接传送给 A 与 B。

上述方法优缺点：

- 方法 1、2 需人工传递，对链路加密而言合理；对于端到端加密，由于大量密钥动态产生，密钥分配难度大；
- 方法 3 一旦暴露一个密钥，后续密钥都暴露了；
- 方法 4 适合于端到端加密，有许多变体，需要**可信第三方**。

问题的规模取决于通信的个数

- 在网络层或 IP 层实现端到端加密，要求每对主机都要有一个密钥，n 台主机需要 $n(n-1)/2$ 个密钥；
- 在应用层实现端到端加密，要求每对通信用户或进程有一个密钥，m 个用户需要 $m(m-1)/2$ 个密钥，一般来说 $m \geq n$。

广泛应用的是方法 4 的变体，即层次式密钥，其结构为：

- 有一个密钥分配中心 KDC (Key Distribution Center) 负责密钥分发（主机、进程、用户）；
- 每个用户与 KDC 共享一个密钥 (master key)，用于密钥分配。

密钥分级 (hierarchy)：典型的应用中，密码分为主密钥和会话密钥两级，其中

- **会话密钥**用于数据加密，是临时性的；
- **主密钥**由 KDC 和用户共享，是长期的。

其主要过程是 KDC 生成会话密钥 (session key)，然后用主密钥加密会话密钥并将加密后的会话密钥传给用户；用户收到会话密钥后用会话密钥加密数据并进行通信；因此 n 个用户需要 KDC 中含有 n 个主密钥。

基于对称加密的几个密钥分配协议及其攻击

在下文中，令 `E(m, K)` 表示用密钥 `K` 加密消息 `m`；`D(c, K)` 表示用密钥 `K` 解密密文 `c`。

密钥分配协议 1

```
(1) A send "Request (A, B)" to KDC;
(2) KDC respond "E(Ks, Ka) || E(Ks, Kb)" to A;
(3) A send "E(Ks, Kb)" to B.
```

最终，A 和 B 各自可以解密出会话密钥 `Ks` 并通信。

- **攻击 1**：攻击者 Eve 假扮成 KDC 并返回 `Ca, Cb`，则 A 与 B 解密出不同的会话密钥，无法进行通信；不包含对 KDC 的实体认证。
- **攻击 2**：攻击者 Eve 截获 A 给 KDC 发送的消息并将消息篡改为 `"Request (A, E)"`，等 KDC 返回加密信息后，Eve 截获 A 给 B 发送的消息（实际上即为 `"E(Ks, Ke)"`）并得到会话密钥，而 B 无法解密出相同的会话密钥；此时 A 认为其在与 B 通信，但实际上其在与 Eve 通信；不包含对 B 的实体认证 。
- **攻击 3**：（更强的中间人攻击）攻击者 Eve 同时假装 A 与 B 通信，并假装 B 与 A 通信；与攻击 2 类似，只需要将攻击 2 在 B 与 KDC 之间再运行一次即可（此时 Eve 与 A，Eve 与 B 的会话密钥不同）。

密钥分配协议 2

```
(1) A send "Request (A, B)" to KDC;
(2) KDC respond "E(Request || Ks || E(Ks, Kb), Ka)" to A;
(3) A send "E(Ks, Kb)" to B.
```

A 可通过验证 `Request` 确认其在与 KDC 通信，同时 Eve 无法解密 KDC 发送给 A 的消息，从而无法假扮成 A 与 B 通信。最终，A 和 B 各自可以解密出会话密钥 `Ks` 并通信。

- **攻击**：攻击者 Eve 截获上次 KDC 返回给 A 的消息，并假扮 KDC，返回相同的消息；则 A 与 B 的多次通信的会话密钥并未改变，通过密码分析进行攻击；不包含对消息的认证（新鲜性）。

密钥分配协议 3

```
(1) A send "Request (A, B) || N1" to KDC;
(2) KDC respond "E(Request || N1 || Ks || E(Ks, Kb), Ka)" to A;
(3) A send "E(Ks, Kb)" to B.
```

A 可以通过验证 `N1` 确认是本次发送的消息。最终，A 和 B 各自可以解密出会话密钥 `Ks` 并通信。

- **攻击**：攻击者 Eve 假扮 A 向 KDC 发消息请求与 B 通信，获得 Eve 与 B 的会话密钥；此时 B 认为其在 A 通信，但实际上其在与 Eve 通信；不包含对 A 的实体认证.

