

---

# Rechnernetze I

SoSe 2019

Roland Wismüller  
Universität Siegen  
roland.wismueller@uni-siegen.de  
Tel.: 0271/740-4050, Büro: H-B 8404

Stand: 10. März 2020

---

# Rechnernetze I

SoSe 2019

## 9 Anwendungsprotokolle



## Inhalt

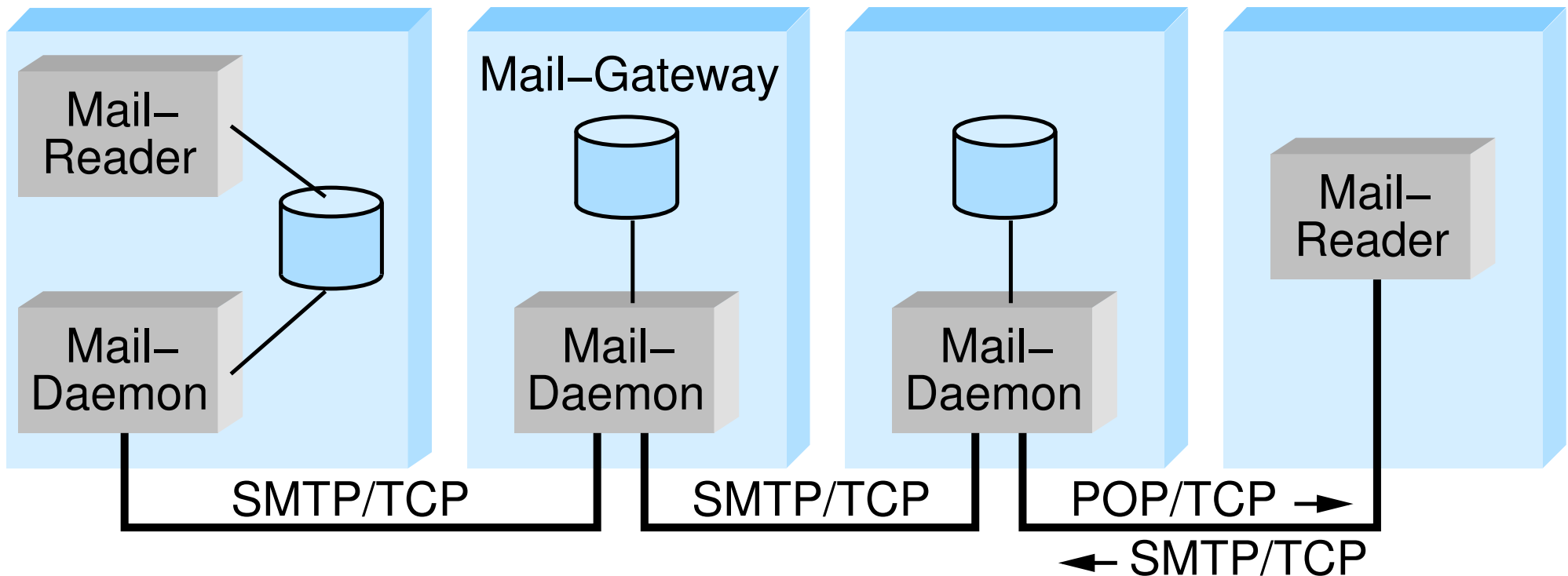
- ➔ SMTP (*Simple Mail Transport Protocol*)
- ➔ HTTP (*Hypertext Transport Protocol*)
- ➔ DNS (*Domain Name Service*)
  
- ➔ Peterson, Kap. 9.1, 9.2.1 – 9.2.2
- ➔ CCNA, Kap. 10



### Protokolle für Emails

- ➔ RFC 822 und MIME: Format einer Email
  - ➔ Header (ASCII)
  - ➔ Rumpf (ASCII)
    - ➔ binäre Daten (z.B. Bilder) werden durch MIME in ASCII codiert
  
- ➔ **SMTP** (*Simple Mail Transport Protocol*)
  - ➔ regelt Weitergabe der Emails zwischen Rechnern
  - ➔ textbasiertes Protokoll
    - ➔ vermeidet Darstellungsprobleme
    - ➔ erleichtert Test und Debugging
  
- ➔ **POP** (*Post Office Prot.*) / **IMAP** (*Internet Message Access Prot.*)
  - ➔ Herunterladen der Emails von einem entfernten Server

## Transport von Emails über Mail-Gateways



- ➔ Eigene Speichervermittlung, da
  - ➔ Zielrechner beim Sender nicht immer bekannt
  - ➔ Zielrechner nicht immer erreichbar



### Beispiel zum Aufbau einer Email

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="-----417CAWVF...
From: Alice Smith <Alice@cicso.com>
To: Bob@cs.Princeton.edu
Subject: proposed material
Date: Mon, 07 Sep 1988 19:45:19 -0400
```

Header

```
-----417CAWVFNCVKRZKAZCFHKWFHQ
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit
```

```
Bob, here is the jpeg image I promised.
Alice
```

Rumpf

```
-----417CAWVFNCVKRZKAZCFHKWFHQ
Content-Type: image/jpeg; name="beach.jpg"
Content-Transfer-Encoding: base64
```

*... ASCII-codiertes JPEG-Bild ...*



### Beispiel eines SMTP Dialogs

(Client, *Server*)

```
HELO cs.princeton.edu
250 Hello daemon@mail.cs.princeton.edu [128.12.169.24]
MAIL FROM: <Alice@cs.princeton.edu>
250 OK
RCPT TO: <Bob@cisco.com>
250 OK
RCPT TO: <Tom@cisco.com>
550 No such user here
DATA
354 Start mail input; end with <CRLF>.<CRLF>
Blah blah blah ...
... etc. etc. etc.
.
250 OK
QUIT
221 Closing connection
```

} siehe vorige Folie

- ➔ HTTP ist wie SMTP textbasiert
- ➔ Kommandos (u.a.):

GET <URL>	Anfrage nach einem Dokument
HEAD <URL>	Anfrage nach Metainformation
POST <URL>	Senden von Information an den Server
PUT <URL>	Speichern eines Dokuments
DELETE <URL>	Löschen eines Dokuments

- ➔ Beispiel-Anfrage:

