Work in progress

Informationssicherheit

Ute Bormann, TI2

2023-10-13

Inhalt

1. Sicherheitsziele und lokale Zugangspolitik
2. Symmetrische Verschlüsselung
3. Asymmetrische Verschlüsselung

Teil 1:
Sicherheitsziele und lokale Zugangspolitik

Informationssicherheit (Security)

- Ziel: Reibungsloses Erfüllen der beabsichtigten Aufgaben des Systems
- Angriffe → Verwundbarkeit
  - versehentlich/absichtlich

	versehentlich	absichtlich
Hardware	Speisen/Getränke	Zerstörung
	Wasserschäden	Herbeiführung von Störungen
	Mäuse...	Diebstahl
Software/Daten	Bugs	Zerstörung
	Bedienfehler	Fälschung
		„Diebstahl“ (Kopie)

Informationssicherheit (Security)

- Ziel: Reibungsloses Erfüllen der beabsichtigten Aufgaben des Systems
- Angriffe → Verwundbarkeit
  - versehentlich/absichtlich

	versehentlich	absichtlich
Hardware	Speisen/Getränke	Zerstörung
	Wasserschäden	Herbeiführung von Störungen
	Mäuse...	Diebstahl
Software/Daten	Bugs	Zerstörung
	Bedienfehler	Fälschung
		„Diebstahl“ (Kopie)
		Schadprogramme:
		• Trojanische Pferde
		• Viren
		• Würmer

Sicherheitsziele

- Geheimhaltung (Secrecy/Confidentiality)

  Nur autorisierte Personen erhalten Kenntnis

- Unversehrtheit (Integrity)

  Nur autorisierte Personen können ändern/erzeugen

- Verfügbarkeit (Availability)

  Autorisierte Personen können immer zugreifen

Sicherheitsziele

- Geheimhaltung (Secrecy/Confidentiality)

  Nur autorisierte Personen erhalten Kenntnis

- Unversehrtheit (Integrity)

  Nur autorisierte Personen können ändern/erzeugen

- Verfügbarkeit (Availability)

  Autorisierte Personen können immer zugreifen

- außerdem: Authentisierung als Voraussetzung

Sicherheitsziele

- Geheimhaltung (Secrecy/Confidentiality)

  Nur autorisierte Personen erhalten Kenntnis

- Unversehrtheit (Integrity)

  Nur autorisierte Personen können ändern/erzeugen

- Verfügbarkeit (Availability)

  Autorisierte Personen können immer zugreifen

- außerdem: Authentisierung als Voraussetzung

Klassifikation von Angriffen

- Abfangen (Interception)
  - Anzapfen von Leitungen
  - Manipulation von Druckern... (Kopie anlegen)
  - Manipulation von Tastaturen... (Geheimzahl)
  - Nachfrage bei Geheimnisträgern...
- Modifikation (Modification) / Fabrikation (Fabrication)
  - Daten verfälschen...
  - Programme verändern...
  - Transaktionen einfügen... (Wiedereinspielen)
- Unterbrechung (Interruption)
  - Zerstörung/Wegnahme von Geräten
  - Löschen von Daten/Programmen
  - Blockieren des Systems
- Maskerade
  - Ausgeben für anderen

Vorgehen

- Vermeidung von Angriffen
- Abwehr von Angriffen → Entdecken/Verfolgen
- Schadensbegrenzung

→ Realisierung einer Sicherheitspolitik

  - Was ist das „Normalverhalten“?
  - Welche Sicherheitsziele sind einzuhalten?

→ Daraus Strategien ermittelbar

Lokale Zugangspolitik

- Zugangsbeschränkung:

  Nur autorisierte Personen kommen „rein“

- Einschränkung des Zugriffs:

  Personen dürfen nur bestimmte Dinge tun

➔ setzt Authentisierung der Personen voraus

Authentisierung (Identifizierung)

- Wissen (Kennwort)

Probleme:

- Durchprobieren ➔ u.U. „Denial of Service“
- Erraten
- Erfragen
- Aufzeichnungen lesen
- Ablage im Rechner lesen
- Eintippen abfangen

Who is who?

Authentisierung (Identifizierung)

- Wissen (Kennwort)

Probleme:

- Durchprobieren ➔ u.U. „Denial of Service“
- Erraten
- Erfragen
- Aufzeichnungen lesen
- Ablage im Rechner lesen
- Eintippen abfangen

Who is who?

- Besitz („Schlüssel“)
  - meist Chipkarten

  ➔ aktivieren mit PIN (Wissen)

Authentisierung (Identifizierung)

- Wissen (Kennwort)

Probleme:

- Durchprobieren ➔ u.U. „Denial of Service“
- Erraten
- Erfragen
- Aufzeichnungen lesen
- Ablage im Rechner lesen
- Eintippen abfangen

Who is who?

- Besitz („Schlüssel“)
  - meist Chipkarten

  ➔ aktivieren mit PIN (Wissen)

- Persönliche Merkmale
  - Fingerabdrücke
  - Augenhintergrund
  - Sprache ➔ Aufzeichnungen?
  - ...

Autorisierung

- Zugriffshierarchie
  - z.B. Super-User vs. „normale“ Nutzer

  → oft zu inflexibel

Wer darf was?

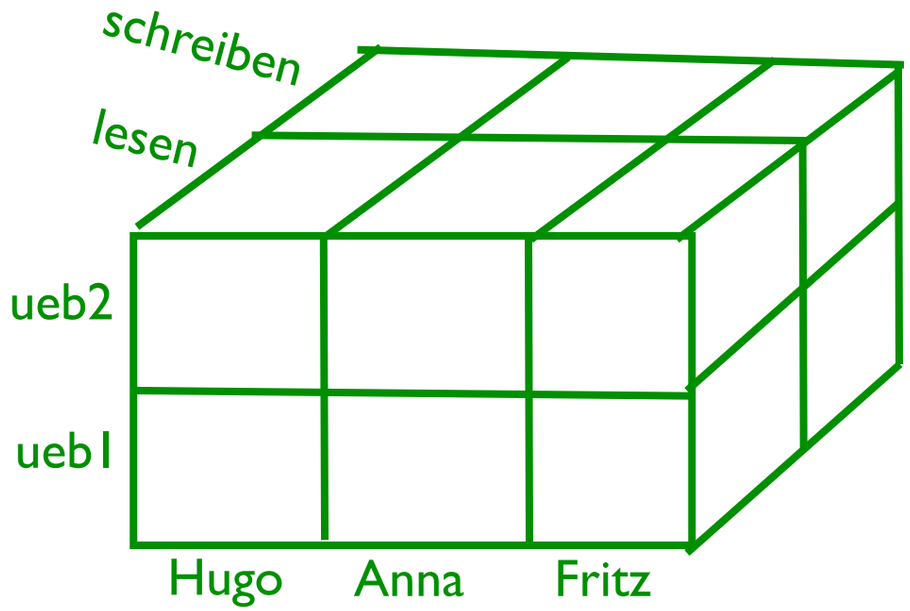
Autorisierung

Wer darf was?

- Zugriffshierarchie
  - z.B. Super-User vs. „normale“ Nutzer

  ➔ oft zu inflexibel
- Subjekt-Objekt-Funktions-Modell
  - Wer darf worauf was?

  ➔ dreidimensionale Zugriffskontroll-Matrix



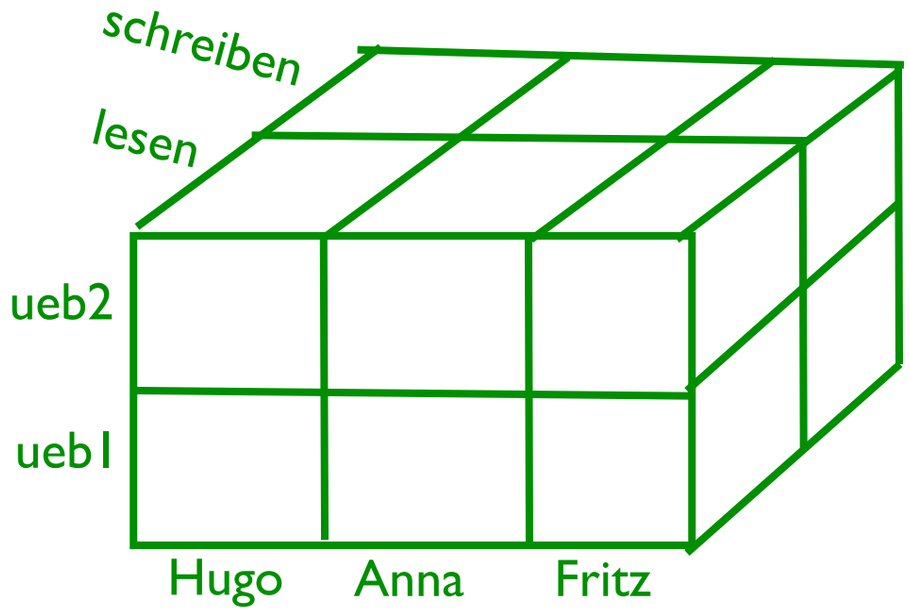
Autorisierung

- Zugriffshierarchie
  - z.B. Super-User vs. „normale“ Nutzer

  ➔ oft zu inflexibel
- Subjekt-Objekt-Funktions-Modell
  - Wer darf worauf was?