密钥分配协议 4

```
(1) A send "Request (A, B) || N1" to KDC;
(2) KDC respond "E(Request || N1 || Ks || E(Ks || ID(A), Kb), Ka)" to A;
(3) A send "E(Ks || ID(A), Kb)" to B.
```

B 可以通过验证 `ID(A)` 确认是在与 A 的通信。最终，A 和 B 各自可以解密出会话密钥 `Ks` 并通信。

- **攻击**：攻击者 Eve 截获 A 之前向 B 发送的消息，并假扮 A 向 B 发送消息，则 B 将会重用密钥；AB 通信中不包含对消息的认证（新鲜性）。

最终的密钥分配协议（原型为 Needham-Schroeder 协议，是 Kerberos 的基础）

```
(1) A send "ID(A) || ID(B) || N1" to KDC;
(2) KDC respond "E(ID(A) || ID(B) || N1 || Ks || E(Ks || ID(A), Kb), Ka)" to
A;
(3) A send "E(Ks || ID(A), Kb)" to B;
(4) B send "E(N2, Ks)" to A;
(5) A send "E(f(N2), Ks)" to B.
```

进一步地，B 可以通过验证 N2 保证消息的新鲜性（A 与 B 可以公开约定函数 `f()`）。是一个较为安全的密钥分配协议。此外，可以通过前端处理器（如加密网卡）产生对用户透明的密钥生成方案。

层次式密钥分配方案：网络规模很大时，单个 KDC 不实际，可以建立一系列 KDC，各个 KDC 之间存在层次关系，最底层的负责某一区域，不同区域之间的通信通过上一层 KDC 进行。其具有如下优点：

- 主密钥的分配工作量减小；
- 整个系统鲁棒性强；单个本地 KDC 出错或被破坏，其破坏不会影响到全局。

会话密钥的生命周期：会话密钥的分配过程会给网络造成负担；

- 对于面向连接的协议：一个连接一个会话密钥；如果连接时间很长，则可以选择协议数据单元 PDU 序号循环周期作为会话密钥的寿命长度；
- 对于无连接的协议：最好一个交互过程一个会话密钥；固定时间短或一定数量的交互量后更换一次会话密钥。

分散式密钥控制：（适用于没有可信第三方的小范围场景）在小的范围，可以假定所有用户之间共享密钥，任意两个人之间共享一个主密钥，则需要 $n(n-1)/2$ 个主密钥；每个节点需要保存 $(n-1)$ 个主密钥，可以产生很多会话密钥；由于会话密钥生命周期短，所以不容易被攻破。具体来说，分配方式如下：

```
A send "Request (A, B) || N1" to B;
B receive and send "E(Ks || Request || ID(B) || f(N1) || N2, Kab)" to B;
A send "E(f(N2), Ks)" to B.
```

其中，`Kab` 为 A、B 之间的主密钥；`f()` 是 A 与 B 约定的函数。

密钥的使用方式

- 几种不同的会话密钥：数据加密密钥、个人识别号 PIN 加密密钥、文件加密密钥等等；
- 使用校验码的比特位：一种基于 DES 的区分密钥的方式 (56 + 8)
    - 一个 bit 指示是会话密钥还是主密钥；一个 bit 指示是否可用于加密；一个 bit 指示是否可用于解密；其他 bit 保留。
    - **优点**：简单；
    - **缺点**：灵活性和功能性受限。
- **使用控制向量的方式**：控制向量 (CV) 经过 Hash 函数后与主密钥加，并用此加密会话密钥得到加密的会话密钥；从主密钥也可以根据控制向量解密会话密钥。
    - **优点**：CV 的长度无限制，可实现对密钥的任意复杂控制；CV 始终是明文形式，可以多次运用对密钥的控制要求。