

# Design und Realisierung von E-Business- und Internet-Anwendungen

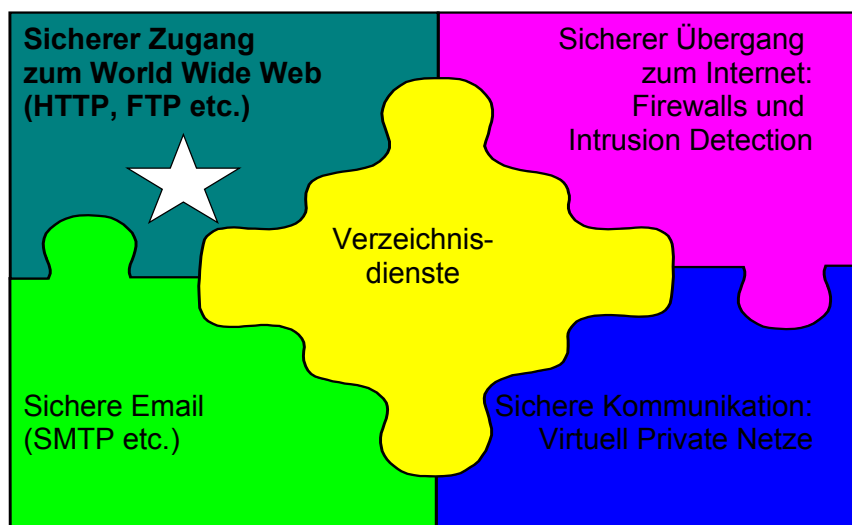
## „Web-Zugang und Internet Sicherheit“

Dr. Stephen Heilbronner et al.  
Prof. Dr. Heinz-Gerd Hegering

SoSe 2006

DREIA  
Dr. S. Heilbronner  
Dr. M. Nerb et al.  
(C) 2006  
Seite 2

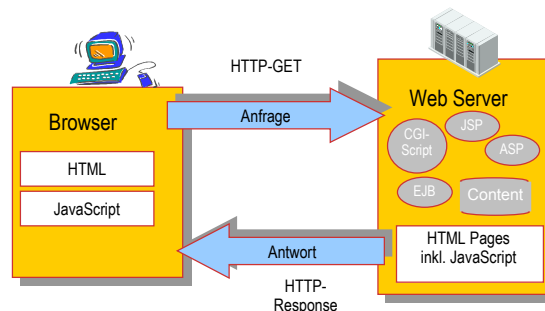
### Sicherheitsdienste großer IT-Infrastrukturen => Überblick



## Web-Zugang Grundprinzip

### ■ Zugriff auf WWW-Server durch WWW-Clients:

- 1. Browser
- 2. andere, „automatische“ Programme



### ■ Formate:

- Nicht nur HTML !
- Auch: WML, XML, beliebige Dateiformate

## Web-Zugriff Veränderte Nutzung des HTTP-Protokolls

### ■ Ursprüngliche Verwendung von HTTP:

- Übertragung statischer HTML-Seiten bzw. Dateien
- keine Unterscheidung zw. Anfragenden

### ■ Heutige Web-Zugriffe sind....

- ... nicht nur mehr vom Typ „Request/Response“.
- finden meist in einem längerdauerndem Kontext („Session“) statt, z.B. zur
  - Individualisierung der ansonsten anonymem Anfragenden bzgl.
    - Spracheinstellungen
    - „Theming“ der Webseite (Layout-Einstellungen)
  - Identifizierung / Authentisierung
    - „Zuordnung langlebiger Merkmale/Ressourcen“

## Web-Zugang Entwicklung des HTTP Protokolls

- „Hypertext Transfer Protocol“  
Protokoll-Typ: „Request/Response“
- HTTP 1.0 nur gedacht nur für „kurze“ Verbindungen:  
⇒ 3-Way TCP-Handshake beim Verbindungsaufbau aufwendig
- HTTP 1.1 lässt die TCP-Verbindung nach Übertragung persistent bestehen:  
⇒ Keine Handshakes beim Abbau/Wiederaufbau
  - Effizientere und schnellere Übertragung kleiner Informationsmengen (⇒ weniger Verzögerungen (latency) )
- Integration mit HTTP Proxies:
  - Zieladresse nicht mehr direkt adressiert mit TCP/IP
  - Alle Adressierungsinfo daher nur im HTTP-Header

Weitere Literatur:  
[http://de.wikipedia.org/wiki/Hypertext\\_Transfer\\_Protocol](http://de.wikipedia.org/wiki/Hypertext_Transfer_Protocol)  
[http://en.wikipedia.org/wiki/HTTP\\_cookie](http://en.wikipedia.org/wiki/HTTP_cookie)

## Web-Zugang „Web Sniffer“

For more information on HTTP see RFC 2616.

