Übungen zu Einführung in Rechnernetze
7. Übung

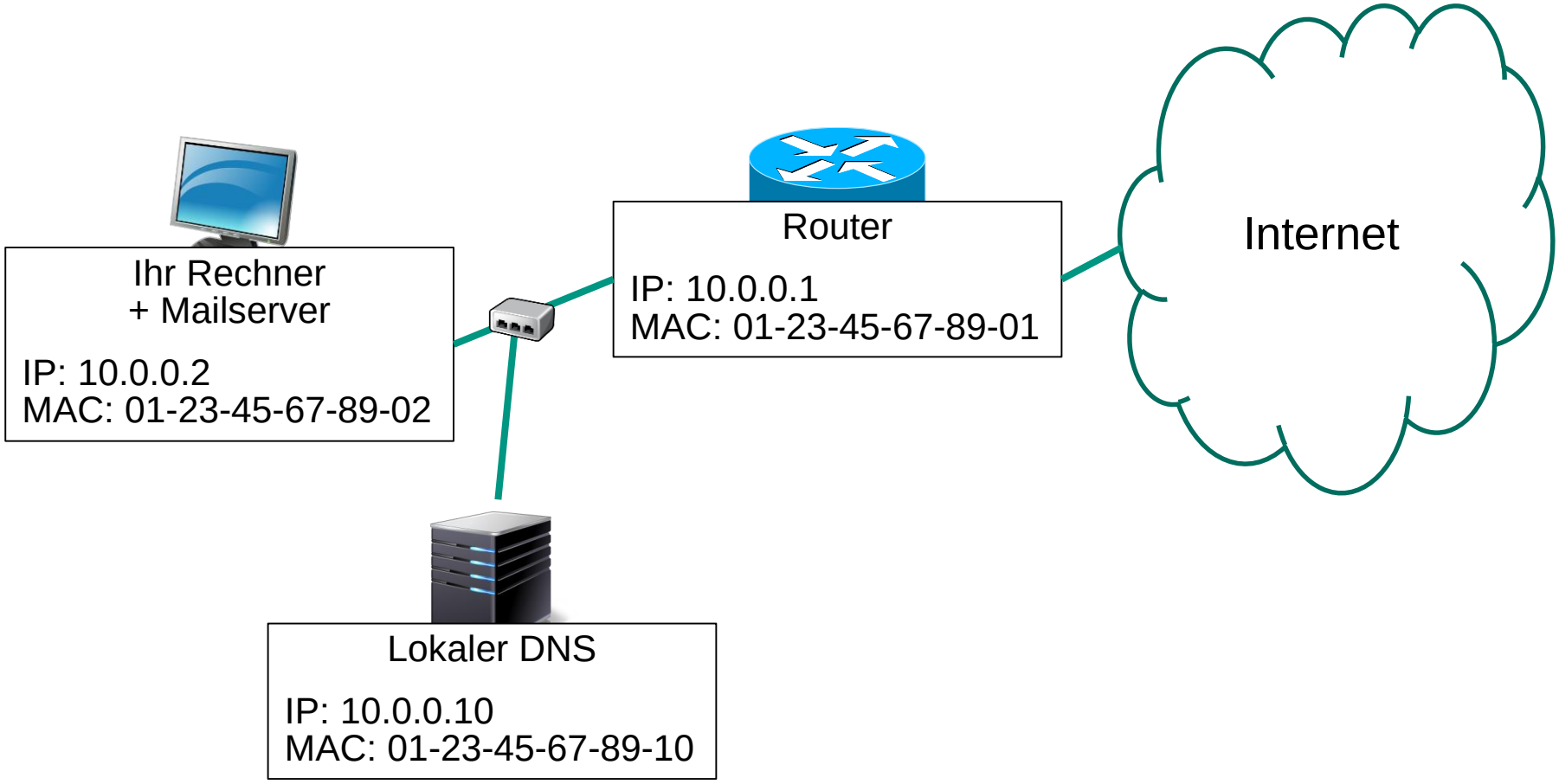
Tim Gerhard, Matthias Flittner
[tim.gerhard, flittner]@kit.edu

Institut für Telematik, Prof. Zitterbart

Einführung in Rechnernetze – 7. Übungsblatt

1. E-Mail im Zusammenspiel
2. Sicherheit
3. Web of Trust

Aufgabe 1: Schichtenübergreifendes Zusammenspiel



- Szenario: SMTP von Ihrem Rechner zu GMail

a) E-Mail-Format

```
Date: Tue, 11 Jul 2017 13:00:00 +0200
To: Alice <schickmirnichts@gmail.com>
From: Admin <admin@myawesomemailserver.com>
Subject: Hallo!

Hallo Alice,
wie geht es dir? Habe schon lange nichts mehr gehoert!
Gruesse!
```

b) ARP (Wer ist LDNS?)

- Ethernet-Hub im lokalen Netz
  - direkte Kommunikation zwischen Geräten

- Anfrage
  - Sender-MAC: 01-23-45-67-89-02
  - Sender-IP: 10.0.0.2
  - Empfänger-MAC: ff-ff-ff-ff-ff-ff
  - Empfänger-IP: 10.0.0.10

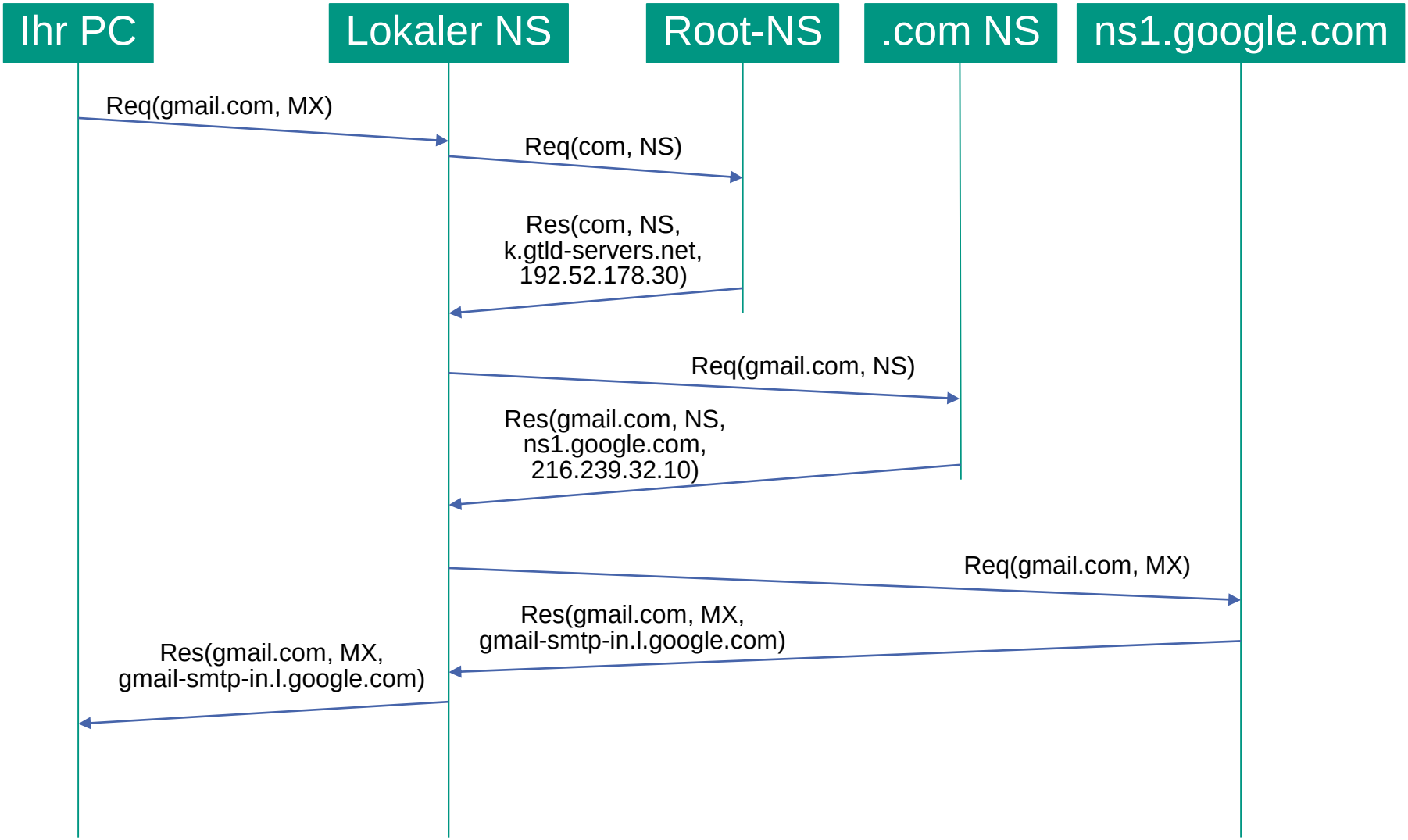
- Antwort
  - Sender-MAC: 01-23-45-67-89-10
  - Sender-IP: 10.0.0.10
  - Empfänger-MAC: 01-23-45-67-89-02
  - Empfänger-IP: 10.0.0.2

