加密文件系统设计

高嘉蕊, 14307130345
计算机科学与技术学院

2017 年 6 月 26 日

1 Server

服务器端只相当于物理存储，没有目录，全部扁平化拉平。保证服务器无法得到除了加密之后的文件内容之外的任何信息。同时对服务器的行为进行时时监测。

2 File System

2.1 文件系统整体设计

在同一个文件系统中，为每个用户建立单独文件夹，对文件夹的创建目录或文件等操作真实在文件夹中操作，同时本地保存文件备份。基于 Fuse 重写各种命令保证保证用户只能回退到 root，不能进入其他用户文件夹。用户分享过程中，为每一个分组单独创建文件夹。对每个分享组的用户而言，在自己的目录下也可以看到这个文件，但是实际文件的备份存储位置为分组中。

对每一个用户，生成一对用于 RSA 加密的密钥 (n_1, e_1, d_1)，一个用于 DES 加密的密钥 k，用户负责保存私钥 (n_1, d_1)。对每一个分享组，生成两对用于 RSA 加密的密钥 $(n_1, e_1, d_1), (n_2, e_2, d_2), (n_3, e_3, d$ 以及用于储存这些密钥的 RSA 密钥的 $(n_4, e_4, d_4), (n_5, e_5, d_5), (n_6, e_6, d_6)$，一个用于 DES 加密的密钥 k，分享组中有读权限的用户保存 (n_4, d_4)，有写权限的用户保存 (n_5, d_5)，有读写权限的用户保存 (n_6, d_6)。

在 root 目录中，保存七个文件：

- 全部用户公钥 (n_1, e_1)；
- 全部用户非分享文件的 (n_1, e_1) 加密后的 DES 密钥 K；
- 全部分享组的三组公钥 $(n_4, e_4), (n_5, e_5), (n_6, e_6)$；
- 全部分享组的 (n_4, e_4) 加密后的 $(n_1, d_1), (n_2, d_2)$；
- 全部分享组的 (n_5, e_5) 加密后的 $(n_1, d_1), (n_3, d_3)$；
- 全部分享组的 (n_6, e_6) 加密后的 $(n_1, d_1), (n_2, d_2), (n_3, d_3)$；
- 全部分享文件的 (n_2, e_2) 加密后的 DES 密钥 K。

每个用户目录中，保存两个文件：

- 全部共享文件的全部路径名；
- 全部共享文件的全部路径名与其映射的共享文件实际存储位置。

2.2 加密过程

2.2.1 文件名加密

用户使用公钥 (n_1, e_1) 加密文件全部路径，例如 userA/path/file1.txt，作为上传服务器的名字，如果出现与服务器上的文件重名（可能性非常小），则要求用户重命名才可以上传。保证每个文件的路径名是不同的。

分享组使用公钥 (n_1, e_1) 加密。其余同上。

2.2.2 文件内容加密

使用 DES 加密，密钥为 k。k 无需用户保存，使用 RSA 加密用户/分享组公钥 (n_1, e_1) 对 k 加密，之后保存在 root 目录下的文件中即可。

2.2.3 用户上传文件

- 用户用自己的私钥 (n_1, d_1) 解密得到 root 文件中自己的 DES 密钥 k，用 k 对文件内容进行加密。
- 计算加密后文件（部分文件）的 hash 值，同时把这个值用自己的私钥 (n_1, d_1) 签名，加到文件的 header 中（文件的最开始部分，不改 inode）。
- 对文件全部路径用公钥 (n_1, e_1) 加密，之后上传到文件服务器。如果文件服务器上出现重名，要求用户对文件重命名。
- 把加密上传的文件同时备份在本地。

2.2.4 用户读文件

- 对想读的文件的全部路径用 (n_1, e_1) 加密，在 server 寻找对应文件下载下来。
- 对文件进行验证，使用 (n_1, e_1) 对前一次写的签名进行解密，得到的字符串应当为文件（部分文件）的 hash 值。如果不不符，认为服务器或者其他原因文件被非法修改，则重新上传本地备份的文件到服务器。
- 如果验证成功，则认为读取到正确文件。使用私钥 (n_1, d_1) 解密得到 root 文件中自己的 DES 密钥 k，用 k 对文件解密。

2.2.5 用户写文件

- 首先按照 Sec2.2.4读文件。
- 完成写文件操作之后，按照 Sec2.2.3上传。
- 更新本地加密的备份文件。

2.2.6 用户分享文件

下面以用户 userA 把 test.txt 分享给 userB 读权限，userC 写权限，userD 读写权限为例：

- userA 选择 test.txt 为分享文件，则如果之前上述三个人没有共享过文件，则创新新文件夹 group_userA_userB_userC，同时生成对应的 7 个密钥，更新 root 中的文件。
- 更新三个用户自己目录下存储的共享文件对应表，保证可以找到共享文件的实际存储位置。
- userA 对按照 Sec2.2.4读取文件，并删除服务器上的对应文件。
- userA 使用分享组的 (n_2, d_2) 对分享组的 DES 密钥解密得到 k，对 test.txt 加密。
- 计算加密后 test.txt（部分文件）的 hash 值，同时把这个值用分享组的另一对私钥 (n_3, d_3) 加密签名，加到 test.txt 的 header 中。
- 删除 A 用户中原来的加密后的 test.txt 的备份，把新加密好的 test.txt 被分到路径 group_userA_userB_userC/test.txt 中。
- 对 test.txt 的新路径公钥 (n_1, e_1) 加密，之后上传到文件服务器。如果文件服务器上出现重名，要求用户对文件重命名。
- userA 把私钥 (n_4, d_4) 告诉有读权限的 userB，把 (n_5, d_5) 告诉有写权限的 userC，把 (n_6, d_6) 告诉有读写权限的 userD。

2.2.7 共享文件读写

通过 Sec2.2.6可以看到，用户需要有 (n_1, d_1) 才能读文件名字，同时再有 (n_2, d_2) 才能解密得到 DES 的 k 读文件，再得到 (n_3, d_3) 才能正确签名。为了让用户保存尽可能少的私钥信息，文件系统生成新的 RSA 密钥对，把用户需要保存的多个私钥加密存储在文件系统之中，用户只需要保存一个私钥。

注意在共享这里文件系统在：读文件、文件上传、备份文件更新之前都要进行签名检验。其余操作类似用户文件读写。

3 完成需求

由于服务器相当于物理存储，因此除了加密之后的文件内容外，得不到任何其他信息。

文件系统除了实现基本需求外，由于它只保存相关公钥、加密过的 DES 密钥、加密过的备份文件。所以，不仅恶意用户无法破坏文件系统，即使恶意用户破坏了文件系统，也只能得到其他用户的文件目录和文件夹名信息。

4 存在问题

- 用户需要保存的密钥太多，包括自己的私钥，以及每个分享组的私钥；
- 因为想要恢复被服务器非法更改的文件，只能进行文件本地备份。