HTTP(S)-URL:   (EN allowed)

HTTP version:  HTTP/1.1  HTTP/1.0 (with Host header)  HTTP/1.0 (without Host header)

Raw HTML view  Accept-Encoding: gzip - Request type:  GET  POST  HEAD  TRACE

User agent:

### HTTP Request Header

```
Connect to 145.97.39.155 on port 80 ... ok
GET /wiki/Hypertext_Transfer_Protocol HTTP/1.1 [en:de]
Host: de.wikipedia.org [en:de]
Connection: close [en:de]
Accept-Encoding: gzip [en:de]
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5 [en:de]
Accept-Language: de,en-us;q=0.7,en;q=0.3 [en:de]
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.0 [en:de]
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8.0.1) Gecko/20060128 SeaMonkey/1.0 Web-Sniffer/1.0.24 [en:de]
Referer: https://web-sniffer.net/ [en:de]
```

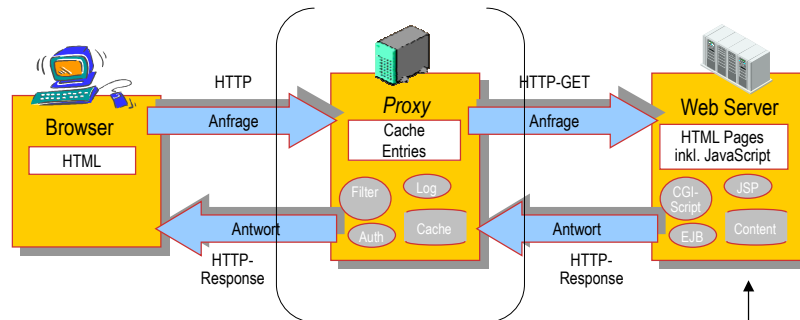
### HTTP Response Header

Name	Value	Delim
HTTP Status Code: HTTP/1.0 200 OK		
Date:	Mon, 01 May 2006 20:02:18 GMT	CRLF
Server:	Apache	CRLF
X-Powered-By:	PHP/5.1.2	CRLF
Content-Language:	de	CRLF
ETag:	W/"de/wiki/pocache:ldhash:2179-0110D01de2-20060428101018"	CRLF
Vary:	Accept-Encoding, Cookie	CRLF
Cache-Control:	private, s-maxage=0, max-age=0, must-revalidate	CRLF
Last-Modified:	Fri, 28 Apr 2006 10:10:18 GMT	CRLF
Content-Encoding:	gzip	CRLF
Content-Type:	text/html; charset=utf-8	CRLF
X-Cache:	MISS from sn7.wikimedia.org	CRLF
X-Cache-Lookup:	MISS from sn7.wikimedia.org/80	CRLF
X-Cache:	MISS from fachaia.knams.wikimedia.org	CRLF
X-Cache-Lookup:	MISS from fachaia.knams.wikimedia.org/80	CRLF
Connection:	close	CRLF

Content (encoded: 13.68 KiB / decoded: 48.04 KiB)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="de" lang="de" dir="ltr">
  <head>
```

## Web-Zugang Web-Architektur mit Proxies (HTTP)



In der Praxis:

- Kette oft mehrfach wiederholt: „Proxy Chaining“
- Großzügige Proxy-Auslegung wichtig für:
  - Multimedia-Streaming in Echtzeit
  - „Pre-pushed content“

Mehr zu dessen  
Architektur  
im Juni

## Web-Zugang Proxies (für HTTP)

- HTTP ist
  - entweder anonym, oder
  - Authentisierungs-Information im Header
- Autorisierende Proxies unterbrechen den Fluß zum Server
  - erscheinen dem Client genauso wie geschützte Server
  - verlangen Authentisierung vom Benutzer
  - filtern diese vor Weitergabe wieder heraus
    - echter Server benötigt dann eventuell weitere Autorisierung

## Web-Zugang Exkurs: AAA

- **Authentisierung**
  - Feststellung der Identität
  - Implementierung:
    - Abfrage Benutzername/Passwort
    - Keycard
- **Autorisierung**
  - Festlegung der Nutzungsrechte
  - hier: Welche weiteren Zugriffe sind erlaubt ?
- **Accounting**
  - Aufzeichnung verrechnungsrelevanter Nutzungsdaten
  - Aggregation der Daten
  - Ziel: Verrechnung der Nutzen

## AAA: Übertragung von Autorisierungs-Information

- **Cookies**
  - Kleine „Stücke von Information“
  - Inhalte vom Server festgelegt
  - Browser stellt sie bestimmten Servern zur Verfügung
  - lange Lebensdauer
- **HTTP-Basic/Digest**
  - Authentisierungsinformation im HTTP-Header
- **Oder: Im URL kodiert**
  - „Session-ID's“



Wie sieht so etwas aus ? ...

DREIA  
Dr. S. Hellbrunner  
Dr. M. Nerb  
(C) 2006  
Seite 12

http://leia.muclab.de:8080/manage

### Site Error

An error was encountered while publishing this resource.

### Unauthorized

**You are not authorized to access this resource.**

Troubleshooting Suggestions

- The URL may be incorrect.
- The parameters passed to this resource may be incorrect.
- A resource that this resource relies on may be encountered.

For more detailed information about the error, please refer to the error message.

If the error persists please contact the site maintainer. Thank you.

**Authorization Dialog**

You need to supply a username and a password to access this site.

Site: **Zope at leia.muclab.de**

Username: admin2

Password: \*\*\*\*\*

OK Cancel

leia.muclab.de contacted. Waiting for reply...

DREIA  
Dr. S. Hellbrunner  
Dr. M. Nerb  
(C) 2006  
Seite 12

**K - <capture> - Ethereal**

No.	Time	Source	Destination	Protocol	Info
7	0.001677	qui-gon.muclab.de	leia.muclab.de	TCP	service-ctrl > http-alt [ACK] Seq=465730
8	0.004155	qui-gon.muclab.de	leia.muclab.de	HTTP	GET /manage HTTP/1.1
9	0.004191	leia.muclab.de	qui-gon.muclab.de	TCP	http-alt > service-ctrl [ACK] Seq=519057
10	0.018531	leia.muclab.de	qui-gon.muclab.de	HTTP	HTTP/1.1 401 Unauthorized
11	0.019286	qui-gon.muclab.de	leia.muclab.de	TCP	service-ctrl > http-alt [ACK] Seq=483730
12	0.019327	leia.muclab.de	qui-gon.muclab.de	HTTP	Continuation
13	0.020670	qui-gon.muclab.de	leia.muclab.de	TCP	service-ctrl > http-alt [ACK] Seq=483730
14	8.389589	qui-gon.muclab.de	leia.muclab.de	TCP	service-ctrl > http-alt [RST, ACK] Seq=4
15	8.390577	qui-gon.muclab.de	leia.muclab.de	TCP	opentable > http-alt [SYN] Seq=499628242
16	8.390627	leia.muclab.de	qui-gon.muclab.de	TCP	http-alt > opentable [SYN, ACK] Seq=5366
17	8.391235	qui-gon.muclab.de	leia.muclab.de	TCP	opentable > http-alt [ACK] Seq=499628243
18	8.392766	qui-gon.muclab.de	leia.muclab.de	HTTP	GET /manage HTTP/1.1
19	8.392803	leia.muclab.de	qui-gon.muclab.de	TCP	http-alt > opentable [ACK] Seq=536677737

Transmission Control Protocol, Src Port: opentable (2368), Dst Port: http-alt (8080), Seq: 499628243, Ack: 519057

Hypertext Transfer Protocol