```
GET index.html HTTP/1.1  
Host: www.cs.princeton.edu
```



### ➔ Ergebniscodes:

1xx	Informativ	Anfrage erhalten
2xx	Erfolg	Anfrage empfangen u. akzeptiert
3xx	Weiterleitung	Weitere Aktionen notwendig
4xx	Client-Fehler	Syntaxfehler, nicht erfüllbar, ...
5xx	Server-Fehler	Anfrage OK, Problem im Server

### ➔ Beispiel-Antwort:

HTTP/1.1 301 Moved Permanently

Location: <http://www.cs.princeton.edu/cs/index.html>



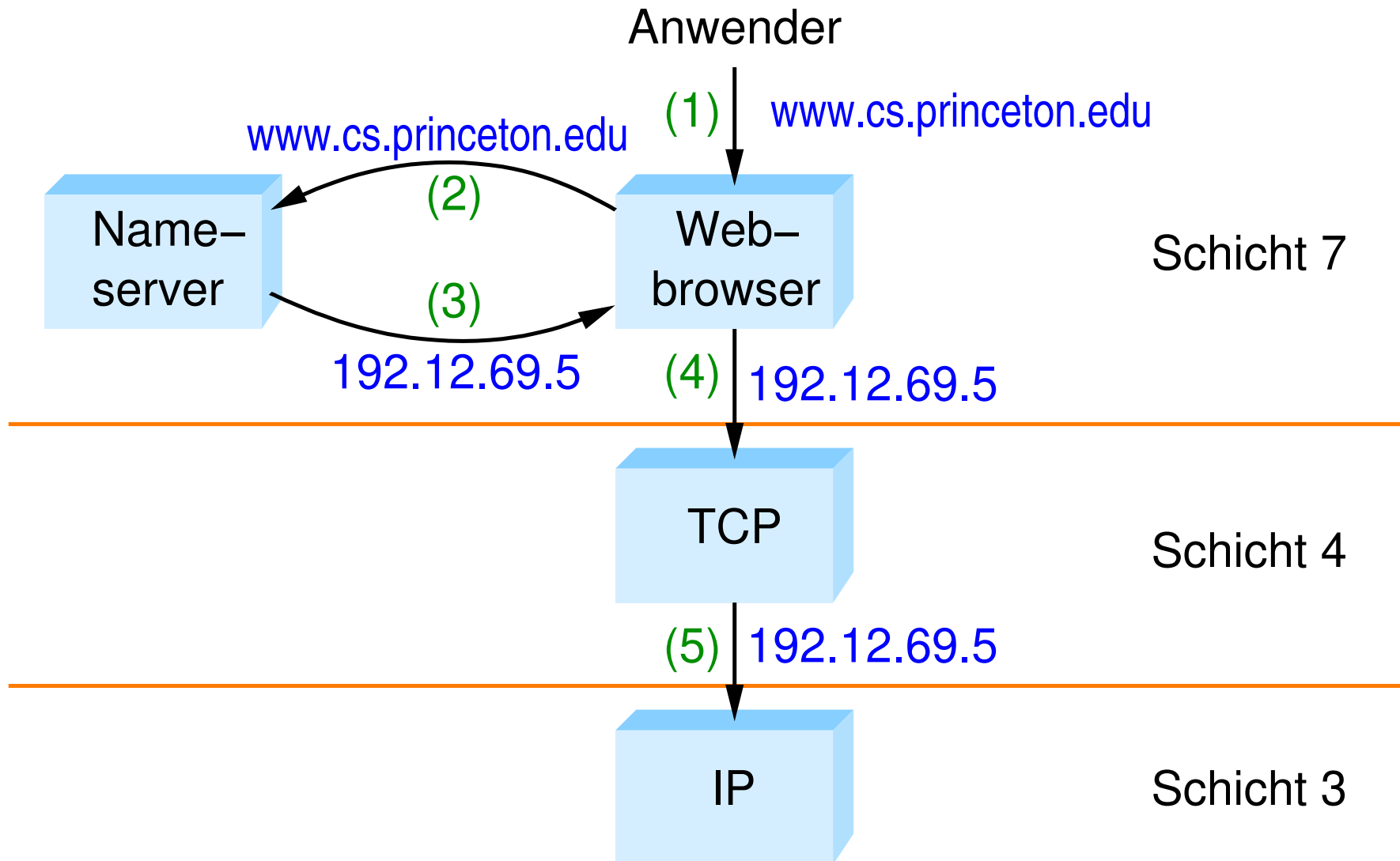
- ➔ In HTTP 1.0:
  - ➔ neue TCP-Verbindung für jedes Seiten-Element
    - ➔ z.B. Frame, eingebettetes Bild, ...
- ➔ Ab HTTP 1.1: persistente Verbindungen möglich
  - ➔ mehrere Anfragen / Antworten über dieselbe TCP-Verbindung