  ➔ dreidimensionale Zugriffskontroll-Matrix

Wer darf was?



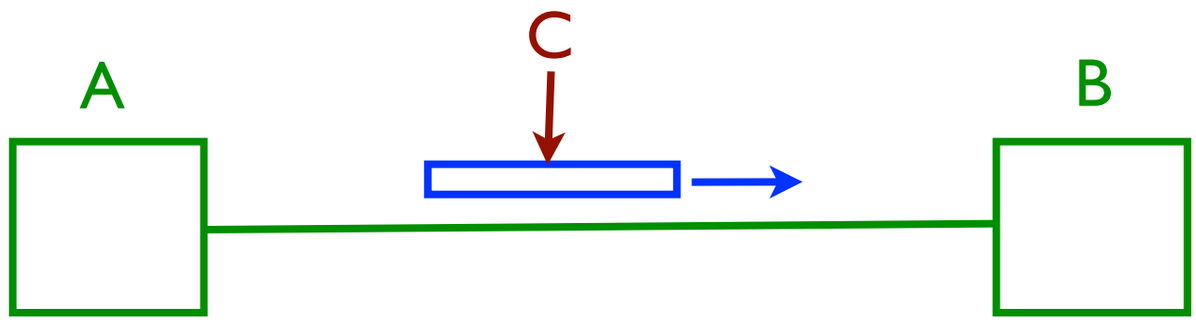
- Vereinfachungen (➔ klassisches Unix):
  - globale Eigentümerrechte
  - Benutzergruppe vs. „Rest der Welt“
- Varianten:
  - Zugriffskontroll-Listen bei Objekten (➔ ACL)
  - Ausweise (Capabilities) bei Subjekten

Fragen – Teil 1

- Nenne einige absichtliche und unabsichtliche Angriffe auf Hardware, Software und/oder Daten.
- Welche grundsätzlichen *Sicherheitsziele* kann man unterscheiden?
- Was ist eine *Sicherheitspolitik*?
- Auf welche verschiedenen Arten kann sich ein Benutzer authentifizieren?
- Welche Komponenten enthält eine Zugriffskontrollmatrix? Wie ordnen sich die Dateizugriffsrechte in Unix in dieses Schema ein?

Teil 2:
Symmetrische Verschlüsselung

Sicherheit in Rechnernetzen

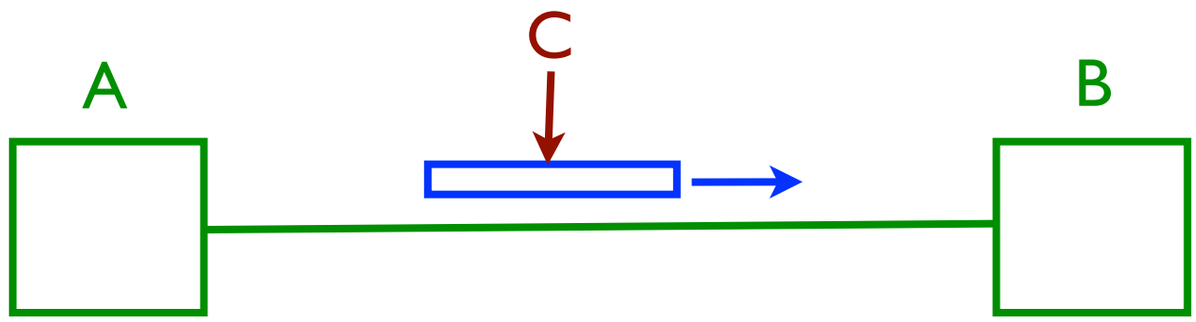


Zusätzliche Gefahren (gegenüber lokalen Systemen)

a) „Eindringlinge“ (Intruder) kann man nicht aussperren

➔ Kommunikationsmedien können angezapft/gestört werden

Sicherheit in Rechnernetzen



Zusätzliche Gefahren (gegenüber lokalen Systemen)

a) „Eindringlinge“ (Intruder) kann man nicht aussperren

➔ Kommunikationsmedien können angezapft/gestört werden

b) Kommunikationspartner kennen sich u.U. kaum

➔ vertrauen sich nicht unbedingt

Sicherheitsziele erweitern um:

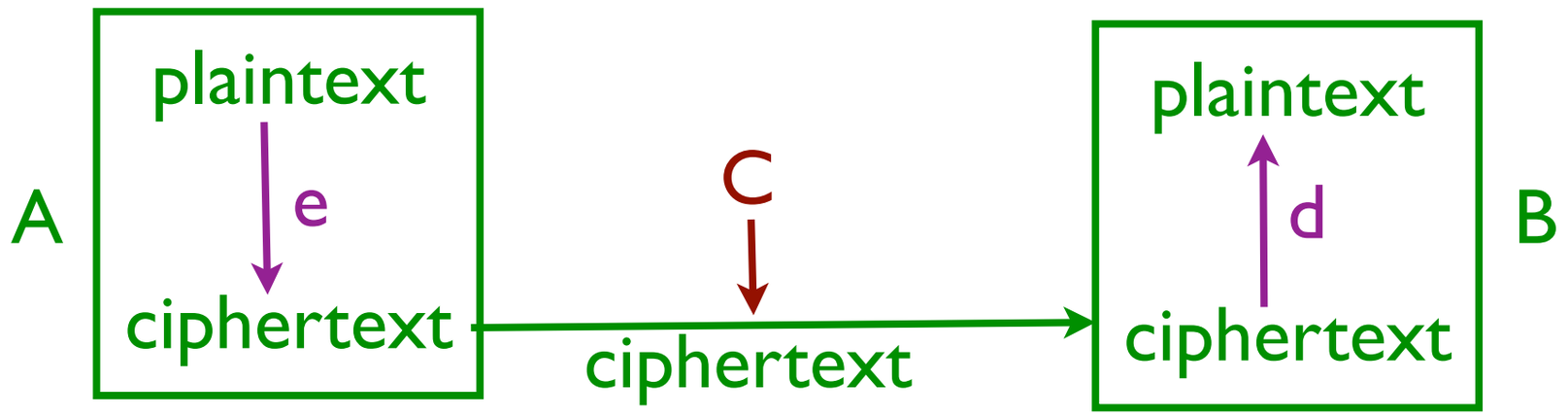
- Ad a):
    - Kein unbefugtes Lesen durch C
      → Geheimhaltung
    - Kein unerkanntes Abändern/Wiedereinspielen durch C
      → Integrität
    - Gefahr des Störens durch C reduzieren
      → Verfügbarkeit
- Ad b):
    - Kein Abstreiten des Versands durch A
    - Kein Abstreiten des Empfangs durch B
      → Nichtabstreitbarkeit
- In beiden Fällen: Keine unerkannte Maskerade zulassen
  → Authentisierung
- Lokale Verfahren nicht ausreichend (Probleme mit Passwortaustausch)

- Komplexere Methoden erforderlich

Ziel	Methode
• Geheimhaltung	• Verschlüsselung
• Integrität	• Verschlüsselung
• Authentisierung	• [Verschlüsselung] „digitale Unterschriften“
• Nichtabstreitbarkeit	• z.T. durch Authentisierung • auch Verfahren mit Einbeziehung von „Notaren“

Verschlüsselungstechnik (Grundtechnik für all das)

