Lehrstuhl für Netzarchitekturen und Netzdienste
Fakultät für Informatik
Technische Universität München

Grundlagen Rechnernetze und Verteilte Systeme

IN0010, SoSe 2019

Übungsblatt 12

22. Juli – 26. Juli 2019

Hinweis: Mit * gekennzeichnete Teilaufgaben sind ohne Lösung vorhergehender Teilaufgaben lösbar.

Aufgabe 1 Domain Name System (DNS)

Hinweis: Angelehnt an Endterm 2015

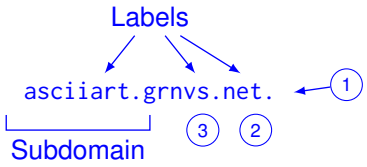
Zentrale Aufgabe des Domain Name Systems (DNS) ist es, menschenlesbare Namen auf IP-Adressen abzubilden, die dann für die Wegwahl auf der Netzwerkschicht verwendet werden können. Bei dem Namen `asciiart.grnvs.net.` handelt es sich um einen sog. *Fully Qualified Domain Name (FQDN)*.

a)* Was ist der Unterschied zwischen einem vollqualifizierten Domain Name (FQDN) und einem nicht-(voll)qualifizierten?

Ein FQDN endet stets mit `.`, d. h. der Wurzel des Name Spaces. Ein nicht-qualifizierter Domain Name hingegen kann ein einzelnes Label oder eine geordnete Liste durch Punkte getrennter Labels sein, die relativ zu einer anderen Wurzel als `.` zu sehen sind.

b)* Benennen Sie die einzelnen Bestandteile des FQDNs, sofern es dafür gängige Bezeichnungen gibt.

1. Root (Beginn des Namensraums)
2. Top Level Domain (TLD)
3. Second Level Domain



Da im Alltag zumeist nicht explizit zwischen einem „FQDN“ (also mit terminierenden Punkt) und „Domain Name“ (also ohne terminierenden Punkt) unterschieden wird, da es kontextabhängig klar ist, was von beiden gerade gemeint ist, werden wir[1] im Folgenden auch nur noch dann den Root-Punkt setzen, wenn wir dies besonders hervorheben bzw. deutlich machen wollen.

In Abbildung 1 sind ein PC sowie eine Reihe von Servern dargestellt. Wir nehmen an, dass PC1 den Router als Resolver nutzt. Der Router wiederrum nutzt einen Resolver von Google unter der IP-Adresse 8.8.8.8 zur Namensauflösung. Ferner nehmen wir an, dass der Google-Resolver gerade neu gestartet wurde (also insbesondere keine Resource Records gecached hat) und rekursive Namensauflösung anbietet.

Die autoritativen Nameserver für die jeweiligen Zonen sind in Tabelle 1 gegeben.

Zone	autoritativer Nameserver
`.`	`d.root-servers.net.`
`com.`, `net.`	`a.gtld-servers.net.`
`google.com.`	`ns1.google.com.`
`grnvs.net.`	`bifrost.grnvs.net.`

Tabelle 1: Zonen mit zugehörigen autoritativen Nameservern

c)* Erläutern Sie den Unterschied zwischen einem *Resolver* und einem *Nameserver*.

Nameserver sind autoritativ für eine oder mehrere Zonen („Bereiche“), d. h. sie besitzen eine gültige und aktuelle Kopie der gesamten Zone, für die sie autoritativ sind.

[1] for the sake of notational brevity

Prof. Dr.-Ing. Georg Carle carle@tum.de
Dr.-Ing. Stephan Günther, Henning Stubbe grnvs@net.in.tum.de

Resolver hingegen extrahieren mittels eine Reihe iterativer Anfragen an die jeweils autoritativen Nameserver die benötigten Information aus dem DNS und geben diese an den anfragenden Client zurück. Resolver können Einträge für begrenzte Zeit cachen, so dass bei erneuter Anfrage derselben Resource Records der Prozess nicht wiederholt werden muss.

d)* Welche Funktion erfüllen `d.root-servers.net` und `a.gtld-servers.net`?