  - ➔ deutlich effizienter (Verbindungsaufbau)
  - ➔ Problem: Server muß viele Verbindungen offenhalten
    - ➔ wie lange? ⇒ Timeout-Mechanismus notwendig
- ➔ Einsatz von Caches (Proxy-Server)
  - ➔ nahe beim Client, speichert häufig besuchte Seiten



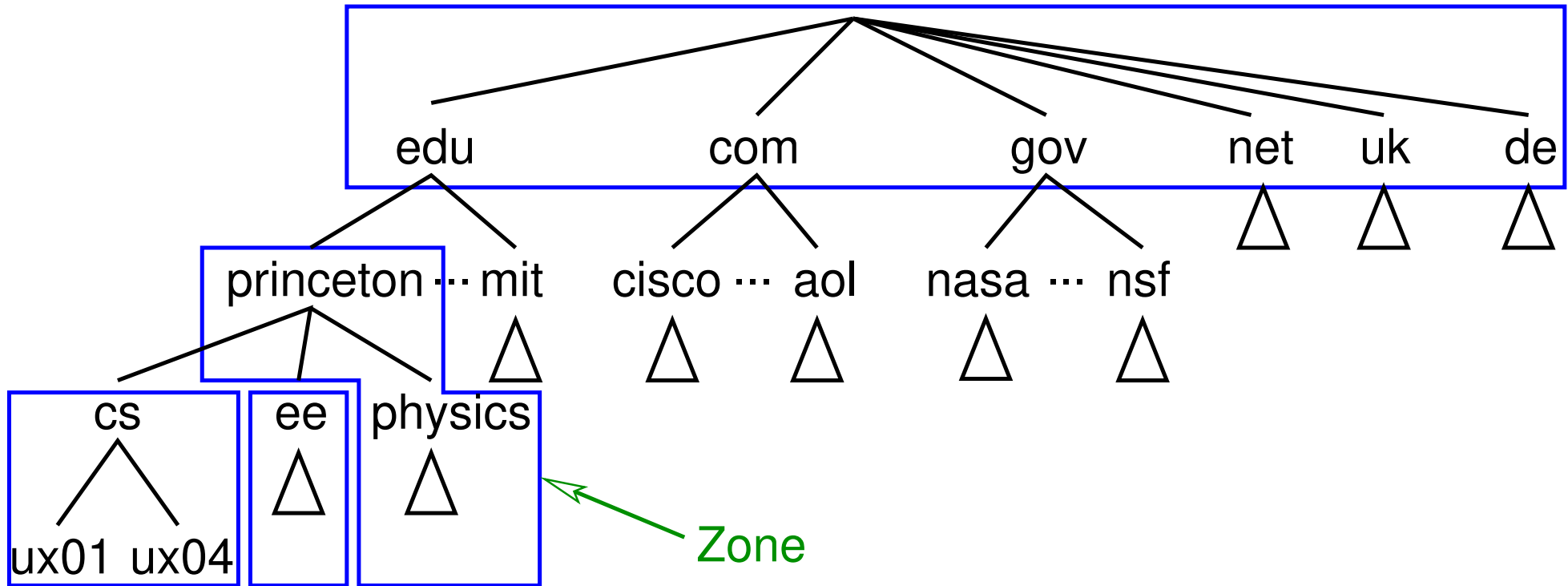
- ➔ Das Internet-Protokoll (IP) arbeitet mit numerischen Adressen
  - ➔ einheitliches Format (Länge)
  - ➔ identifizieren das Netz des Hosts (logische Adressen)
    - ➔ wichtig für das Routing von Paketen
  - ➔ nicht benutzerfreundlich
- ➔ Benutzer verwenden (Host-/Rechner-)Namen:
  - ➔ benutzerfreundlich
  - ➔ unterschiedliche Länge
  - ➔ nicht für das Routing nutzbar
