English Edition

9. File Systems (4)

- 9.5 Practice: Windows
- 9.6 Theory

/home/esser/Daten/Dozent/Folien/bs2-esser-10-english.odp

```
Sep 19 14:27:41 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 20 01:00:01 amd64 /usr/sbin/cron[29278]: (root) CMD (/sbin/evlogmgr -c "severity=DEBUG")
Sep 20 01:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 20 02:00:01 amd64 /usr/sbin/cron[30103]: (root) CMD (/sbin/evlogmgr -c 'age > "30d"')
Sep 20 02:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 20 12:46:44 amd64 sshd[6516]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62004
Sep 20 12:46:44 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 20 12:48:41 amd64 sshd[6609]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62105
Sep 20 12:54:44 amd64 sshd[6694]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62514
Sep 20 15:27:35 amd64 sshd[9077]: Accepted rsa for esser from ::ffff:87.234.201.207 port 64242
Sep 20 15:27:35 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 20 16:37:11 amd64 sshd[10102]: Accepted rsa for esser from ::ffff:87.234.201.207 port 63375
Sep 20 16:37:11 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 20 16:38:10 amd64 sshd[10140]: Accepted rsa for esser from ::ffff:87.234.201.207 port 63546
Sep 21 01:00:01 amd64 /usr/sbin/cron[17055]: (root) CMD (/sbin/evlogmgr -c "severity=DEBUG")
Sep 21 01:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 21 02:00:01 amd64 /usr/sbin/cron[17878]: (root) CMD (/sbin/evlogmgr -c 'age > "30d"')
Sep 21 02:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 21 17:43:26 amd64 sshd[31088]: Accepted rsa for esser from ::ffff:87.234.201.207 port 63397
Sep 21 17:43:26 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 21 17:53:39 amd64 sshd[31269]: Accepted rsa for esser from ::ffff:87.234.201.207 port 64391
Sep 21 18:43:26 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 21 19:43:26 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 22 01:00:01 amd64 /usr/sbin/cron[4674]: (root) CMD (/sbin/evlogmgr -c "severity=DEBUG")
Sep 22 01:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 22 02:00:01 amd64 /usr/sbin/cron[5499]: (root) CMD (/sbin/evlogmgr -c 'age > "30d"')
Sep 22 02:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 22 20:23:21 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 23 01:00:01 amd64 /usr/sbin/cron[24739]: (root) CMD (/sbin/evlogmgr -c "severity=DEBUG")
Sep 23 01:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 23 02:00:01 amd64 /usr/sbin/cron[[illegible]555]: (root) CMD (/sbin/evlogmgr -c 'age > "30d"')
Sep 23 02:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 23 18:04:05 amd64 sshd[6554]: Accepted publickey for esser from ::ffff:192.168.1.5 port 59771 ssh2
Sep 23 18:04:05 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 23 18:04:34 amd64 sshd[6606]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62093
Sep 24 01:00:01 amd64 /usr/sbin/cron[12436]: (root) CMD (/sbin/evlogmgr -c "severity=DEBUG")
Sep 24 01:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 24 02:00:01 amd64 /usr/sbin/cron[13253]: (root) CMD (/sbin/evlogmgr -c 'age > "30d"')
Sep 24 02:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 24 11:15:48 amd64 sshd[20998]: Accepted rsa for esser from ::ffff:87.234.201.207 port 64456
Sep 24 11:15:48 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 24 13:49:08 amd64 sshd[23197]: Accepted rsa for esser from ::ffff:87.234.201.207 port 61330
Sep 24 13:49:08 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 24 15:42:07 amd64 kernel: snd_seq_midi_event: unsupported module, tainting kernel.
Sep 24 15:42:07 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 24 15:42:07 amd64 kernel: snd_seq_oss: unsupported module, tainting kernel.
Sep 24 20:25:31 amd64 sshd[29399]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62566
Sep 24 20:25:31 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 25 01:00:02 amd64 /usr/sbin/cron[662]: (root) CMD (/sbin/evlogmgr -c "severity=DEBUG")
Sep 25 01:00:02 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 25 02:00:01 amd64 /usr/sbin/cron[1484]: (root) CMD (/sbin/evlogmgr -c 'age > "30d"')
Sep 25 02:00:02 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 25 10:59:25 amd64 sshd[8889]: Accepted rsa for esser from ::ffff:87.234.201.207 port 64183
Sep 25 10:59:25 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 25 10:59:47 amd64 sshd[8921]: Accepted rsa for esser from ::ffff:87.234.201.207 port 64253
Sep 25 11:30:02 amd64 sshd[9372]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62029
Sep 25 11:59:25 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 25 14:05:37 amd64 sshd[11554]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62822
Sep 25 14:05:37 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 25 14:06:10 amd64 sshd[11586]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62951
Sep 25 14:07:17 amd64 sshd[11608]: Accepted rsa for esser from ::ffff:87.234.201.207 port 63392
Sep 25 14:08:33 amd64 sshd[11630]: Accepted rsa for esser from ::ffff:87.234.201.207 port 63709
Sep 25 15:25:33 amd64 sshd[12930]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62778
```