Der Root-Nameserver ist autoritativ für die Rootzone, d. h. er kennt die Nameserver, welche für die einzelnen TLDs verantwortlich sind, so z. B. `a.gtld-servers.net` als einen der autoritativen Nameserver für net-Domains.

`a.gtld-servers.net` kennt wiederum die zuständigen Nameserver für alle Second-Level-Domains unterhalb der net-TLD.

e)* Erklären Sie den Unterschied zwischen iterativer und rekursiver Namensauflösung.

Rekursive Namensauflösung bedeutet, dass eine DNS-Anfrage an einen Resolver gestellt wird. Dieser wird das endgültige Ergebnis zurücksenden.

Bei iterativer Auflösung hingegen werden schrittweise die autoritativen Nameserver der einzelnen Zonen angefragt.

f) Zeichnen Sie in Abbildung 1 alle DNS-Nachrichten (Requests / Responses) ein, die ausgetauscht werden, sobald PC1 auf `asciiart.grnvs.net.` zugreift. Nummerieren Sie die Nachrichten gemäß der Reihenfolge, in der sie zwischen den einzelnen Knoten ausgetauscht werden.

s. Abbildung 1.

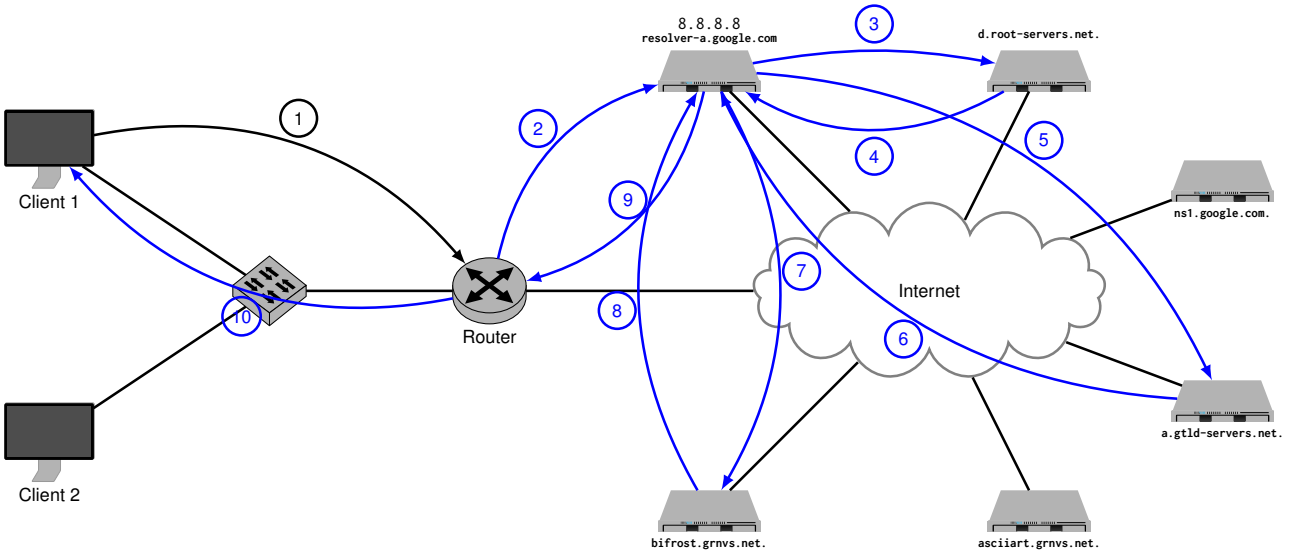


Abbildung 1: Vorlage zu Aufgabe 1f)

g)* Wie wird im DNS sichergestellt, dass kein bösartiger Nameserver Anfragen für andere Domänen beantwortet? (Wir gehen davon aus, dass keine Man-in-the-Middle-Angriffe möglich sind.)

