Project 2 Design Doc - CS161

yiwen song (cs161-jp), Huadian Henry Zhang (cs161-sy)

1 Design Summary

1.1 User

User stores a location of the file database and a PRNG seed to derive encryption and MAC keys for the file database, and the username (for anti-spoofing). (Encrypted with a symmetric key encrypted and signed with public key). Located at HASH(username.userdata.DERexport(K_{user}^{-1})).

1.2 File Database

Name is randomly generated. Encryption and MAC keys are derived from User file (generated from random PRNG seed). Contents - contains a dictionary of the tuple:

- Filename
- Enc Key
- Mac Key

Each of these point to either a file descriptor or a sharing descriptor.

1.3 File Descriptor

Each FD contains:

1. the number of blocks in the file
2. a nonce
3. pointer to the latest transaction log entry (if any)
4. encryption key and MAC key of that log entry
5. encrypted and MAC shared list using the owner's PRNG seed

1.4 Share List

Each ShareList file is associated with a File Descriptor, and is located at the location obtained by $HMAC_{userdata.prngseed}(fileDescriptorName.sharing)$. The file is encrypted with the owner's File Database encryption and MAC keys, and thus makes it accessible to only the owner.

The ShareList contains a map of usernames to tuples of SharedDescriptor filename, encryption key, and MAC key.

1.5 Shared Descriptor

Contains:

1. Filename of parent
2. Enc Key of parent
3. MAC key of parent

Points to parent, which is either a shared descriptor or a file descriptor.

1.6 Transaction Log

Contains:

1. Nonse of the file descriptor update
2. Block numbers of the changes in that file
3. File descriptor and keys for the next descriptor and log entry

1.7 Data Block

Contains:

1. File name
2. Block Number
3. Data

"Put" at the file descriptor location + '.' + block number + '.' + data. Uses the same ENC and MAC keys as the file descriptor.

1.8 How to Share

Take the file descriptor, and generate a shared descriptor that points to the descriptor. Add entry in shared list (tuple as described), and return encrypted with location of new descriptor and key.

1.9 How to Receive

Take the message and add the descriptor location and key to your database.

1.10 How to Revoke

Re-generate key for file descriptor, relocate file descriptor, and re-encrypt data. Give everyone who was not revoked a new shared descriptor with this information. (Place at same location).

1.11 Diff Optimizations

Use transaction logging. When a user tries to upload, he goes through the transaction log and looks for which patch he owns (if any). He keeps a running total of all blocks he has changed since he last downloaded the file, and reuploads all blocks uploaded at those numbers that have changed.

2 Prevented Attacks

2.1 Old-share key attack

If Alice shares file f with Bob then revokes Bob's access, Bob is unable to access Alice's new updates after revokation, even if he stores the old key. This is because when Alice revokes Bob's access, Alice copies the entire file to a new location and changes the key for the file. If Bob were to try to use the old key, he is unable to decrypt this new file.

2.2 Revoked Child Attack

Alice shares a file f with Bob then Bob shares the same file with Carol. When Alice revokes Bob, Carol should not be able to get new updates to the file. This is possible because Carol's block points to Bob's block, which points to Alice's file. When Alice revokes Bob, Carol does not have access to the new key because Alice does not update Bob's block.

2.3 Share-stealing attack

When Alice sends a message to share a file with Bob, no one else should be able to receive this share. This is possible because Alice encrypts the share message with Bob's public key.

2.4 Non-owner revocation

When Alice shares a file with Bob and Carol, Bob should not be able to revoke Carol's access. Our implementation uses an encrypted share list, with a key that only Alice can access. There is also a MAC for this (also a key owned by just Alice). This means that if Bob changes anything, he is unable to calculate the MAC and we will raise an IntegrityError.

2.5 Buffer Overflow Attack

The client is written in Python, which is a memory-safe language.

2.6 Datablock Swap Attack

If a malicious server swaps two data blocks in a file, we should be able to detect that. Since we check the block number in the value