- ➔ Daher: Umsetzung von Namen auf IP-Adressen:  
**DNS (*Domain Name Service*)**



## Ablauf der Umsetzung von Namen in Adressen



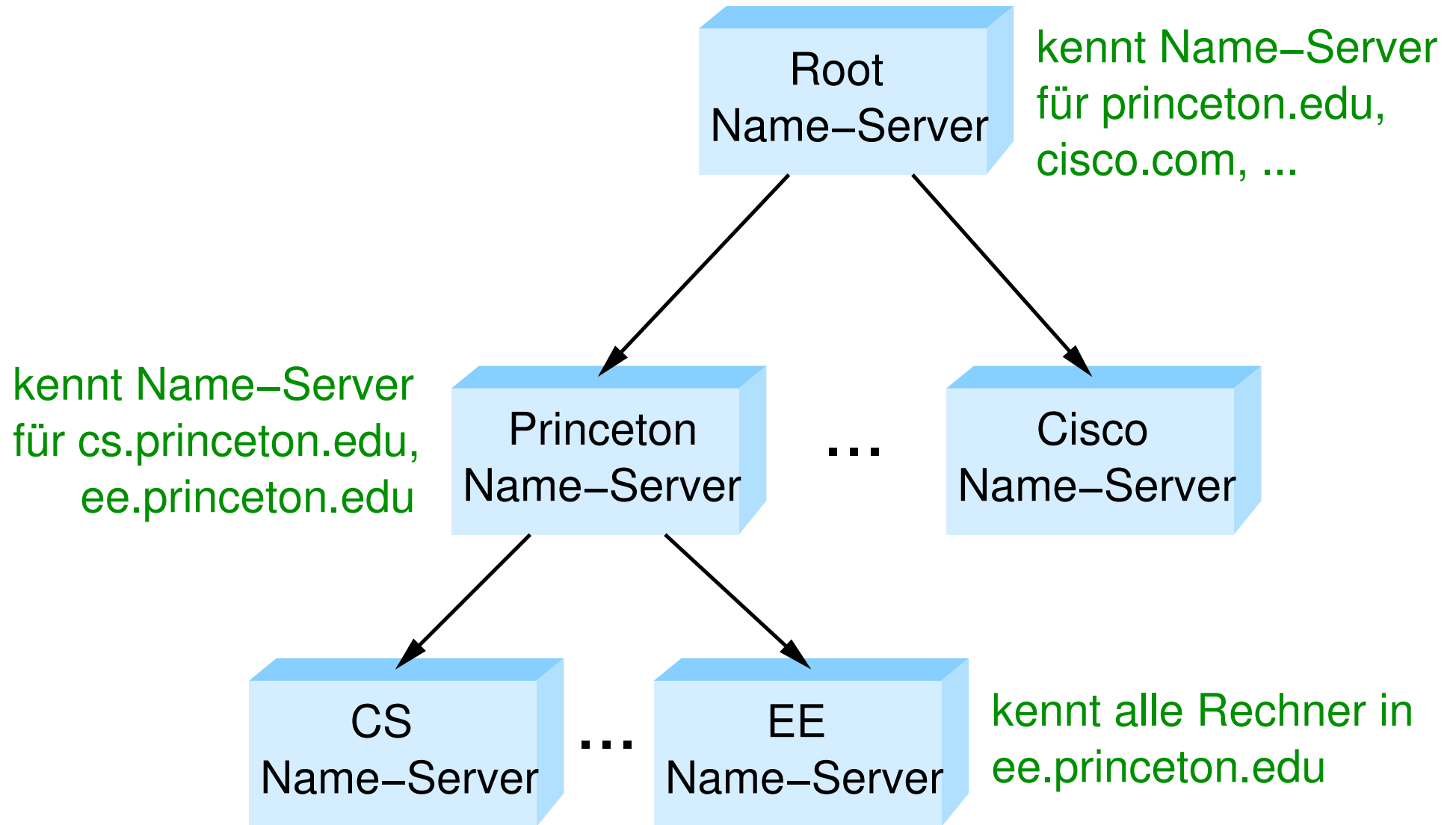
## DNS realisiert einen hierarchischen Namensraum



- ➔ Für jede Zone ist ein Name-Server zuständig
- ➔ Hierarchie von Name-Servern



## Hierarchie von Name-Servern bei DNS





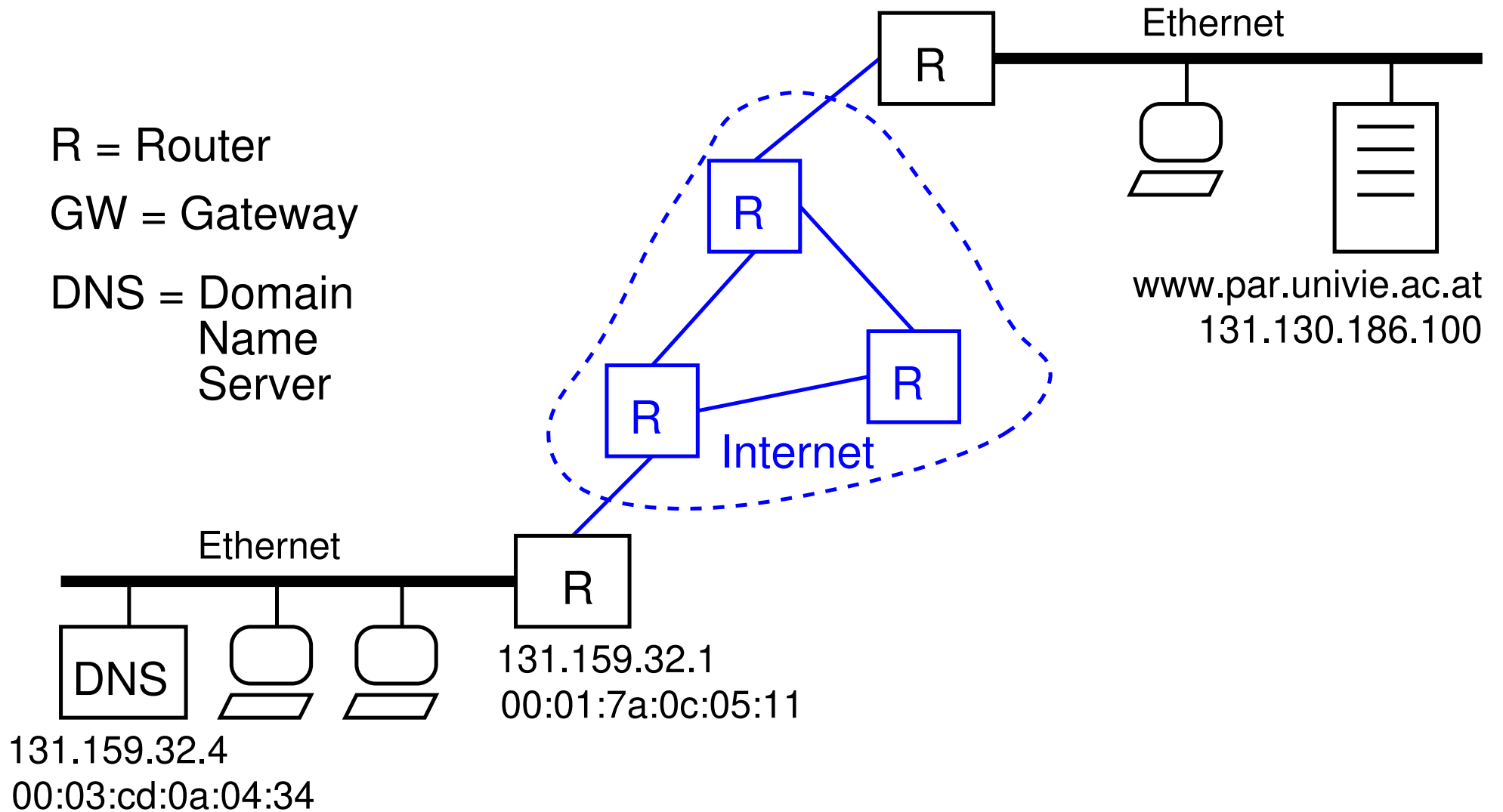
### Hierarchische Namensauflösung mit DNS

- ➔ Jeder Rechner kennt nur seinen lokalen Name-Server
- ➔ Lokale Name-Server
  - ➔ lösen lokale Namen auf
  - ➔ kennen Root-Name-Server (über Konfigurationsdatei)
- ➔ Server führen Cache mit bereits aufgelösten Namens-Adreß-Paaren