Dies wird lediglich indirekt dadurch sichergestellt, dass während der iterativen Namensauflösung stets nur die jeweils autoritativen Nameserver kontaktiert werden. Sofern die

- Antwort des Rootservers zuverlässig war und
- die Antwort auf dem Weg vom Rootserver zum anfragenden Nameserver nicht modifiziert wurde

kann ein bösartiger Nameserver keine falschen Antworten liefern – eben da er nie gefragt wird.

Selbstverständlich wird auf diese Weise nicht verhindert, dass DNS-Antworten mittels Man-in-the-Middle-Attacken abgefangen und modifiziert werden können. Dagegen helfen lediglich kryptographische Verfahren, wie sie in der DNSSEC-Erweiterung zu finden sind (nicht in der Vorlesung behandelt).

Aufgabe 2 All in a nutshell

In dieser Aufgabe wollen wir noch einmal alles nachvollziehen, was geschieht, wenn Sie auf Ihrem Computer die Webseite `www.google.de` aufrufen. Wir treffen dabei lediglich die Annahme, dass in Ihrem privaten Netz ARP- und DNS-Caches noch leer sind (d. h. etwaige Caches ab dem ersten Router können als gefüllt angenommen werden). Die Netzwerktopologie ist in Abbildung 2 dargestellt. Ihr Router übersetzt bei Bedarf private in öffentliche IP-Adressen sowie Portnummern (NAT). Auf Ihrem Computer sei der Google-Resolver mit der IPv4-Adresse 8.8.8.8 konfiguriert, der rekursive Anfragen erlaubt.

Es sollen nun für **jeden Link** – also jeden Abschnitt zwischen jeweils zwei Geräten (z. B. von PC zu SW) – einige ausgewählte Felder der Nachrichten notiert werden, die im jeweiligen Schritt über diesen Link versendet werden. Da dies insbesondere für MAC-Adressen etwas Schreibarbeit ist, kürzen wir Adressen mit der Bezeichnung `<Gerätename>.<Interface>.<Typ>` ab, z. B. stehe `RA.eth0.MAC` für die MAC-Adresse von Interface `eth0` an Router `RA`.

Sie finden in den Abbildungen 3 – 5 vorgedruckte Tabellen. Eine Zeile entspricht dabei einer Nachricht, die über den jeweiligen Link gesendet wird. Die erste Spalte bezeichnet den Link, also z. B. vom PC zum Switch oder vom Switch zum Router. Die übrigen Spalten entsprechen verschiedenen Schichten des ISO/OSI-Modells. Diese sind jeweils in die relevanten Headerfelder der üblicherweise verwendeten Protokolle unterteilt. Je nach Nachricht sind nicht alle Spalten oder Unterzeilen pro Spalte auszufüllen. **Streichen Sie deutlich nicht benötigte Felder.** Ein Beispiel ist bereits in der Tabelle eingetragen.

Einige Header verfügen über ein Protokoll-Feld, in dem das Protokoll der nächsthöheren Schicht angegeben wird. Üblicherweise stehen Zahlencodes für die jeweiligen Protokolle. Es ist **nicht** notwendig, diese Zahlencodes anzugeben. Stattdessen reicht es, das verwendete Protokoll anzugeben, z. B. IPv4, TCP oder UDP. Bei einigen Header-Feldern gibt es gewisse Freiheiten, z. B. bei Portnummern oder der initialen TTL. Wählen Sie in diesen Fällen **sinnvolle** Werte.