```

GET /manage HTTP/1.1\r\n
Connection: Keep-Alive\r\n
User-Agent: Mozilla/5.0 (compatible; Konqueror/2.1.1; X11)\r\n
Pragma: no-cache\r\n
Cache-control: no-cache\r\n
Accept: text/*;q=1.0, image/png;q=1.0, image/jpeg;q=1.0, image/gif;q=1.0, image/*;q=0.8, */*;q=0.5\r\n
Accept-Encoding: x-gzip; q=1.0, gzip; q=1.0, identity\r\n
Accept-Charset: iso-8859-1;q=1.0, */*;q=0.9, utf-8;q=0.8\r\n
Accept-Language: en\r\n
Host: leia.muclab.de:8080\r\n
Authorization: Basic YWRtaAM4yOjEgMw==\r\n
\r\n

```

0000 00 a0 d2 1a b9 35 08 00 20 8f 88 9a 08 00 45 00 .....5.. ..E.  
0010 01 f5 6d 50 40 00 3f 06 c5 b8 3e 9d c4 dc 3e 9d ..nP8.? ..>...  
0020 c4 e3 09 40 1f 90 1d c7 b8 d3 1f fd 0d 69 80 18 ...@.....  
0030 7d 78 75 31 00 00 01 01 08 0a 06 67 e8 e6 00 86 }xdl.....g....  
0040 5c 0c 47 45 54 20 2f 6d 61 6e 61 67 65 20 48 54 \.GET /m anage HT

Filter: [ ] Reset [ ] File: <capture> Drops: 0

## Web-Proxies

### Server Backends: Anforderungen

- Abfrage von Informationen für Authentisierung und Autorisierung
  - Benutzer
  - Jeweilige Rechte
  - Nutzungszeiten
  - Sonstige Vorbelegungen (GUI)
- Aufzeichnung der Nutzungsdaten
  - Accounting
  - Leistungsmanagement
- Prüfung von Inhalten
  - Angefragte URLs
  - Empfangene Daten

## Web-Proxies

### Server-Backends: Implementierungen

- Datenbank
  - Vorzuhaltende Information
    - Autorisierung: Name/Passwort
    - Autorisierung: Welche Bereiche dürfen erreicht werden?
    - .....
  - Zugriffsprotokolle
    - ODBC für SQL-Datenbank
    - RADIUS (Remote Access and DialIn User Service)
    - LDAP (Lightweight Directory Access Protocol)
- Logging
  - Logdatei aus Performanz-Gründen (keine DB!)
- Weitere Dienste
  - Spezielle Protokolle

## Typisches Nutzerverhalten 2001 bis 2004: Ziele aus großen IT-Infrastrukturen

2004

Top Websites Yesterday:					
destination	request	%	Byte	%	hit-%
*.ebay.de	3032	2.44	99970K	13.43	0.66
*.uni-kl.de	143	0.11	91385968	11.99	0.00
*	133	0.11	27819751	3.65	97.74
*.ebaystatic.com	25216	20.26	24799285	3.25	66.86
*.hp.com	66	0.05	22314733	2.93	34.85
*.ebayimg.com	2382	1.91	21111267	2.77	25.65
*.herkeley.edu	114	0.09	20421144	2.68	0.00
*.smc.com	349	0.28	17355549	2.28	59.03
*.condirect.de	4300	3.45	17172862	2.25	1.00
*.t-online.de	3260	2.62	13278385	1.74	32.82
<error>	6481	5.21	12263399	1.61	13.56
*.gmx.net	1899	1.53	11962863	1.57	66.72
*.praline.de	1325	1.06	10141176	1.33	50.79
*.ebay.com	3103	2.49	8927071	1.17	58.07
*.mobile.de	978	0.79	8708235	1.14	35.99
*.web.de	1539	1.24	8012137	1.05	15.85
*	1355	1.09	6998993	0.92	79.63
*.monthbilder.de	267	0.21	6690062	0.88	43.77

Destination	Request	%	Bytes	%	hit-%
<error>	33683	12.84	39721881	3.41	5.38
*.t-online.de	8765	3.34	19083028	1.64	66.77
*.bild.de	8282	3.16	55732135	4.79	46.61
*.doubleclick.net	5061	1.93	7087115	0.61	9.27
*.web.de	4917	1.87	24289001	2.09	35.67
*.akamai.net	4884	1.86	7574273	0.65	87.24