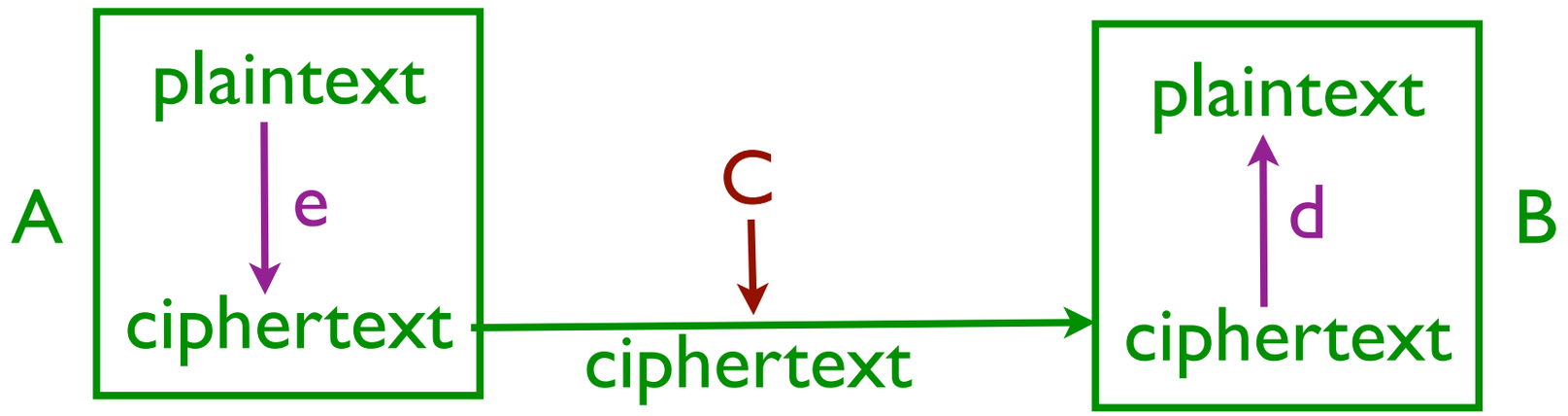
- Grundidee:



  - C darf Entschlüsselungsverfahren (d) nicht kennen
    - ➔ gelesener Ciphertext wertlos
    - ➔ Geheimhaltung erzielt
  - (C kann durchaus Wissen über e haben)

Verschlüsselungstechnik (Grundtechnik für all das)

- Grundidee:



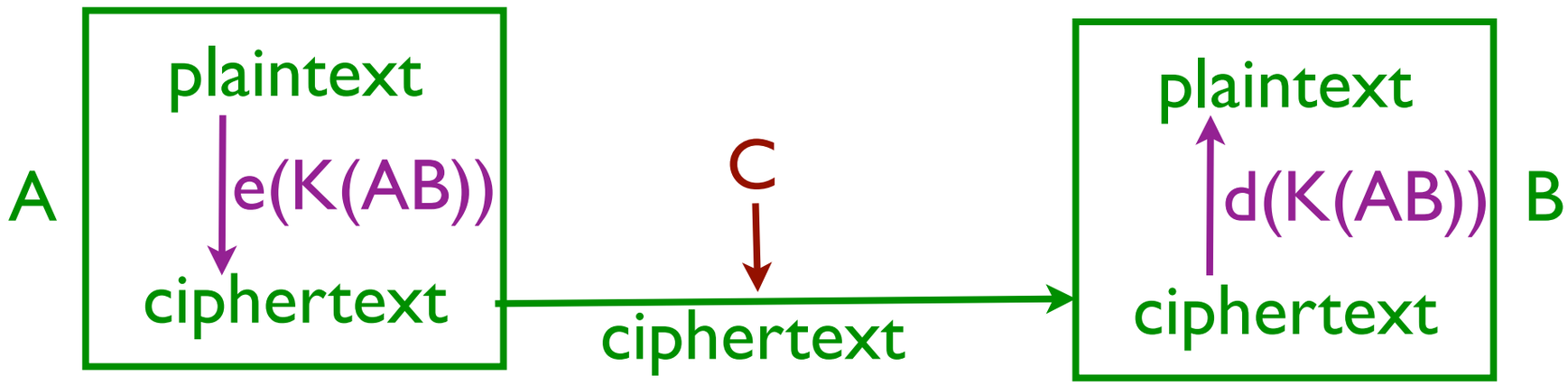
  - C darf Entschlüsselungsverfahren (d) nicht kennen
    - ➔ gelesener Ciphertext wertlos
    - ➔ Geheimhaltung erzielt
  - (C kann durchaus Wissen über e haben)

- Verfeinerung
  - Verfahren zur Ver-/Entschlüsselung bekannt
  - Aber durch Schlüssel parametrisiert
  - 2 Varianten: symmetrische/asymmetrische Verschlüsselung

Symmetrische Verschlüsselung

- A und B verwenden denselben Schlüssel (key) K(AB)

  ➔ Sitzungsschlüssel



- C darf diesen Schlüssel nicht kennen

  ➔ Schlüsselverteilungsproblem

  - Telefon, Kurier,...
  - spezielle Protokolle wie z.B. Diffie-Hellman...

Diffie-Hellman

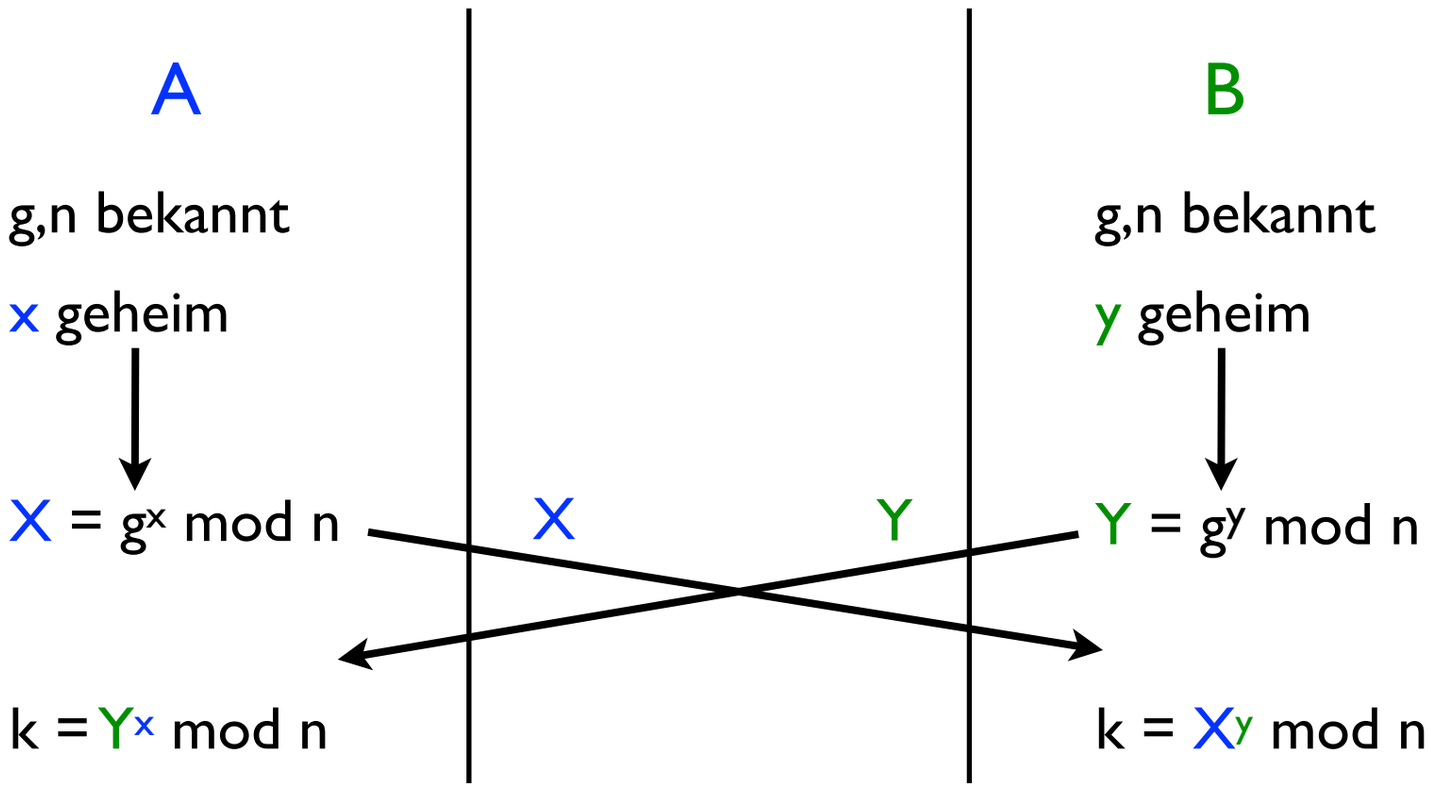
- Asymmetrisches Verfahren zur Verteilung von Sitzungsschlüsseln
- Kein Authentisierungsmechanismus ! (Kein Schutz gegen „Man-in-the-Middle-Angriff“)