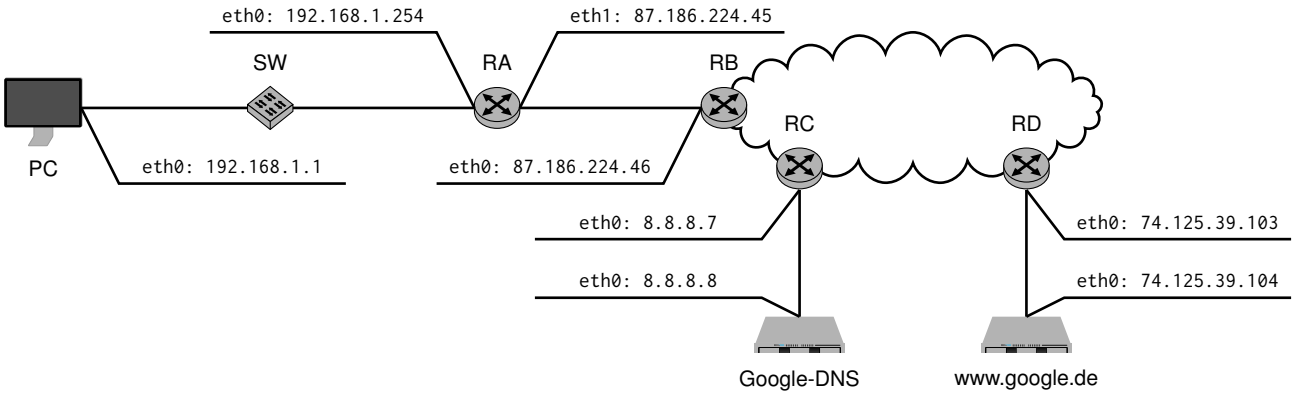


Abbildung 2: Netztopologie zu Aufgabe 2. Die relevanten Links sind mit den Ziffern 1 – 5 gekennzeichnet.

a)* Füllen Sie nun die Vordrucke in den Abbildungen 3 – 5 aus. Brechen Sie **nach dem ersten** an `www.google.de` übermittelten Paket auf dem Link von PC an SW ab.

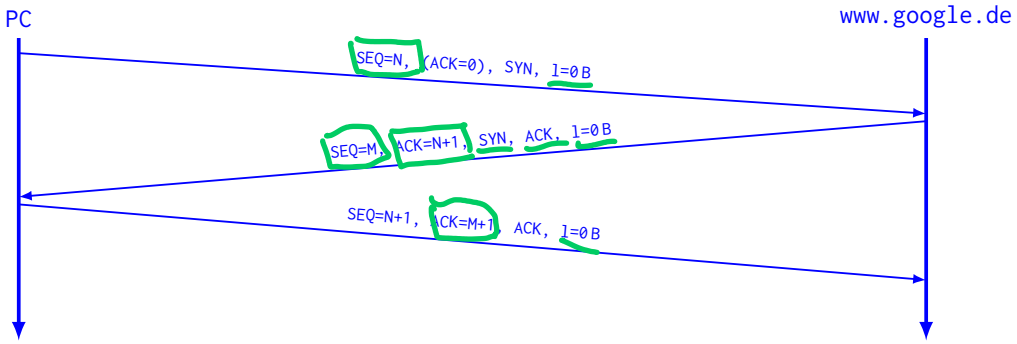
Hinweise:

- Der Well-Known Port für DNS ist UDP 53.
- Wir nehmen an, dass sich zwischen Router RB und RC insgesamt 10 weitere Router befinden. Dies ist für die Bestimmung der TTL entscheidend.
- In die Spalte „Schicht 7“ tragen sie einfach das Anwendungsprotokoll, ggf. den Typ der Nachricht (z. B. Request / Reply) sowie stichpunktartig den Inhalt der übermittelten Nachricht ein (z. B. „DNS-Request“ oder „DNS-Response“).

Siehe Abbildungen 3 – 5.

Die vorangegangene Teilaufgabe hat detailliert die Vorgänge bis zum Beginn des TCP-Verbindungsaufbaus dargestellt. Im Folgenden wollen wir uns auf die TCP-Verbindung und Datenübertragung konzentrieren. Aus diesem Grund betrachten wir ab jetzt nur noch die logische Verbindung zwischen dem PC und `www.google.de` in Form eines einfachen Weg-Zeit-Diagramms **ohne** die dazwischenliegenden Knoten. Sie können Serialisierungszeiten vernachlässigen. Gehen Sie außerdem davon aus, dass während der gesamten Übertragung keine Segmentverluste auftreten.