Windows File Access (1)

- On a low level Windows provides the same file access functions as Linux / Unix, and they have the same syntax
  - fd=**open** (), **close** (fd)
  - **read** (fd,...), **write** (fd,...)
- **fopen**, **fread** exist on Windows as well
- however, the standard way is to use Windows-specific file functions

9.5 Practice: File Access with Windows

```
Sep 19 14:20:18 amd64 sshd[20494]: Accepted rsa for esser from ::ffff:87.234.201.207 port 61557
Sep 19 14:27:41 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 20 01:00:01 amd64 /usr/sbin/cron[29278]: (root) CMD (/sbin/evlogmgr -c "severity=DEBUG")
Sep 20 01:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 20 02:00:01 amd64 /usr/sbin/cron[30103]: (root) CMD (/sbin/evlogmgr -c 'age > "30d"')
Sep 20 02:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 20 12:46:44 amd64 sshd[6516]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62004
Sep 20 12:46:44 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 20 12:48:41 amd64 sshd[6609]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62105
Sep 20 12:54:44 amd64 sshd[6694]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62514
Sep 20 15:27:35 amd64 sshd[9077]: Accepted rsa for esser from ::ffff:87.234.201.207 port 64242
Sep 20 15:27:35 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 20 16:37:11 amd64 sshd[10102]: Accepted rsa for esser from ::ffff:87.234.201.207 port 63375
Sep 20 16:37:11 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 20 16:38:10 amd64 sshd[10140]: Accepted rsa for esser from ::ffff:87.234.201.207 port 63546
Sep 21 01:00:01 amd64 /usr/sbin/cron[17055]: (root) CMD (/sbin/evlogmgr -c "severity=DEBUG")
Sep 21 01:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 21 02:00:01 amd64 /usr/sbin/cron[17878]: (root) CMD (/sbin/evlogmgr -c 'age > "30d"')
Sep 21 02:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 21 17:43:26 amd64 sshd[31088]: Accepted rsa for esser from ::ffff:87.234.201.207 port 63397
Sep 21 17:43:26 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 21 17:53:39 amd64 sshd[31269]: Accepted rsa for esser from ::ffff:87.234.201.207 port 64391
Sep 21 18:43:26 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 21 19:43:26 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 22 01:00:01 amd64 /usr/sbin/cron[4674]: (root) CMD (/sbin/evlogmgr -c "severity=DEBUG")
Sep 22 01:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 22 02:00:01 amd64 /usr/sbin/cron[5499]: (root) CMD (/sbin/evlogmgr -c 'age > "30d"')
Sep 22 02:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 22 20:23:21 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 23 01:00:01 amd64 /usr/sbin/cron[24739]: (root) CMD (/sbin/evlogmgr -c "severity=DEBUG")
Sep 23 01:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 23 02:00:01 amd64 /usr/sbin/cron[[illegible]555]: (root) CMD (/sbin/evlogmgr -c 'age > "30d"')
Sep 23 02:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 23 18:04:05 amd64 sshd[6554]: Accepted publickey for esser from ::ffff:192.168.1.5 port 59771 ssh2
Sep 23 18:04:05 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 23 18:04:34 amd64 sshd[6606]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62093
Sep 24 01:00:01 amd64 /usr/sbin/cron[12436]: (root) CMD (/sbin/evlogmgr -c "severity=DEBUG")
Sep 24 01:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 24 02:00:01 amd64 /usr/sbin/cron[13253]: (root) CMD (/sbin/evlogmgr -c 'age > "30d"')
Sep 24 02:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 24 11:15:48 amd64 sshd[20998]: Accepted rsa for esser from ::ffff:87.234.201.207 port 64456
Sep 24 11:15:48 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 24 13:49:08 amd64 sshd[23197]: Accepted rsa for esser from ::ffff:87.234.201.207 port 61330
Sep 24 13:49:08 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 24 15:42:07 amd64 kernel: snd_seq_midi_event: unsupported module, tainting kernel.
Sep 24 15:42:07 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 24 15:42:07 amd64 kernel: snd_seq_oss: unsupported module, tainting kernel.
Sep 24 20:25:31 amd64 sshd[29399]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62566
Sep 24 20:25:31 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 25 01:00:02 amd64 /usr/sbin/cron[662]: (root) CMD (/sbin/evlogmgr -c "severity=DEBUG")
Sep 25 01:00:02 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 25 02:00:01 amd64 /usr/sbin/cron[1484]: (root) CMD (/sbin/evlogmgr -c 'age > "30d"')
Sep 25 02:00:02 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 25 10:59:25 amd64 sshd[8889]: Accepted rsa for esser from ::ffff:87.234.201.207 port 64183
Sep 25 10:59:25 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 25 10:59:47 amd64 sshd[8921]: Accepted rsa for esser from ::ffff:87.234.201.207 port 64253
Sep 25 11:30:02 amd64 sshd[9372]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62029
Sep 25 11:59:25 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 25 14:05:37 amd64 sshd[11554]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62822
Sep 25 14:05:37 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 25 14:06:10 amd64 sshd[11586]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62951
Sep 25 14:07:17 amd64 sshd[11608]: Accepted rsa for esser from ::ffff:87.234.201.207 port 63392
Sep 25 14:08:33 amd64 sshd[11630]: Accepted rsa for esser from ::ffff:87.234.201.207 port 63709
Sep 25 15:25:33 amd64 sshd[12930]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62778
```