Diffie-Hellman

- Asymmetrisches Verfahren zur Verteilung von Sitzungsschlüsseln
- Kein Authentisierungsmechanismus ! (Kein Schutz gegen „Man-in-the-Middle-Angriff“)

Grober Ablauf

- Zwei Zahlen g und n allgemein bekannt, g < n
- Jeder Partner denkt sich große, geheime Zahl aus, x bzw. y

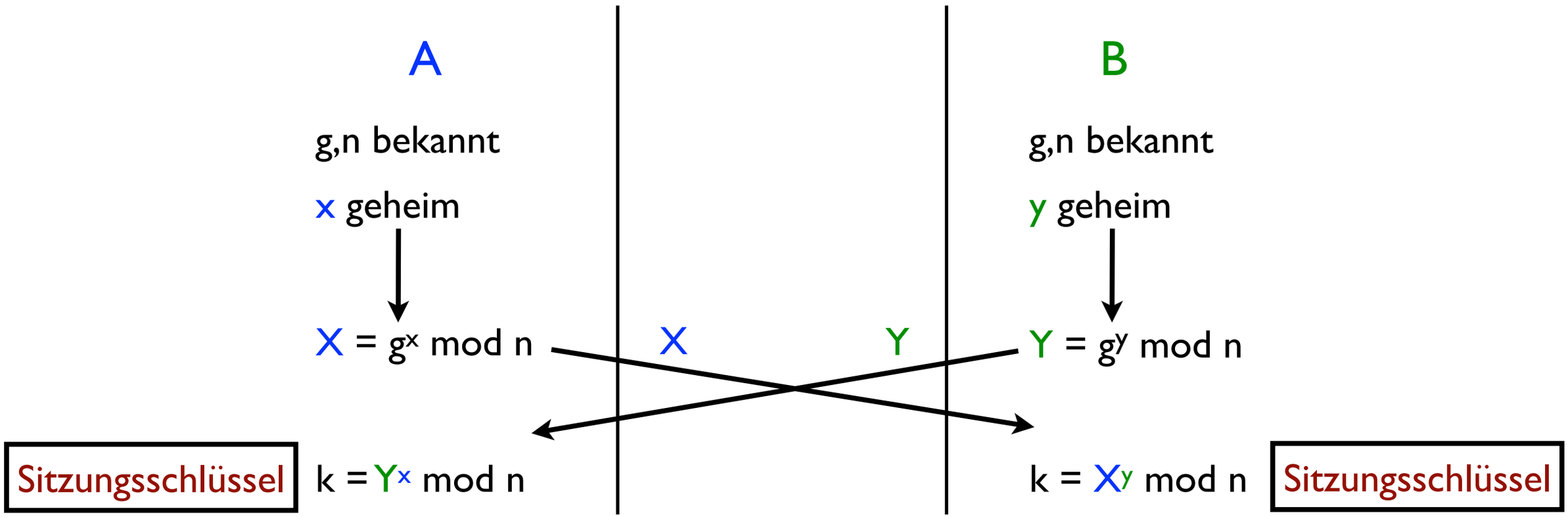


Diffie-Hellman

- Asymmetrisches Verfahren zur Verteilung von Sitzungsschlüsseln
- Kein Authentisierungsmechanismus ! (Kein Schutz gegen „Man-in-the-Middle-Angriff“)

Grober Ablauf

- Zwei Zahlen g und n allgemein bekannt, g < n
- Jeder Partner denkt sich große, geheime Zahl aus, x bzw. y



⇒ funktioniert, weil: $(g^x \bmod n)^y \bmod n = (g^y \bmod n)^x \bmod n$

⇒ x,y darf aus X,Y nicht herleitbar sein

- Beispiel (natürlich viel zu kleine Zahlen):

  g = 3 (bekannt)

  n = 7 (bekannt)

  A: x = 5 (geheim)

  B: y = 7 (geheim)

- Initial:

  A: $g^x \bmod n = 3^5 \bmod 7 = 243 \bmod 7 = 5 = X$

  B: $g^y \bmod n = 3^7 \bmod 7 = 2187 \bmod 7 = 3 = Y$

- Nach dem Austausch von X und Y:

  A: $Y^x \bmod n = 3^5 \bmod 7 = 243 \bmod 7 = 5 = k$

  B: $X^y \bmod n = 5^7 \bmod 7 = 78125 \bmod 7 = 5 = k$

Verfahren zur symmetrischen Verschlüsselung

- 2 klassische Verfahren (+ Varianten)

a) Substitutionschiffren (Ersetzungschiffren)

Beispiel-Chiffre

Øψ ◆⌟ψψØ⊗ ψ∇†¶ ⌟◆◆Ø

◆Ø§¥Ø ©§Ø∈ Ø∇⊗∇∂Ø

ςØ∨¥ §⊗℘ Ø∇⊗∇∂Ø ◆Ø§¥Ø

©§Ø∈ ⌟◆◆Ø ξØ∇¥, ⌟√Ø∈

⊗∇†¶¥ ⌟◆◆Ø ◆Ø§¥Ø ©§Ø∈

⌟◆◆Ø ξØ∇¥ ξ§♥ ⊗⌟∈∈Ø⊗

¶⌟◆¥Ø⊗.

- Häufigkeit von Buchstaben
  - Im Deutschen: E, N, I, S, R, A, D, T, H, U, ...
  - Im Beispiel:

- Kurze Wörter:
  - an am ab im in du da er es ob zu um Ei Au...
  - der den dem des die das dass

    wer wen wem wie was

    als aus auf und man ich sie ein...

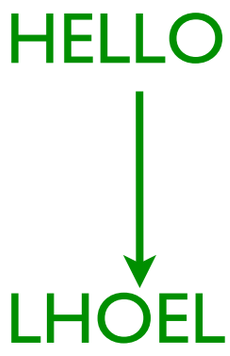
- Buchstabenkombinationen
  - ch, sch, st, ...cht(s), ...
  - Endungen: ...en, ...ion, ...eit, ...

Verfahren zur symmetrischen Verschlüsselung

- 2 klassische Verfahren (+ Varianten)

a) Substitutionschiffren (Ersetzungschiffren)

b) Transpositionschiffren (Umstellungschiffren)



Kleine Aufgabe

- Folgende Texte sind durch einfache Umstellungschiffren entstanden. Was bedeuten sie?

a) GÜNBU

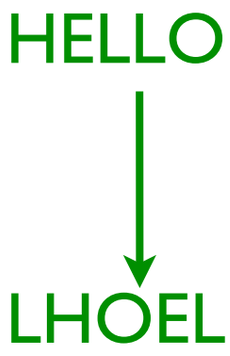
b) SCHACHGEFRAPE

Verfahren zur symmetrischen Verschlüsselung

- 2 klassische Verfahren (+ Varianten)

a) Substitutionschiffren (Ersetzungschiffren)

b) Transpositionschiffren (Umstellungschiffren)

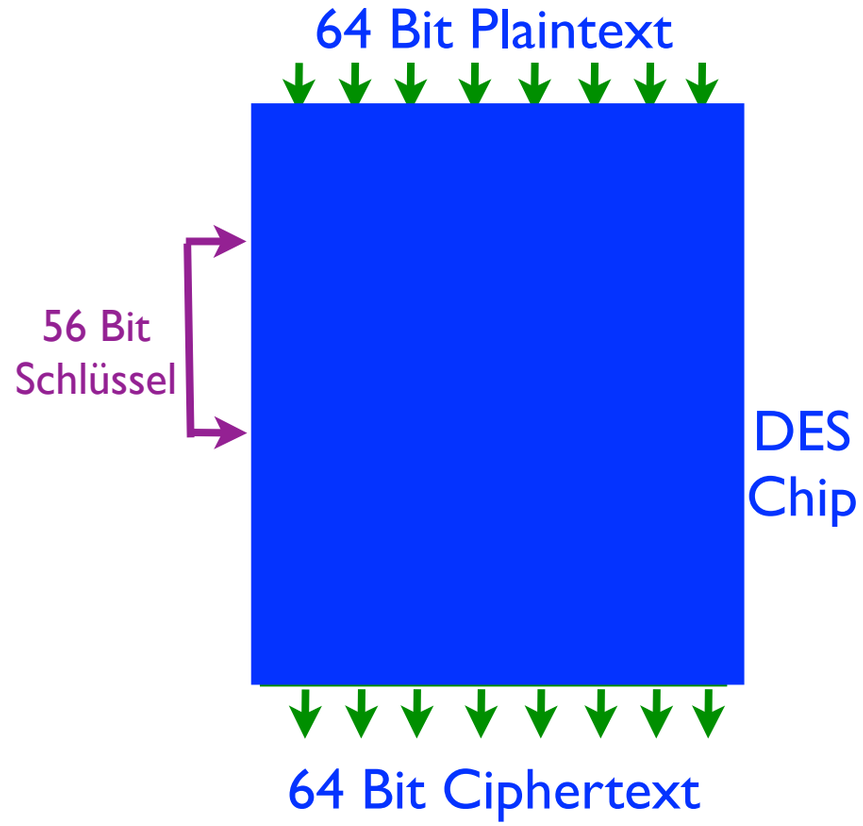


➔ viel zu leicht zu knacken:

- durchprobieren
- Häufigkeitsverteilung von Buchstaben/Wörtern
- teilweise bekannter Klartext

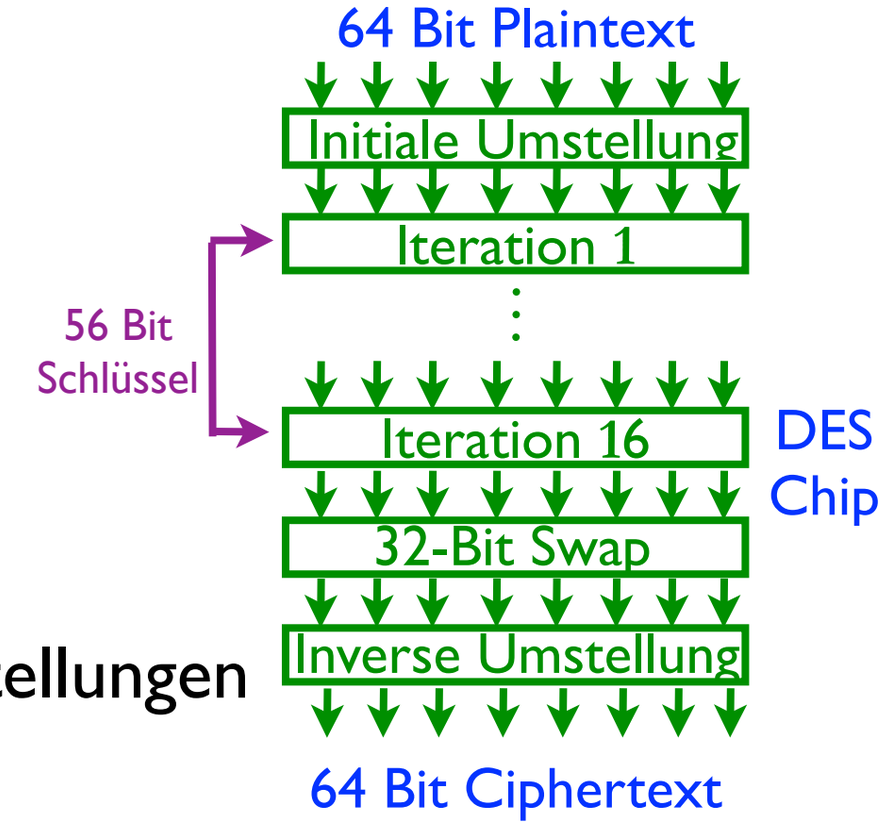
DES (Data Encryption Standard, 1976)

- Bestimmte Kombination von Umstellungen und Ersetzungen
- Grundsätzliche Funktionsweise bekannt
- Über 56-Bit-Schlüssel parametrisiert
- Im Grundsatz Verschlüsselung von 64-Bit-Blöcken



DES (Data Encryption Standard, 1976)

- Bestimmte Kombination von Umstellungen und Ersetzungen
- Grundsätzliche Funktionsweise bekannt
- Über 56-Bit-Schlüssel parametrisiert
- Im Grundsatz Verschlüsselung von 64-Bit-Blöcken



Jede der Iterationen:

- Linke Hälfte: rechte Hälfte des vorigen Schritts
- Rechte Hälfte: Bestimmte Kombination von Umstellungen und Ersetzungen. Grundlage:
  - veränderter Schlüssel
  - rechte Hälfte des vorigen Schritts
  - sowie linke Hälfte des vorigen Schritts damit XORen
- Vergleichsweise einfach zu realisieren
  → in Realzeit durchführbar

- Allerdings 56-Bit-Schlüssel (mittlerweile) viel zu kurz
  → DES ist zu knacken…

Seit Okt. 2000: AES (Advanced Encryption Standard)

- Weiterhin bestimmte (andere) Kombination von Umstellungen und Ersetzungen
- 128-Bit-Blöcke verschlüsseln
- Parametrisiert mit 128/192/256-Bit-Schlüssel

DES/AES in mehreren Arbeitsmodi verwendbar:

a) Block-Modus

- Klartext in 64/128-Bit-Blöcke aufteilen
- Jeder Block wird isoliert über DES/AES verschlüsselt

  ⇒ Wiederholte Muster erkennbar

DES/AES in mehreren Arbeitsmodi verwendbar:

a) Block-Modus

- Klartext in 64/128-Bit-Blöcke aufteilen
- Jeder Block wird isoliert über DES/AES verschlüsselt

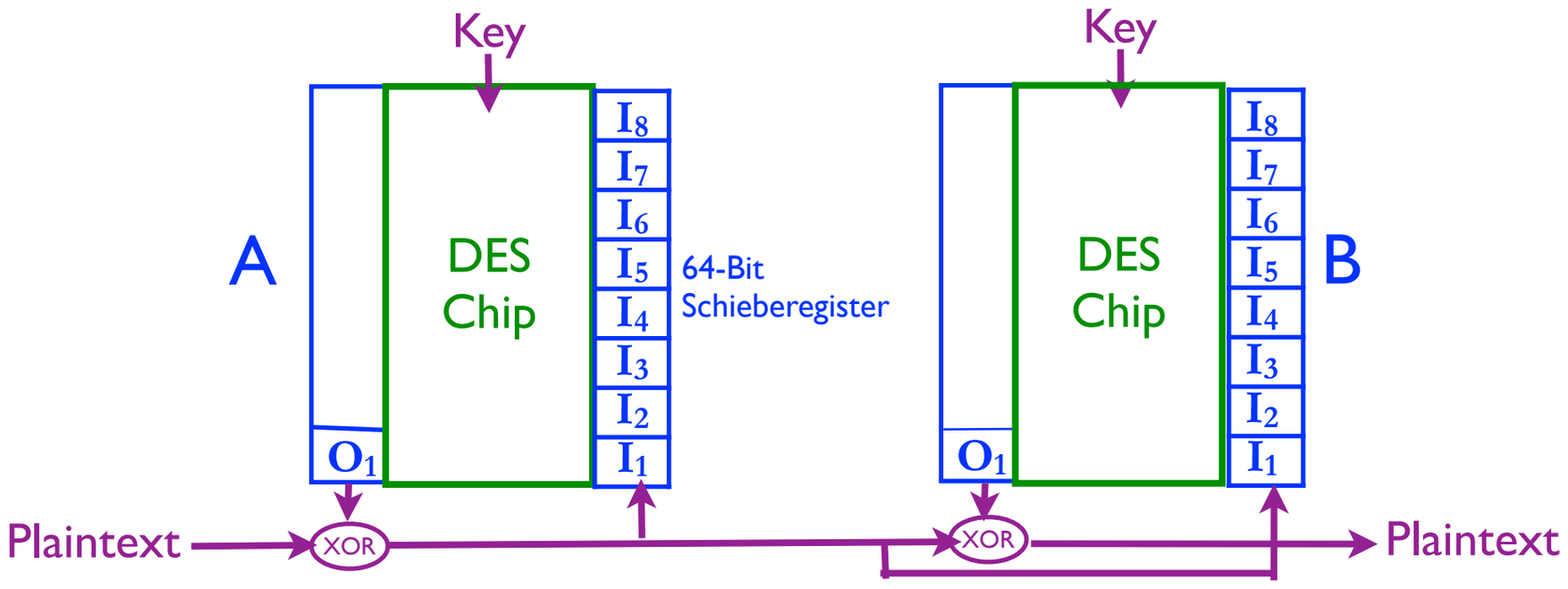
  ⇒ Wiederholte Muster erkennbar

b) Strom-Modus

- Eingabebytes durchlaufen ein Schieberegister
- Ausgabebyte O1 wird in Eingabestrom zurückgekoppelt (XOR-Verknüpfung)

  ⇒ O1 basiert auf gesamter „Geschichte“

  ⇒ Keine Erkennung von Mustern möglich (insbesondere bei Wahl eines geeigneten Initialisierungsvektors IV)



Fragen – Teil 2

- Charakterisiere *symmetrische* Verschlüsselungsverfahren. Wie können sie zur Realisierung einer Vertraulichkeit eingesetzt werden?
- Was ist Diffie-Hellman?