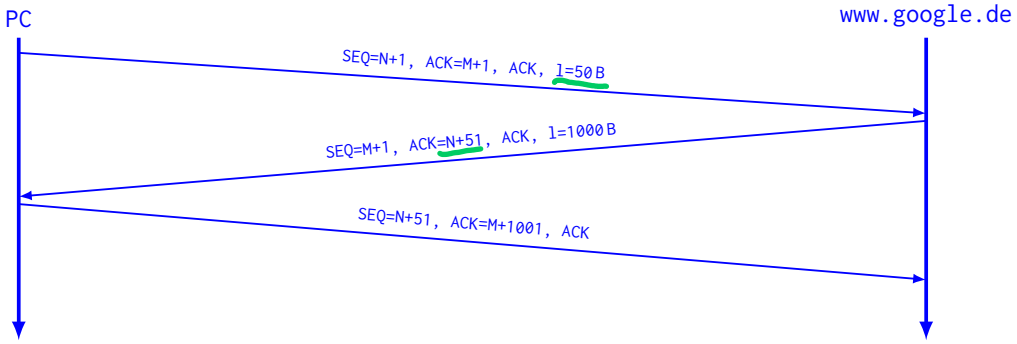
b)* Skizzieren Sie ein Weg-Zeit-Diagramm, welches den TCP-Verbindungsaufbau darstellt. Geben Sie für jedes übermittelte Segment die Sequenznummer, Bestätigungsnummer, die gesetzten[2] Flags sowie die Länge *l* der transportieren Nutzdaten an.



Die Bestätigungsnummer des ersten Segments hat keinerlei Auswirkung, da das ACK-Flag nicht gesetzt ist (was soll auch beim ersten Segment bestätigt werden?). Die initialen Sequenznummern beider Teilnehmer sind prinzipiell beliebig, d. h. $0 \leq N, M, \leq 2^{32} - 1$.

Der PC fordert nun die Webseite an, die auf `www.google.de` gehostet wird. Dazu sendet der PC eine `HTTP-GET`-Nachricht, welche aus Sicht von Schicht 4 eine Nutzdatenlänge von $l_1 = 50\,\text{B}$ habe. Der Webserver wird daraufhin die Webseite an den PC senden, welche eine Länge $l_2 = 1000\,\text{B}$ habe. Die ausgehandelte MSS[3] sei größer als l_2.

c) Skizzieren Sie ein Weg-Zeit-Diagramm, welches die TCP-Verbindungsphase darstellt. Gehen Sie von den in Teilaufgabe b) ausgehandelten Sequenznummern aus. Geben Sie für jedes übermittelte Segment die Sequenznummer, Bestätigungsnummer, die gesetzten Flags sowie die Länge *l* der im Segment transportierten Nutzdaten an.

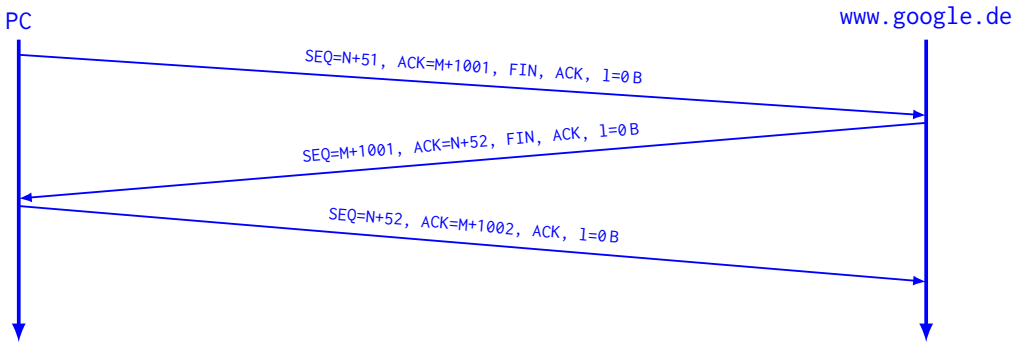


[2] Ein Bit-Flag gilt als „gesetzt“, wenn es logisch 1 ist.