  - ➔ begrenzte Lebensdauer der Einträge
- ➔ Alternativen, falls Zuordnung nicht im Cache:
  - ➔ Nachfrage bei anderem Servern (*recursive query*)
  - ➔ antworte mit Adresse eines anderen Servers, erneute Anfrage des Clients (*nonrecursive query*)

# 9.4 Zusammenfassung



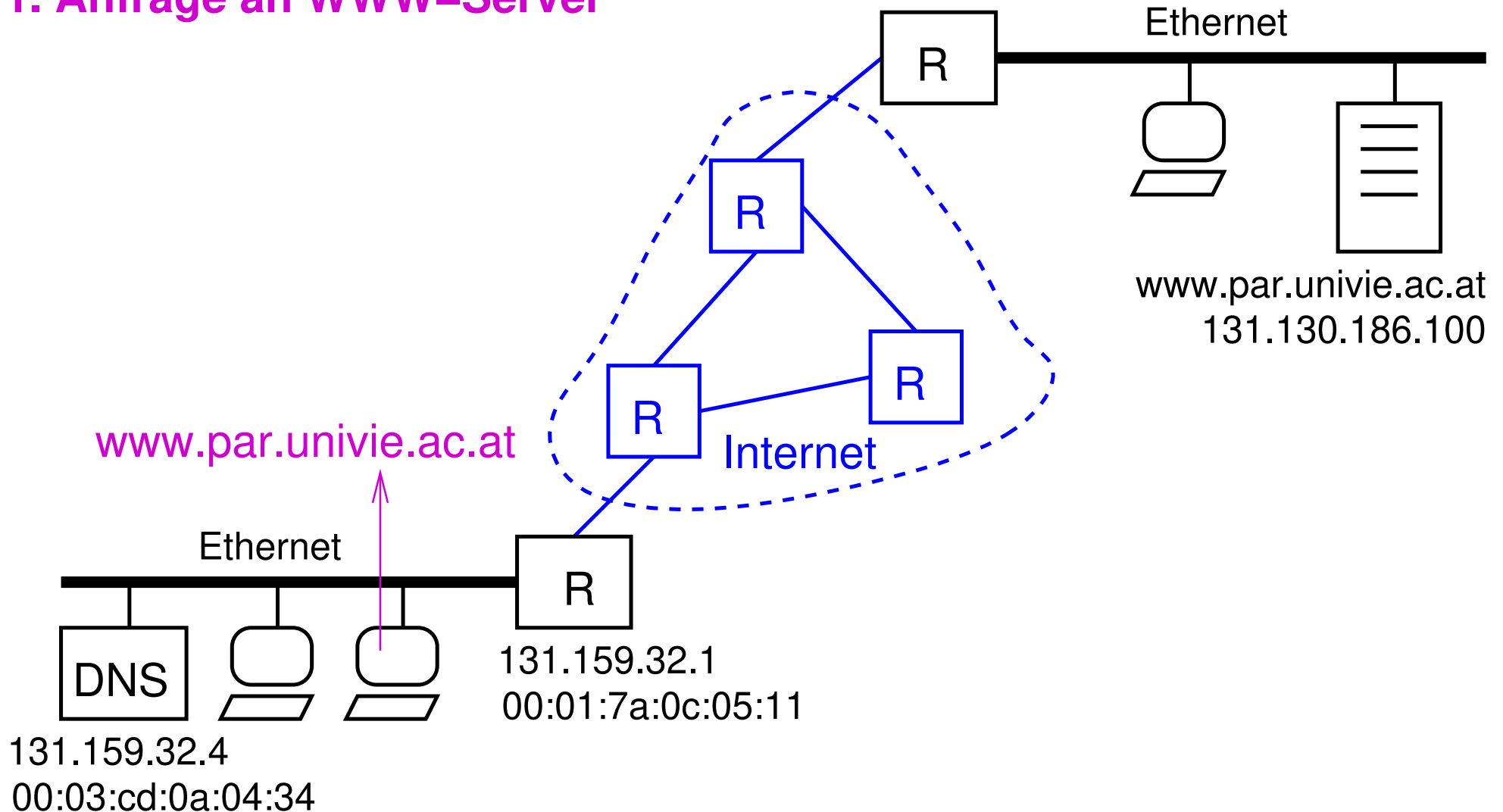
## Beispiel: Ablauf einer Web-Server-Anfrage





## Beispiel: Ablauf einer Web-Server-Anfrage

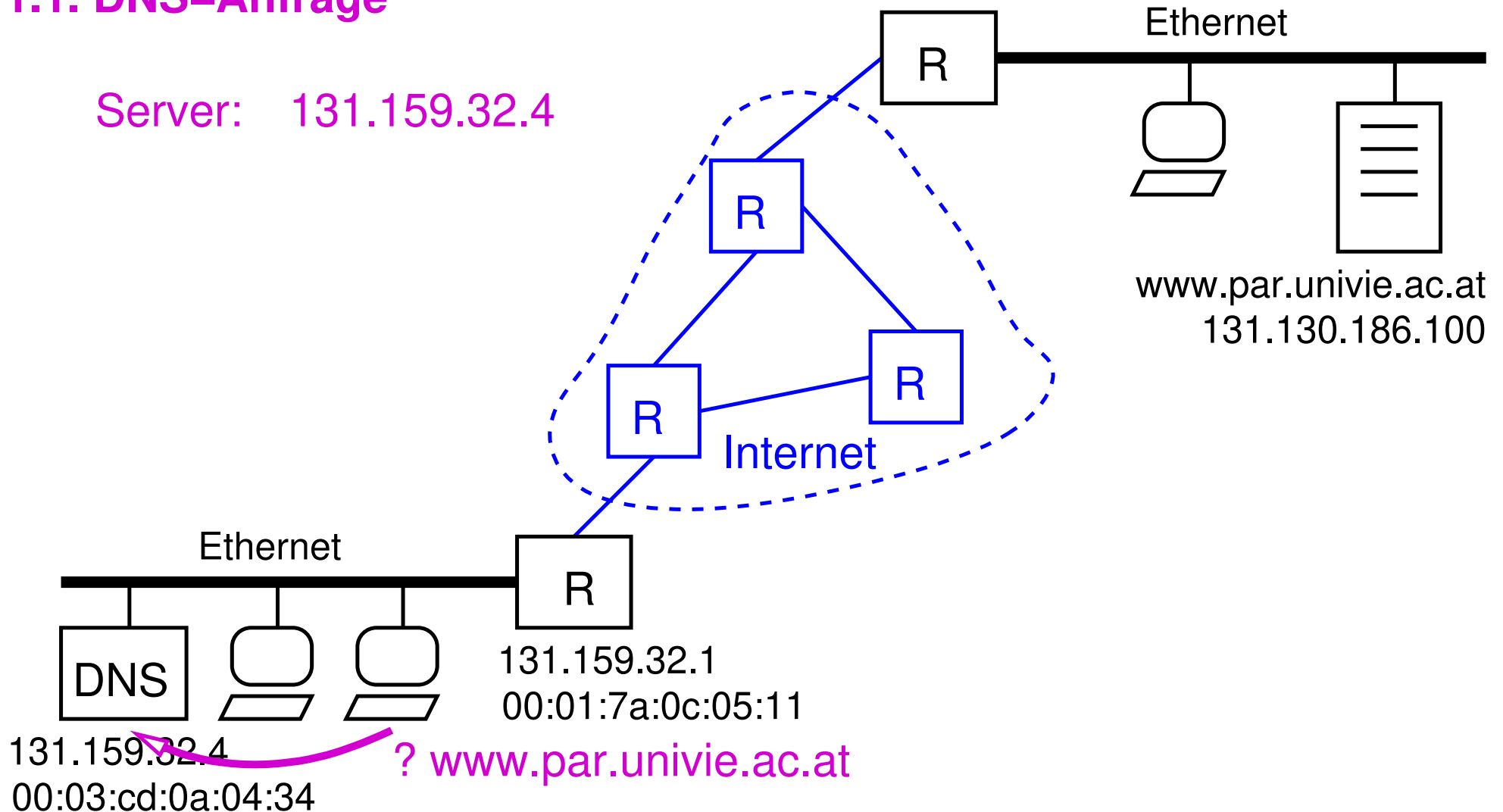
### 1. Anfrage an WWW-Server



## Beispiel: Ablauf einer Web-Server-Anfrage

### 1.1. DNS-Anfrage

Server: 131.159.32.4

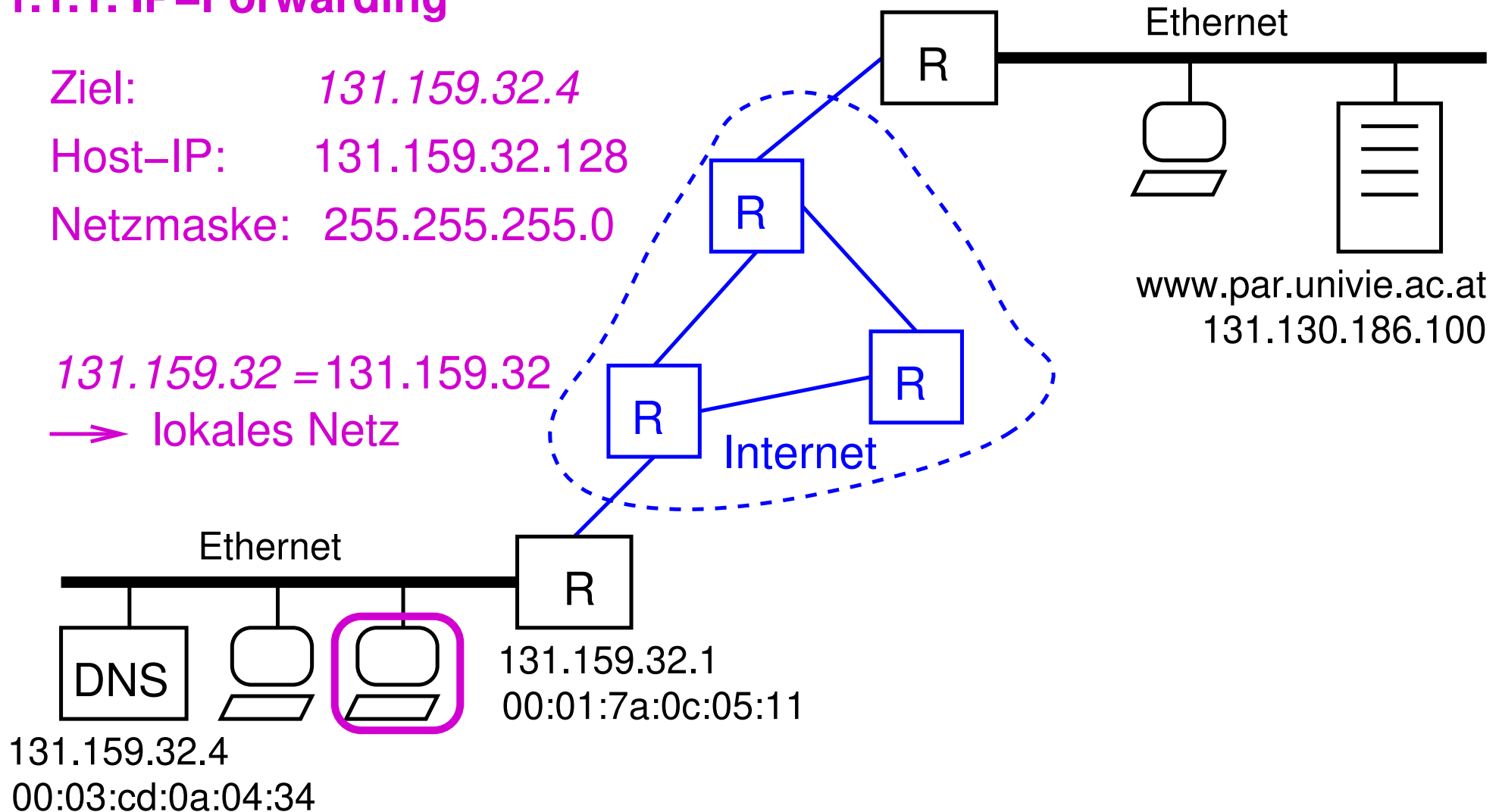