Windows File Access (2)

- Windows standard functions are object-based:
  - handle = **CreateFile ()** creates / opens a file
  - **ReadFile** (handle, ...) reads from a file
  - **WriteFile** (handle, ...) writes into a file
  - **CloseFile** (handle) closes a file
  - **SetFilePointer** (handle,...) change read/write position
  - **WaitForSingleObject** (handle) wait for termination of a file I/O operation

Windows File Access (3)

- open / create file: **CreateFile**

```
HANDLE CreateFile(
  LPCTSTR lpFileName,
  DWORD dwDesiredAccess,
  DWORD dwShareMode,
  LPSECURITY_ATTRIBUTES lpSecurityAttributes,
  DWORD dwCreationDisposition,
  DWORD dwFlagsAndAttributes,
  HANDLE hTemplateFile);
```

example 1: open a file for reading

```
hFile = CreateFile("ONE.TXT",          // open ONE.TXT
        GENERIC_READ,                   // open for reading
        0,                              // do not share
        NULL,                           // no security
        OPEN_EXISTING,                  // existing file only
        FILE_ATTRIBUTE_NORMAL,          // normal file
        NULL);                          // no attr. template
```

Windows File Access (4)

example 2: open a file in append mode

```
hAppend = CreateFile("TWO.TXT", // open TWO.TXT
  GENERIC_WRITE,                // open for writing
  0,                            // do not share
  NULL,                         // no security
  OPEN_ALWAYS,                  // open or create
  FILE_ATTRIBUTE_NORMAL,        // normal file
  NULL);                        // no attr. template
```

append data of ONE.TXT to TWO.TXT; with locking

```
do {
  if (ReadFile(hFile, buff, 4096, &dwBytesRead, NULL)) {
    dwPos = SetFilePointer(hAppend, 0, NULL, FILE_END);
    LockFile(hAppend, dwPos, 0, dwPos + dwBytesRead, 0);
    WriteFile(hAppend, buff, dwBytesRead,&dwBytesWritten, NULL);
    UnlockFile(hAppend, dwPos, 0, dwPos + dwBytesRead, 0);
  }
} while (dwBytesRead == 4096);
// Close both files.
CloseHandle(hFile);
CloseHandle(hAppend);
```

Windows File Access (5)

- reading: **ReadFile**

```
BOOL ReadFile(
  HANDLE hFile,
  LPVOID lpBuffer,
  DWORD nNumberOfBytesToRead,
  LPDWORD lpNumberOfBytesRead,
  LPOVERLAPPED lpOverlapped);
```

- writing: **WriteFile**

```
BOOL WriteFile(
  HANDLE hFile,
  LPCVOID lpBuffer,
  DWORD nNumberOfBytesToWrite,
  LPDWORD lpNumberOfBytesWritten,
  LPOVERLAPPED lpOverlapped);
```

Windows File Access (6)

- example: copy a file and convert the data

```
#include <windows.h>
#include <stdio.h>
#define BUFSIZE 512

int main() {
  HANDLE hFile, hTempFile;
  DWORD dwRetVal, dwBytesRead, dwBytesWritten, dwBufSize=BUFSIZE;
  UINT uRetVal; BOOL fSuccess;
  char szTempName[BUFSIZE], buffer[BUFSIZE], lpPathBuffer[BUFSIZE];

  // Open the existing file.
  hFile = CreateFile("original.txt",        // file name
                     GENERIC_READ,          // open for reading
                     0,                     // do not share
                     NULL,                  // default security
                     OPEN_EXISTING,         // existing file only
                     FILE_ATTRIBUTE_NORMAL, // normal file
                     NULL);                 // no template
  if (hFile == INVALID_HANDLE_VALUE) {
      printf ("CreateFile failed with error %d.\n", GetLastError()); return (1);
  }
```