c) DNS

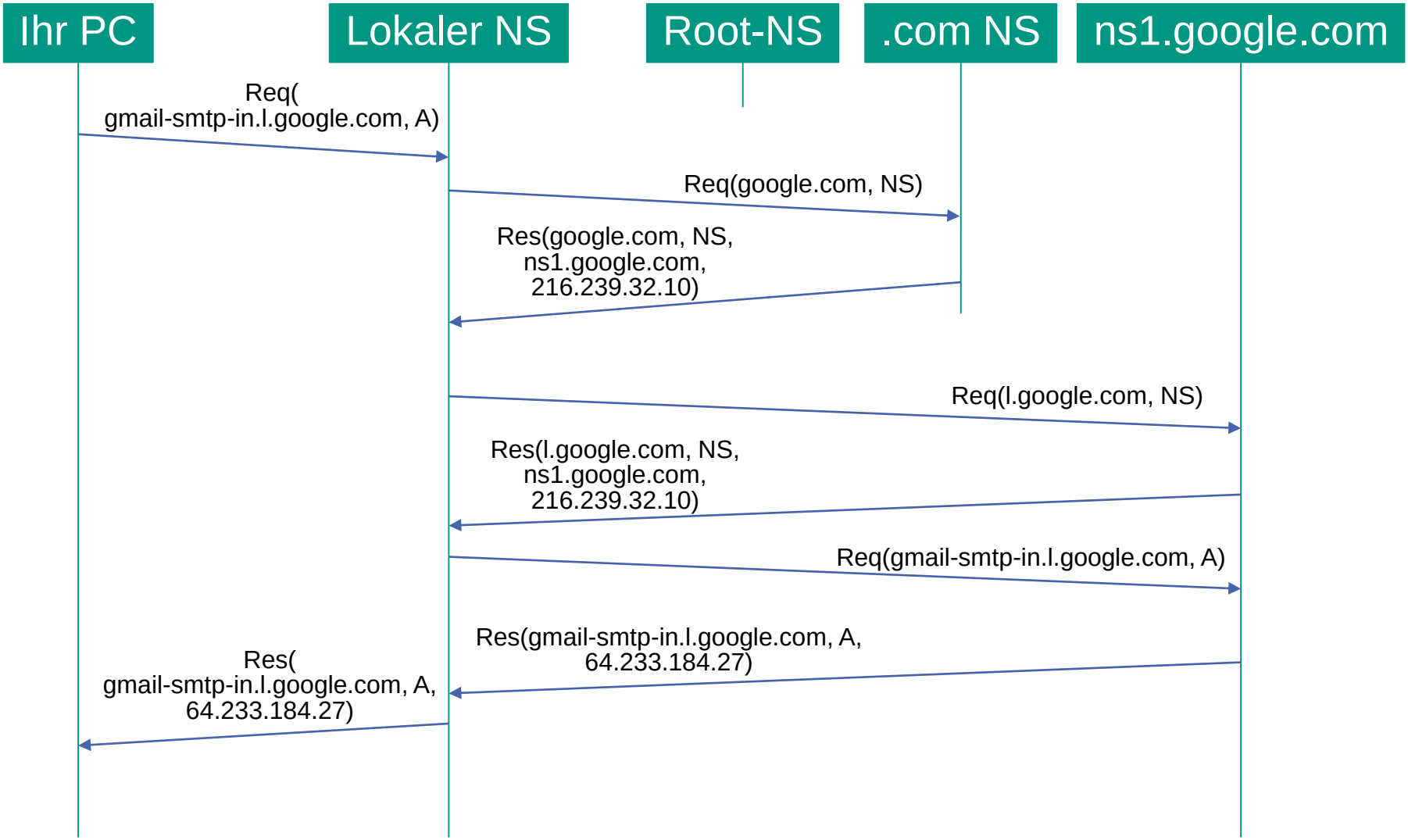
- DNS-Konfiguration von GMail

Domäne + RR		Wert
K-Root-Server Karlsruhe		193.0.14.129
com	NS	k.gtld-servers.net
gmail.com	NS	ns1.google.com
gmail.com	MX	gmail-smtp-in.l.google.com
google.com	NS	ns1.google.com
l.google.com	NS	ns1.google.com
gmail-smtp-in.l.google.com	A	64.233.184.27
ns1.google.com	A	216.239.32.10
k.gtld-servers.net	A	192.52.178.30

