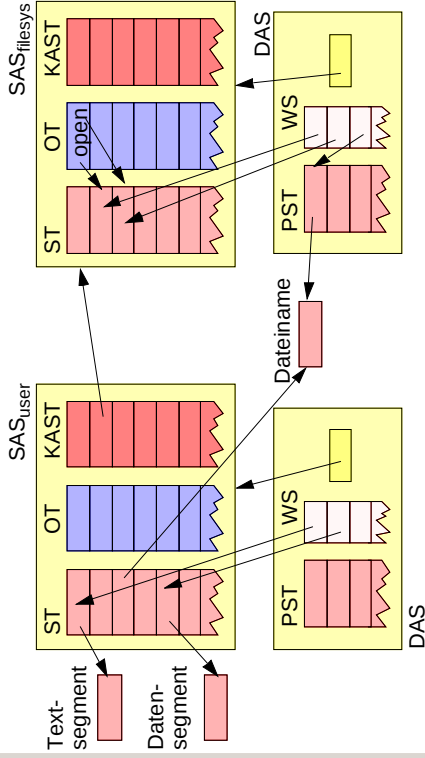


## 2 Beispielaufruf (2)

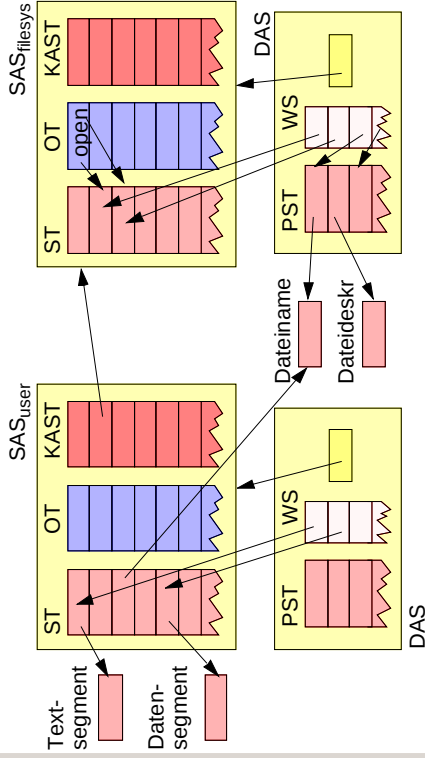
- SAS des Benutzers ruft Operation „open“ des SAS des Dateisystems auf



- ◆ für den Aufruf von „open“ wird ein neues DAS erzeugt

## 2 Beispielaufruf (3)

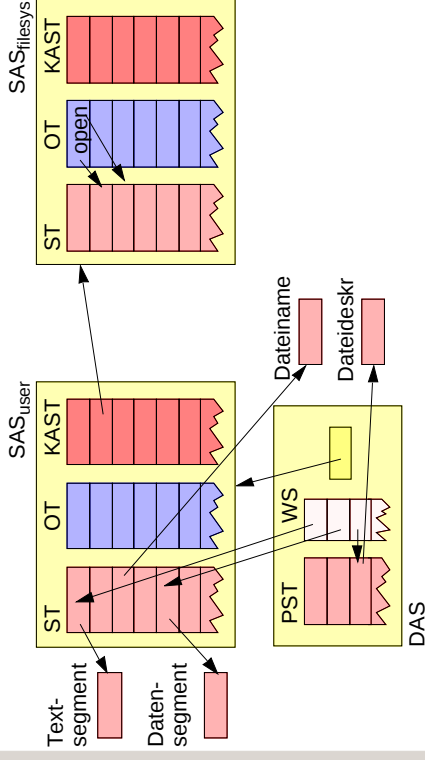
- SAS des Benutzers ruft Operation „open“ des SAS des Dateisystems auf