## Beispiel: Ablauf einer Web-Server-Anfrage

### 1.1.1. IP-Forwarding

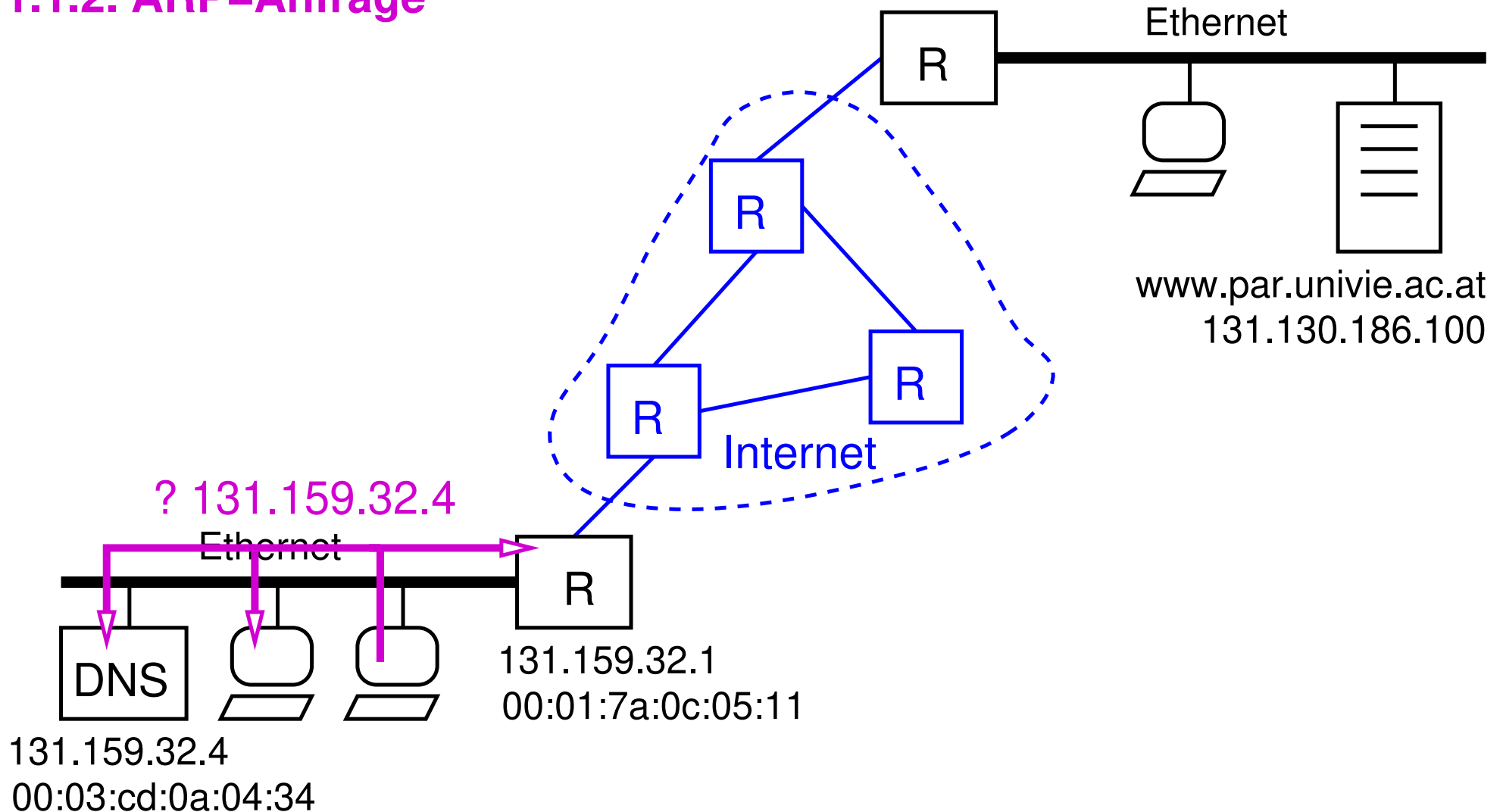
Ziel: 131.159.32.4  
Host-IP: 131.159.32.128  
Netzmaske: 255.255.255.0

$131.159.32 = 131.159.32$   
→ lokales Netz



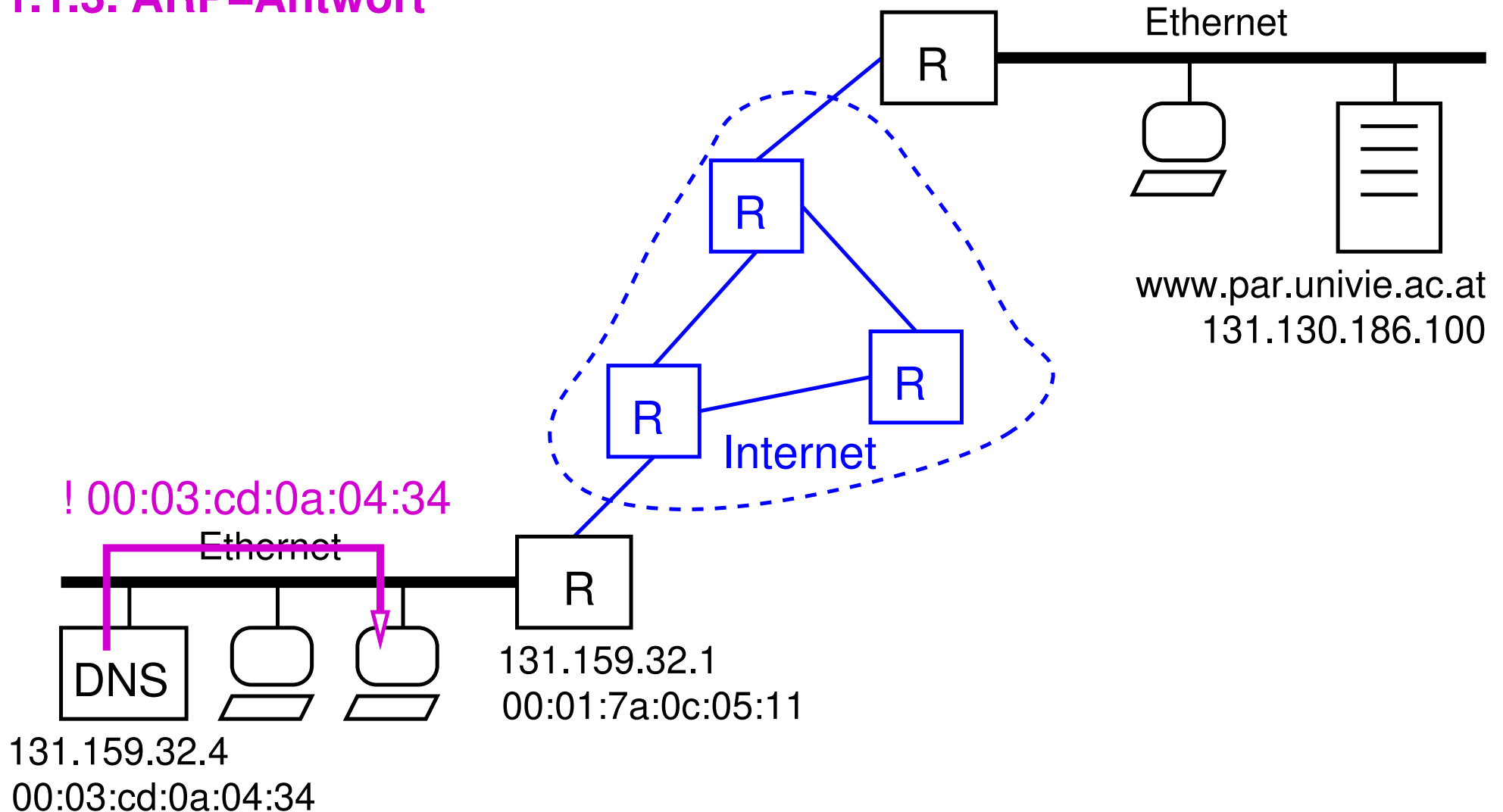
## Beispiel: Ablauf einer Web-Server-Anfrage

### 1.1.2. ARP-Anfrage



## Beispiel: Ablauf einer Web-Server-Anfrage

### 1.1.3. ARP-Antwort

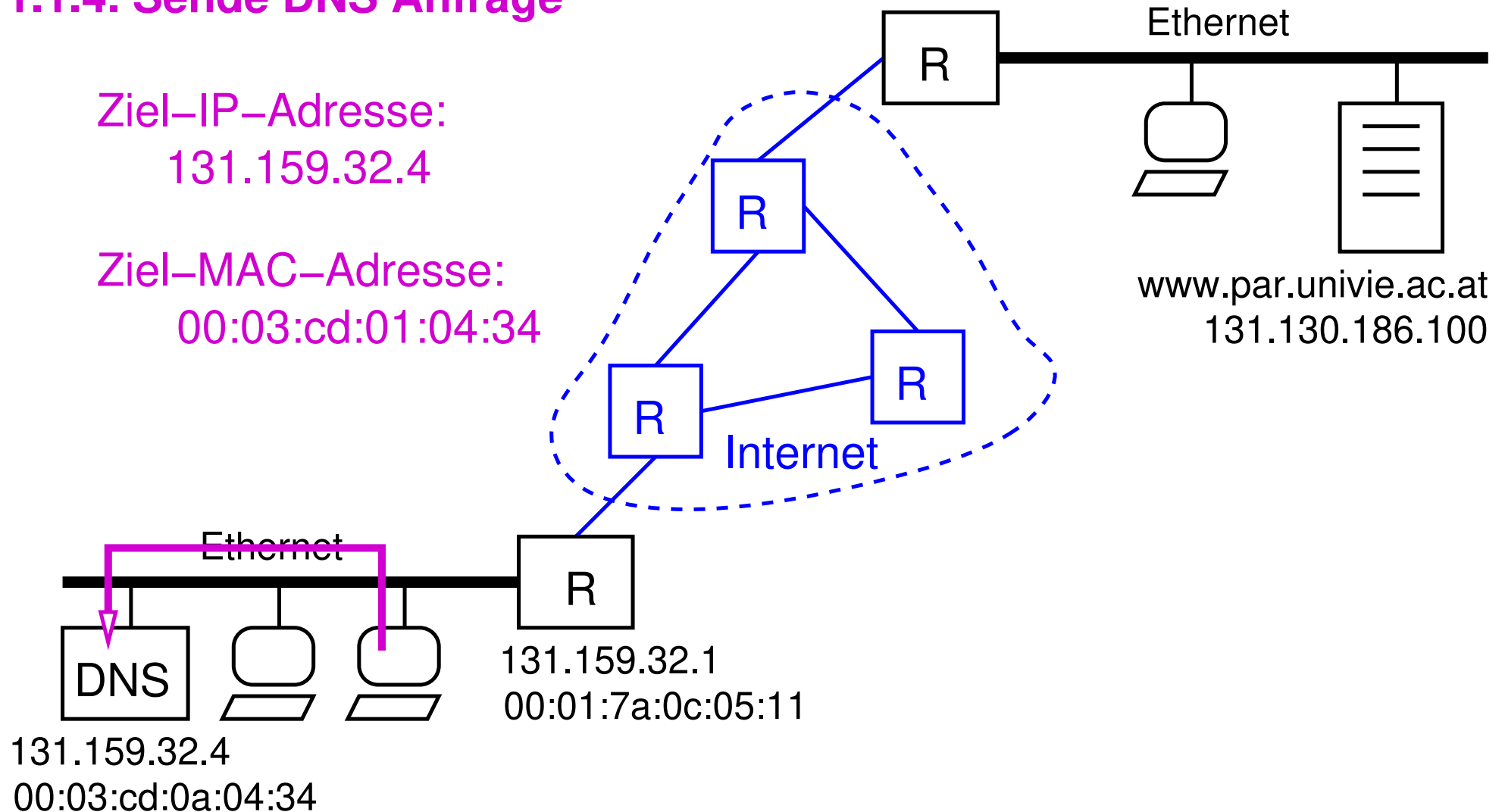


## Beispiel: Ablauf einer Web-Server-Anfrage

### 1.1.4. Sende DNS Anfrage

Ziel-IP-Adresse:  
131.159.32.4

Ziel-MAC-Adresse:  
00:03:cd:01:04:34

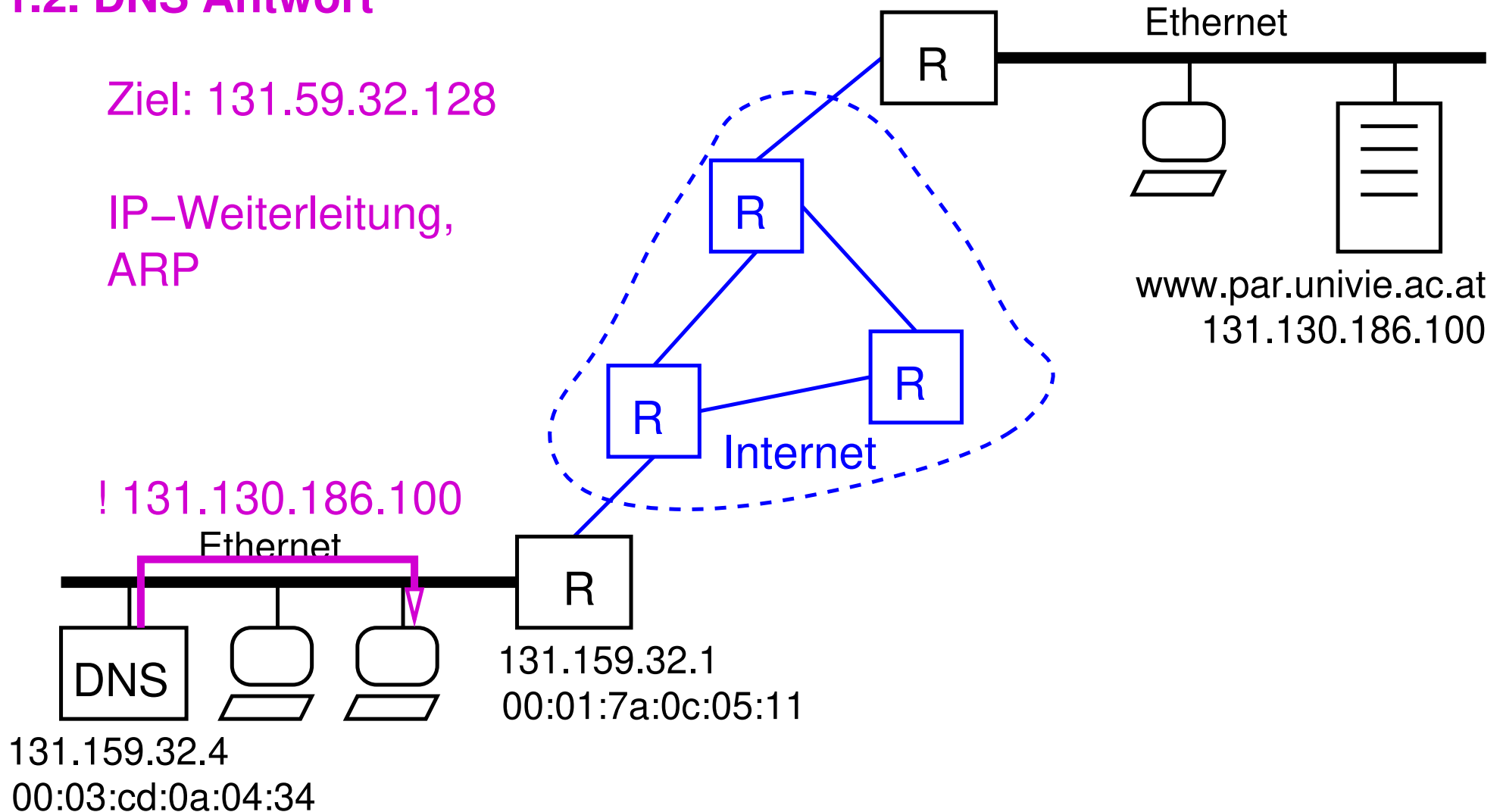


## Beispiel: Ablauf einer Web-Server-Anfrage

### 1.2. DNS Antwort

Ziel: 131.59.32.128