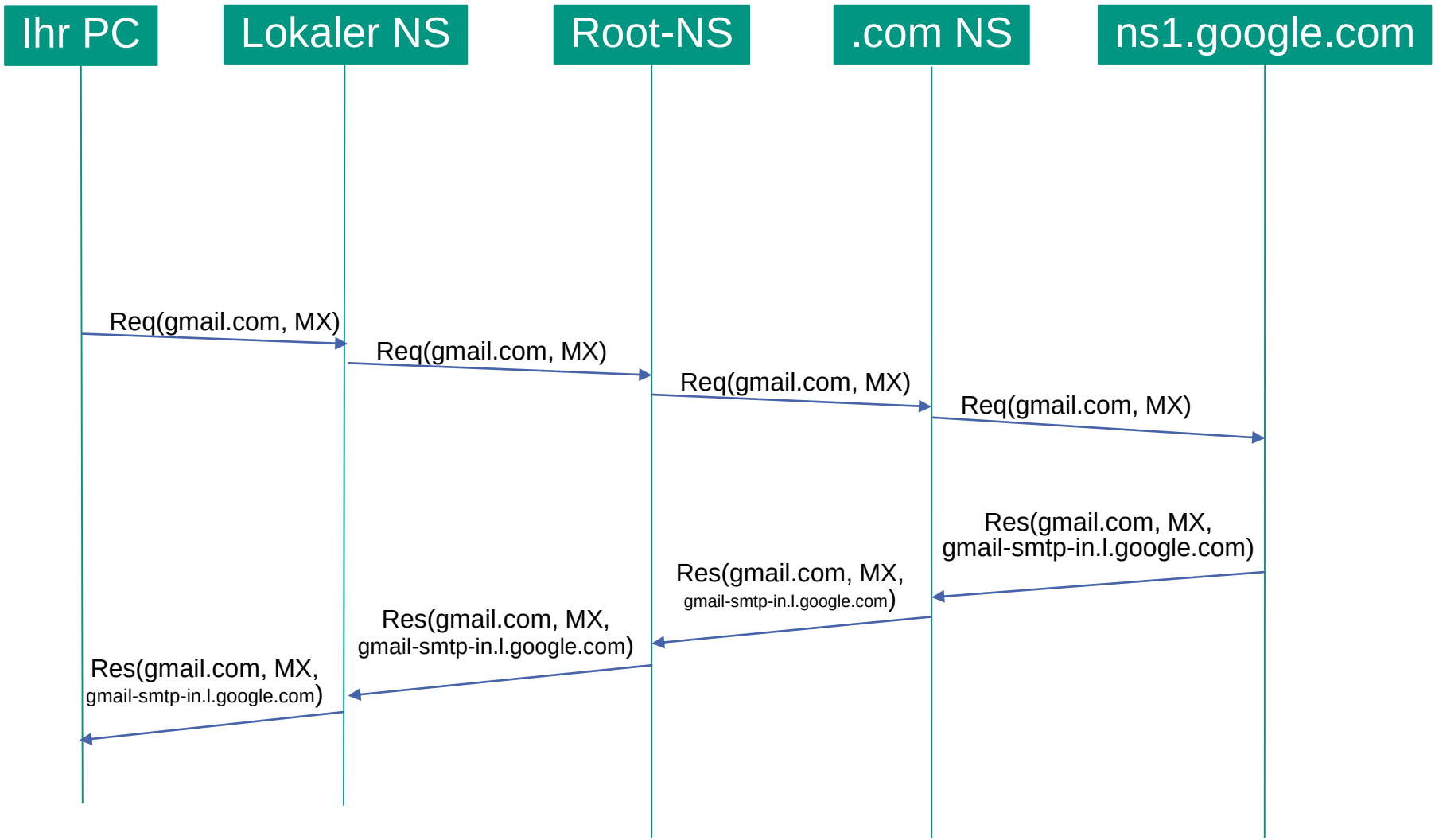
c) DNS: Iterative Anfrage - MX



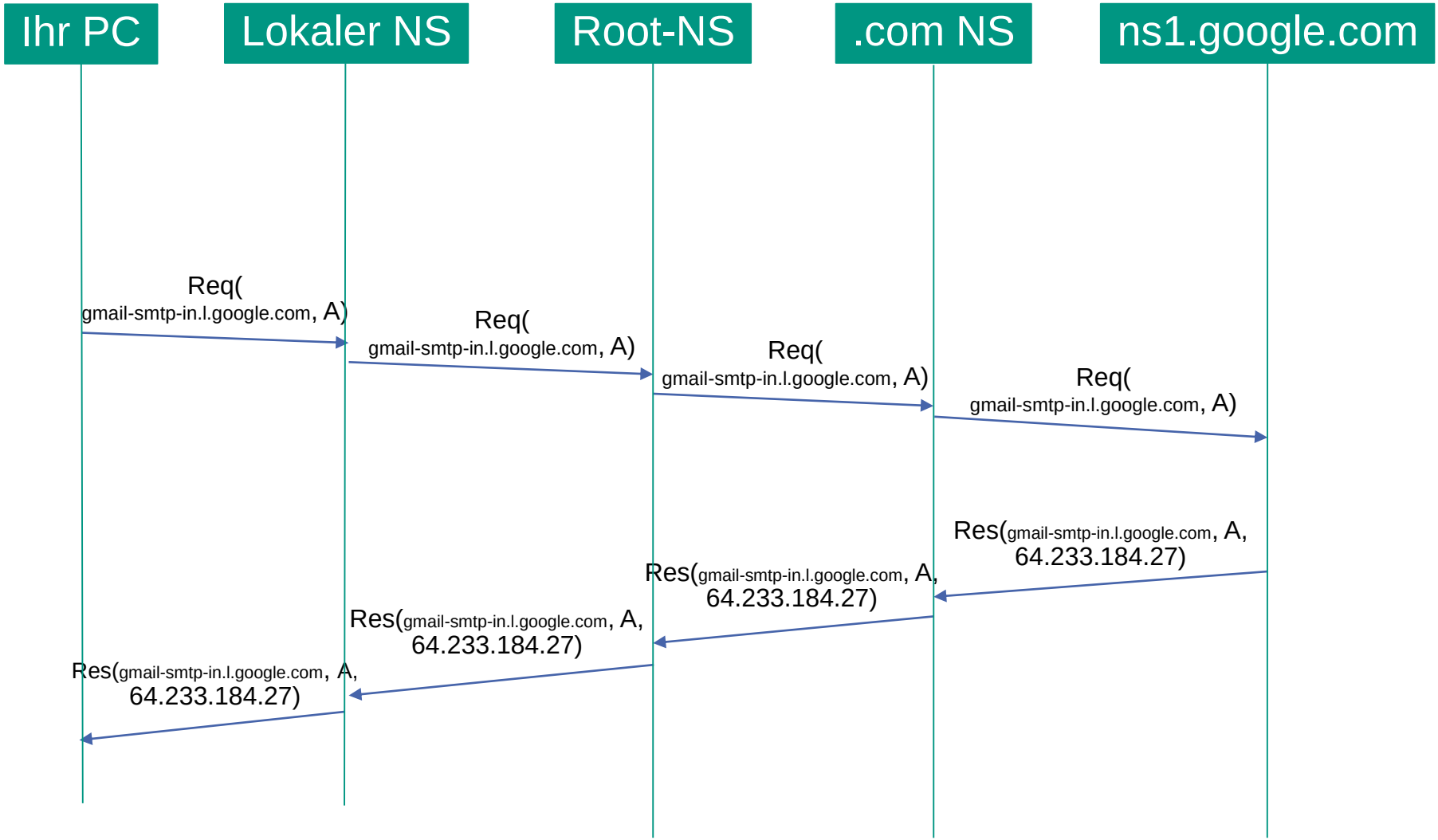
c) DNS: Iterative Anfrage - A



d) DNS: Rekursive Anfragen - MX



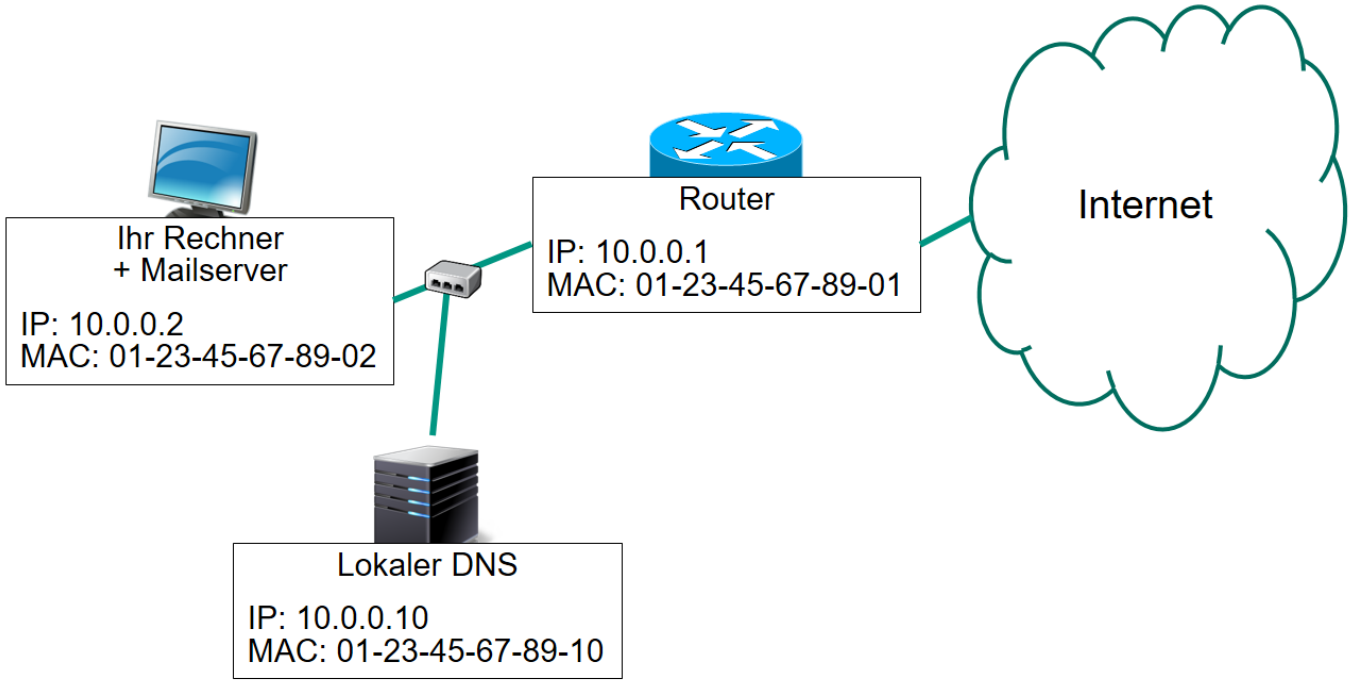
d) DNS: Rekursive Anfragen - A



e) MAC-Adressen für IP-Pakete

- IP-Pakete werden über den Router gesendet
- Ethernet-Netz endet beim Router

- Ziel-MAC-Adresse ist also die des Routers
  - Wird über ARP bestimmt.