Windows File Access (7)

```
// Get the temp path.
dwRetVal = GetTempPath(dwBufSize,     // length of the buffer
                       lpPathBuffer); // buffer for path
if (dwRetVal > dwBufSize || (dwRetVal == 0)) {
    printf ("GetTempPath failed with error %d.\n", GetLastError()); return (2);
}

// Create a temporary file.
uRetVal = GetTempFileName(lpPathBuffer, // directory for tmp files
                          "NEW",        // temp file name prefix
                          0,            // create unique name
                          szTempName);  // buffer for name
if (uRetVal == 0) {
    printf ("GetTempFileName failed with error %d.\n", GetLastError()); return (3);
}

// Create the new file to write the upper-case version to.
hTempFile = CreateFile((LPTSTR) szTempName, // file name
                       GENERIC_READ | GENERIC_WRITE, // open r-w
                       0,                    // do not share
                       NULL,                 // default security
                       CREATE_ALWAYS,        // overwrite existing
                       FILE_ATTRIBUTE_NORMAL,// normal file
                       NULL);                // no template
if (hTempFile == INVALID_HANDLE_VALUE) {
    printf ("CreateFile failed with error %d.\n", GetLastError()); return (4);
}
```

Windows File Access (8)

```
// Read BUFSIZE blocks to the buffer. Change all characters in the buffer to
// upper case. Write the buffer to the temporary file.
do {
    if (ReadFile(hFile, buffer, BUFSIZE, &dwBytesRead, NULL)) {
        CharUpperBuff(buffer, dwBytesRead);
        fSuccess = WriteFile(hTempFile, buffer, dwBytesRead,
                             &dwBytesWritten, NULL);
        if (!fSuccess) {
            printf ("WriteFile failed with error %d.\n", GetLastError()); return (5);
        }
    } else {
        printf ("ReadFile failed with error %d.\n", GetLastError()); return (6);
    }
} while (dwBytesRead == BUFSIZE);
// Close the handles to the files.
fSuccess = CloseHandle (hFile);
if (!fSuccess) {
   printf ("CloseHandle failed with error %d.\n", GetLastError()); return (7);
}
fSuccess = CloseHandle (hTempFile);
if (!fSuccess) {
   printf ("CloseHandle failed with error %d.\n", GetLastError()); return (8);
}
// Move the temporary file to the new text file.
fSuccess = MoveFileEx(szTempName, "allcaps.txt", MOVEFILE_REPLACE_EXISTING);
if (!fSuccess) {
    printf ("MoveFileEx failed with error %d.\n", GetLastError()); return (9);
}
return (0);
}
```

Windows File Access (9)

- jump to a specific position within the file

```
DWORD SetFilePointer(
  HANDLE hFile,
  LONG lDistanceToMove,
  PLONG lpDistanceToMoveHigh,
  DWORD dwMoveMethod
);
```

hFile: [in] A handle to the file that has a file pointer to be moved.

lDistanceToMove: [in] The low order 32-bits of a signed value that specifies the number of bytes to move the file pointer.

lpDistanceToMoveHigh: [in, out, optional] A pointer to the high order 32-bits of the signed 64-bit distance to move. If you do not need the high order 32-bits, this pointer must be set to NULL.

dwMoveMethod: [in] The starting point for the file pointer move:

FILE_BEGIN	The starting point is 0 (zero) or the beginning of the file.
FILE_CURRENT	The starting point is the current value of the file pointer.
FILE_END	The starting point is the current end-of-file position.

Searching for Files (1)

- hFind = **FindFirstFile** ()

```
HANDLE FindFirstFile(
  LPCTSTR lpFileName,
  LPWIN32_FIND_DATA lpFindFileData
);
```

lpFileName: [in] A pointer to a null-terminated string that specifies a valid directory or path, and file name that can contain wildcard characters, for example, an asterisk (*) or a question mark (?).

lpFindFileData: [out] A pointer to the WIN32_FIND_DATA structure that receives information about a found file or subdirectory.

- **FindNextFile** (hFind)
- **FindClose** (hFind)

```
BOOL FindNextFile(
  HANDLE hFindFile,
  LPWIN32_FIND_DATA lpFindFileData
);
```

hFindFile: [in] Search handle returned by a previous call to the FindFirstFile or FindFirstFileEx function.

lpFindFileData: [out] Pointer to the WIN32_FIND_DATA structure that receives information about the found file or subdirectory. The structure can be used in subsequent calls to FindNextFile to indicate from which file to continue the search.

http://msdn2.microsoft.com/en-us/library/aa364418.aspx, http://msdn2.microsoft.com/en-us/library/aa364428.aspx

