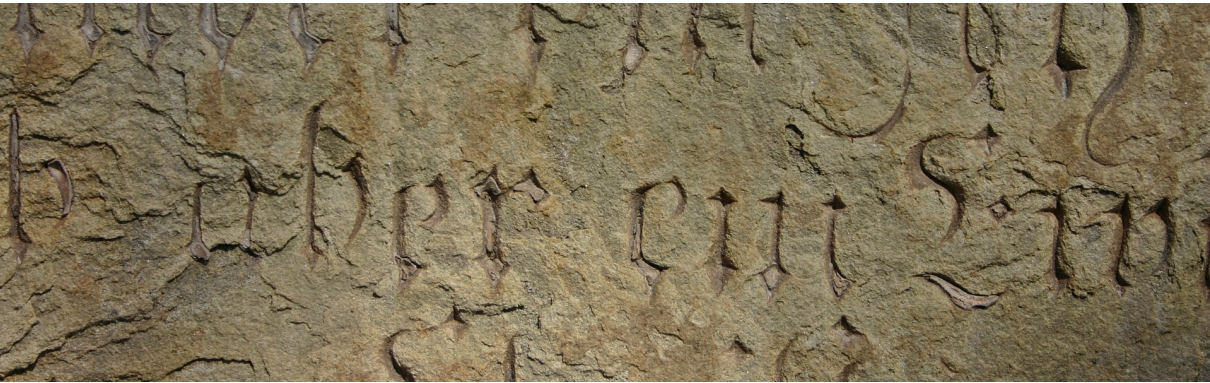

Zeichensätze und Rechteverwaltung

Tutorium 12

Betriebssysteme SS2024

Inhaltsverzeichnis

ASCII

- Wurde 1963 in den USA eingeführt
- 7-Bit Code (erstes Bit ist immer 0)
- Unterstützt keine weiteren Sprachen neben Englisch
- Viele wichtige Zeichen (z.B. deutsche Umlaute) fehlen

ISO 8859

- Verschiedene Erweiterungen des ASCII Codes
- Aktuell existieren 15 normierte Zeichensätze
- 8-Bit Code, wobei die Positionen
  - 00_{16} bis $7F_{16}$ ASCII entsprechen
  - 80_{16} bis $9F_{16}$ nicht definiert sind
  - $A0_{16}$ bis FF_{16} regionale Sonderzeichen enthalten

Unicode

- Wird vom Unicode-Konsortium verwaltet
- Die aktuelle Version 15.1 definiert 149.878 Zeichen
- Wird kontinuierlich um neue Zeichen erweitert
- Es existieren verschiedene Kodierungsformate:
  - UTF-8: Kodierung in ein bis vier 1 Byte großen Blöcken
  - UTF-16: Kodierung in ein oder zwei 2 Byte große Blöcken
  - UTF-32: Keine Kodierung notwendig

UTF-8 Kodierung

- Darstellung von Unicode mit variabler Länge
- Die Kodierung erfolgt mit Hilfe folgender Tabelle

Unicode	UTF-8
U+000000 - U+00007F	0xxxxxxx
U+000080 - U+0007FF	110xxxxx 10xxxxxx
U+000800 - U+00FFFF	1110xxxx 10xxxxxx 10xxxxxx
U+010000 - U+10FFFF	11110xxx 10xxxxxx 10xxxxxx 10xxxxxx

UTF-8 Kodierung - Beispiele

- Bestimmen Sie die UTF-8 Darstellung für den Unicode U+000240:

UTF-8 Kodierung - Beispiele

- Bestimmen Sie die UTF-8 Darstellung für den Unicode U+000240:
  Binärdarstellung von 000240_{16}:

UTF-8 Kodierung - Beispiele

- Bestimmen Sie die UTF-8 Darstellung für den Unicode U+000240:

  Binärdarstellung von 000240_{16}:
  $000240_{16} = 00000000\ 00000010\ 01000000_2$

  Einfügen in 2-Byte breite Kodierung da $000080 < 000240 < 0007FF$:

UTF-8 Kodierung - Beispiele

- Bestimmen Sie die UTF-8 Darstellung für den Unicode U+000240:

  Binärdarstellung von 000240_{16}:
  $000240_{16} = 00000000\ 00000010\ 01000000_2$

  Einfügen in 2-Byte breite Kodierung da $000080 < 000240 < 0007FF$:
  110xxxxx 10xxxxxx $\rightarrow$ 11001001 10000000

UTF-8 Kodierung - Beispiele

- Bestimmen Sie die UTF-8 Darstellung für den Unicode U+000240:

  Binärdarstellung von 000240_{16}:
  $000240_{16} = 00000000\ 00000010\ 01000000_2$

  Einfügen in 2-Byte breite Kodierung da $000080 < 000240 < 0007FF$:
  $110xxxxx\ 10xxxxxx \rightarrow 11001001\ 10000000$
- Bestimmen Sie die UTF-8 Darstellung für den Unicode U+000876:

UTF-8 Kodierung - Beispiele

- Bestimmen Sie die UTF-8 Darstellung für den Unicode U+000240:

  Binärdarstellung von 000240_{16}:
  $000240_{16} = 00000000\ 00000010\ 01000000_2$

  Einfügen in 2-Byte breite Kodierung da $000080 < 000240 < 0007FF$:
  $110xxxxx\ 10xxxxxx \rightarrow 11001001\ 10000000$
- Bestimmen Sie die UTF-8 Darstellung für den Unicode U+000876:

  Binärdarstellung von 000876_{16}:

UTF-8 Kodierung - Beispiele

- Bestimmen Sie die UTF-8 Darstellung für den Unicode U+000240:

  Binärdarstellung von 000240_{16}:
  $000240_{16} = 00000000\ 00000010\ 01000000_2$

  Einfügen in 2-Byte breite Kodierung da $000080 < 000240 < 0007FF$:
  $110xxxxx\ 10xxxxxx \rightarrow 11001001\ 10000000$

- Bestimmen Sie die UTF-8 Darstellung für den Unicode U+000876:

  Binärdarstellung von 000876_{16}:
  $000876_{16} = 00000000\ 00001000\ 01110110_2$

  Einfügen in 3-Byte breite Kodierung da $000800 < 000876 < 00FFFF$:

UTF-8 Kodierung - Beispiele

- Bestimmen Sie die UTF-8 Darstellung für den Unicode U+000240:

  Binärdarstellung von 000240_{16}:
  $000240_{16} = 00000000\ 00000010\ 01000000_2$

  Einfügen in 2-Byte breite Kodierung da $000080 < 000240 < 0007FF$:
  110xxxxx 10xxxxxx $\rightarrow$ 11001001 10000000

- Bestimmen Sie die UTF-8 Darstellung für den Unicode U+000876:

  Binärdarstellung von 000876_{16}:
  $000876_{16} = 00000000\ 00001000\ 01110110_2$

  Einfügen in 3-Byte breite Kodierung da $000800 < 000876 < 00FFFF$:
  1110xxxx 10xxxxxx 10xxxxxx $\rightarrow$ 11100000 10100001 10110110

Berechtigungen

- Ein Betriebssystem verwaltet Zugriffsberechtigungen für Ressourcen
- Einschränkung der Berechtigungen:
  - Nicht jeder Nutzer sollte alles tun dürfen
  - Nicht jeder Prozess sollte alles tun dürfen
  - Stattdessen können Rechte vergeben, geändert oder entzogen werden
- Zugriffsmöglichkeiten
  - *datenzentriert* : lesender und schreibender Zugriff
  - *operationszentriert* : ausführen von beliebigen Operationen

Rechteverwaltung

- Bei Betriebssystemen gibt es drei Akteuere bei der Rechteverwaltung
  - *Subjekte* - z.B. Benutzer
  - *Objekte* - z.B. Dateien, Verzeichnisse, ...
  - *Operationen* - z.B. Schreibzugriffe, Lesezugriffe, ...