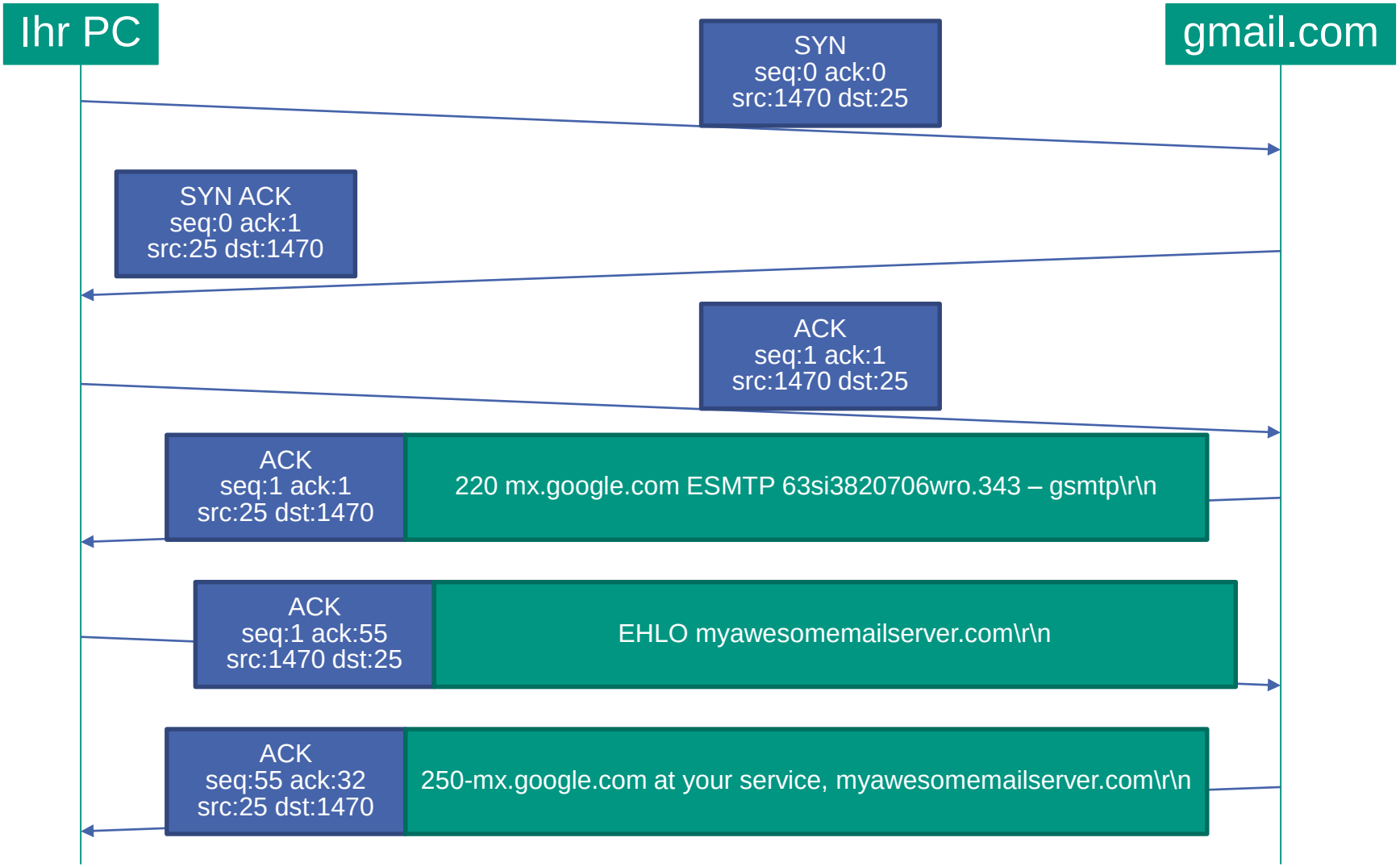


g) SMTP und TCP

- Annahme: 1 SMTP-Nachricht pro TCP-Segment

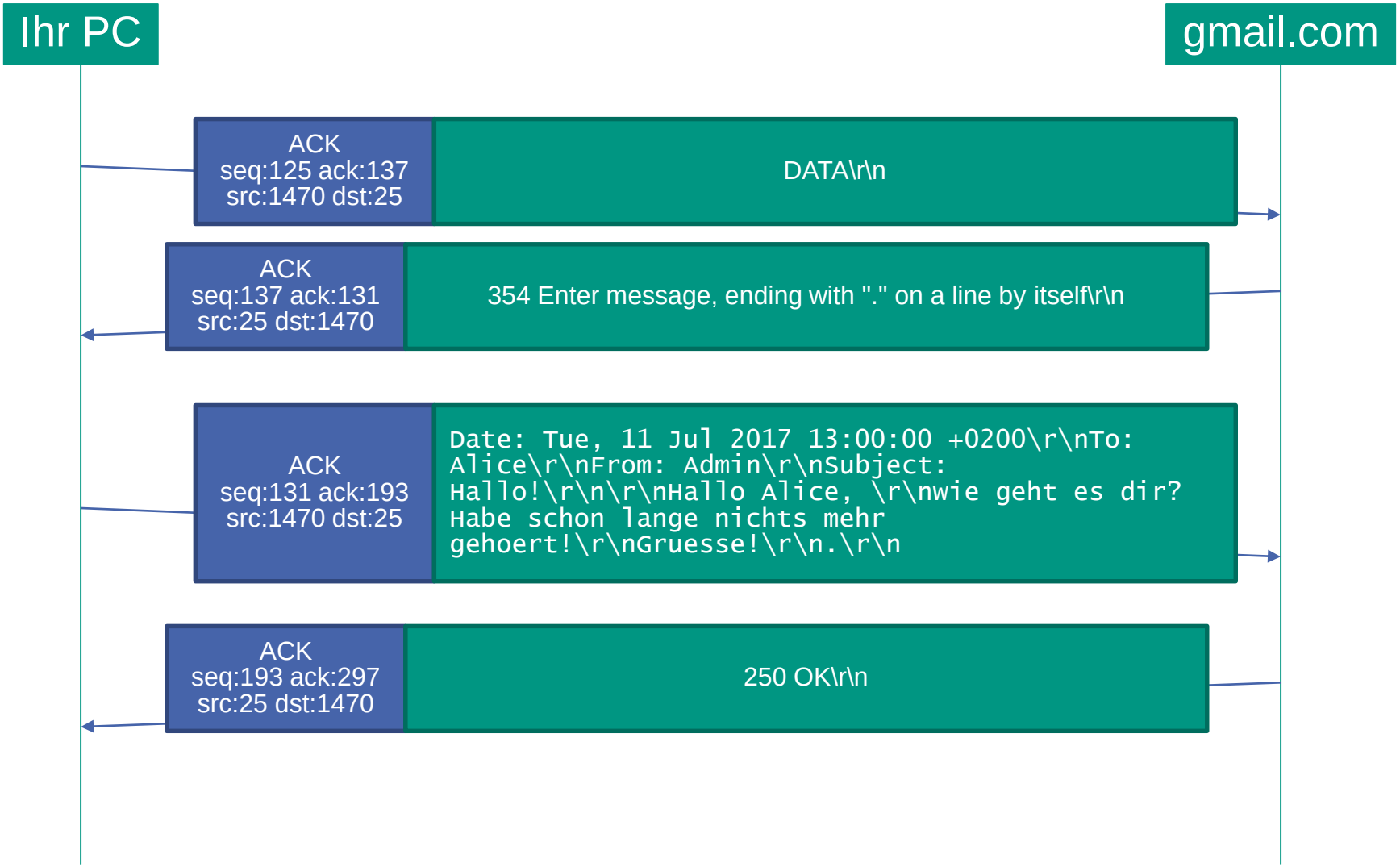
- SMTP-Nachrichten immer abwechselnd
  - Max. 1 TCP-Segment gleichzeitig unterwegs
    - Staukontrolle kann vernachlässigt werden!
    - Timeout für Paketverlust kritisch (einziger Indikator!)

f) SMTP und TCP

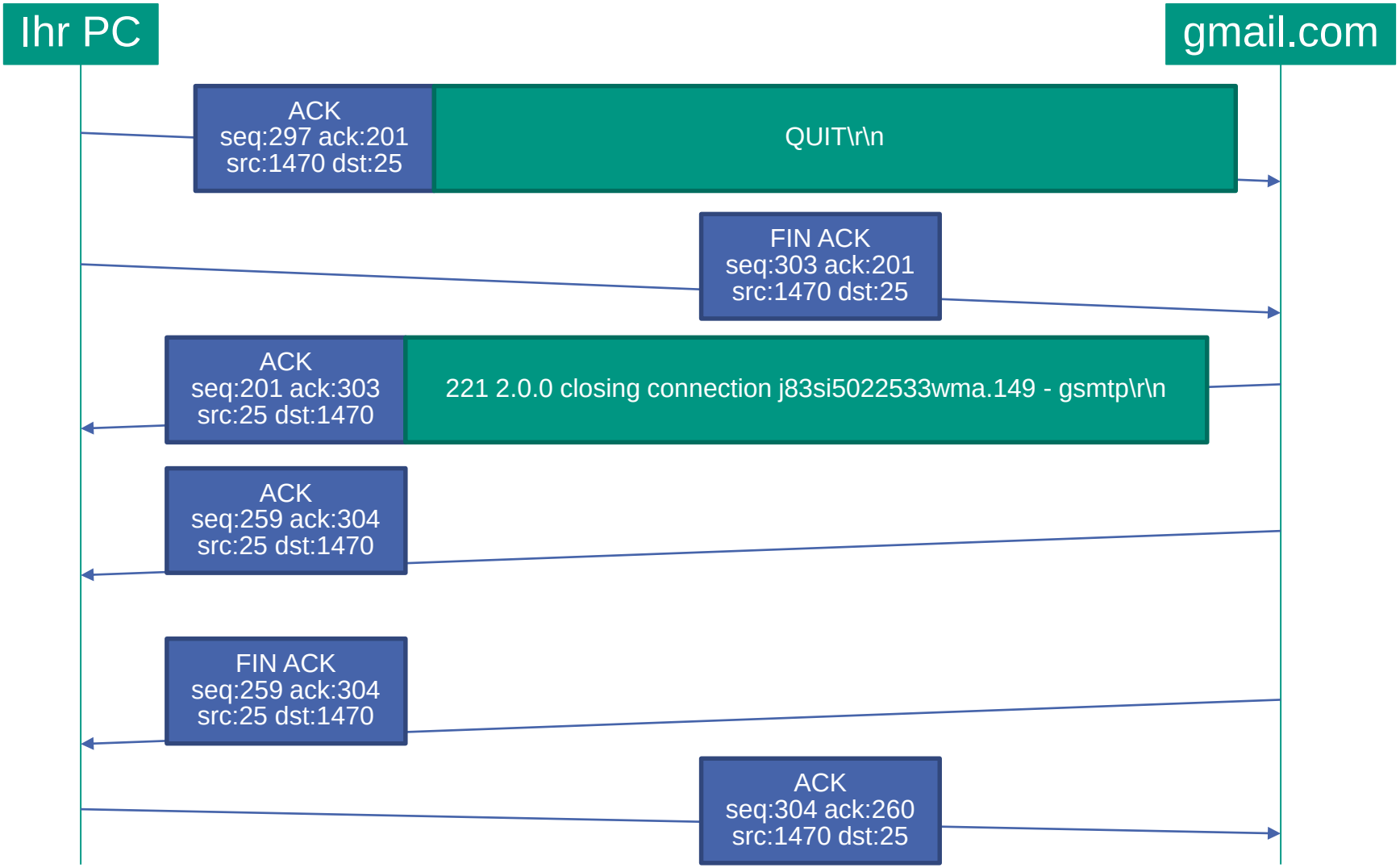


f) SMTP und TCP

f) SMTP und TCP



f) SMTP und TCP



Einführung in Rechnernetze – 7. Übungsblatt

1. E-Mail im Zusammenspiel
2. Sicherheit
3. Web of Trust

Aufgabe 2 (a)

- Nehmen Sie an, Alice und Bob einigen sich über einen sicheren Kanal auf einen gemeinsamen symmetrischen Schlüssel S. Sie wollen diesen für die Integritätssicherung verwenden: Sobald Alice eine Nachricht an Bob versendet, berechnet sie den SHA-1-Hashwert H_{Send} ihres Nachrichtentextes, verschlüsselt diesen mit S und hängt das Ergebnis $S(H_{Send})$ an ihre Nachricht. Sobald Bob die Nachricht erhält, entschlüsselt er $S(H_{Send})$ mit Hilfe von S und erhält H_{Send}. Jetzt berechnet er den SHA-1-Hashwert H_{Recv} des erhaltenen Nachrichtentextes. Falls $H_{Send} = H_{Recv}$ gilt, weiß er, dass er die Originalnachricht von Alice unmanipuliert erhalten hat. Da der Schlüssel S symmetrisch ist, funktioniert das Verfahren für Nachrichten von Bob an Alice identisch. Gehen Sie davon aus, dass S nur Alice und Bob bekannt ist.