Teil 3:
Asymmetrische Verschlüsselung

Asymmetrische Verschlüsselung

- Ver- und Entschlüsselung mit verschiedenen Schlüsseln

  ⇒ jeder Teilnehmer hat eigenes Schlüsselpaar:

  - öffentlicher Schlüssel (public key)
  - geheimer Schlüssel (secret key)

Asymmetrische Verschlüsselung

- Ver- und Entschlüsselung mit verschiedenen Schlüsseln

  ⇒ jeder Teilnehmer hat eigenes Schlüsselpaar:

  - öffentlicher Schlüssel (public key)
  - geheimer Schlüssel (secret key)

⇒ Verschlüsseln mit dem einen.
Entschlüsseln mit dem anderen.

Asymmetrische Verschlüsselung

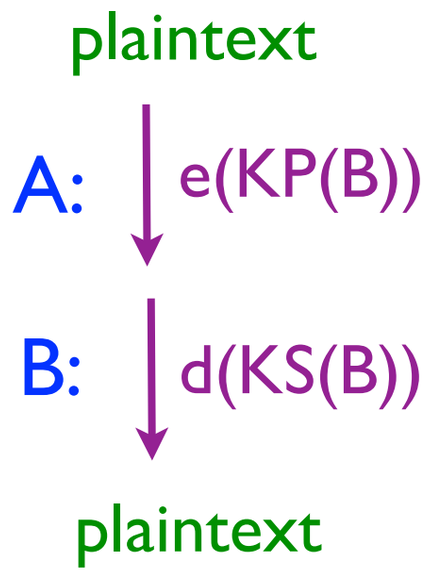
- Ver- und Entschlüsselung mit verschiedenen Schlüsseln
  ⇒ jeder Teilnehmer hat eigenes Schlüsselpaar:
  - öffentlicher Schlüssel (public key)
  - geheimer Schlüssel (secret key)

- Geheimhaltung von Nachricht A→B ?
  ⇒ Nutzung des Schlüsselpaars von B
  - Verschlüsselung (bei A): öffentlicher Schlüssel von B ⇒ KP(B)
  - Entschlüsselung (bei B): geheimer Schlüssel von B ⇒ KS(B)

Asymmetrische Verschlüsselung

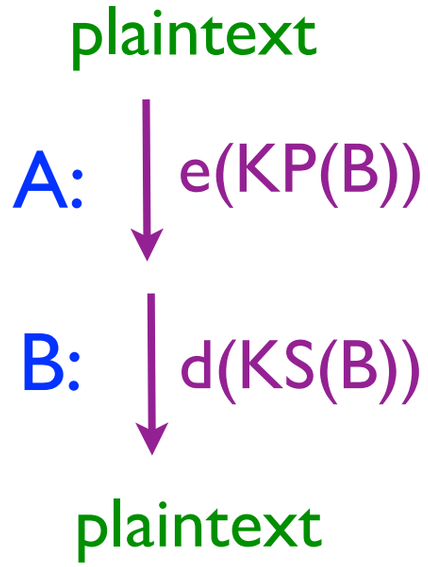
- Ver- und Entschlüsselung mit verschiedenen Schlüsseln
  ⇒ jeder Teilnehmer hat eigenes Schlüsselpaar:
  - öffentlicher Schlüssel (public key)
  - geheimer Schlüssel (secret key)
- Geheimhaltung von Nachricht A→B ?
  ⇒ Nutzung des Schlüsselpaars von B
  - Verschlüsselung (bei A): öffentlicher Schlüssel von B ⇒ KP(B)
  - Entschlüsselung (bei B): geheimer Schlüssel von B ⇒ KS(B)
- Schlüsselverteilproblem?
  - Nur Eigentümer darf geheimen Schlüssel besitzen (z.B. auf Chipkarte)
  - Jeder darf öffentlichen Schlüssel besitzen
    ⇒ Ist er korrekt? ⇒ z.B. Nutzung von Zertifikaten (später)

- 2 Varianten der Anwendung der Schlüssel:

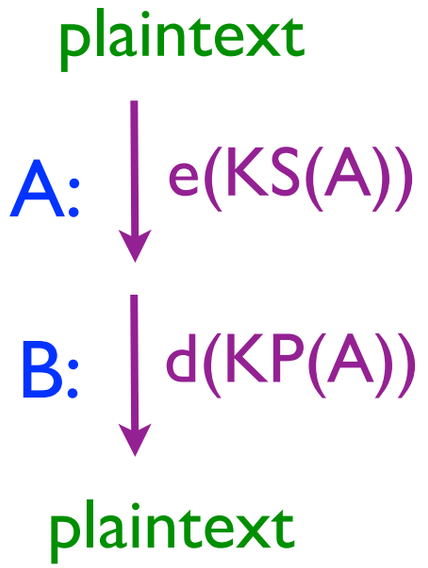


⇒ Geheimhaltung (A ➔ B)

- 2 Varianten der Anwendung der Schlüssel:

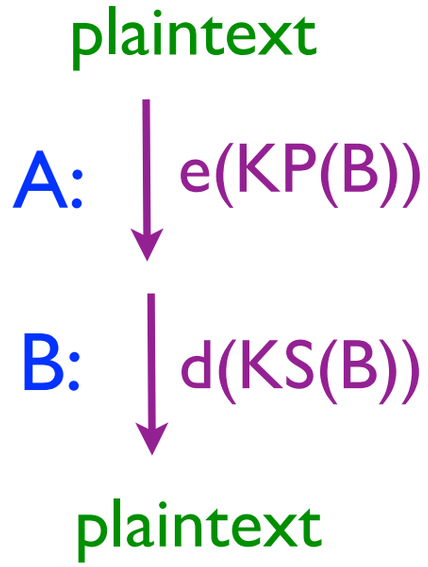


⇒ Geheimhaltung (A ➔ B)

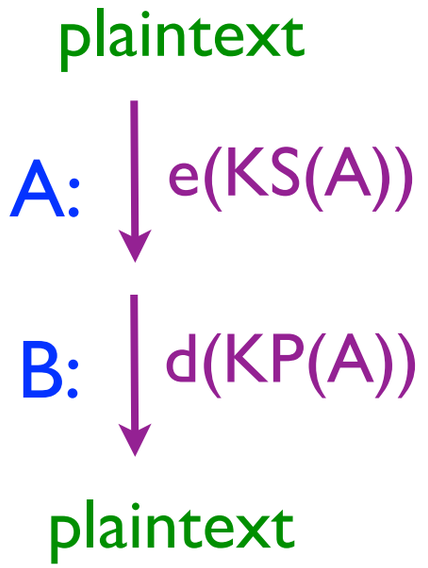


⇒ Integrität (A ➔ B)
Authentisierung (von A)

- 2 Varianten der Anwendung der Schlüssel:



⇒ Geheimhaltung (A → B)



⇒ Integrität (A → B)
Authentisierung (von A)

- Wichtiger klassischer Beispielalgorithmus:
  RSA (Rivest, Shamir, Adleman)
  - Basiert auf der Schwierigkeit, sehr große Zahlen in Primfaktoren zu zerlegen (Zahlentheorie)
  - Allerdings relativ aufwendig/langsam
- Auch alternative Algorithmen (z.B. ECC-Varianten...)