Rechteverwaltung

- Bei Betriebssystemen gibt es drei Akteuere bei der Rechteverwaltung
  - *Subjekte* - z.B. Benutzer
  - *Objekte* - z.B. Dateien, Verzeichnisse, ...
  - *Operationen* - z.B. Schreibzugriffe, Lesezugriffe, ...
- Rechte können für jeden der Akteure definiert werden
  - Für jedes *Subjekt* kann festgelegt werden, welche Operationen auf welchen Objekten ausgeführt werden können.
    - Implementierung typischerweise mittels *Capabilities*

Rechteverwaltung

- Bei Betriebssystemen gibt es drei Akteuere bei der Rechteverwaltung
  - *Subjekte* - z.B. Benutzer
  - *Objekte* - z.B. Dateien, Verzeichnisse, ...
  - *Operationen* - z.B. Schreibzugriffe, Lesezugriffe, ...
- Rechte können für jeden der Akteure definiert werden
  - Für jedes *Subjekt* kann festgelegt werden, welche Operationen auf welchen Objekten ausgeführt werden können.
    - Implementierung typischerweise mittels *Capabilities*
  - Für jedes *Objekt* kann festgelegt werden, welche Operationen auf welchen Objekten ausgeführt werden können.
    - Implementierung typischerweise mittels *Access Control Lists*

Rechteverwaltung

- Bei Betriebssystemen gibt es drei Akteuere bei der Rechteverwaltung
  - *Subjekte* - z.B. Benutzer
  - *Objekte* - z.B. Dateien, Verzeichnisse, ...
  - *Operationen* - z.B. Schreibzugriffe, Lesezugriffe, ...
- Rechte können für jeden der Akteure definiert werden
  - Für jedes *Subjekt* kann festgelegt werden, welche Operationen auf welchen Objekten ausgeführt werden können.
    - Implementierung typischerweise mittels *Capabilities*
  - Für jedes *Objekt* kann festgelegt werden, welche Operationen auf welchen Objekten ausgeführt werden können.
    - Implementierung typischerweise mittels *Access Control Lists*
  - Für jede *Operation* kann festgelegt werden, welche Subjekte diese auf welchen Objekten ausführen können.
    - Sehr selten implementiert

Benutzerverwaltung

- Benutzer werden durch einen Benutzernamen identifiziert
- Die Identität muss durch ein Authentisierungsverfahren nachgewiesen werden
- Die Anmeldedaten werden in einer geschützten Datenbank gespeichert
  - Die Datenbank muss vor Manipulation und unerwünschtem Auslesen geschützt sein
  - Passwortänderungen müssen trotzdem möglich sein
  - Die Datenbank muss für den Anmeldeprozess lesbar sein

Benutzerverwaltung

- Benutzer werden durch einen Benutzernamen identifiziert
- Die Identität muss durch ein Authentisierungsverfahren nachgewiesen werden
- Die Anmeldedaten werden in einer geschützten Datenbank gespeichert
  - Die Datenbank muss vor Manipulation und unerwünschtem Auslesen geschützt sein
  - Passwortänderungen müssen trotzdem möglich sein
  - Die Datenbank muss für den Anmeldeprozess lesbar sein
- Beispiel: UNIX/Linux Betriebssysteme
  - Speicherung von Nutzer Informationen in der Datei `/etc/passwd`
    $\rightarrow$ öffentlich lesbar
  - Speicherung von Passwort Hashwerten in der Datei `/etc/shadow`
    $\rightarrow$ nur vom Benutzer `root` oder der Benutzern in der Gruppe `sudo` lesbar

UNIX/Linux Rechteverwaltung

- Berechtigungen sind an Inodes geknüpft
- Inodes haben einen Eigentümer der durch eine User-ID identifiziert wird
- Inodes gehören genau einer Gruppe an die durch die Group-ID identifiziert wird

UNIX/Linux Rechteverwaltung

- Berechtigungen sind an Inodes geknüpft
- Inodes haben einen Eigentümer der durch eine User-ID identifiziert wird
- Inodes gehören genau einer Gruppe an die durch die Group-ID identifiziert wird
- Pro Inode werden jeweils drei Berechtigungen für Eigentümer `User`, die Gruppe `Group`, und alle anderen vergeben `Others`
- Bei Dateien gibt es hierbei das Leserecht (r), Schreibrecht (w), und Ausführungsrecht (x)
- Bei Verzeichnissen gibt es ebenfalls das Leserecht (r) und Schreibrecht (w), sowie das Durchgangsrecht (x)
- Die Berechtigungen können nur vom Eigentümer, vom Benutzer `root` oder Nutzern der Gruppe `sudo` geändert werden