- ◆ „open“ erzeugt neues Segment für den Dateideskriptor

## 2 Beispielaufruf (4)

- SAS des Benutzers ruft Operation „open“ des SAS des Dateisystems auf

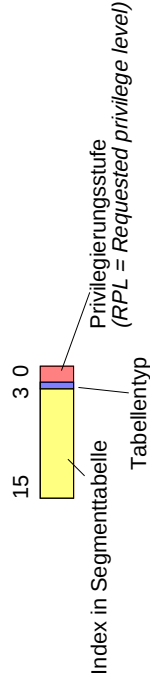


- ◆ Ergebnissegment wird an den Aufrufer zurückgegeben

## 3 Beispiel: Pentium

- Privilegierungsstufen
- ◆ Stufe 0: höchste Privilegien (privilegierte Befehle, etc.): BS Kern
- ◆ Stufe 1: BS Treiber
- ◆ Stufe 2: BS Erweiterungen
- ◆ Stufe 3: Benutzerprogramme

- Segmentselektoren enthalten Privilegierungsstufe

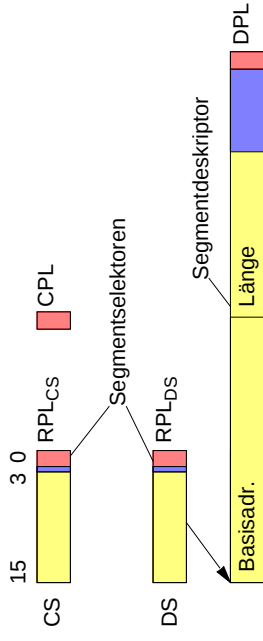


- Codesegmentselektor (CS) bestimmt aktuelle Privilegierungsstufe

- ◆ CPL = *Current privilege level*

### 3 Beispiel: Pentium (2)

- Datenzugriff (z.B. auf Datensegment DS)



- ◆ DPL = *Descriptor privilege level*
- ◆ CPL ist normalerweise gleich RPL<sub>CS</sub>
- ◆ Zugriff wird erlaubt, wenn:  $DPL \geq \max(CPL, RPL_{DS})$
- ◆ ansonsten wird Unterbrechung ausgelöst (Schutzverletzung)

SP I

Systemprogrammierung I

© Franz J. Hauck, Universität Erlangen-Nürnberg, IMMD IV, 1998

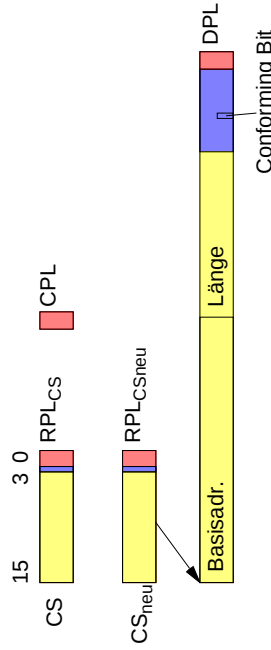
I-Security.doc 1998-02-03 16:35

Reproduktion oder andere Verwendung dieser Vorlesung ist ohne schriftliche Genehmigung der Universität Erlangen-Nürnberg strafbar.

L36

### 3 Beispiel: Pentium (3)

- Sprünge in andere Codesegmente (*Far Call*)



- ◆ Sprung wird erlaubt, falls:  $DPL = CPL$  oder Conforming Bit gesetzt und  $DPL \leq CPL$
- ◆ Im Falle von  $DPL \leq CPL$  wird jedoch CPL nicht geändert (Codesegment hat höheres Privileg, CPL bleibt aber unverändert)

SP I

Systemprogrammierung I

© Franz J. Hauck, Universität Erlangen-Nürnberg, IMMD IV, 1998

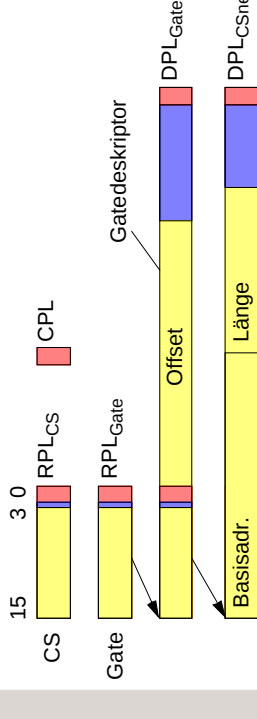
I-Security.doc 1998-02-03 16:35

Reproduktion oder andere Verwendung dieser Vorlesung ist ohne schriftliche Genehmigung der Universität Erlangen-Nürnberg strafbar.