[3] Die MSS (Maximum Segment Size) gibt die maximale Größe eines Segments an. Sie bezieht sich dabei lediglich auf die Nutzdaten. Bestätigungen beispielsweise sind Segmente der Länge null, welche lediglich aus einem TCP-Header bestehen.

Hinweis: Will der PC die Verbindung unmittelbar nach dem Erhalt der Nachricht abbauen, so kann das dritte Segment bereits das FIN-Flag gesetzt haben.

d) Skizzieren Sie ein Weg-Zeit-Diagramm, welches den TCP-Verbindungsabbau darstellt. Dieser werde vom PC initiiert. Gehen Sie dabei von den in Teilaufgabe c) ausgehandelten Sequenznummern aus. Geben Sie für jedes übermittelte Segment die Sequenznummer, Bestätigungsnummer, die gesetzten Flags sowie die Länge *l* der im Segment transportieren Nutzdaten an.



An dieser und den vorherigen beiden Teilaufgaben ist noch einmal zu sehen, dass TCP einzelne Bytes, nicht aber Segmente bestätigt. Segmente, welche ein SYN oder FIN Flag gesetzt haben, werden dabei wie Segmente mit genau 1 B Nutzdaten behandelt.

Der Verbindungsabbau kann auch in Form von vier statt drei Nachrichten stattfinden, d. h. `www.google.de` bestätigt zunächst den erhalt des `FIN`-Flags ohne dabei selbst das `FIN`-Flag zu setzen. Dies könnte beispielsweise dann der Fall sein, wenn zwar der `PC` keine Daten mehr an `www.google.de` zu übertragen hat, `www.google.de` aber noch unbestätigte Segmente an den `PC` hat. Derartige TCP-Verbindungen werden als „halb-offen“ bezeichnet, da eine Datenübertragung in eine der beiden Richtungen noch immer möglich ist.

Lehrstuhl für Netzarchitekturen und Netzdienste
Fakultät für Informatik
Technische Universität München

Link		Schicht 2		Schicht 3		Schicht 4		Schicht 7
From	PC	Src	PC.eth0.MAC	Src		Src		
		Dst	ff:ff:ff:ff:ff:ff	Dst		Dst		
To	SW	Prot	ARP	Prot		Flags		
		Op	Request	TTL		SEQ		
						ACK		
From	SW	Src	PC.eth0.MAC	Src		Src		
		Dst	ff:ff:ff:ff:ff:ff	Dst		Dst		
To	RA	Prot	ARP	Prot		Flags		
		Op	Request	TTL		SEQ		
						ACK		
From	RA	Src	RA.eth0.MAC	Src		Src		
		Dst	PC.eth0.MAC	Dst		Dst		
To	SW	Prot	ARP	Prot		Flags		
		Op	Reply	TTL		SEQ		
						ACK		
From	SW	Src	RA.eth0.MAC	Src		Src		
		Dst	PC.eth0.MAC	Dst		Dst		
To	PC	Prot	ARP	Prot		Flags		
		Op	Reply	TTL		SEQ		
						ACK		
From	PC	Src	PC.eth0.MAC	Src	192.168.1.1	Src	51827	DNS Request
		Dst	RA.eth0.MAC	Dst	8.8.8.8	Dst	53	Who is www.google.de?
To	SW	Prot	IPv4	Prot	UDP	Flags		
				TTL	64	SEQ		
						ACK		

Abbildung 3: Vordruck zu Aufgabe 2

Prof. Dr.-Ing. Georg Carle
carle@tum.de

Dr.-Ing. Stephan Günther, Henning Stubbe
grnvs@net.in.tum.de

Lehrstuhl für Netzarchitekturen und Netzdienste
Fakultät für Informatik
Technische Universität München

