L15: Transactions

Frans Kaashoek & Nickolai Zeldovich
6.033 Spring 2012
Bank account transfer

xfer(bank, a, b, amt):
    bank[a] = bank[a] − amt
    bank[b] = bank[b] + amt
Bank account transfer

\[ \text{xfer}(\text{bank}, a, b, \text{amt}): \]

\[ \text{bank}[a] = \text{bank}[a] - \text{amt} \]
\[ \text{bank}[b] = \text{bank}[b] + \text{amt} \]

\[ \text{xfer}(\text{bank}, a, b, 50): \]

\[ a = 100, b = 100 \]
\[ a = 50, b = 100 \]
\[ a = 50, b = 150 \]
Bank account transfer

\[ \text{xfer}(\text{bank}, a, b, \text{amt}): \]
\[
\text{bank}[a] = \text{bank}[a] - \text{amt} \\
\text{bank}[b] = \text{bank}[b] + \text{amt}
\]

\[ \text{audit}(\text{bank}): \]
\[
\text{sum} = 0 \\
\text{for acct in bank:} \\
\quad \text{sum} = \text{sum} + \text{bank}[\text{acct}] \\
\text{return sum}
\]
Bank account transfer

\texttt{audit(bank):}

\texttt{xfer(bank, a, b, amt):}
\begin{align*}
    \text{bank}[a] &= \text{bank}[a] - \text{amt} \quad \text{\textcolor{red}{sum=200}} \\
    \text{bank}[b] &= \text{bank}[b] + \text{amt} \quad \text{\textcolor{red}{sum=150}} \\
\end{align*}

\texttt{audit(bank):}
\begin{align*}
    \text{sum} &= 0 \\
    \text{for acct in bank:} \\
        \quad \text{sum} &= \text{sum} + \text{bank}[\text{acct}] \\
    \text{return sum} \\
\end{align*}

\texttt{sum=200}
\texttt{sum=150}
\texttt{sum=200}
Eventual goal: transactions
all-or-nothing & before-or-after atomicity

\[ \text{xfer(bank, a, b, amt):} \]
  \[ \text{begin} \]
  \[ \text{bank}[a] = \text{bank}[a] - \text{amt} \]
  \[ \text{bank}[b] = \text{bank}[b] + \text{amt} \]
  \[ \text{commit} \]

\[ \text{audit(bank):} \]
  \[ \text{begin} \]
  \[ \text{sum} = 0 \]
  \[ \text{for acct in bank:} \]
    \[ \text{sum} = \text{sum} + \text{bank}[\text{acct}] \]
  \[ \text{commit} \]
  \[ \text{return sum} \]
Strawman implementation

```python
xfer(bankfile, a, b, amt):
    bank = read_accounts(bankfile)
    bank[a] = bank[a] – amt
    bank[b] = bank[b] + amt
    write_accounts(bankfile)
```
Shadow copy

`xfer(bankfile, a, b, amt):
    bank = read_accounts(bankfile)
    bank[a] = bank[a] - amt
    bank[b] = bank[b] + amt
    write_accounts("#bankfile")
    rename("#bankfile", bankfile)`
File system data structures

directory data blocks:
  filename “bank” → inode 12
  filename “#bank” → inode 13

inode 12:
  data blocks: 3, 4, 5
  refcount: 1

inode 13:
  data blocks: 6, 7, 8
  refcount: 1
rename(“#bank”, “bank”)

directory data blocks:
  filename “bank” → inode 12
  filename “#bank” → inode 13

inode 12:
  data blocks: 3, 4, 5
  refcount: 1

inode 13:
  data blocks: 6, 7, 8
  refcount: 1
rename(“#bank”, “bank”)

directory data blocks:
    filename “bank” → inode 12
    filename “#bank” → inode 13

inode 12:
    data blocks: 3, 4, 5
    refcount: 1

inode 13:
    data blocks: 6, 7, 8
    refcount: 2
rename(“#bank”, “bank”)

directory data blocks:
  filename “bank” → inode 13
  filename “#bank” → inode 13

inode 12:
  data blocks: 3, 4, 5
  refcount: 1

inode 13:
  data blocks: 6, 7, 8
  refcount: 2
rename(“#bank”, “bank”)

directory data blocks:
  filename “bank” → inode 13
  filename “#bank” → inode 13

inode 12:
  data blocks: 3, 4, 5
  refcount: 0

inode 13:
  data blocks: 6, 7, 8
  refcount: 2
rename(""#bank", "bank")

directory data blocks:
   filename "bank" → inode 13
   filename "#bank" → inode 13

inode 12:
   data blocks: 3, 4, 5
   refcount: 0

inode 13:
   data blocks: 6, 7, 8
   refcount: 2
rename("#bank", "bank")

directory data blocks:
  filename “bank” → inode 13
  filename “#bank” → inode 13

inode 12:
  data blocks: 3, 4, 5
  refcount: 0

inode 13:
  data blocks: 6, 7, 8
  refcount: 1
Recovery after crash

```python
salvage(disk):
    for inode in disk.inodes:
        inode.refcnt =
            find_all.refs(disk.root_dir, inode)

    if exists("#bank"):
        unlink("#bank")
```