L37

### 3 Beispiel: Pentium (4)

- Kontrolltransfer mit einem Gate



- ◆ Gatedeskriptoren stehen wie Segmentdeskriptoren in der Segmenttabelle
- ◆ Gatedeskriptor enthält Segmentselektor für das Codesegment und einen Offset zu diesem Segment, an dem der Einsprungpunkt liegt
- ◆ Kontrolltransfer (*CALL* Aufruf) wird erlaubt, falls:  $DPL_{Gate} \geq \max(CPL, RPL_{Gate})$  und  $CPL \geq DPL_{CSneu}$

SP I

Systemprogrammierung I

© Franz J. Hauck, Universität Erlangen-Nürnberg, IMMD IV, 1998

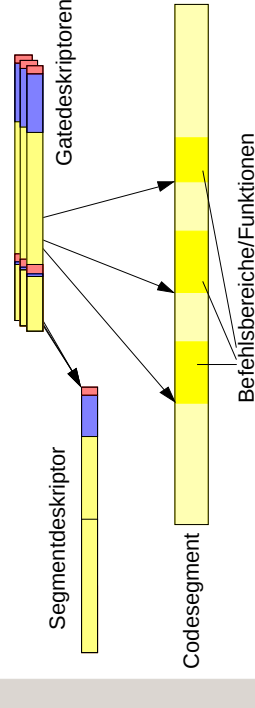
I-Security.doc 1998-02-03 16:35

Reproduktion oder andere Verwendung dieser Vorlesung ist ohne schriftliche Genehmigung der Universität Erlangen-Nürnberg strafbar.

L38

### 3 Beispiel: Pentium (5)

- Gates erlauben den kontrollierten Sprung in ein privilegierten Befehlsbereich



- ◆ es existiert auch der entsprechende Rücksprung
- ◆ für jede Privilegiierungsstufe gibt es einen eigenen Stack; dieser wird mit umgeschaltet
- ◆ Parameter werden automatisch auf den neuen Stack kopiert (Anzahl wird im Gatedeskriptor vermerkt)

SP I

Systemprogrammierung I

© Franz J. Hauck, Universität Erlangen-Nürnberg, IMMD IV, 1998

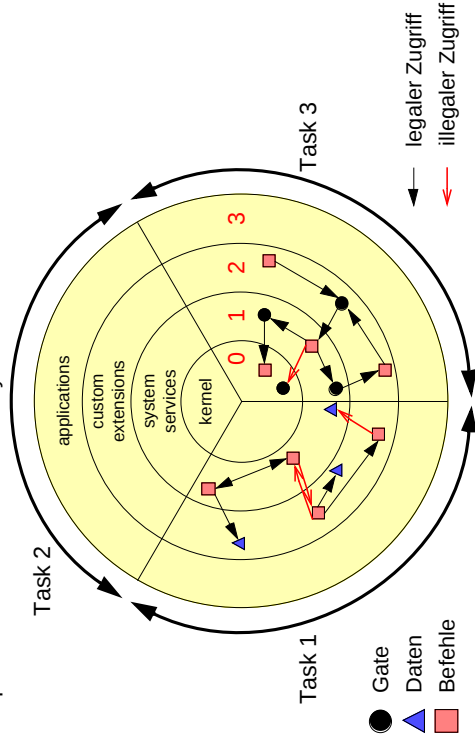
I-Security.doc 1998-02-03 16:35

Reproduktion oder andere Verwendung dieser Vorlesung ist ohne schriftliche Genehmigung der Universität Erlangen-Nürnberg strafbar.