Link		Schicht 2		Schicht 3		Schicht 4		Schicht 7
From	SW	Src	PC.eth0.MAC	Src	192.168.1.1	Src	51827	DNS Request
		Dst	RA.eth0.MAC	Dst	8.8.8.8	Dst	53	Who is www.google.de?
To	RA	Prot	IPv4	Prot	UDP	Flags		
				TTL	64	SEQ		
						ACK		
From	RA	Src	RA.eth1.MAC	Src	87.186.224.45	Src	38218	DNS Request
		Dst	RB.eth0.MAC	Dst	8.8.8.8	Dst	53	Who is www.google.de?
To	RB	Prot	IPv4	Prot	UDP	Flags		
				TTL	63	SEQ		
						ACK		
From	RC	Src	RC.eth0.MAC	Src	87.186.224.45	Src	38218	DNS Request
		Dst	DNS.eth0.MAC	Dst	8.8.8.8	Dst	53	Who is www.google.de?
To	Google DNS	Prot	IPv4	Prot	UDP	Flags		
				TTL	51	SEQ		
						ACK		
From	Google DNS	Src	DNS.eth0.MAC	Src	8.8.8.8	Src	53	DNS Reply
		Dst	RC.eth0.MAC	Dst	87.186.224.45	Dst	38218	www.google.de is
To	RC	Prot	IPv4	Prot	UDP	Flags		74.125.39.104!
				TTL	64	SEQ		
						ACK		
From	RB	Src	RB.eth0.MAC	Src	8.8.8.8	Src	53	DNS Reply
		Dst	RA.eth1.MAC	Dst	87.186.224.45	Dst	38218	www.google.de is
To	RA	Prot	IPv4	Prot	UDP	Flags		74.125.39.104!
				TTL	52	SEQ		
						ACK		

Abbildung 4: Vordruck zu Aufgabe 2

Prof. Dr.-Ing. Georg Carle
carle@tum.de

Dr.-Ing. Stephan Günther, Henning Stubbe
grnvs@net.in.tum.de

Lehrstuhl für Netzarchitekturen und Netzdienste
Fakultät für Informatik
Technische Universität München

Link		Schicht 2		Schicht 3		Schicht 4		Schicht 7
From	RA	Src	RA.eth0.MAC	Src	8.8.8.8	Src	53	DNS Reply
		Dst	PC.eth0.MAC	Dst	192.168.1.1	Dst	51827	www.google.de is
To	SW	Prot	IPv4	Prot	UDP	Flags		74.125.39.104!
				TTL	51	SEQ		
						ACK		
From	SW	Src	RA.eth0.MAC	Src	8.8.8.8	Src	53	DNS Reply
		Dst	PC.eth0.MAC	Dst	192.168.1.1	Dst	51827	www.google.de is
To	PC	Prot	IPv4	Prot	UDP	Flags		74.125.39.104!
				TTL	51	SEQ		
						ACK		
From	PC	Src	PC.eth0.MAC	Src	192.168.1.1	Src	58392	
		Dst	RA.eth0.MAC	Dst	74.125.39.104	Dst	80	
To	SW	Prot	IPv4	Prot	TCP	Flags	SYN	
				TTL	64	SEQ	0	
						ACK	0	
From		Src		Src		Src		
		Dst		Dst		Dst		
To		Prot		Prot		Flags		
				TTL		SEQ		
						ACK		
From		Src		Src		Src		
		Dst		Dst		Dst		
To		Prot		Prot		Flags		
				TTL		SEQ		
						ACK		

Abbildung 5: Vordruck zu Aufgabe 2

Prof. Dr.-Ing. Georg Carle
carle@tum.de

Dr.-Ing. Stephan Günther, Henning Stubbe
grnvs@net.in.tum.de

Aufgabe 3 SMTP (Hausaufgabe)

Für daheim: Mailversand von der Kommandozeile

Sie haben in der Vorlesung gelernt, dass mithilfe des Programms `telnet` bzw. mit dem s_client von OpenSSL Verbindungen zu Webservern aufgebaut werden können.

a) * Ihre Aufgabe ist nun, sich mithilfe des s_client mit dem SMTP-Servers Ihres Mailproviders zu verbinden und sich selbst eine e-Mail zu schicken. Die notwendigen Schritte können Sie sich aus dem RFC 5321 erschließen. Alternativ ist es auch möglich entsprechende Tutorials zu konsultieren. Der initiale Befehl könnte folgendermaßen aussehen: `openssl s_client -crlf -connect <smtp.server.org>:465`