*.xxxxxxxxxxxxxxxxx.de	4098	1.56	14681246	1.26	68.64
*.sueddeutsche.de	3768	1.44	13554644	1.16	38.96
*.consors.de	3133	1.19	6986643	0.60	69.04
*.boerse.de	2831	1.08	11167450	0.96	72.45
*.lycos.de	2796	1.07	22830497	1.96	65.24
*.microsoft.com	1898	0.72	14627786	1.26	54.74
*.br-online.de	1761	0.67	3545059	0.30	73.25
*.ebay.com	1623	0.62	2222164	0.19	91.44
*.yyyyyyyyyyyyyyyyy.de	1599	0.61	3464164	0.30	49.47
*.gmx.net	1483	0.57	11860845	1.02	0.20
other: 2691 2nd-level-domains	160337	61.13	854876K	75.20	48.87
Sum	262293	100.00	1136733K	100.00	43.35

2001

## Web-Proxies Ein paar Gedanken zu Optimierungspotentialen....

- Hit/Miss-Rate:
  - Anzahl: ca. 1/3 Treffer
  - Größe: ca. 1/4 Treffer
  - 2/3 aller Anfragen werden verlangsamt
- Nutzung über Tageszeit
  - Mittags NICHT weniger :-)
- Server-seitige Optimierung der Übertragung?
  - Header „LAST-MODIFIED“ mitschicken
  - Explizite Informationen zu „EXPIRES“ (z.B. in 10 Minuten)
  - Grafiken/Inhalte browser-spezifisch aufbereiten
  - Inhalte komprimieren (GZ)

Top Websites Yesterday:					
destination	request	%	Byte	%	hit-%
*.ebay.de	3032	2.44	99970K	13.43	0.66
*.uni-kl.de	143	0.11	91385968	11.99	0.00
*	133	0.11	27819751	3.65	97.74
*.ebaystatic.com	25216	20.26	24799285	3.25	66.86
*.hp.com	66	0.05	22314733	2.93	34.85
*.ebayimg.com	2382	1.91	21111267	2.77	25.65
*.herkeley.edu	114	0.09	20421144	2.68	0.00
*.smc.com	349	0.28	17355549	2.28	59.03
*.condirect.de	4300	3.45	17172862	2.25	1.00
*.t-online.de	3260	2.62	13278385	1.74	32.82
<error>	6481	5.21	12263399	1.61	13.56
*.gmx.net	1899	1.53	11962863	1.57	66.72
*.praline.de	1325	1.06	10141176	1.33	50.79
*.ebay.com	3103	2.49	8927071	1.17	58.07
*.mobile.de	978	0.79	8708235	1.14	35.99
*.web.de	1539	1.24	8012137	1.05	15.85
*	1355	1.09	6998993	0.92	79.63
*.monthbilder.de	267	0.21	6690062	0.88	43.77



## Caching Proxies Statische Auslegung

- Plattenplatzbedarf:
  - Statistische Fragen
    - Wie häufig wird auf welche Seiten zugegriffen?
    - Wie schnell veralten welche Seiten?
  - Nutzen vs. Verwaltungsaufwand berücksichtigen
  - Typische, sinnvolle Größe ??
  
- Welche Schlüsse zieht man aus der Beobachtung:
  - 80 % der Seiten im Cache veralten innerhalb eines Tages...
  - Bringt ein großer Cache wirklich so viel ..... (nein, aber ....)

## Caching Proxies Dynamische Auslegung

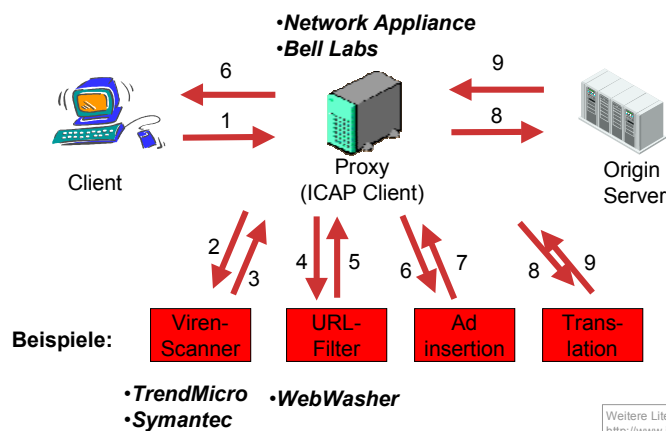
- Anzahl Prozessoren
  - Anzahl parallel laufender Zugriffe (HTTP 1.1 vs 1.0)
  - Wieviel kann ein Prozessor davon abwickeln ?
    - Wie ist das Verhalten bei Überlast ?
    - Toleranz der Benutzer ?
  - Entscheidend sind Zusatzdienste:
    - Virenschanning für HTTP/FTP => hohe Prozessorlast
    - Multimedia-Streaming
  
- Weitere limitierende Faktoren
  - Zugangsbandbreiten eingehend
  - Zugangsbandbreiten abgehend
  - Zugriffskarakteristik für Hintergrundspeicher