Searching for Files (2)

source: http://msdn2.microsoft.com/
en-us/library/aa365200.aspx

```c
#define _WIN32_WINNT 0x0501
#include <windows.h>
#include <stdio.h>
#include <strsafe.h>
#include <malloc.h>
#define BUFSIZE MAX_PATH

int main(int argc, char *argv[]) {
   WIN32_FIND_DATA FindFileData;
   HANDLE hFind = INVALID_HANDLE_VALUE;
   DWORD dwError; LPSTR DirSpec;
   size_t length_of_arg;

   DirSpec = (LPSTR) malloc (BUFSIZE);

   // Check for command-line parameter; otherwise, print usage.
   if(argc != 2) { printf("Usage: Test <dir>\n"); return 2; }

   // Check that the input is not larger than allowed.
   StringCbLength(argv[1], BUFSIZE, &length_of_arg);
   if (length_of_arg > (BUFSIZE - 2)) {
      printf("Input directory is too large.\n");
      return 3;
   }

   printf ("Target directory is %s.\n", argv[1]);
```

Searching for Files (3)

```c
   // Prepare string for use with FindFile functions.  First, copy the
   // string to a buffer, then append '\*' to the directory name.
   StringCbCopyN (DirSpec, BUFSIZE, argv[1], length_of_arg+1);
   StringCbCatN (DirSpec, BUFSIZE, "\\*", 3);

   // Find the first file in the directory.
   hFind = FindFirstFile(DirSpec, &FindFileData);

   if (hFind == INVALID_HANDLE_VALUE) {
      printf ("Invalid file handle. Error is %u.\n", GetLastError()); return (-1);
   } else {
      printf ("First file name is %s.\n", FindFileData.cFileName);

      // List all the other files in the directory.
      while (FindNextFile(hFind, &FindFileData) != 0) {
         printf ("Next file name is %s.\n", FindFileData.cFileName);
      }
      dwError = GetLastError();
      FindClose(hFind);
      if (dwError != ERROR_NO_MORE_FILES) {
         printf ("FindNextFile error. Error is %u.\n", dwError); return (-1);
      }
   }

   free(DirSpec);
   return (0);
}
```

Windows: stdin, stdout, stderr

- there are three standard file descriptors (as on Linux/Unix)
  - Standard Input (0, stdin)
  - Standard Output (1, stdout)
  - Standard Error Output (2, stderr)
- These can be accessed the same way as on a Linux system:

```c
#include <io.h>     /* Linux: <unistd.h> */
void main() {
  if (write(1, "Here is some data\n", 18) != 18)
    write(2, "A write error has occurred on file descriptor 1\n",46);
}
```