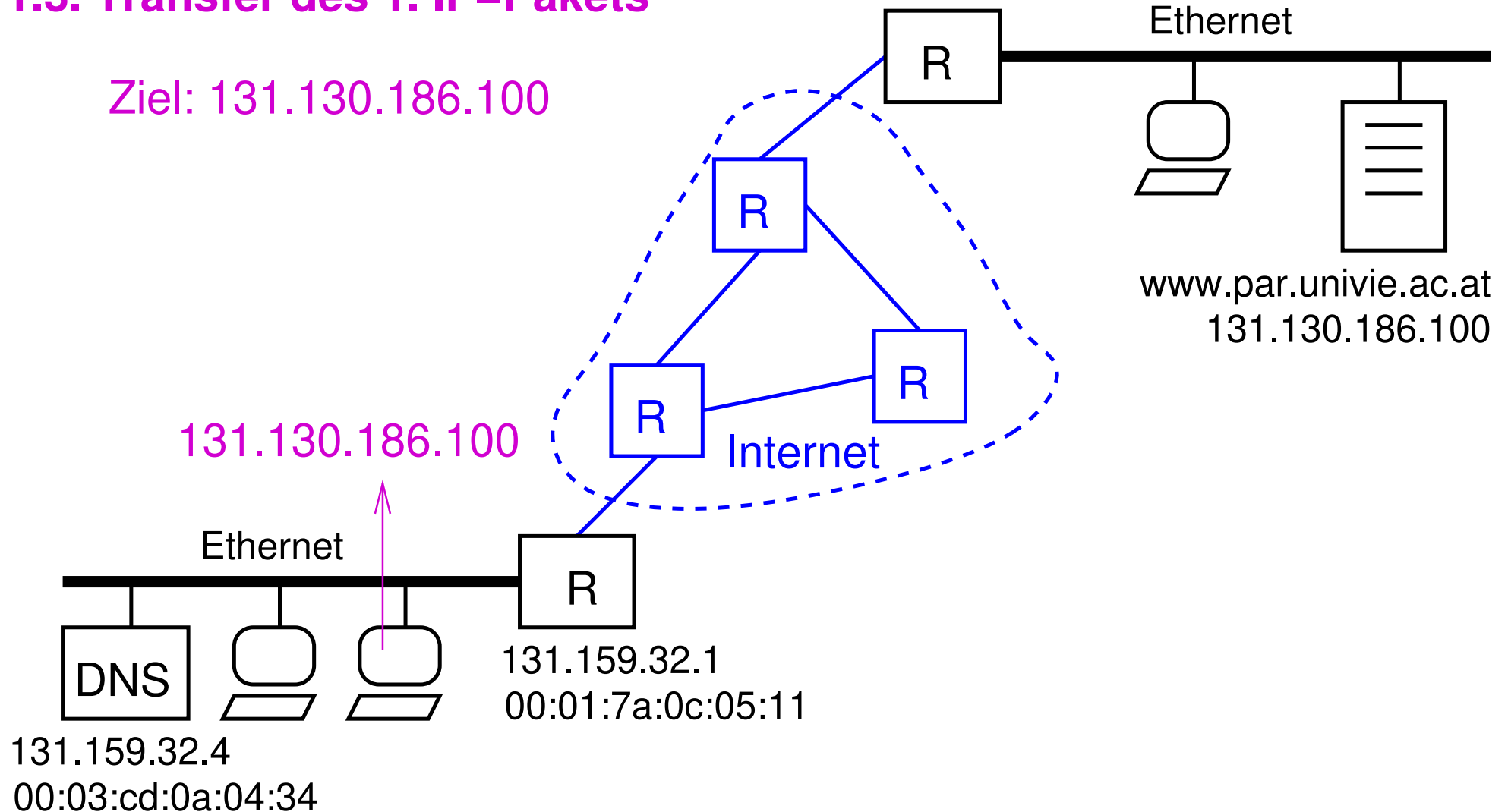
IP-Weiterleitung,  
ARP



## Beispiel: Ablauf einer Web-Server-Anfrage

### 1.3. Transfer des 1. IP-Pakets

Ziel: 131.130.186.100





## Beispiel: Ablauf einer Web-Server-Anfrage

### 1.3.1. IP-Forwarding

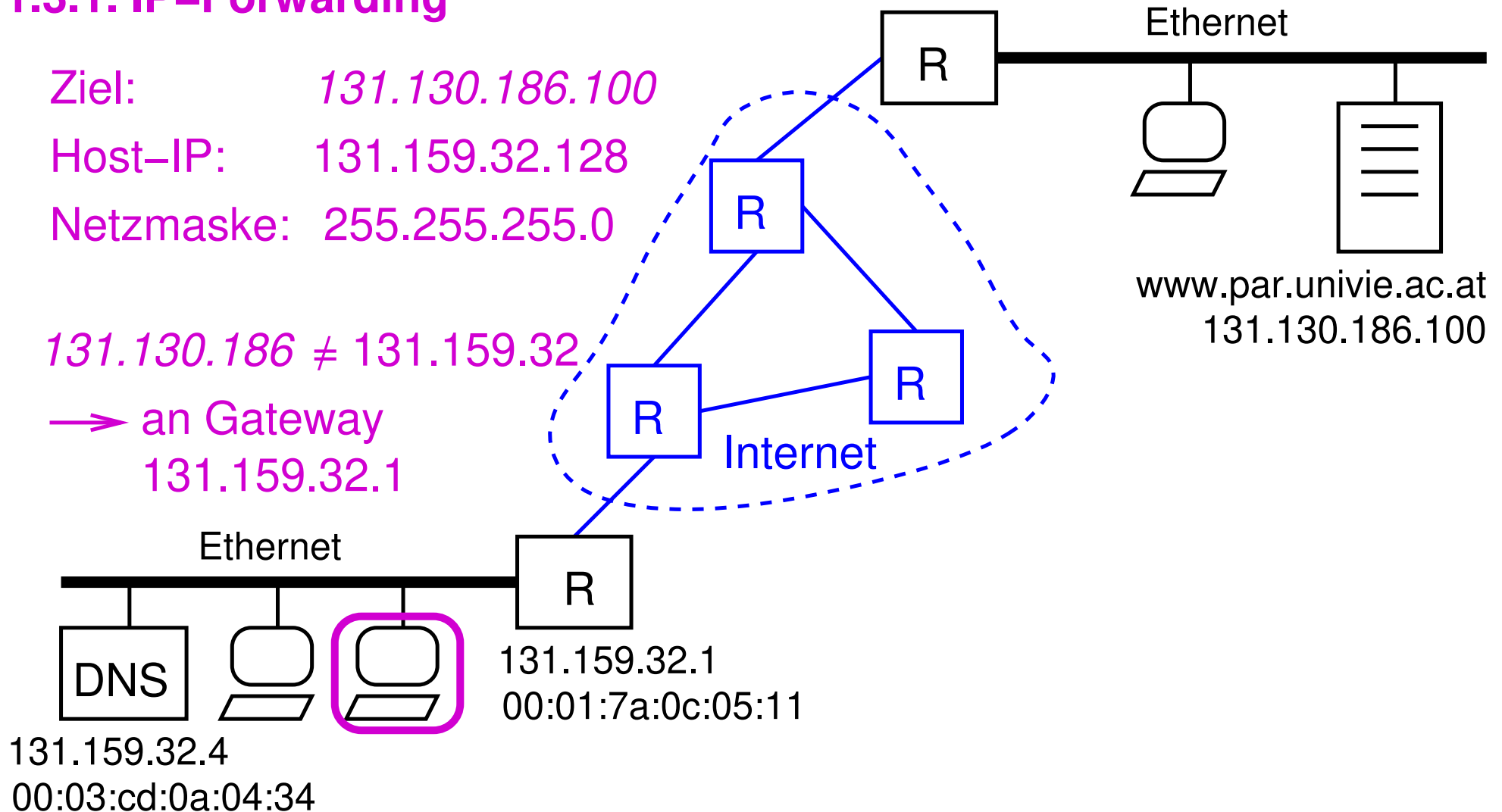
Ziel: *131.130.186.100*

Host-IP: *131.159.32.128*

Netzmaske: *255.255.255.0*

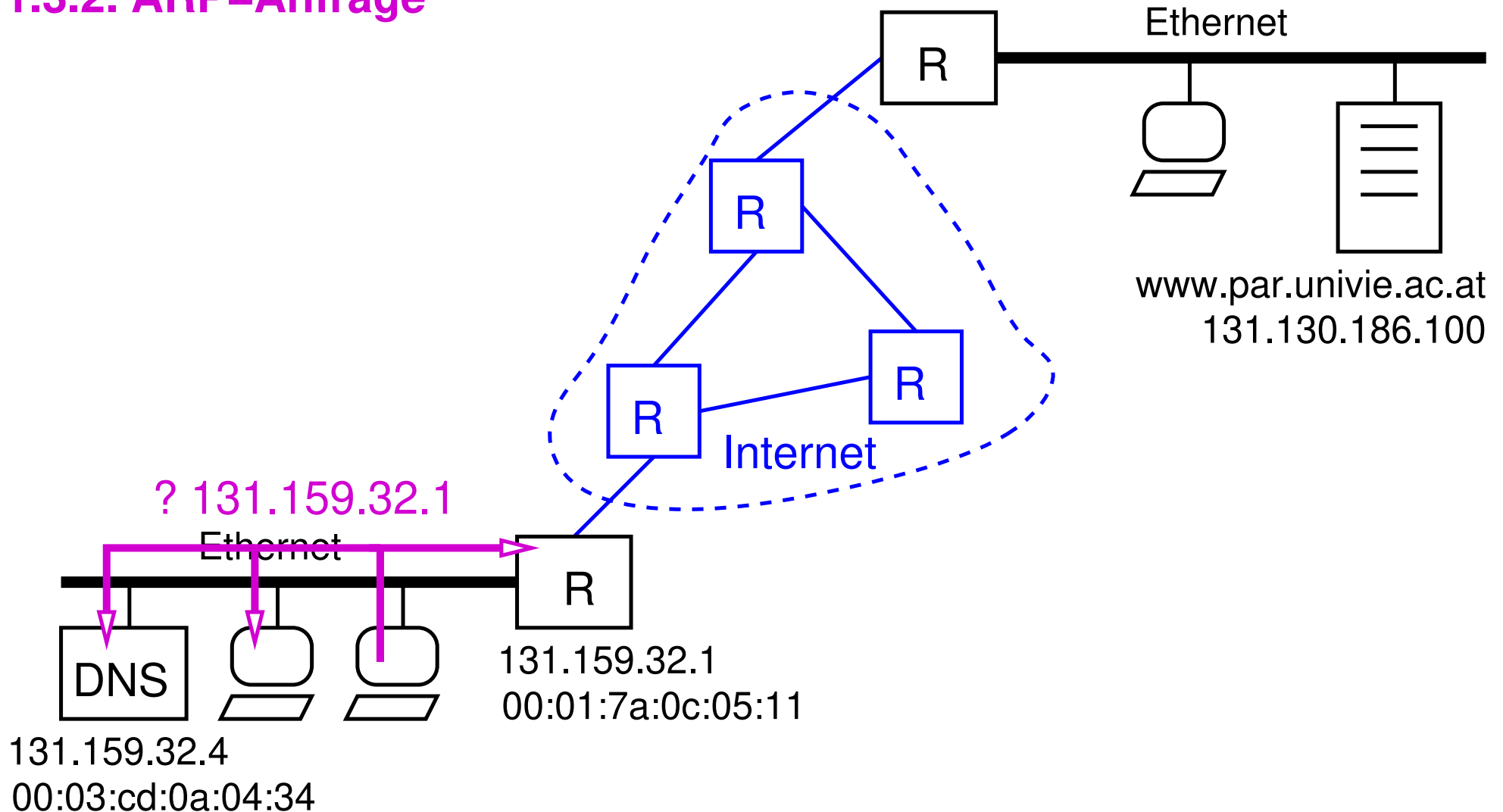
*131.130.186 ≠ 131.159.32*

→ an Gateway  
*131.159.32.1*



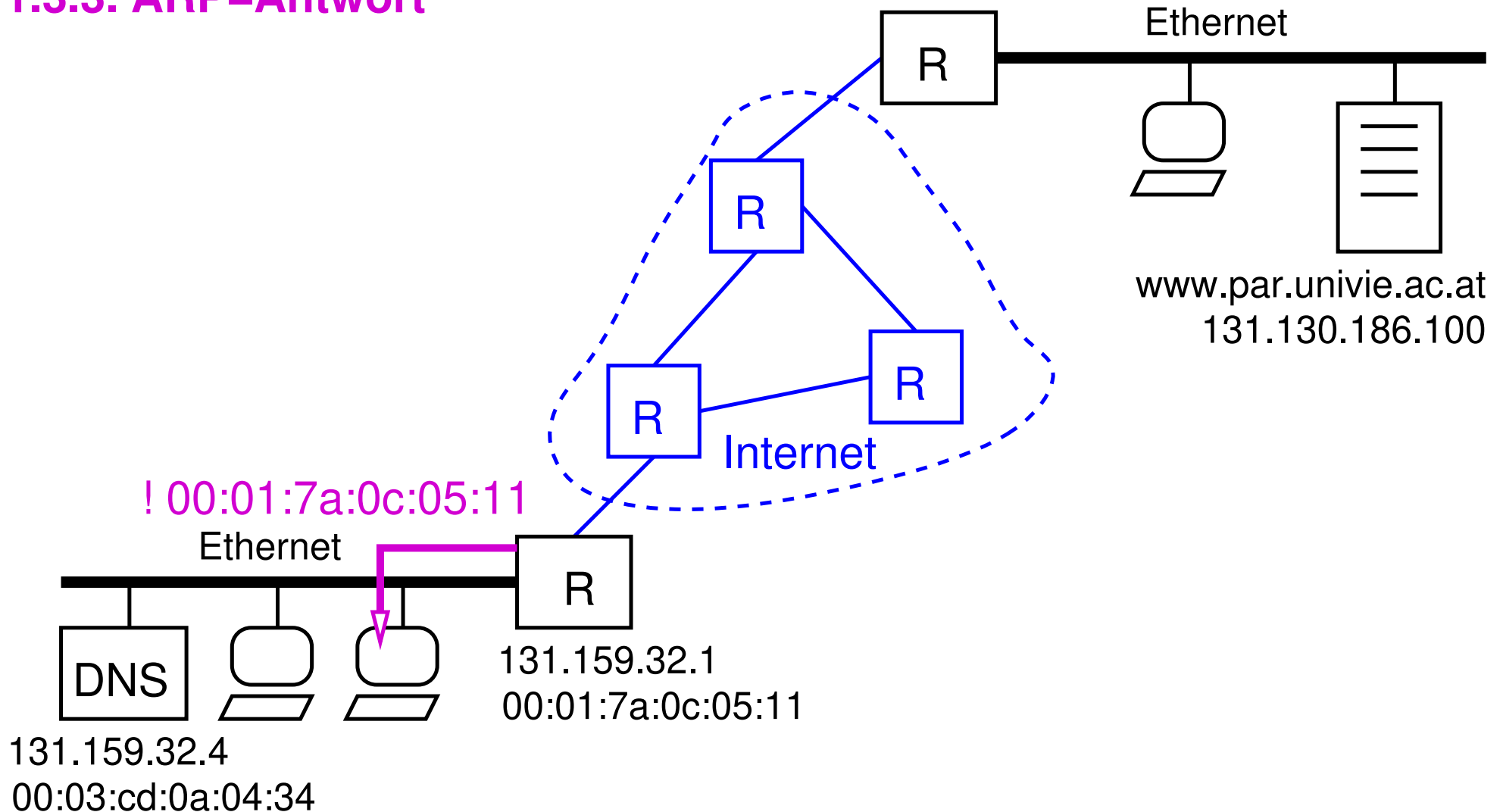
## Beispiel: Ablauf einer Web-Server-Anfrage

### 1.3.2. ARP-Anfrage



## Beispiel: Ablauf einer Web-Server-Anfrage

### 1.3.3. ARP-Antwort

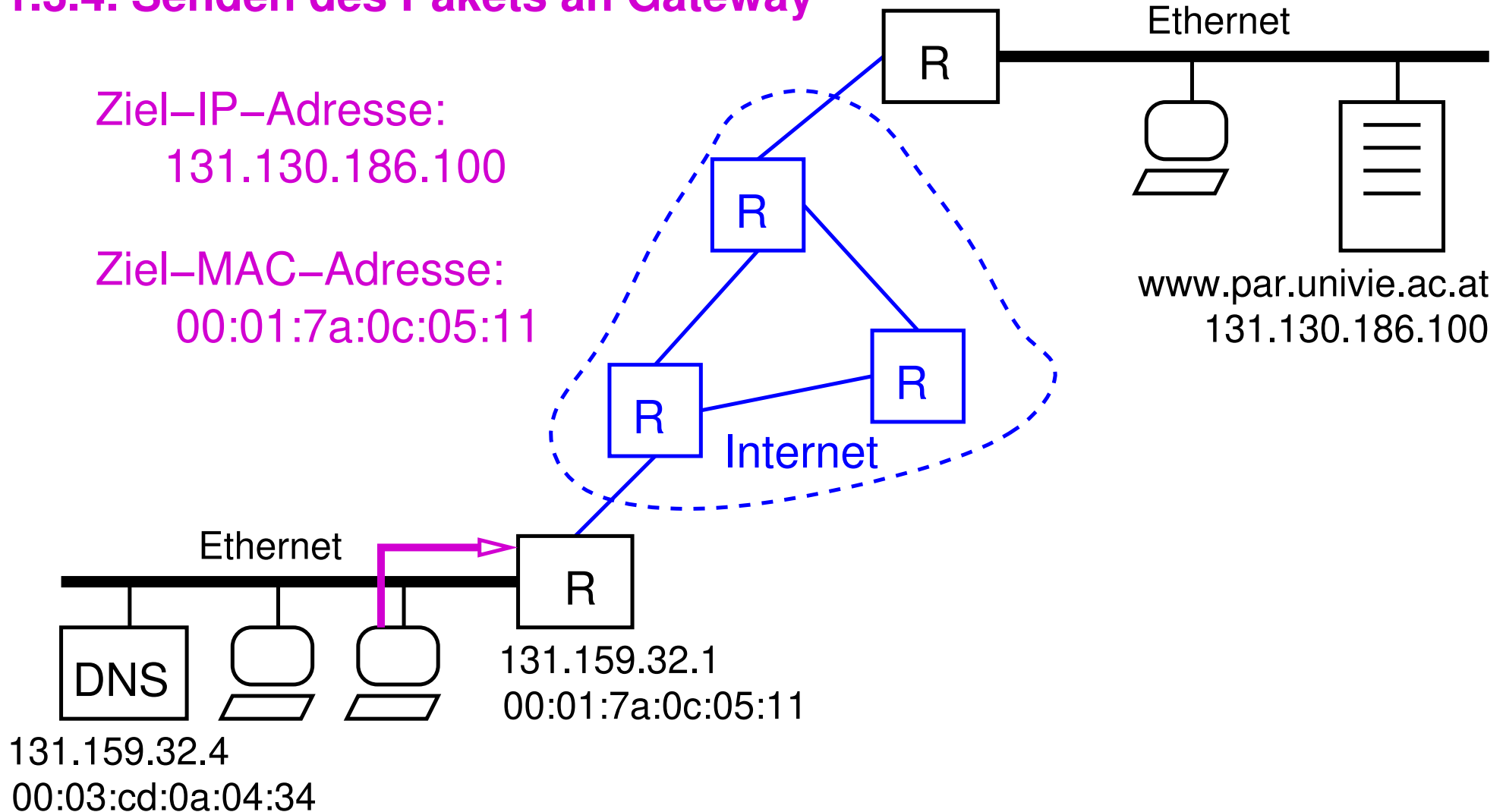