- Welche Nachteile hat dieses Verfahren dennoch, verglichen mit der Verwendung von asymmetrischer Kryptographie, wie Sie sie von digitalen Signaturen kennen

Aufgabe 2 (a)

Msg_{Send} „Hi Bob. Ich schulde dir hiermit 100 EUR. Alice.“

Msg_{Recv}

H_{Send} c33eca12914a39d7a53 4eeaab9fda50b86cce4cc

c33eca12914a39d7a53 4eeaab9fda50b86cce4cc H_{Recv}

=

$S(H_{Send})$ 97284928310843902853 84320858379120289272

c33eca12914a39d7a53 4eeaab9fda50b86cce4cc $S(H_{Send})$

Aufgabe 2 (a)

Msg_{Send} „Hi Bob. Ich schulde dir hiermit 100 EUR. Alice.“ „Hi Bob. Ich schulde dir hiermit 1 EUR. Alice.“

H_{Send} c33eca12914a3920aa13d4b9ccd8a9f427
4eeaab9fda50b86ce4c1dbffb9faa98df0

(H_{Send}) 97284928310843902853
84320858379120289272 **?**

Aufgabe 2 (a)

(✗) Authentizität

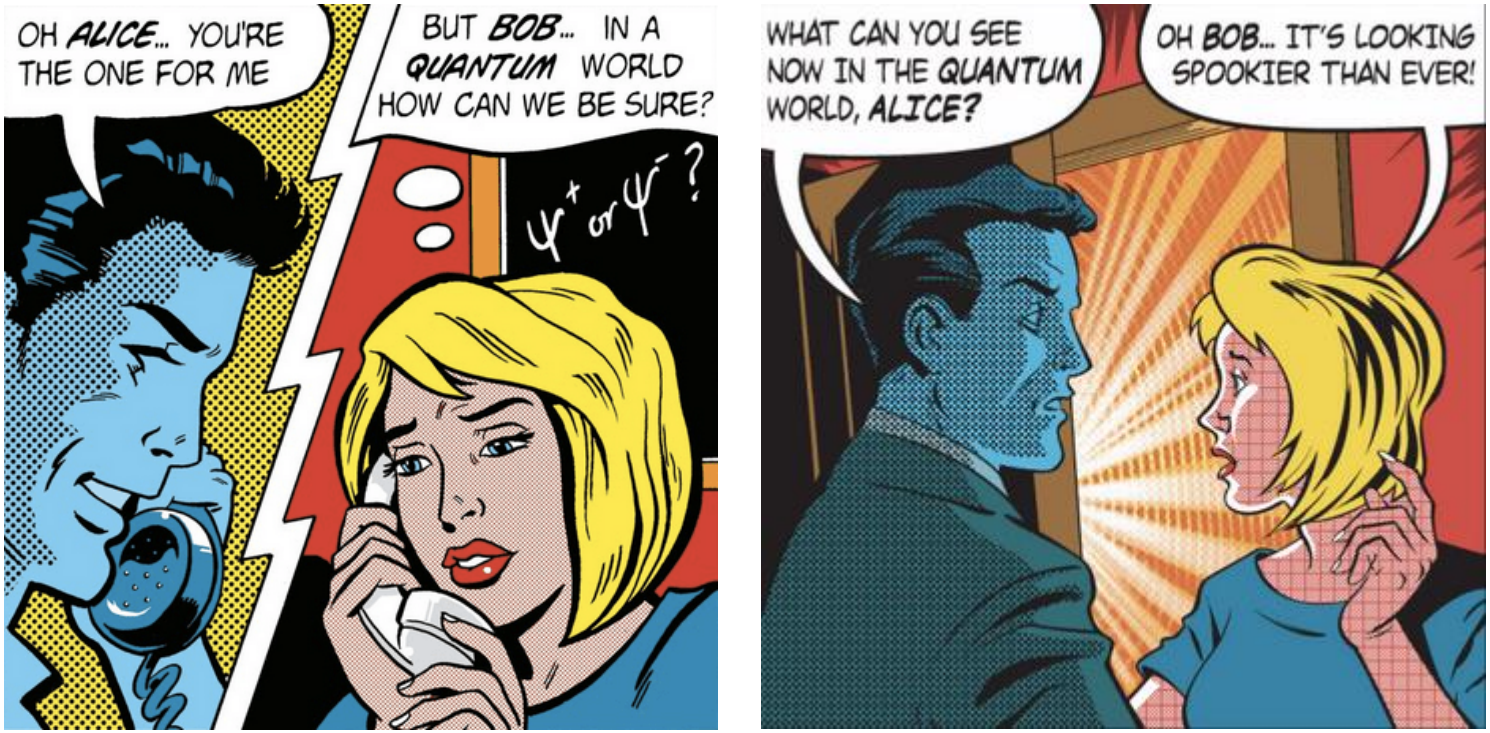
(✓) Integrität

✗ **„Nichtabstreitbarkeit“**
Nachweisbar
Nicht fälschbar
Kann nicht geleugnet werden

Msg_{Send}	„Hi Bob. Ich schulde dir hiermit 100 EUR. Alice.“	„Hi Bob. Ich schulde dir hiermit 1000 EUR. Alice.“	Msg_{Recv}
H_{Send}	c33eca12914a39d7a53 4eeaab9fda50b86cce4cc	4cc9b1223c07fc5448a09 1325cac1d54e48c1b88	H_{Recv}
$S(H_{Send})$	97284928310843902853 84320858379120289272	38734792398734987[illegible]9 23978937481527398437	$S(H_{Send})$

Einschub: Alice & Bob

- Geht auf RSA (Rivest, Shamir, Adleman) zurück
- Seit 1978 untrennbar ☺
- Weitere Charaktere: Carol, Eve, Peggy, Mallory, Trent, Trudy, …



- Mehr zum Thema: http://cryptocouple.com/
  - A History of The World's Most Famous Cryptographic Couple

Aufgabe 2 (b)

- Wenn das Verschlüsselungsverfahren selbst öffentlich bekannt, standardisiert und verfügbar ist, woraus erfolgt dann die Sicherheit?

- Kerckhoffs's principle
  - Ein Verfahren bleibt sicher, selbst wenn alles bis auf den Schlüssel bekannt wird (*kein Zwang*)
  - Bereits beim Entwurf des Verfahrens soll man davon ausgehen, dass das Verfahren dem Angreifer bekannt ist
  - ➔ Geheimhaltung des Schlüssels

- Schneier's Law
  - “Anyone, from the most clueless amateur to the best cryptographer, can create an algorithm that he himself can't break”
  - Verfahren sollte vielmehr durch die Forschungscommunity als sicher angesehen werden
  - ➔ Offenlegung des Verfahrens

- Gegenteil: Security through obscurity

Aufgabe 2 (c)

- Folgt aus dem erfüllten Schutzziel Vertraulichkeit automatisch Datenintegrität?
- Folgt aus dem erfüllten Schutzziel Datenintegrität automatisch Vertraulichkeit?

Aufgabe 2 (c)

- **Vertraulichkeit**
  - Übertragene Daten sollen nur berechtigten Instanzen zugänglich sein, d.h. es darf kein unautorisierter Informationsgewinn stattfinden
  - *Bausteine:* Symmetrische/asymmetrische Verschlüsselung

- **Integrität**
  - Daten dürfen nicht unautorisiert und **unbemerkt** manipuliert werden
  - *Bausteine:* Message Authentication Codes (Hashfunktionen mit Schlüssel)

- Folgt aus Vertraulichkeit automatisch Integrität?
  - **Nein.**
  - Angreifer kann Chiffretext abhängig von der Art der Verschlüsselung manipulieren, ohne ihn zu entziffern.
  - Beispiel: Löschen von einzelnen Sätzen

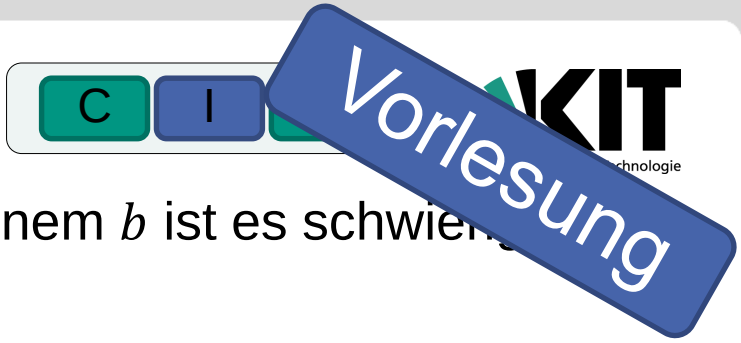
- Folgt aus Integrität automatisch Vertraulichkeit?
  - **Nein.**
  - Daten können signiert, aber dennoch unverschlüsselt sein.

Fazit: Schutzziele sind in der Regel unabhängig voneinander und benötigen eigene Bausteine.

Aufgabe 2 (d)

- Warum bietet eine kryptographische Hashfunktion einen besseren Integritätscheck als die TCP-Prüfsumme?

Hashfunktion: Eigenschaften



- Einwegfunktion/Urbildresistenz: zu gegebenem b ist es schwierig, ein a zu finden, so dass gilt $H(a) = b$
  - Anschauliches Beispiel: Telefonbuch
    - Nummer zu Name leicht; Name zu Nummer nicht leicht
- Schwache Kollisionsresistenz
  - Für gegebenes a ist es schwierig, ein $\bar{a} \neq a$ zu finden, so dass gilt $H(a) = H(\bar{a})$
- Starke Kollisionsresistenz
  - Es ist schwierig, zwei verschiedene Werte a und $\bar{a}$ aus der Urbildmenge A zu finden, so dass gilt $H(a) = H(\bar{a})$
- Beliebige Länge der Nachricht
  - $H(m)$ kann auf Nachrichten beliebiger Länge angewandt werden
- Hashwert fester Länge
  - $H(m)$ liefert Hashwert z einer festen Länge
- Effizient
  - $H(m)$ ist relativ einfach zu berechnen
- Beispiele: MD5, SHA-1 (gelten als nicht mehr sicher), SHA-2, SHA-3

Aufgabe 2 (e)

- TCP-Prüfsumme
  - Einer-Komplement der Einer-Komplement-Summe von 16-Bit-Worten

- Beispiel: Text in 8-Bit ASCII-Code
- „Deutschland hat im Halbfinale 1 : 7 gespielt.“

Zeichen	Hexadezimal wert (ASCII)	Binär	
„1 “	0x3120	0001 0001 0001 0101	
„7 “	0x3220	0010 0000 0001 0101	
		0011 0001 0010 1010	Einerkomplement-Summe
		1100 1110 1101 0101	Einerkomplement (➔ Prüfsumme)

Aufgabe 2 (f)

- Kann man das Urbild eines Hashwerts einer kryptographischen Hashfunktion wieder herstellen? Wann gilt eine Hashfunktion also als „sicher“? Begründen Sie Ihre Antwort.