9.6 Theory

```
Sep 19 14:27:41 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 20 01:00:01 amd64 /usr/sbin/cron[29278]: (root) CMD (/sbin/evlogmgr -c "severity=DEBUG")
Sep 20 01:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 20 02:00:01 amd64 /usr/sbin/cron[30103]: (root) CMD (/sbin/evlogmgr -c 'age > "30d"')
Sep 20 02:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 20 12:46:44 amd64 sshd[6516]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62004
Sep 20 12:46:44 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 20 12:48:41 amd64 sshd[6609]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62105
Sep 20 12:54:44 amd64 sshd[6694]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62514
Sep 20 15:27:35 amd64 sshd[9077]: Accepted rsa for esser from ::ffff:87.234.201.207 port 64242
Sep 20 15:27:35 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 20 16:37:11 amd64 sshd[10102]: Accepted rsa for esser from ::ffff:87.234.201.207 port 63375
Sep 20 16:37:11 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 20 16:38:10 amd64 sshd[10140]: Accepted rsa for esser from ::ffff:87.234.201.207 port 63546
Sep 21 01:00:01 amd64 /usr/sbin/cron[17055]: (root) CMD (/sbin/evlogmgr -c "severity=DEBUG")
Sep 21 01:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 21 02:00:01 amd64 /usr/sbin/cron[17878]: (root) CMD (/sbin/evlogmgr -c 'age > "30d"')
Sep 21 02:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 21 17:43:26 amd64 sshd[31088]: Accepted rsa for esser from ::ffff:87.234.201.207 port 63397
Sep 21 17:43:26 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 21 17:53:39 amd64 sshd[31269]: Accepted rsa for esser from ::ffff:87.234.201.207 port 64391
Sep 21 18:43:26 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 21 19:43:26 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 22 01:00:01 amd64 /usr/sbin/cron[4674]: (root) CMD (/sbin/evlogmgr -c "severity=DEBUG")
Sep 22 01:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 22 02:00:01 amd64 /usr/sbin/cron[5499]: (root) CMD (/sbin/evlogmgr -c 'age > "30d"')
Sep 22 02:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 22 20:23:21 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 23 01:00:01 amd64 /usr/sbin/cron[24739]: (root) CMD (/sbin/evlogmgr -c "severity=DEBUG")
Sep 23 01:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 23 02:00:01 amd64 /usr/sbin/cron[25555]: (root) CMD (/sbin/evlogmgr -c 'age > "30d"')
Sep 23 02:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 23 18:04:05 amd64 sshd[6554]: Accepted publickey for esser from ::ffff:192.168.1.5 port 59771 ssh2
Sep 23 18:04:05 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 23 18:04:34 amd64 sshd[6606]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62093
Sep 24 01:00:01 amd64 /usr/sbin/cron[12436]: (root) CMD (/sbin/evlogmgr -c "severity=DEBUG")
Sep 24 01:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 24 02:00:01 amd64 /usr/sbin/cron[13253]: (root) CMD (/sbin/evlogmgr -c 'age > "30d"')
Sep 24 02:00:01 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 24 11:15:48 amd64 sshd[20998]: Accepted rsa for esser from ::ffff:87.234.201.207 port 64456
Sep 24 11:15:48 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 24 13:49:08 amd64 sshd[23197]: Accepted rsa for esser from ::ffff:87.234.201.207 port 61330
Sep 24 13:49:08 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 24 15:42:07 amd64 kernel: snd_seq_midi_event: unsupported module, tainting kernel.
Sep 24 15:42:07 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 24 15:42:07 amd64 kernel: snd_seq_oss: unsupported module, tainting kernel.
Sep 24 20:25:31 amd64 sshd[29399]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62566
Sep 24 20:25:31 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 25 01:00:02 amd64 /usr/sbin/cron[662]: (root) CMD (/sbin/evlogmgr -c "severity=DEBUG")
Sep 25 01:00:02 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 25 02:00:01 amd64 /usr/sbin/cron[1484]: (root) CMD (/sbin/evlogmgr -c 'age > "30d"')
Sep 25 02:00:02 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 25 10:59:25 amd64 sshd[8889]: Accepted rsa for esser from ::ffff:87.234.201.207 port 64183
Sep 25 10:59:25 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 25 10:59:47 amd64 sshd[8921]: Accepted rsa for esser from ::ffff:87.234.201.207 port 64253
Sep 25 11:30:02 amd64 sshd[9372]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62029
Sep 25 11:59:25 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 25 14:05:37 amd64 sshd[11554]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62822
Sep 25 14:05:37 amd64 syslog-ng[7653]: STATS: dropped 0
Sep 25 14:06:10 amd64 sshd[11586]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62951
Sep 25 14:07:17 amd64 sshd[11608]: Accepted rsa for esser from ::ffff:87.234.201.207 port 63392
Sep 25 14:08:33 amd64 sshd[11630]: Accepted rsa for esser from ::ffff:87.234.201.207 port 63709
Sep 25 15:25:33 amd64 sshd[12930]: Accepted rsa for esser from ::ffff:87.234.201.207 port 62778
```

Alle images in this subsection are taken from **William Stallings**: „Operating Systems“, 5th edition (2005), chapter 12; slides are based on slides by **Patricia Roy**, Manatee Community College (with small modifications)

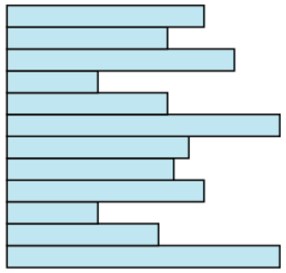
File Organization (1)

Criteria for File Organization

- Short access time
  - Needed when accessing a single record
  - Not needed for batch mode
- Ease of update
  - Files on CD-ROM are not updated
- Economy of storage
  - there should be minimum redundancy in the data
  - use redundancy to speed access, e.g. an index
- Simple maintenance
- Reliability

File Organization (2)

- The Pile
  - Data are collected in the order they arrive
  - Purpose is to accumulate a mass of data and save it
  - Records may have different fields
  - No structure
  - Record access is by exhaustive search



Variable-length records
Variable set of fields
Chronological order

(a) Pile File

File Organization (3)

- Example for **Pile**: contents of typical configuration files (here: Wine configuration)

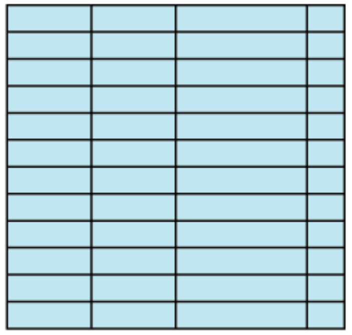
```
[Drive A]
"Type" = "floppy"
"Path" = "/media/floppy"
"Label" = "/media/floppy"
"Device" = "/dev/fd0"

[Drive C]
"Path" = "${HOME}/.wine/fake_windows"
"Type" = "hd"
"Label" = "fake_windows"
"Filesystem" = "win95"

[Drive D]
"Path"="/media/cdrecorder"
"Label"="cdrecorder"
"Type"="cdrom"
"FS"="win95"
;"Device"="/dev/cdrecorder"

[Drive E]
"Path"="/media/cdrom"
"Label"="cdrom"
"Type"="cdrom"
"FS"="win95"
;"Device"="/dev/cdrom"
```