Sollten Sie CRAM-MD5 verwenden wollen, könnte Ihnen die folgende Bash-Funktion zur Berechnung der Response auf eine vom Server gegebene Challenge helfen:

```
cram() {
  challenge=$1
  username=$2
  challenge=$(echo -n $challenge|base64 -d)
  echo "Challenge is: $challenge"
  read -sp "Password for $username: " password
  echo ""
  hash=$(echo -n "$challenge" |openssl md5 -hmac "$password" -hex|cut -d" " -f 2 )
  response=$(echo -n "$username $hash" |base64)
  echo "Response for server is: "
  echo $response
}
```

Rufen sie Sie dann folgendermaßen auf:
`cram <CHALLENGE> <USERNAME>`

220 gmx.com (mrgmx003) Nemesis ESMTP Service ready
EHLO <FQDN of localhost>
250-gmx.com Hello <FQDN of localhost> [XXX.XXX.XXX.XX]
250-SIZE 69920427
250 AUTH LOGIN PLAIN
AUTH LOGIN
334 VXNlcm5hbWU6
<output of: echo -n "<USERNAME>" | base64>
334 UGFzc3dvcmQ6
<output of: echo -n "<PASSWORD>" | base64>
235 Authentication succeeded
mail from:<YOUR AUTHENTICATED MAIL ADDRESS>
6250 Requested mail action okay, completed
rcpt to:<YOUR RECIPIENT ADDRESS>
250 OK
data
354 Start mail input; end with <CRLF>.<CRLF>
From: <SENDER>
To: <RECIPIENT>
Subject: <SUBJECT>

Hello world
.
250 Requested mail action okay, completed: id=0Mcmmn-1XHes72NnN-00Hxc8
QUIT
DONE
–
Verwendet man CRAM-MD5, nimmt man statt AUTH LOGIN
AUTH CRAM-MD5

334 <CHALLENGE>
<RESPONSE>
235 Authentication succeeded

b) Ermitteln Sie, welche Authentifizierungsmethoden der von Ihnen gewählte SMTP-Server anbietet und machen Sie sich den Unterschied zwischen ihnen klar.

- PLAIN erwartet den Usernamen und das Passwort base64-codiert in der Form “NULusernameNULpassword”.
- LOGIN erwartet zuerst den Usernamen base64-codiert und als anschließende Nachricht das Passwort, ebenfalls base64-codiert
- MD5-CRAM ist eine Challenge-Response-Authentifizierung. Der Server sendet base64-codiert einen eindeutigen String, welcher in der Form “username hmac(”challenge“, ”password“)” zurückgesendet werden muss. So wird vermieden, dass das Passwort ungehasht übertragen wird. Die übertragenen Authentifizierungsinformationen können aufgrund der stets wechselnden Challenge auch nicht für Replay-Angriffe genutzt werden.
- SCRAM-SHA-1 ist ebenfalls eine Challenge-Response-Authentifizierung. Sie hat gegenüber MD5-CRAM zum Vorteil, dass der Server während des Dialoges dem Client beweist, dass er tatsächlich im Besitz des (gehashten) Passwortes des Clients ist. Dies ist eine Maßnahme zur Erkennung von MitM-Angriffen. Zusätzlich kann der Server die Passwörter der Clients (im Gegensatz zur Verwendung von CRAM-MD5) beliebig stark gehasht und gesalted in seiner Datenbank abspeichern. Bei Dialog muss dem Client vor dem Austausch der Authentifizierungsinformationen die Hashfunktion (inkl. Rundenanzahl) und der Salt mitgeteilt werden, damit der Client den entsprechenden Hash ausrechnen und damit weiterarbeiten kann.

c) * Warum ist überhaupt eine Authentifizierung notwendig? Welche Probleme werden dadurch behoben oder zumindest verringert? Mailserver ohne Authentifizierung nehmen jegliche Versandaufträge an. Daher können sie zum SPAM-Versand missbraucht werden.