Hashfunktion: Eigenschaften

- Einwegfunktion/Urbildresistenz: zu gegebenem b ist es schwierig, ein a zu finden, so dass gilt $H(a) = b$
  - Anschauliches Beispiel: Telefonbuch
    - Nummer zu Name leicht; Name zu Nummer nicht leicht
- Schwache Kollisionsresistenz
  - Für gegebenes a ist es schwierig, ein $\bar{a} \neq a$ zu finden, so dass gilt $H(a) = H(\bar{a})$
- Starke Kollisionsresistenz
  - Es ist schwierig, zwei verschiedene Werte a und $\bar{a}$ aus der Urbildmenge A zu finden, so dass gilt $H(a) = H(\bar{a})$
- Beliebige Länge der Nachricht
  - $H(m)$ kann auf Nachrichten beliebiger Länge angewandt werden
- Hashwert fester Länge
  - $H(m)$ liefert Hashwert z einer festen Länge
- Effizient
  - $H(m)$ ist relativ einfach zu berechnen
- Beispiele: MD5, SHA-1 (gelten als nicht mehr sicher), SHA-2, SHA-3

Einschub: Coin Tossing

- Alice und Bob telefonieren miteinander
- Möchten einen Münzwurf durchführen

- Alice nimmt ein Telefonbuch zur Hand
  - Sucht einen beliebigen Namen und Nummer heraus
  - Verrät Bob diesen nicht

- Bob nimmt eine Münze mit Kopf und Zahl
  - Führt Münzwurf durch
  - Verrät Alice das Ergebnis nicht

- Am Telefon
  - Bob sagt Namen
  - Alice das Ergebnis des Münzwurfs
    - kann Telefonnummer zu Namen nicht kennen
  - Ergebnis von Alice wird auf letzte Ziffer der Telefonnummer aufaddiert

➔ Zufallsergebnis erzeugt (kein Einfluss von Bob oder Alice möglich)

Aufgabe 2 (g)

- Worauf, auf welcher Annahme, beruht die Sicherheit von RSA?

- Das man aus dem öffentlichen Schlüssel $K_D(n, d)$ den privaten Schlüssel $K_E(e)$ nicht berechnen kann
  - Zerlege n in Primfaktoren p und q
  - Berechne $z = (p-1) * (q-1)$
  - Finde zu d und z ein passendes e sodass gilt: $e * d \; mod \; z \equiv 1$
  - ➔ privater Schlüssel: e

- Annahme ein solcher effizienter Polynomialzeit Algorithmus zur Primfaktorzerlegung von großen Zahlen existiert nicht

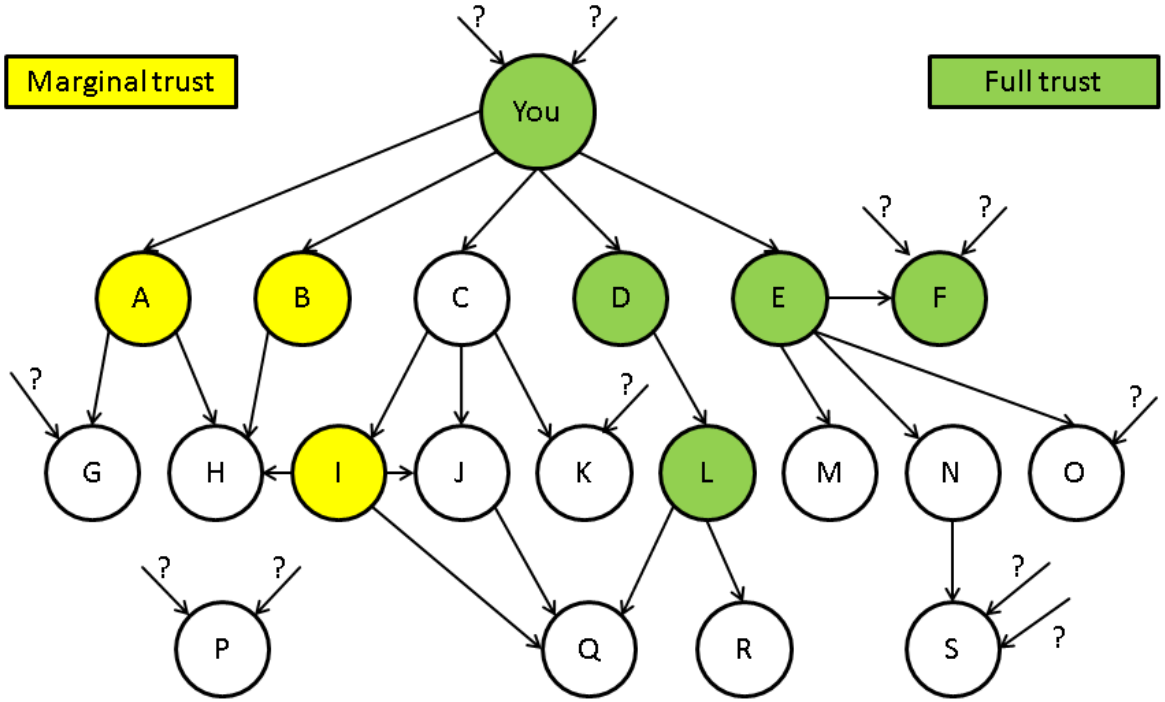
- In der Komplexitätstheorie: $P \neq NP$

Einführung in Rechnernetze – 7. Übungsblatt

1. E-Mail im Zusammenspiel
2. Sicherheit
3. Web of Trust

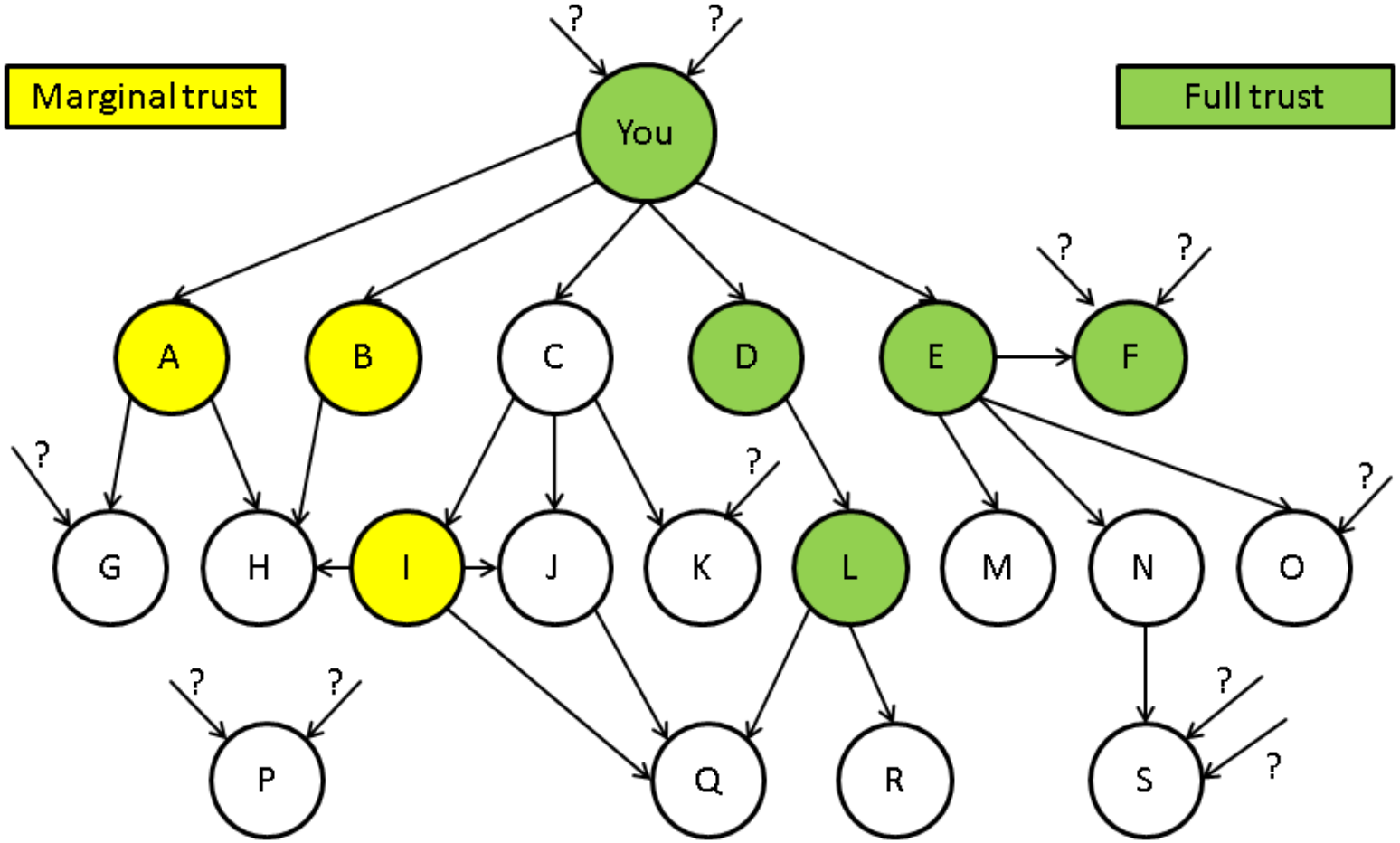
Web of Trust

- Studierende der Vorlesung Einführung in Rechnernetze haben untereinander ein **Web of Trust** aufgebaut um in Zukunft Mails vertraulich und authentisch austauschen zu können.
- In folgender Abbildung werden die Signaturen der Studierenden untereinander dargestellt. In gelb ist in der Abbildung gekennzeichnet in welche Nutzer Sie teilweise vertrauen haben und mit grün in welche Sie volles vertrauen haben.