File Organization (4)

- The Sequential File
  - Fixed format used for records
  - Records are the same length
  - All fields the same (order and length)
  - Field names and lengths are attributes of the file
  - One field is the key field
    - Uniquely identifies the record
    - Records are stored in key sequence

File Organization (5)

- The Sequential File
  - New records are placed in a log file or transaction file
  - Batch update is performed to merge the log file with the master file



Fixed-length records
Fixed set of fields in fixed order
Sequential order based on key field

(b) Sequential File

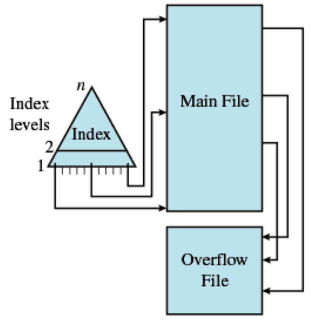
File Organization (6)

- example for **sequential file**:

```
struct data {
  int id;            // length  4
  char name[30];     // length 30
  char vorname[30];  // length 30
};

int blocksize = sizeof(struct data);
                     // 30+30+4=64

struct data buffer;

main() {
  printf ("block size: %d \n", blocksize);
}
```

File Organization (7)

- Indexed Sequential File
  - Index provides a lookup capability to quickly reach the vicinity of the desired record
    - Contains key field and a pointer to the main file
    - Index is searched to find highest key value that is equal to or precedes the desired key value
    - Search continues in the main file at the location indicated by the pointer



(c) Indexed Sequential File

File Organization (8)

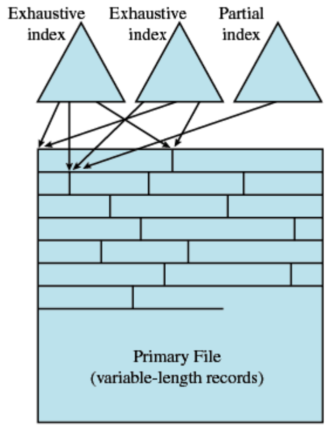
- Comparison of sequential and indexed sequential files
  - Example: a file contains 1 million records
  - On average 500,000 accesses are required to find a record in a sequential file
  - If an index contains 1000 entries, it will take an average 500 accesses to find the key, followed by 500 accesses in the main file.
  - Now on average it takes 1000 accesses (that's a factor of 1/500)

File Organization (9)

- Indexed Sequential File
  - New records are added to an overflow file
  - Record in main file that precedes it is updated to contain a pointer to the new record
  - The overflow is merged with the main file during a batch update
  - Multiple indexes for the same key field can be created to increase efficiency

File Organization (10)

- Indexed File
  - Uses multiple indexes for different key fields
  - May contain an exhaustive index that contains one entry for every record in the main file
  - May contain a partial index (only entries for records in which this field exists)



(d) Indexed File

Secondary Memory Management (1)

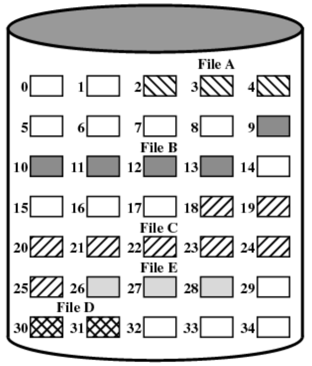
- Space must be allocated to files
- Must keep track of the space available for allocation

- Pre-Allocation
  - Need the maximum size for the file at the time of creation
  - Difficult to reliably estimate the maximum potential size of the file
  - Tend to over-estimate file size so as not to run out of space

Secondary Memory Management (2)

- Contiguous allocation
  - Single set of blocks is allocated to a file at the time of creation
  - Only a single entry in the file allocation table
    - Starting block and length of the file
- External fragmentation will occur
  - Need to perform compaction

Secondary Memory Management (3)

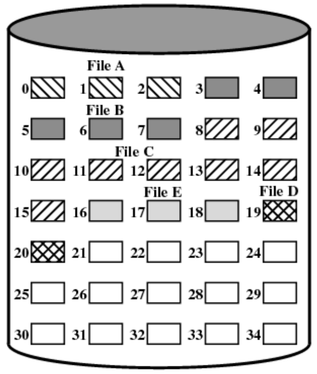


File Allocation Table

File Name	Start Block	Length
File A	2	3
File B	9	5
File C	18	8
File D	30	2
File E	26	3

Figure 12.7 Contiguous File Allocation

Secondary Memory Management (4)



File Allocation Table

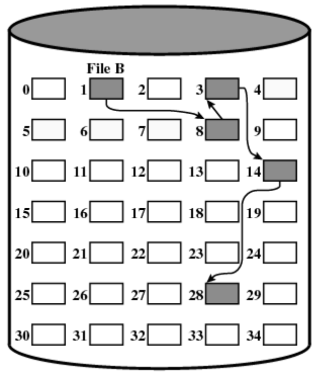
File Name	Start Block	Length
File A	0	3
File B	3	5
File C	8	8
File D	19	2
File E	16	3