L39

### 3 Beispiel: Pentium (6)

- Beispiel eines realisierten Schutzsystems

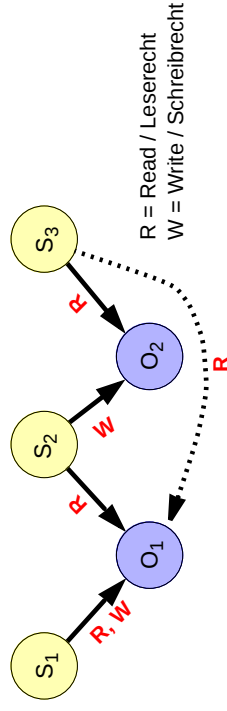


SP I

L40

### I.5 Capability-basierte Systeme

- Ein Benutzer (Subjekt) erhält eine Referenz auf ein Objekt
- ◆ die Referenz enthält alle Rechte, die das Subjekt an dem Objekt besitzt
- ◆ bei der Nutzung der Capability (Zugriff auf das Objekt) werden die Rechte überprüft



Subjekte und Objekte; Weitergabe einer Capability (O<sub>1</sub> von S<sub>2</sub> nach S<sub>3</sub>)

SP I

L41

### I.5 Capability-basierte Systeme (2)

#### ★ Vorteile

- ◆ keine Speicherung von Rechten beim Objekt oder Subjekt nötig; Capability enthält Zugriffsrechte
  - ◆ leichte Vergabe von individuellen Rechten
  - ◆ einfache Weitergabe von Zugriffsrechten möglich
- #### ▲ Nachteile
- ◆ Weitergabe nicht kontrollierbar
  - ◆ Rückruf von Zugriffsrechten nicht möglich
  - ◆ Capability muß vor Fälschung und Verfälschung geschützt werden (z.B. durch kryptographische Mittel oder durch Speicherverwaltung)

SP I

L42

### 1 Beispiel: Hydra

- Hydra ist ein Capability-basiertes Betriebssystem
  - ◆ entwickelt Mitte der Siebziger Jahre an der Carnegie-Mellon University
  - ◆ lief auf einem speziellen Multiprozessor namens C.mmp
  - ◆ Capability-Mechanismen sind integraler Bestandteil des Betriebssystems
- Objekte in Hydra werden durch Capabilities angesprochen und geschützt
- ◆ Objekte haben einen Typ (z.B. Prozeduren, Prozesse, Semaphoren, Datei etc.)
  - ◆ benutzerdefinierte Typen sind möglich
  - ◆ generische Operationen für alle Typen implementiert durch das Betriebssystem
  - ◆ Objekte besitzen eine Liste von Capabilities auf andere Objekte (genannt C-List)
  - ◆ Objekte besitzen einen Datenbereich (implementiert durch geschütztes Segment)

SP I

L43

## 1 Beispiel: Hydra (2)

- Prozesse (Subjekte)
  - ◆ Prozesse besitzen einen aktuellen Kontext, den LNS (*Local name space*)
  - ◆ LNS ist ein Objekt
  - ◆ LNS besitzt einen Aktivitätsträger (Thread)
  - ◆ LNS kann nur auf Objekte zugreifen, die in seiner C-List stehen (mehrstufige Zugriffe, z.B. auf die C-List eines Objekts, dessen Capabilities in der C-List des LNS steht, sind möglich)
- Capabilities
  - ◆ Capabilities haben einen Typ (z.B. Prozedur, LNS etc.)
  - ◆ Prozesse können nur über Systemaufrufe ihre Capabilities bzw. ihre C-List bearbeiten
  - ◆ Capabilities können nicht gefälscht oder verfälscht werden
  - ◆ Betriebssystem kann sicheres Schutzkonzept basierend auf Capabilities implementieren

SP I

Systemprogrammierung I

© Franz J. Hauck, Universität Erlangen-Nürnberg, IMMD IV, 1998

I-Security.doc: 1998-02-03 16:35

I.44

Reproduzieren oder Verändern dieser Unterlagen stellt ein Verstoß gegen die Urheberrechte dar.