Aufgabe 4 (a)

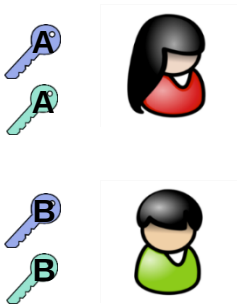
- Welche Art von Schlüsseln muss jeder Nutzer wie erzeugen um beim Web of Trust mitmachen zu können? Wie werden diese Schlüssel untereinander bekannt gemacht?



Beispiel: Nachrichtenversand von Alice an Bob

Vorlesung

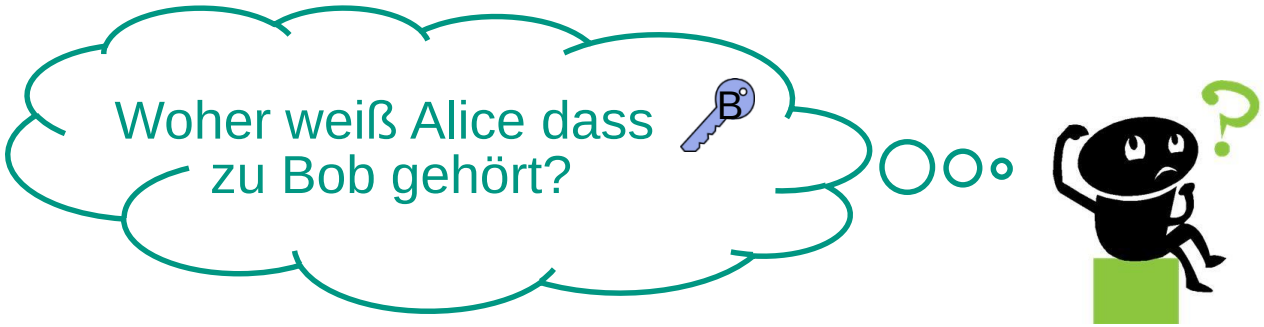
- Schlüsselgenerierung
  - Alice generiert
    - Öffentlichen Schlüssel K_E^A
    - Privaten Schlüssel K_D^A
  - Bob generiert
    - Öffentlichen Schlüssel K_E^B
    - Privaten Schlüssel K_D^B



- Versand der Nachricht
  - Alice
    - Verschlüsselt Nachricht m symmetrisch mit zufälligem Schlüssel K_R
    - Verschlüsselt Schlüssel K_R asymmetrisch mit Bobs öffentlichem Schlüssel K_E^B
    - Sendet Ciphertext der Nachricht und Ciphertext des Schlüssels an Bob
  - Bob
    - Entschlüsselt den Schlüssel K_R asymmetrisch mit privatem Schlüssel K_D^B
    - Entschlüsselt Nachricht m mit symmetrischen Schlüssel K_R

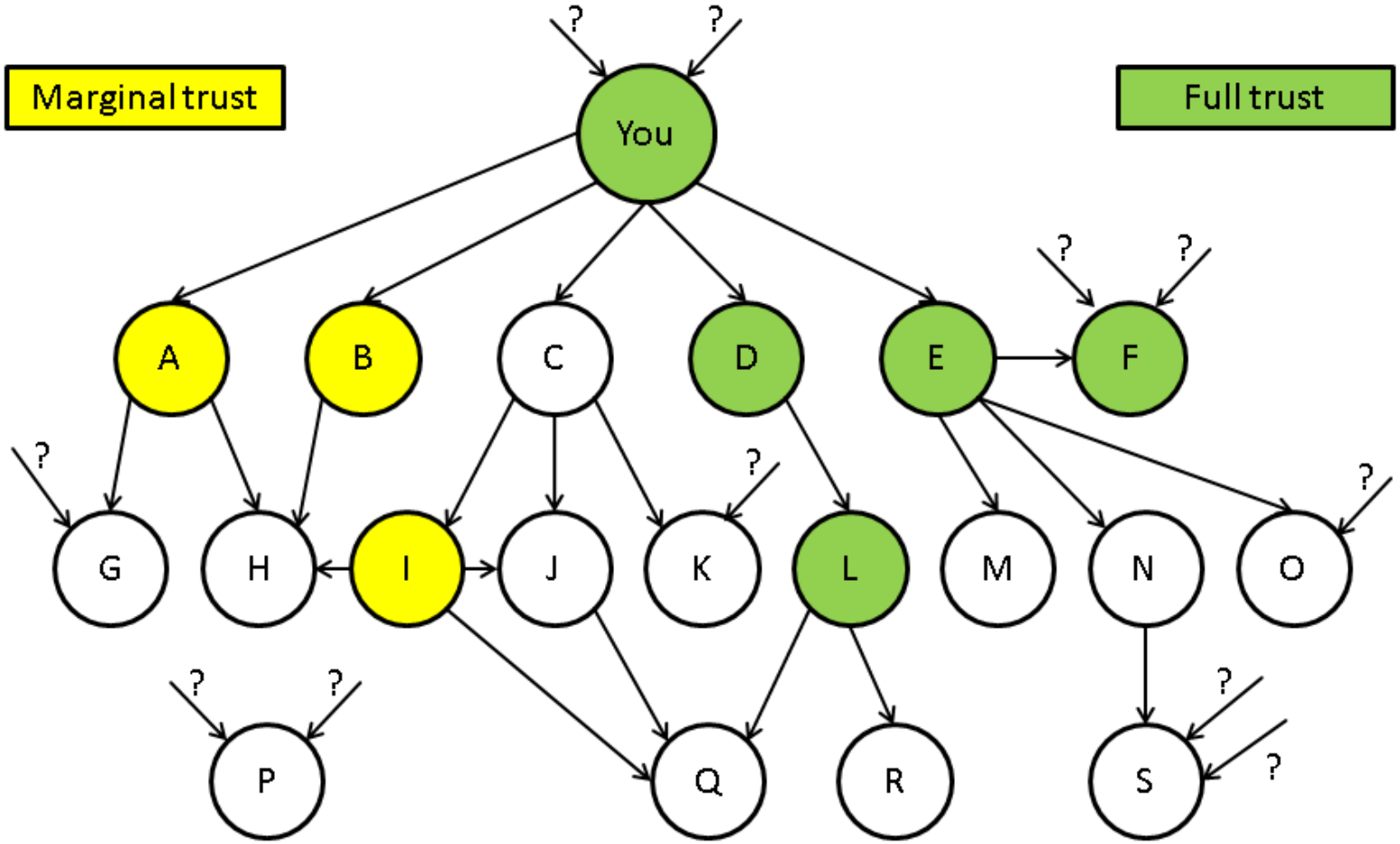
Bekanntgabe öffentlicher Schlüssel

- Verwendung von öffentlichen Schlüsseln
  - Müssen Nutzern vor Versand bekannt sein
  - Zugehörigkeit zu Email-Adresse muss überprüfbar sein
    - Für Authentizität des Absenders
    - Für Authentizität des Empfängers
- Bekanntgabe des zur Email-Adresse gehörenden öffentlichen Schlüssels
  - Persönlich oder per Email
  - Über eine bekannte vertrauenswürdige dritte Partei
  - Eintragung auf wohlbekannten Schlüsselservern
    - z.B. pgp.mit.edu, keyserver.pgp.com, zimmermann.mayfirst.org, …



Aufgabe 4 (b)

- Erläutern Sie wie im Web of Trust prinzipiell überprüft wird ob ein öffentlicher Schlüssel authentisch ist, also ob dieser tatsächlich zu einer Mailadresse gehört.



Web of Trust (WoT)

- Idee
  - Keine zentrale vertrauenswürdige dritte Partei notwendig
  - Dezentraler anarchischer Ansatz
  - Transitive Überprüfung der Authentizität eines öffentlichen Schlüssels

- Abstrahiertes Beispiel
  - Überprüfen ob öffentlicher Schlüssel K_E^X zu Bob gehört (authentisch ist)
  - Suche in Bekanntenkreis ob jemand bereits K_E^X überprüft hat (Signatur)
  - Akzeptiere K_E^X als authentisch bei bekannten Signaturen

- Aber Achtung: Trennung von
  - Vertrauen in Nutzer und deren Signaturen
  - Authentizität eines öffentlichen Schlüssels

Web of Trust (WoT)

- Grad des Vertrauens in andere Nutzer und in deren Signaturen
  - Owner Trust und daraus abgeleitet Signatory Trust
  - Unbekannt, kein, geringes, volles oder absolutes Vertrauen
- Grad der Authentizität eines öffentlichen Schlüssels
  - Unbekannt, keine, teilweise oder volle Authentizität
  - Wird berechnet
- Vorbereitung
  - Nutzer können andere öffentliche Schlüssel signieren
  - Informationen werden über Schlüsselserver an alle verteilt
- Berechnung der (transitiven) Authentizität (lokal)
  - Nutzer bauen untereinander graduelle Vertrauensbeziehungen auf
  - Nutzer vertrauen Signaturen ihrer vertrauenswürdigen Kontakte
  - Authentizität eines öffentlichen Schlüssel wird berechnet (Key Legitimacy)

WoT – Key Legitimacy

- Nutzer kann einstellen ab wann er öffentlichen Schlüssel als authentisch akzeptiert