Figure 12.8 Contiguous File Allocation (After Compaction)

Secondary Memory Management (5)

- Chained allocation
  - Allocation on basis of individual block
  - Each block contains a pointer to the next block in the chain
  - Only single entry in the file allocation table (Starting block and length of file)
  - No external fragmentation
  - Best for sequential files
  - No accommodation of the principle of locality

Secondary Memory Management (6)

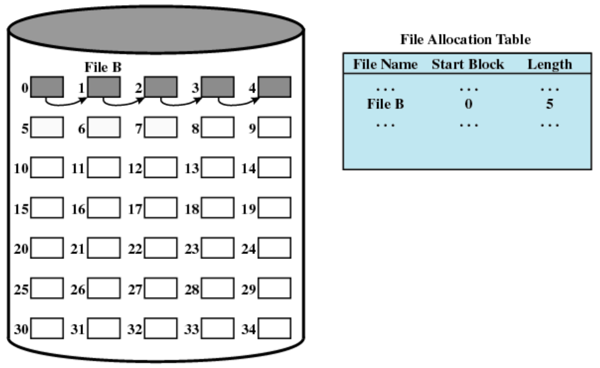


File Allocation Table

File Name	Start Block	Length
...	...	...
File B	1	5
...	...	...

Figure 12.9 Chained Allocation

Secondary Memory Management (7)



File Allocation Table

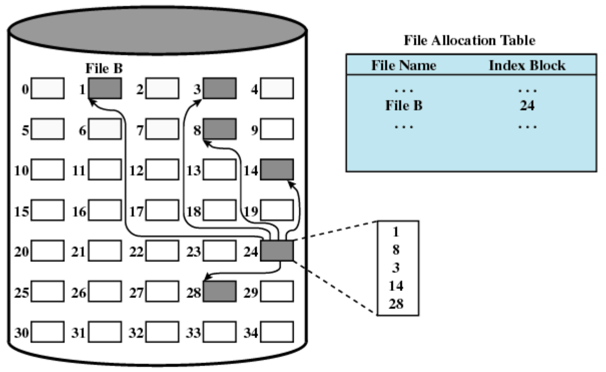
File Name	Start Block	Length
. . .	. . .	. . .
File B	0	5
. . .	. . .	. . .

Figure 12.10 Chained Allocation (After Consolidation)

Secondary Memory Management (8)

- Indexed allocation
  - File allocation table contains a separate one-level index for each file
  - The index has one entry for each portion allocated to the file
    (portions are the smallest units which can be allocated – this size can vary from single blocks to the whole file)
  - The file allocation table contains the block number of the index

Secondary Memory Management (9)

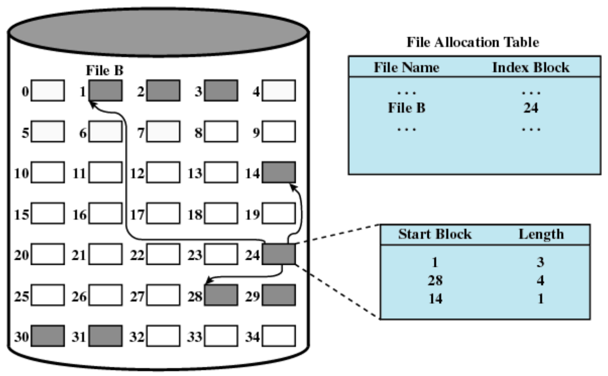


File Allocation Table

File Name	Index Block
. . .	. . .
File B	24
. . .	. . .

Figure 12.11 Indexed Allocation with Block Portions

Secondary Memory Management (10)



File Allocation Table

File Name	Index Block
. . .	. . .
File B	24
. . .	. . .

Start Block	Length
1	3
28	4
14	1

Figure 12.12 Indexed Allocation with Variable-Length Portions

Secondary Memory Management (11)

Management of free disk space

- manage the disk space which is currently unused
- in addition to FAT (file allocation table) there is a **disk allocation table** with information about free areas
- **bit table:** bit vector which each bit representing one disk block:
  0 = free block 1 = used block
  (cf. bit vectors for free main memory management)
  -> bit table may completely fit in RAM (efficient)

Secondary Memory Management (12)

- **free block list:** blocks are numbered; a list contains the numbers of all free blocks
  - list is too big to be kept in memory completely
  - remedy: treat list as a FIFO queue
    - keep the first few 1000 and the last few 1000 queue entries in memory
    - block allocation: take an entry from the queue's head
    - Block deallocation: append an entry to the queue's tail
    - from time to time synchronize heads and tails with data on disk