Grobe Arbeitsweise des RSA-Algorithmus:

- p und q sind Primzahlen ⇒ bei Schlüsselerstellung auswählen
- n = p×q, z = (p-1)×(q-1)
- d ist Zahl, die keine gemeinsamen Primfaktoren mit z hat
  ⇒ bei Schlüsselerstellung auswählen
- e auswählen mit (e×d)mod z = 1 ⇒ bei Schlüsselerstellung auswählen

Grobe Arbeitsweise des RSA-Algorithmus:

- p und q sind Primzahlen ⇒ bei Schlüsselerstellung auswählen
- $n = p \times q$, $z = (p-1) \times (q-1)$
- d ist Zahl, die keine gemeinsamen Primfaktoren mit z hat
  ⇒ bei Schlüsselerstellung auswählen
- e auswählen mit $(e \times d) \bmod z = 1$ ⇒ bei Schlüsselerstellung auswählen

- Verschlüsselung: $C = P^e \pmod n$
- Entschlüsselung: $P = C^d \pmod n$
- Klartext wird als binär kodierte ganze Zahl interpretiert:
  - z.B. P = 10011000 = 128 + 16 + 8 = 152
  - P < n, d.h. max. k Bits auf einmal verschlüsseln mit $2^k < n$

Grobe Arbeitsweise des RSA-Algorithmus:

- p und q sind Primzahlen ⇒ bei Schlüsselerstellung auswählen
- $n = p \times q$, $z = (p-1) \times (q-1)$
- d ist Zahl, die keine gemeinsamen Primfaktoren mit z hat
  ⇒ bei Schlüsselerstellung auswählen
- e auswählen mit $(e \times d) \bmod z = 1$ ⇒ bei Schlüsselerstellung auswählen

- Verschlüsselung: $C = P^e \pmod n$
- Entschlüsselung: $P = C^d \pmod n$
- Klartext wird als binär kodierte ganze Zahl interpretiert:
  - z.B. P = 10011000 = 128 + 16 + 8 = 152
  - P < n, d.h. max. k Bits auf einmal verschlüsseln mit $2^k < n$

$$((P^e \bmod n)^d \bmod n) = P$$

Grobe Arbeitsweise des RSA-Algorithmus:

- p und q sind Primzahlen ⇒ bei Schlüsselerstellung auswählen
- $n = p \times q$, $z = (p-1) \times (q-1)$
- d ist Zahl, die keine gemeinsamen Primfaktoren mit z hat
  ⇒ bei Schlüsselerstellung auswählen
- e auswählen mit $(e \times d) \bmod z = 1$ ⇒ bei Schlüsselerstellung auswählen

- Verschlüsselung: $C = P^e \pmod n$
- Entschlüsselung: $P = C^d \pmod n$
- Klartext wird als binär kodierte ganze Zahl interpretiert:
  - z.B. P = 10011000 = 128 + 16 + 8 = 152
  - P < n, d.h. max. k Bits auf einmal verschlüsseln mit $2^k < n$
- Öffentlicher Schlüssel: (n,e)
- Geheimer Schlüssel: (n,d)

$$((P^e \bmod n)^d \bmod n) = P$$

Grobe Arbeitsweise des RSA-Algorithmus:

- p und q sind Primzahlen ⇒ bei Schlüsselerstellung auswählen
- $n = p \times q$, $z = (p-1) \times (q-1)$
- d ist Zahl, die keine gemeinsamen Primfaktoren mit z hat
  ⇒ bei Schlüsselerstellung auswählen
- e auswählen mit $(e \times d) \bmod z = 1$ ⇒ bei Schlüsselerstellung auswählen

- Verschlüsselung: $C = P^e \pmod n$
- Entschlüsselung: $P = C^d \pmod n$
- Klartext wird als binär kodierte ganze Zahl interpretiert:
  - z.B. P = 10011000 = 128 + 16 + 8 = 152
  - P < n, d.h. max. k Bits auf einmal verschlüsseln mit $2^k < n$
- Öffentlicher Schlüssel: (n,e)
- Geheimer Schlüssel: (n,d)

$$((P^e \bmod n)^d \bmod n) = P$$
$$((P^d \bmod n)^e \bmod n) = P$$

Grobe Arbeitsweise des RSA-Algorithmus:

- p und q sind Primzahlen ⇒ bei Schlüsselerstellung auswählen
- $n = p \times q$, $z = (p-1) \times (q-1)$
- d ist Zahl, die keine gemeinsamen Primfaktoren mit z hat
  ⇒ bei Schlüsselerstellung auswählen
- e auswählen mit $(e \times d) \bmod z = 1$ ⇒ bei Schlüsselerstellung auswählen

- Verschlüsselung: $C = P^e \pmod n$
- Entschlüsselung: $P = C^d \pmod n$
- Klartext wird als binär kodierte ganze Zahl interpretiert:
  - z.B. P = 10011000 = 128 + 16 + 8 = 152
  - P < n, d.h. max. k Bits auf einmal verschlüsseln mit $2^k < n$
- Öffentlicher Schlüssel: (n,e)
- Geheimer Schlüssel: (n,d)

  ⇒ Aus n lassen sich p, q und damit z (und d) nicht ohne weiteres herleiten, wenn Zahlen groß genug gewählt werden

  ⇒ Ingenieurmäßiger Ansatz
  (Frage der Rechnerleistung und der Größe der Zahlen)

- Viel zu einfaches Beispiel
  (entlehnt aus „A.S. Tanenbaum: Computer Networks“):

  $p=3, q=11 \Rightarrow n=33, z=20$

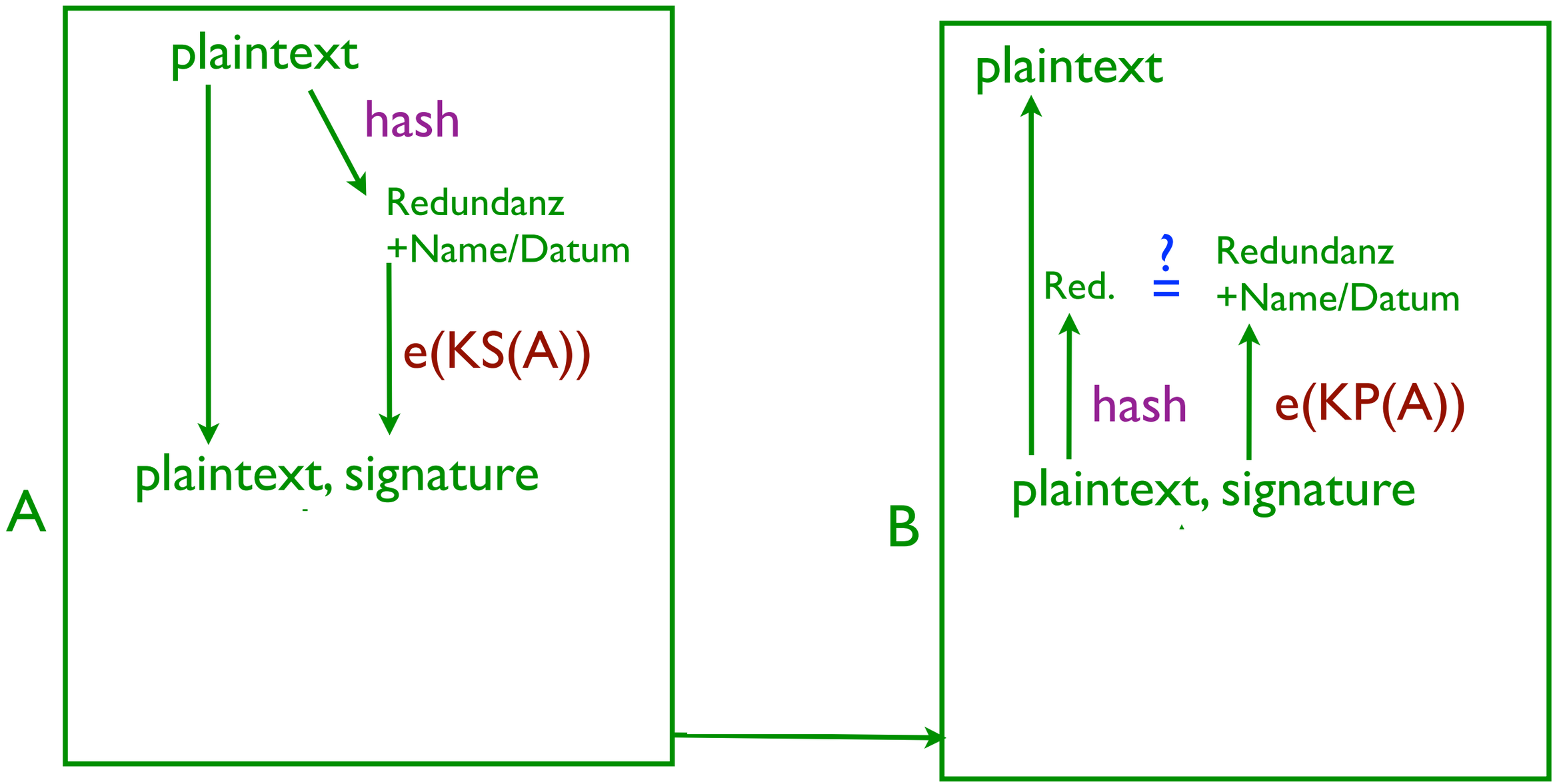
  Wahl von z.B. $d=7$, $e=3$, da $(7\times3) \bmod 20 = 1$