## Beispiel: Ablauf einer Web-Server-Anfrage

### 1.3.4. Senden des Pakets an Gateway

Ziel-IP-Adresse:  
131.130.186.100

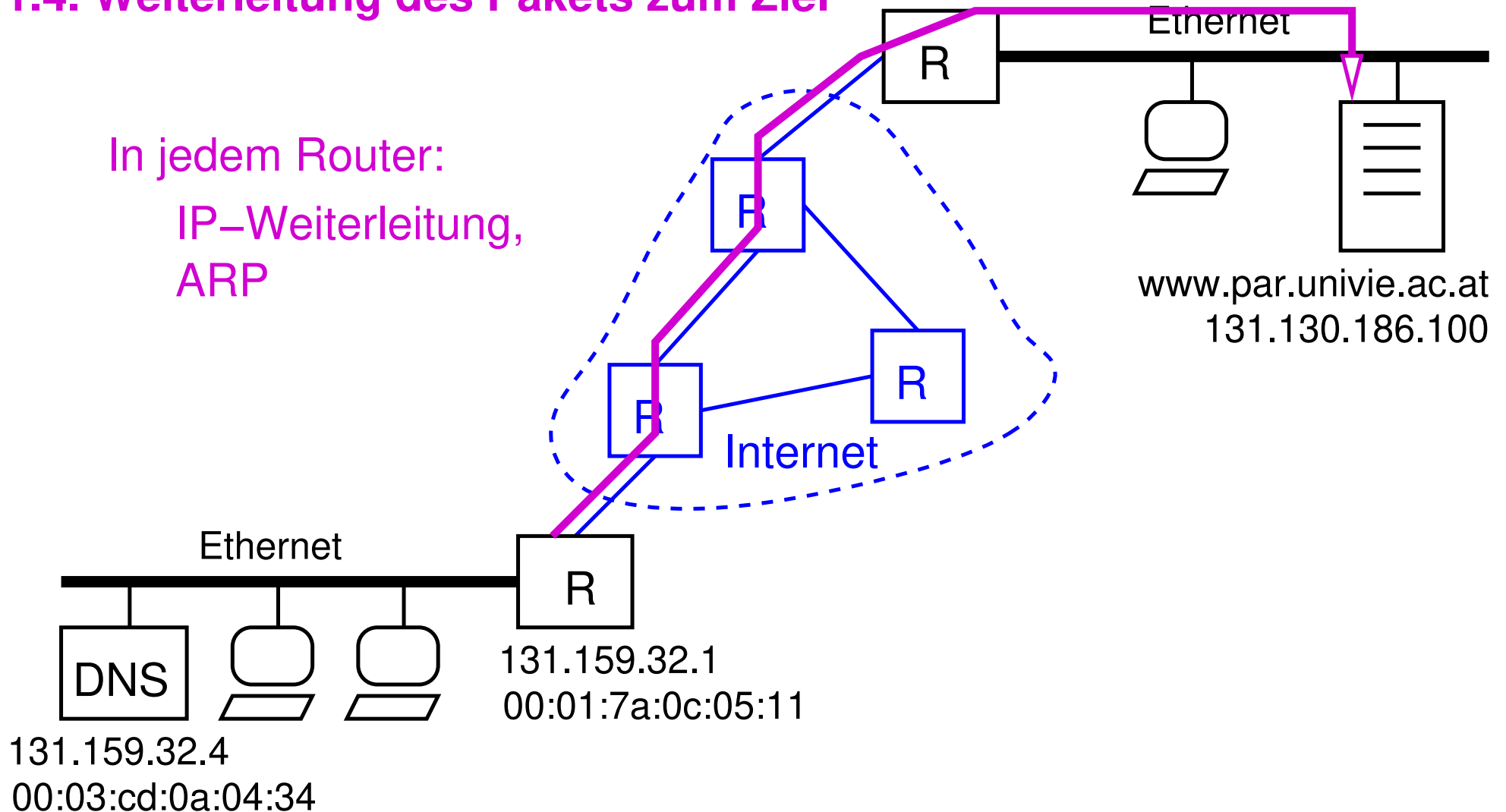
Ziel-MAC-Adresse:  
00:01:7a:0c:05:11



## Beispiel: Ablauf einer Web-Server-Anfrage

### 1.4. Weiterleitung des Pakets zum Ziel

In jedem Router:  
IP-Weiterleitung,  
ARP





- ➔ SMTP: textbasiert, speichervermittelnd
- ➔ HTTP: ebenfalls textbasiert
  - ➔ wie viele andere Anwendungsprotokolle im Internet
- ➔ DNS: Umsetzung von Rechnernamen auf IP-Adressen
  - ➔ Hierarchischer Namensraum + Server-Hierarchie

### Nächste Lektion:

- ➔ Netzwerksicherheit