## 1 Beispiel: Hydra (3)

- Datenbereich
- ◆ Operationen:
  - *GetData*: kopiere Abschnitt aus dem Datenbereich eines Objekts in den Datenbereich des LNS
  - *Putdata*: kopiere Abschnitt aus dem Datenbereich des LNS in den Datenbereich eines Objekts
  - *Adddata*: füge Daten zu dem Datenbereich eines Objekts hinzu
- ◆ Rechte:
  - **GETRTS**: erlaubt den Aufruf von *GetData*
  - **PUTRTS**: erlaubt den Aufruf von *Putdata*
  - **ADDRTS**: erlaubt den Aufruf von *Adddata*
- ◆ Rechte müssen in der Capability zum Objekt gesetzt sein

SP I

Systemprogrammierung I

© Franz J. Hauck, Universität Erlangen-Nürnberg, IMMD IV, 1998

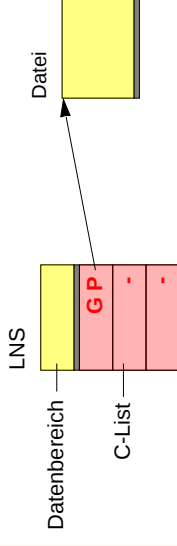
I-Security.doc: 1998-02-03 16:35

I.45

Reproduzieren oder Verändern dieser Unterlagen stellt ein Verstoß gegen die Urheberrechte dar.

## 1 Beispiel: Hydra (4)

- Implementierung von Dateien:
  - ◆ *GetData* erlaubt das Lesen von Daten
  - ◆ *Putdata* erlaubt das Schreiben von Daten
  - ◆ *Adddata* erlaubt das Anhängen von Daten
- Entsprechende Rechte können pro Capability gesetzt werden



SP I

Systemprogrammierung I

© Franz J. Hauck, Universität Erlangen-Nürnberg, IMMD IV, 1998

I-Security.doc: 1998-02-03 16:35

I.46

Reproduzieren oder Verändern dieser Unterlagen stellt ein Verstoß gegen die Urheberrechte dar.

## 1 Beispiel: Hydra (5)

- C-List
- ◆ Operationen:
  - *Load*: kopieren einer Capability aus der C-List eines Objekts in die C-List des LNS
  - *Store*: kopieren einer Capability aus der C-List des LNS in die C-List eines Objekts (dabei können Rechte maskiert werden)
  - *Append*: anfügen einer Capability in die C-List eines Objekts
  - *Delete*: löschen einer Capability aus der C-List eines Objekts
- ◆ Rechte:
  - **LOADRTS**: erlaubt Aufruf von *Load*
  - **STORRTS**: erlaubt Aufruf von *Store*
  - **APPRTS**: erlaubt Aufruf von *Append*
  - **KILLRTS**: erlaubt Aufruf von *Delete*

SP I

Systemprogrammierung I

© Franz J. Hauck, Universität Erlangen-Nürnberg, IMMD IV, 1998

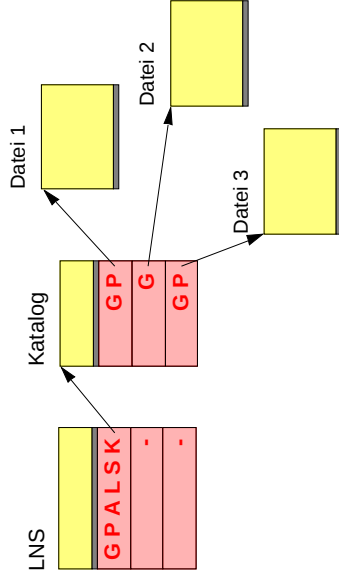
I-Security.doc: 1998-02-03 16:35

I.47

Reproduzieren oder Verändern dieser Unterlagen stellt ein Verstoß gegen die Urheberrechte dar.

## 1 Beispiel: Hydra (6)

- Implementierung von Katalogen:
  - ◆ Load erlaubt das Auflösen von Namen (Aufrufer bekommt die Capability)
  - ◆ Store und Append erlauben das Hinzufügen von Dateien zum Katalog
  - ◆ Delete erlaubt das Austragen von Dateien aus dem Katalog



SP I

Systemprogrammierung I

© Franz J. Hauck, Universität Erlangen-Nürnberg, IMMD IV, 1998

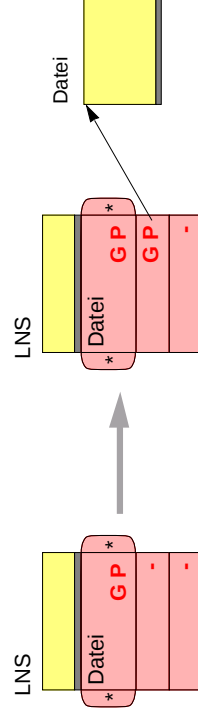
I-Security.doc 1998-02-03 16:35

L.48

Reproduzieren oder Verwenden dieser Vorlesung ist ohne schriftliche Genehmigung der Universität Erlangen-Nürnberg strafbar.