## Virenschanning Implementierung

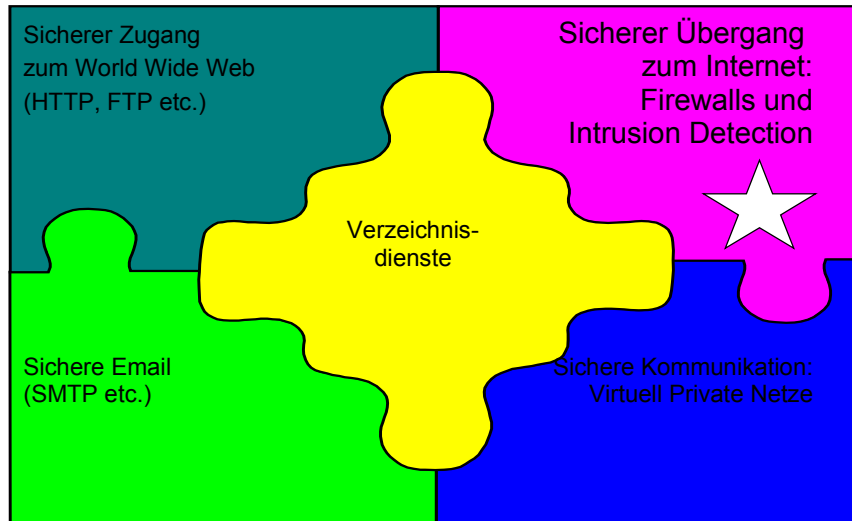
- Scanner im HTTP-Strom
  - unterbricht Zugriff bei
    - Erkennung Virenmuster
    - Zugriff auf bestimmte Seiten
  - Problem:
    - Gesamter Strom muß gefiltert werden (auch HTML)
- besser wäre:  
Proxy „präsentiert“ dem Scanner nur Wichtiges....

## Exkurs Internet Content Adaptation Protocol

- ICAP-Server nur für „bestimmten“ Content registriert
  - somit: Kein Durchschleusen des gesamten HTTP-Stroms



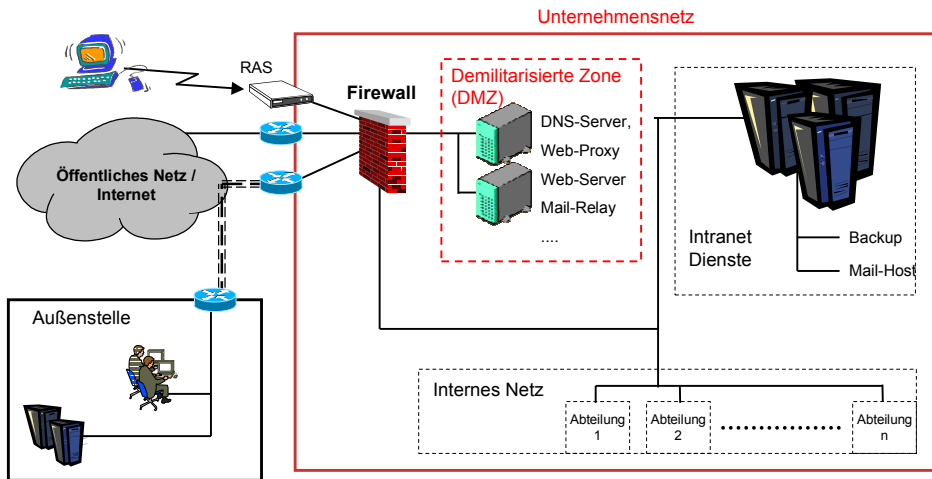
## Sicherheitsdienste für große Firmen => Teil 2: Firewalls



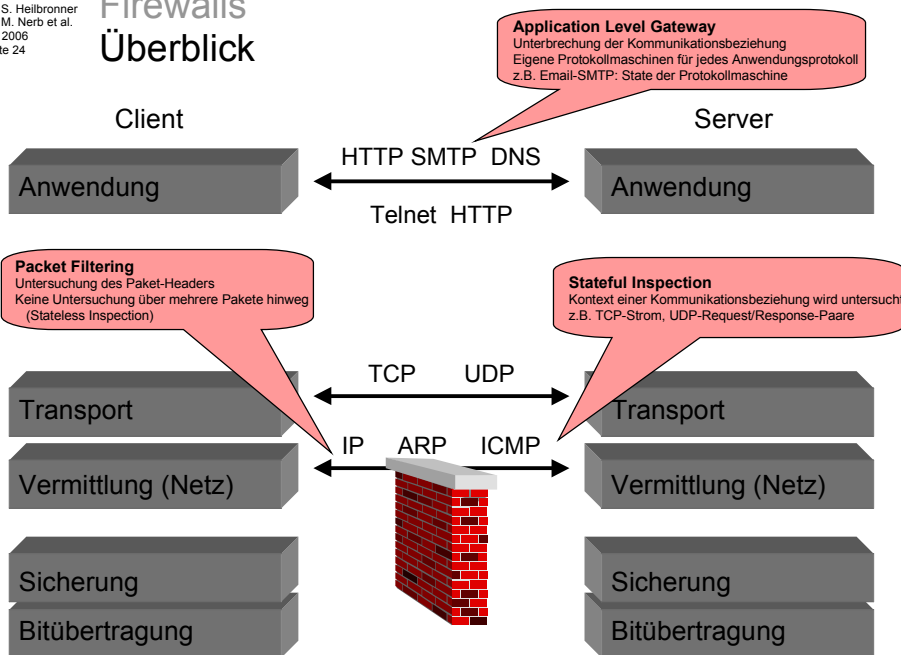
## Firewalls Einsatzzweck

- „*A Firewall helps you to keep **unauthorized** users from accessing your network **resources**.* „
  - Zugriffsrechteverwaltung für Kommunikationsbeziehungen (*Access Control Policy*)
- Grundprinzip:
  - Alles ist (zunächst) prinzipiell gesperrt.
  - Kommunikationsbeziehungen werden einzeln erlaubt.
- => ALLE Bereiche des Netzzugangs werden tangiert!
- Festlegung der Konfiguration in großen IT-Infrastrukturen
  - Iterativer Prozeß in Abstimmung mit vielen Beteiligten
  - Unterliegt ständigem „Change Management“
  - Umgehung durch Tunnelling vermeiden

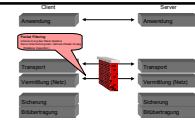
## Internet-Übergang Architektur



## Firewalls Überblick

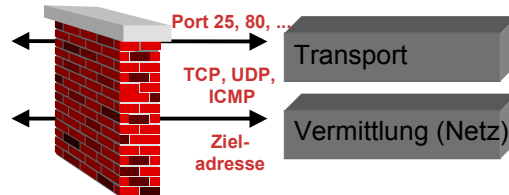


## Firewalls Packet Filtering

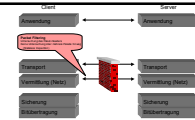


### ■ Filterung erfolgt nach Bitmustern im Paket-Header, z.B.

- IP-Absenderadresse, IP-Zieladresse
- Protokolltyp: TCP / UDP / ICMP ....
- Portnummern als Indiz für Dienst, z.B.
  - Port 80/TCP und 44s/TCP für HTTP /HTTPS
  - Port 25/TCP für SMTP (Email)
  - Port 22/TCP für SSH
  - Port 53/UDP für DNS



## Packet Filtering Bewertung



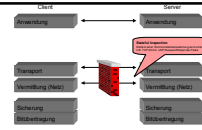
### ■ Vorteile

- transparent, keine spezielle Anpassung im Netzwerk nötig
- flexibel, jedes gängige Client/Server-Protokoll wird unterstützt
- geringe Kosten
- hoher Durchsatz
- in Routern hoch-performant implementierbar

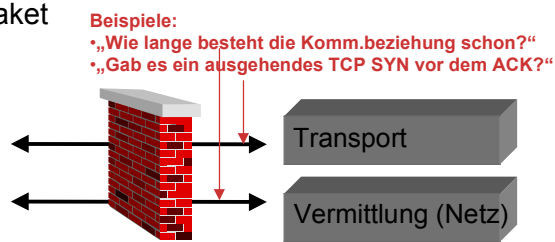
### ■ Nachteile

- Regelsätze starr und schwer zu verwalten
- unzureichende Authentifizierung (IP-Adresse nicht verifizierbar)
- Gefälschte Information in Anwendungsprotokollen (z.B. Mail-Header) können in das interne Netz gelangen.
- Logging und Accounting in Routern nicht üblich

# Stateful Inspection Überblick



- Auch: „Stateful Inspection, Smart Filtering, Adaptive Screening“
- Zustände der „Verbindungen“ werden analysiert, z.B.
  - Verbindungsauf- und abbau
  - Dauer der Verbindung
- Verwendung zusätzlich zum Packet Filtering
- Dynamische Reaktion des Filters wird realisiert, z.B.:
  - Datenpakete werden nur für etablierte Verbindung akzeptiert.
  - Ausgehendes UDP-Paket öffnet ein Zeitfenster für nachfolgende Antwortpakete.
- Beste „einfache“ Lösung