Zusätzliche Berechtigungen an Inodes

- *User S-Bit*
  - Bei *ausführbaren Dateien* findet die Ausführung mit den Berechtigungen des Dateieigentümers statt und nicht mit den Berechtigungen des Nutzers

Zusätzliche Berechtigungen an Inodes

- *User S-Bit*
  - Bei *ausführbaren Dateien* findet die Ausführung mit den Berechtigungen des Dateieigentümers statt und nicht mit den Berechtigungen des Nutzers
- *Group S-Bit*
  - Bei *ausführbaren Dateien* findet die Ausführung mit den Gruppenberechtigungen der Datei statt und nicht mit den Berechtigungen der Gruppe des Nutzers
  - Bei *Verzeichnissen* erhalten neuen Einträge die selbe Gruppenzuordnung wie das Verzeichnis

Zusätzliche Berechtigungen an Inodes

- *User S-Bit*
  - Bei *ausführbaren Dateien* findet die Ausführung mit den Berechtigungen des Dateieigentümers statt und nicht mit den Berechtigungen des Nutzers
- *Group S-Bit*
  - Bei *ausführbaren Dateien* findet die Ausführung mit den Gruppenberechtigungen der Datei statt und nicht mit den Berechtigungen der Gruppe des Nutzers
  - Bei *Verzeichnissen* erhalten neuen Einträge die selbe Gruppenzuordnung wie das Verzeichnis
- *Sticky Bit*
  - Bei *Verzeichnissen* kann ein Nutzer trotz Schreibrecht nur eigene Einträge löschen
  - Bei *ausführbaren Dateien* werden Speicherseiten nicht ausgelagert (**bei modernen Systemen nicht mehr unterstützt**)

Access Control Lists

- Mit Hilfe von POSIX-ACLs können zusätzliche positive Rechte für Benutzer und Gruppen für jeden Inode gespeichert werden
- Eine vorhandene ACL ist an einem `x` hinter den klassischen Rechten erkennbar
- Beispiel:

```
$ ls -l
drw-rw-r--+  alice  users  427 Jul 3 14:37 file.txt
```

Access Control Lists

- Mit Hilfe von POSIX-ACLs können zusätzliche positive Rechte für Benutzer und Gruppen für jeden Inode gespeichert werden
- Eine vorhandene ACL ist an einem `x` hinter den klassischen Rechten erkennbar
- Beispiel:

```
$ ls -l
drw-rw-r--+  alice  users  427 Jul 3 14:37 file.txt
```

- Mit dem Befehl `setfacl` können Einträge in der ACL gesetzt/gelöscht werden
- Mit dem Befehl `getfacl` können Einträge in der ACL angezeigt werden
- Mit Hilfe von Default-ACLs können ACLs an angelegte Dateien und Verzeichnisse vererbt werden

Berechtigungen unter Windows

- Bei modernen Windows Systemen werden *Security Descriptors* für die Speicherung von ACLs verwendet, welche ein NTFS Dateisystem erforden
- Bei FAT Dateisystemen ist lediglich ein einfacher Schreibschutz implementiert
- ACLs sind hier ebenfalls für beliebige Benutzer und Gruppen definierbar

Berechtigungen unter Windows

- Bei modernen Windows Systemen werden *Security Descriptors* für die Speicherung von ACLs verwendet, welche ein NTFS Dateisystem erforden
- Bei FAT Dateisystemen ist lediglich ein einfacher Schreibschutz implementiert
- ACLs sind hier ebenfalls für beliebige Benutzer und Gruppen definierbar
- Erlauben feingranularere Berechtigungsstufen im Vergleich zu UNIX/Linux:
  - `No Access` - Kein Zugriff
  - `List` - Auflistung von Dateien in Verzeichnissen
  - `Read` - Lesen von Dateien, schließt `List` mit ein
  - `Add` - Anlegen von Dateien, schließt `List` mit ein
  - `Read & Add` - Kombination aus `Read` und `Add`
  - `Change` - Änderung/Löschen von Dateien, schließt `Read & Add` mit ein
  - `Full` - Änderung des Eigentümers/Zugriffsrechten, schließt `Change` mit ein