## 1 Beispiel: Hydra (7)

- Objekterzeugung über Erzeugungsschablonen (*Creation Templates*)
  - ◆ Erzeugungsschablone enthält den Typ des neu zu erzeugenden Objektes und eine Rechtemaske
  - ◆ nur die in der Maske angeschalteten Rechte werden dem Aufrufer in einer neuen Capability gegeben



SP I

Systemprogrammierung I

© Franz J. Hauck, Universität Erlangen-Nürnberg, IMMD IV, 1998

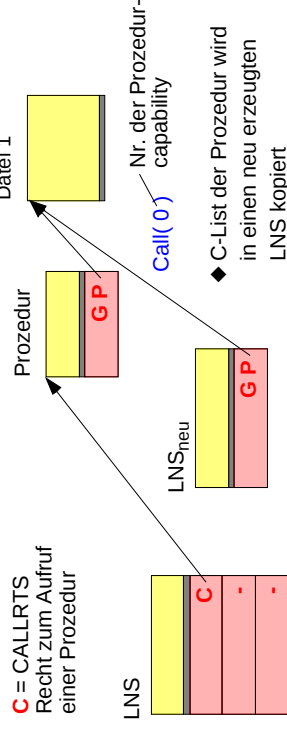
I-Security.doc 1998-02-03 16:35

L.49

Reproduzieren oder Verwenden dieser Vorlesung ist ohne schriftliche Genehmigung der Universität Erlangen-Nürnberg strafbar.

## 2 Hydra Prozeduraufruf

- Prozedur ist ein Objekt, aus dem beim Aufruf ein LNS des laufenden Prozesses erzeugt wird
  - ◆ neuer LNS wird aktueller Kontext (alte LNS stehen auf einem Stack; sie werden wieder aktiviert, wenn Prozedur zu Ende)
- Aufruf einer Prozedur



SP I

Systemprogrammierung I

© Franz J. Hauck, Universität Erlangen-Nürnberg, IMMD IV, 1998

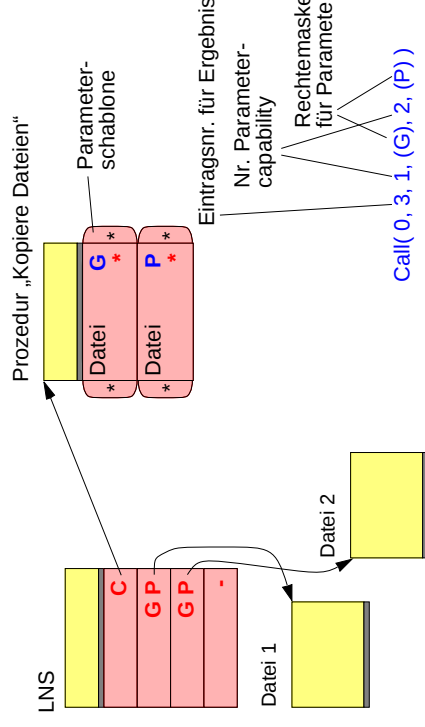
I-Security.doc 1998-02-03 16:35

L.50

Reproduzieren oder Verwenden dieser Vorlesung ist ohne schriftliche Genehmigung der Universität Erlangen-Nürnberg strafbar.

## 2 Hydra Prozeduraufruf (2)

- Übergabe von Parametern
  - ◆ Beispiel: Prozedur zum Kopieren von Dateiinhalten



SP I

Systemprogrammierung I

© Franz J. Hauck, Universität Erlangen-Nürnberg, IMMD IV, 1998

I-Security.doc 1998-02-03 16:35

L.51

Reproduzieren oder Verwenden dieser Vorlesung ist ohne schriftliche Genehmigung der Universität Erlangen-Nürnberg strafbar.