## Aufbau TCP-Verb.

**K -> <capture> - Ethereal**

No.	Time	Source	Destination	Protocol	Info
1	0.000000	leia.nuclab.de	212.184.6.57	TCP	4571 > http [SYN] Seq=4180447360 Ack=0 Min=32120 Len=0
2	0.046438	212.184.6.57	leia.nuclab.de	TCP	http > 4571 [SYN, ACK] Seq=3393461787 Ack=4180447361 W:
3	0.046496	leia.nuclab.de	212.184.6.57	TCP	4571 > http [ACK] Seq=4180447361 Ack=3393461788 Min=32:
4	0.048765	leia.nuclab.de	212.184.6.57	HTTP	GET / HTTP/1.0
5	0.156148	212.184.6.57	leia.nuclab.de	TCP	http > 4571 [ACK] Seq=3393461788 Ack=4180447787 Min=10:
6	0.321248	212.184.6.57	leia.nuclab.de	HTTP	HTTP/1.1 200 OK
7	0.321289	leia.nuclab.de	212.184.6.57	TCP	4571 > http [ACK] Seq=4180447787 Ack=3393462796 Min=32:

Frame 1 (74 on wire, 74 captured)

- [-] Ethernet II
- [-] Internet Protocol
  - Version: 4
  - Header length: 20 bytes
  - Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
  - Total Length: 60
  - Identification: 0x30a6
- [-] Flags: 0x04
  - Fragment offset: 0
  - Time to live: 64
  - Protocol: TCP (0x06)
  - Header checksum: 0x2ba4 (correct)
  - Source: leia.nuclab.de (62.157.196.227)
  - Destination: 212.184.6.57 (212.184.6.57)
- [-] Transmission Control Protocol, Src Port: 4571 (4571), Dst Port: http (80), Seq: 4180447360, Ack: 0, Source port: 4571 (4571)
  - Destination port: http (80)
  - Sequence number: 4180447360
  - Header length: 40 bytes
  - [-] Flags: 0x0002 (SYN)
  - Window size: 32120

0010 00 3c 30 a6 40 00 40 06 2b a4 3e 9d c4 e3 d4 b8 .<@\_@\_+>.....  
 0020 06 39 11 db 00 f9 2c 90 80 00 00 00 00 a0 02 .9..@.....  
 0030 7d 78 26 5d 00 00 02 04 05 b4 04 02 08 0a 00 f5 }x].....  
 0040 28 f2 00 00 00 01 03 05 00 (.....

Filter: [ ] [Reset] Destination Port (tcp.dstport)

**Aufbau  
 TCP-Verb.  
 HTTP-  
 Anfrage**

Wireshark capture showing a GET request. The packet list table is as follows:

No.	Time	Source	Destination	Protocol	Info
1	0.000000	192.168.1.100	192.168.1.1	TCP	4571 > http [SYN] Seq=4180447360 Ack=0 Min=32120 Len=0
2	0.046438	192.168.1.1	192.168.1.100	TCP	http > 4571 [SYN, ACK] Seq=3393461787 Ack=4180447361 Win=32
3	0.046496	192.168.1.100	192.168.1.1	TCP	4571 > http [ACK] Seq=4180447361 Ack=3393461788 Win=32
4	0.048765	192.168.1.100	192.168.1.1	HTTP	GET / HTTP/1.0
5	0.156148	192.168.1.1	192.168.1.100	TCP	http > 4571 [ACK] Seq=3393461788 Ack=4180447787 Min=10:
6	0.321248	192.168.1.100	192.168.1.1	HTTP	HTTP/1.1 200 OK
7	0.321289	192.168.1.100	192.168.1.1	TCP	4571 > http [ACK] Seq=4180447787 Ack=3393462796 Win=32:

The packet details for the selected packet (No. 4) are:

- Transmission Control Protocol, Src Port: 4571 (4571), Dst Port: http (80), Seq: 4180447361, Ack: 3393461788
- Source port: 4571 (4571)
- Destination port: http (80)
- Sequence number: 4180447361
- Next sequence number: 4180447787
- Acknowledgement number: 3393461788
- Header length: 32 bytes
- Flags: 0x0018 (PSH, ACK)
- Window size: 32120