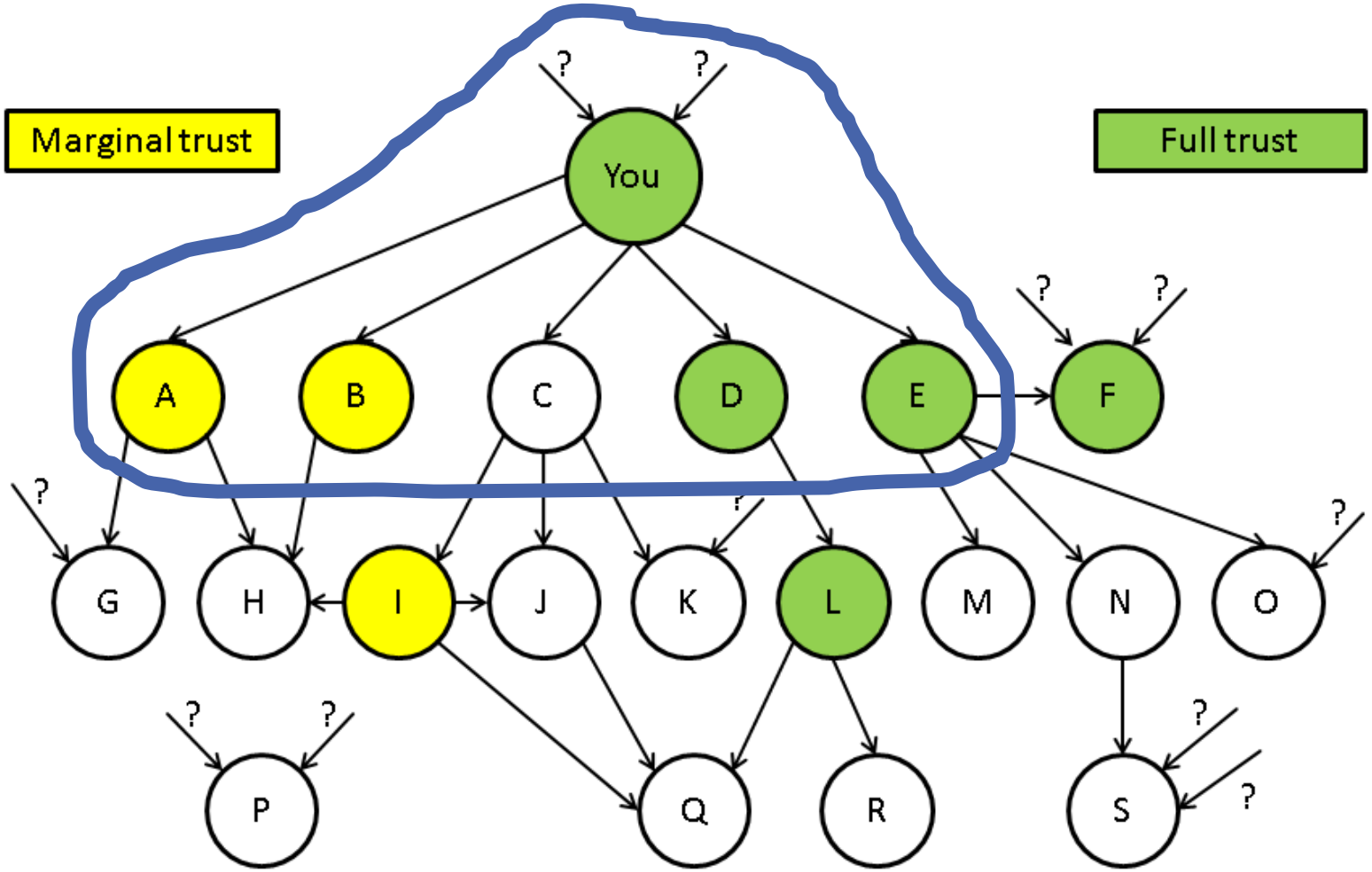
- Durch Berechnung der sogenannten Key Legitimacy L
  - $L := \frac{Signaturen\ mit\ geringem\ Vertrauen}{X} + \frac{Signaturen\ mit\ vollem\ Vertrauen}{Y}$
  - Vertrauen in Signaturen direkt abgeleitet von Vertrauen in Nutzer
  - Parameter X und Y als Integerzahl einstellbar

- Bewertung der Key Legitimacy (Authentizität)
  - $L = 0$ überprüfter Schlüssel nicht authentisch
  - $0 < L < 1$ überprüfter Schlüssel teilweise authentisch
  - $L \geq 1$ überprüfter Schlüssel voll authentisch

- Achtung: Vertrauen in eigenen Schlüssel ist absolut
  - Daraus abgeleitet volle Authentizität selbstsignierter Schlüssel

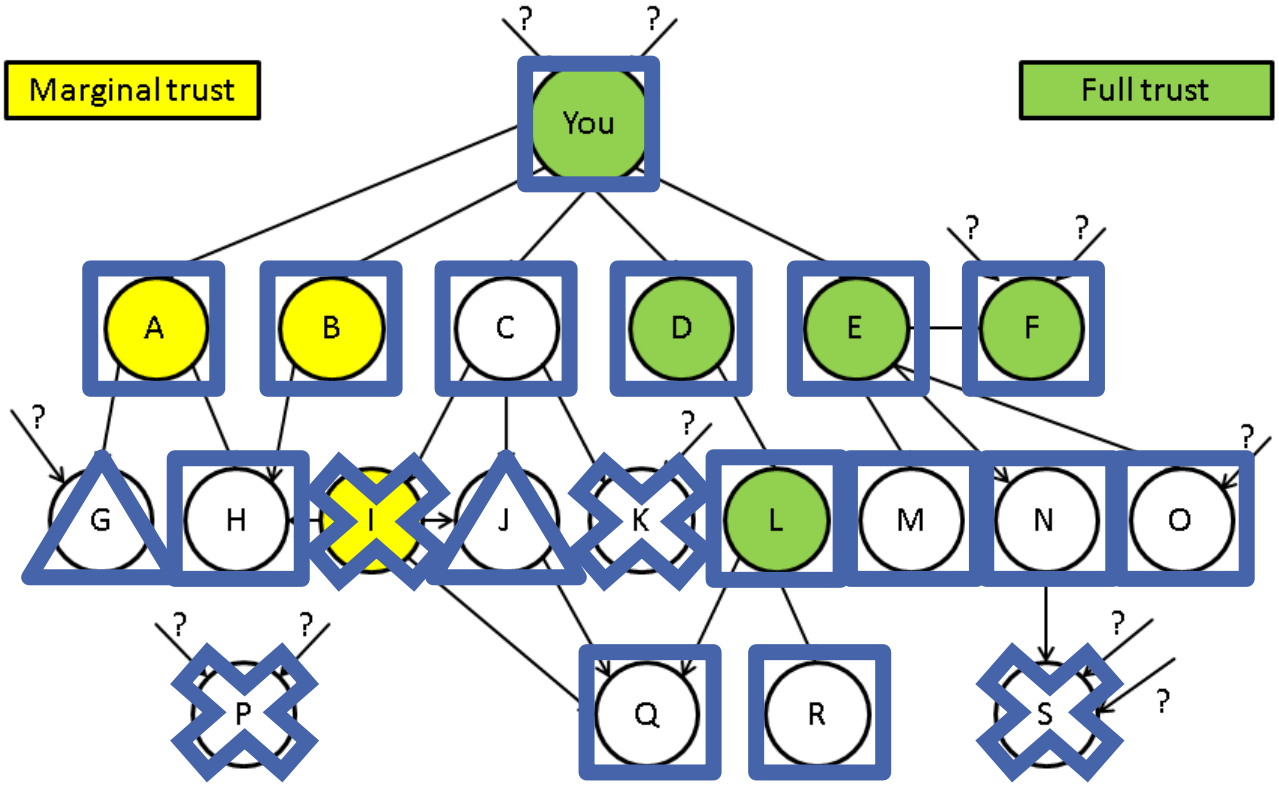
Aufgabe 4 (b)

- Für welche Schlüssel können Sie die Authentizität ohne Beachtung der Key Legitimacy angeben? Und warum?



Aufgabe 4 (c)

- Geben Sie aus Ihrem Blickwinkel für jeden dargestellten Schlüssel das Vertrauen in den Nutzer und die Authentizität des Schlüssels an. Berücksichtigen Sie dazu als Key Legitimacy: zwei teilweise vertrauenswürdige Signaturen oder eine voll vertrauenswürdige Signatur.



Aufgabe 4 (d)

- Sind Vertrauen in Nutzer, Signaturen auf Schlüssel und Authentizität der Abbildung zwischen Schlüssel und Nutzer unabhängig voneinander oder abhängig?

- Vertrauen in Nutzer () Signaturen
- Vertrauen in Nutzer  Authentizität von Schlüsseln
- Signaturen () Authentizität von Schlüsseln

Einführung in Rechnernetze – 7. Übungsblatt

1. E-Mail im Zusammenspiel
2. Sicherheit
3. Web of Trust

Einschub: Crypto War

- TLS Working Group (TLSWG)
- Hitzige (längere) Diskussionen auf der Mailinglist
  - Analyse von verschlüsseltem Verkehr im Rechenzentrum unmöglich
  - Notwendig z.B. für Load-Balancing, Caching, Lawful interception, etc..

➔ Vorschlag: Aufbrechen der TLS Ende-zu-Ende-Sicherheit

  - Server-seitige Auslagerung von Schlüsselmaterial auf dritte Partei zur Berechnung von Session Keys (Key Escrow)
  - Soll „nur“ passive Entschlüsselung ermöglichen
  - Aber: kein Perfect Forward Secrecy mehr (PFS)
  - Aber: bekommt das der Client mit?

- IETF99, Prag, Working Group Meeting
  - u.a. Diskussion eines Drafts dazu
  - Abwägen von Für und Wider (open mic)

➔ Humming: Soll sich die TLSWG damit beschäftigten, ob es möglich sein soll, TLS im Rechenzentrum zu entschlüsseln?

- https://www.youtube.com/watch?v=ms-0PIY1R-8

Fragen?

- Vielen Dank für die Aufmerksamkeit
- Viel Erfolg bei der Klausur! Achtung: Anmeldungs-Deadline

Sonntag, der 06.08.2017