- Checksum: 0xa905 (correct)
- Options: (12 bytes)
- Hypertext Transfer Protocol
  - GET / HTTP/1.0/vr\n
  - User-Agent: Mozilla/5.0 (X11; U; Linux 2.2.18 i686; en-US; rv:0.8.1+) Gecko/20010422/vr\n
  - Accept: \*/\*/vr\n
  - Accept-Language: en, de; q=0.500/vr\n
  - Accept-Encoding: gzip, deflate, compress, identity/vr\n
  - Accept-Charset: ISO-8859-1, utf-8; q=0.667, \*/; q=0.667/vr\n
  - Via: 1.1 192.168.1.1:3128 (Squid/2.3.STABLE4-hno.CVS)/vr\n

**Aufbau  
 TCP-Verb.  
 HTTP-  
 Anfrage  
 Antwort**

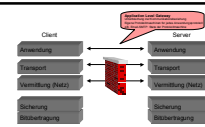
Wireshark capture showing a 200 OK response. The packet list table is as follows:

No.	Time	Source	Destination	Protocol	Info
1	0.000000	192.168.1.100	192.168.1.1	TCP	4571 > http [SYN] Seq=4180447360 Ack=0 Min=32120 Len=0
2	0.046438	192.168.1.1	192.168.1.100	TCP	http > 4571 [SYN, ACK] Seq=3393461787 Ack=4180447361 Win=32
3	0.046496	192.168.1.100	192.168.1.1	TCP	4571 > http [ACK] Seq=4180447361 Ack=3393461788 Win=32
4	0.048765	192.168.1.100	192.168.1.1	HTTP	GET / HTTP/1.0
5	0.156148	192.168.1.1	192.168.1.100	TCP	http > 4571 [ACK] Seq=3393461788 Ack=4180447787 Min=10:
6	0.321248	192.168.1.1	192.168.1.100	HTTP	HTTP/1.1 200 OK
7	0.321289	192.168.1.100	192.168.1.1	TCP	4571 > http [ACK] Seq=4180447787 Ack=3393462796 Win=32:

The packet details for the selected packet (No. 6) are:

- Transmission Control Protocol, Src Port: http (80), Dst Port: 4571 (4571), Seq: 3393461788, Ack: 4180447787
- Source port: http (80)
- Destination port: 4571 (4571)
- Sequence number: 3393461788
- Next sequence number: 3393462796
- Acknowledgement number: 4180447787
- Header length: 32 bytes
- Flags: 0x0018 (PSH, ACK)
- Window size: 10136
- Checksum: 0xc5c7 (correct)
- Options: (12 bytes)
- Hypertext Transfer Protocol
  - HTTP/1.1 200 OK/vr\n
  - Server: Netscape-Enterprise/4.0/vr\n
  - Date: Mon, 30 Apr 2001 14:38:36 GMT/vr\n
  - Content-type: text/html/vr\n
  - Connection: close/vr\n
  - vr\n
  - Data (875 bytes)

## Application Level Gateways Überblick



- Eigene Protokollinstanz für jedes Anwendungsprotokoll
- Typische Manipulationen und Prüfungen
  - Entscheidung, ob Protokollschritte ausgeführt bzw. Daten übertragen werden.
    - Ist dieser Ablauf (Schritte/Inhalte) inhaltlich zulässig?
    - „Wird Email mit diesem Inhalt von diesem Mail-Relay akzeptiert?“
  - Einhaltung des Protokolls
    - Hat sich die andere Protokollinstanz „korrekt“ verhalten?
  - Port-Umsetzung
  - Anonymisierung des Verkehrs

## Vergleich

### Paketfilter

- i.a. nur Daten der Vermittlungs-/Transport-Protokolle werden geprüft und gefiltert
- hohe Performanz, da nicht bis Anwendungsebene geprüft wird
- Regeln werden statisch definiert
- niedrigeres Sicherheitsniveau

### Application Level Gateway

- Anwendungsdaten und -Protokolle werden geprüft und gefiltert
- geringere Performanz, da aufwendige und tiefgreifende Prüfung und Filterung
- Regelwerk kann dynamisch und flexibel angepasst werden
- hohes Sicherheitsniveau



## Rückblick auf Firewalls

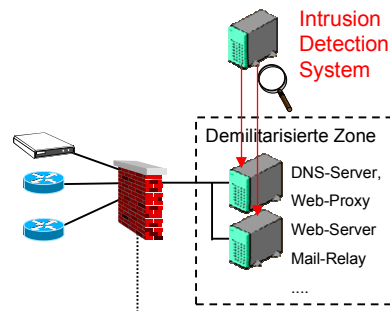
- Trennung und Filterung des internen / externen Netzverkehrs
- Vorteile heutiger Firewalls (bzw. deren Implementierung)
  - einfache Regelung des Netzwerkverkehrs
  - Unterstützung und Prüfung aller wichtigen Protokolle: IP, UDP, TCP, Anwendungsprotokolle
  - Verbergen der internen Netzstruktur (durch „NAT“)
- Nachteile von Firewalls in großen Netzen
  - Regelwerk schnell unübersichtlich
  - häufige Konfigurationsänderungen notwendig
    - Komplexes Change Management
  - bei grossem Netzwerk potentieller Engpaß

## Intrusion Detection System (IDS) Überblick

- Funktion
    - beobachten böswillige Aktivitäten (*malicious activities*)
    - informieren über Aktivitäten (Alarm)
    - initiieren ggf Gegenmaßnahmen (Response)
- Analogie: „Alarmanlage“*
- Räume und Flure werden mit Bewegungsmelder ausgestattet.
  - Fensterscheiben werden auf Druck, Schlag und Risse geprüft
- Typische Bestandteile
    - Agent (auf Host) - Host-basierte ID (1)
    - Sensor (für Netz) - Netz-basierte ID (2)
    - Managementkonsole

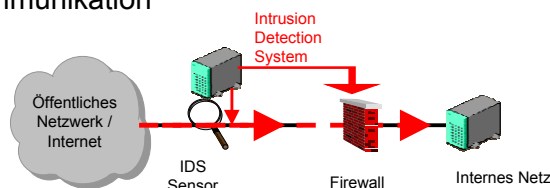
## Host-basierte IDS Prinzip

- Software AUF dem überwachten Host
- Prüfung von:
  - Veränderung von Konfigurations- und Programmdateien
    - Berechnung von Hash-Prüfsummen über Dateien
  - Netzwerkaktivität (Port-Zugriffe)
  - Auswertung von Log-Dateien, Benutzer- und Prozessverhaltens
- Typische Maßnahmen:
  - Alarm an Managementkonsole
  - Sperrung von Diensten oder Benutzer-Accounts
  - „Port-Banner“ simulieren (Unterbrechung TCP-Strom)



## Netz-basiertes IDS Überblick

- IDS Sensor ist **Software auf unabhängigem Host** („Paket-Sniffer“)
- „Intelligente“ Kopplung der Regelwerke von IDS + Firewall
- Prüfung und Analyse der:
  - Datenströme zwischen einzelnen Rechnern / Netzsegmenten
  - Netzlast innerhalb des geprüften Bereichs
- Gegenmaßnahmen (ähnlich host-basierter ID)
  - Alarm an Managementkonsole
  - Terminierung von Verbindungen
  - Aufzeichnen der Kommunikation
  - Situative Änderung der Firewall-Regeln



## Intrusion Detection System Zusammenfassung

- Kontrolle der Hosts und der Netzlast
  - Nutzung zur Erkennung von Angriffen
  - Kein Ersatz für andere Sicherheitsverfahren
- Intelligente“ Kopplung der Regelwerke von IDS + Firewall zur dynamischen Regelanpassung
- Einsetzbar im internen und im externen Netz
- Nachteile:
  - Kopplung mit Firewall oder automatische Gegenmaßnahmen bedürfen der sorgfältigen Analyse ...
  - Hoher Konfigurationsaufwand für Pflege „erlaubter“ Vorgänge

## Das wärs für heute ...

- Fragen / Diskussion
- Verbesserungsvorschläge
- Die Folien von heute kommen auf die Web-Seite der Vorlesung (zusammen mit einigen URLs).
  
- Einen schönen Abend !!!