  $\Rightarrow P<33$, also 5-Bit-weise verschlüsseln ($2^5=32$)

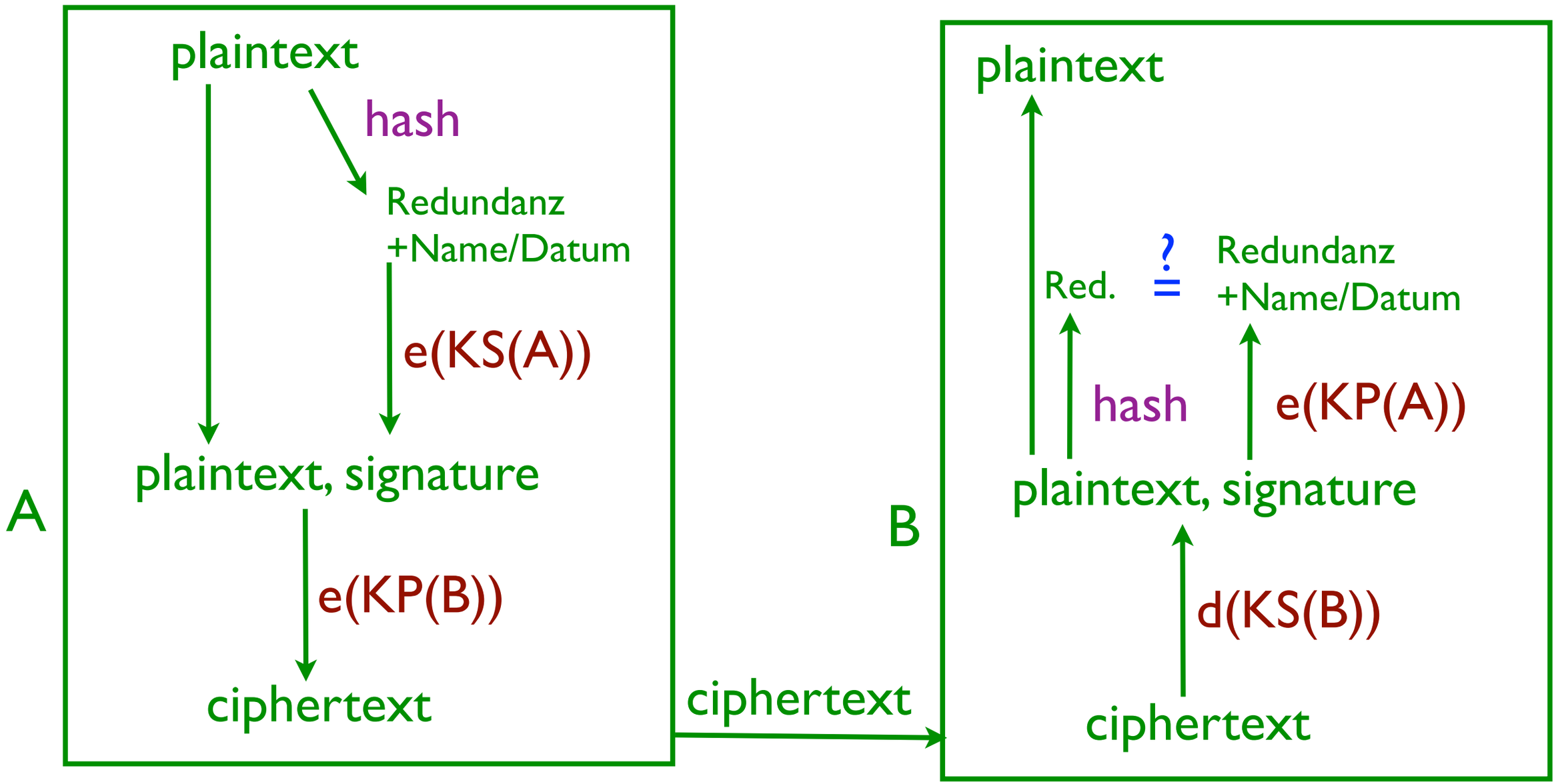
  $\Rightarrow$ mit diesen viel zu kleinen Zahlen letztlich nur einfache monoalphabetische Ersetzungschiffre

plaintext P			ciphertext C				plaintext P
	5-Bit-Buchstaben-nummer	P^3	$P^3 \pmod{33}$		C^7	$C^7 \pmod{33}$	
A	1	1	1	A	1	1	A
N	14	2744	5	E	78125	14	N
N	14	2744	5	E	78125	14	N
E	5	125	26	Z	8031810176	5	E

Digitale Signatur/Unterschrift

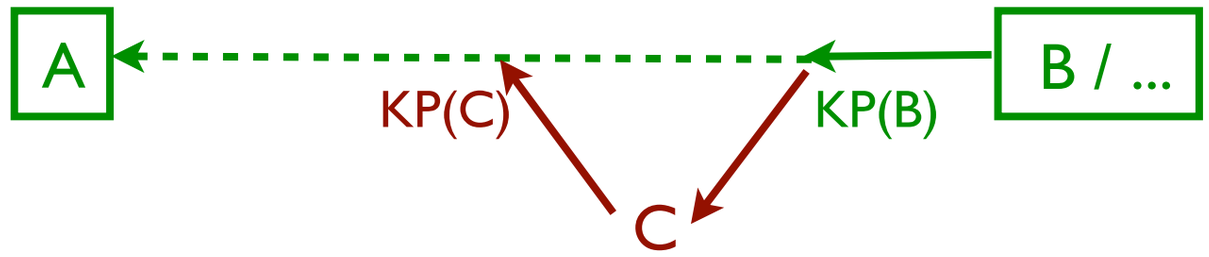


Kombination von Geheimhaltung und Integrität/Authentisierung



Zertifikate

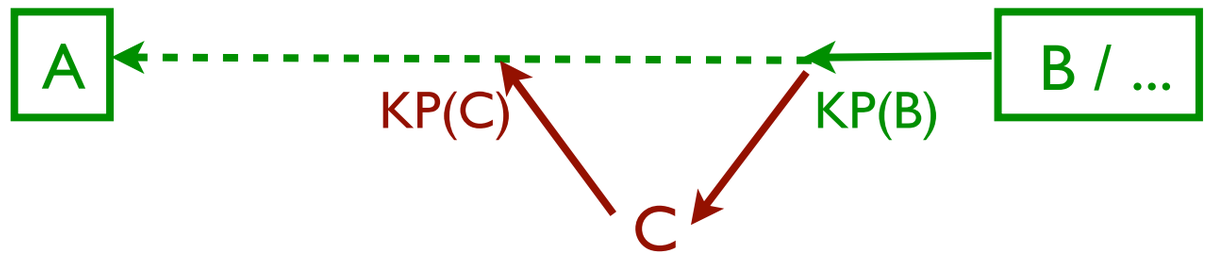
- Woher weiß man, dass ein öffentlicher Schlüssel korrekt ist?



KP(B) muss nicht direkt
von B gesendet worden sein

Zertifikate

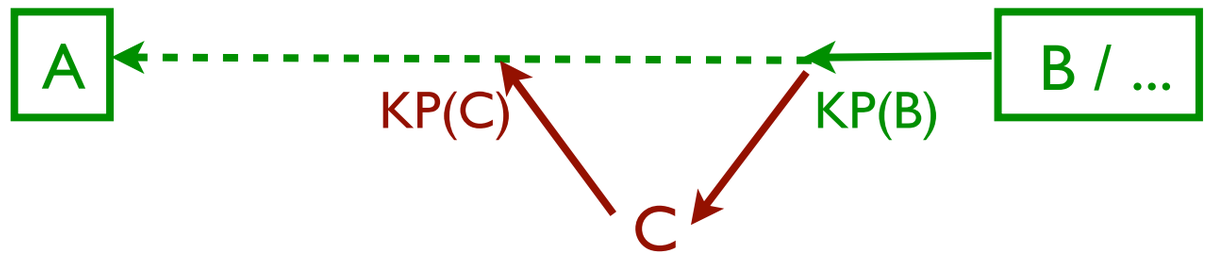
- Woher weiß man, dass ein öffentlicher Schlüssel korrekt ist?



- Vertrauenswürdige Instanz zertifiziert Zuordnung Schlüssel → Eigentümer, indem sie diese mit einer digitalen Unterschrift versehen bestätigt

Zertifikate

- Woher weiß man, dass ein öffentlicher Schlüssel korrekt ist?



- Vertrauenswürdige Instanz zertifiziert Zuordnung Schlüssel → Eigentümer, indem sie diese mit einer digitalen Unterschrift versehen bestätigt

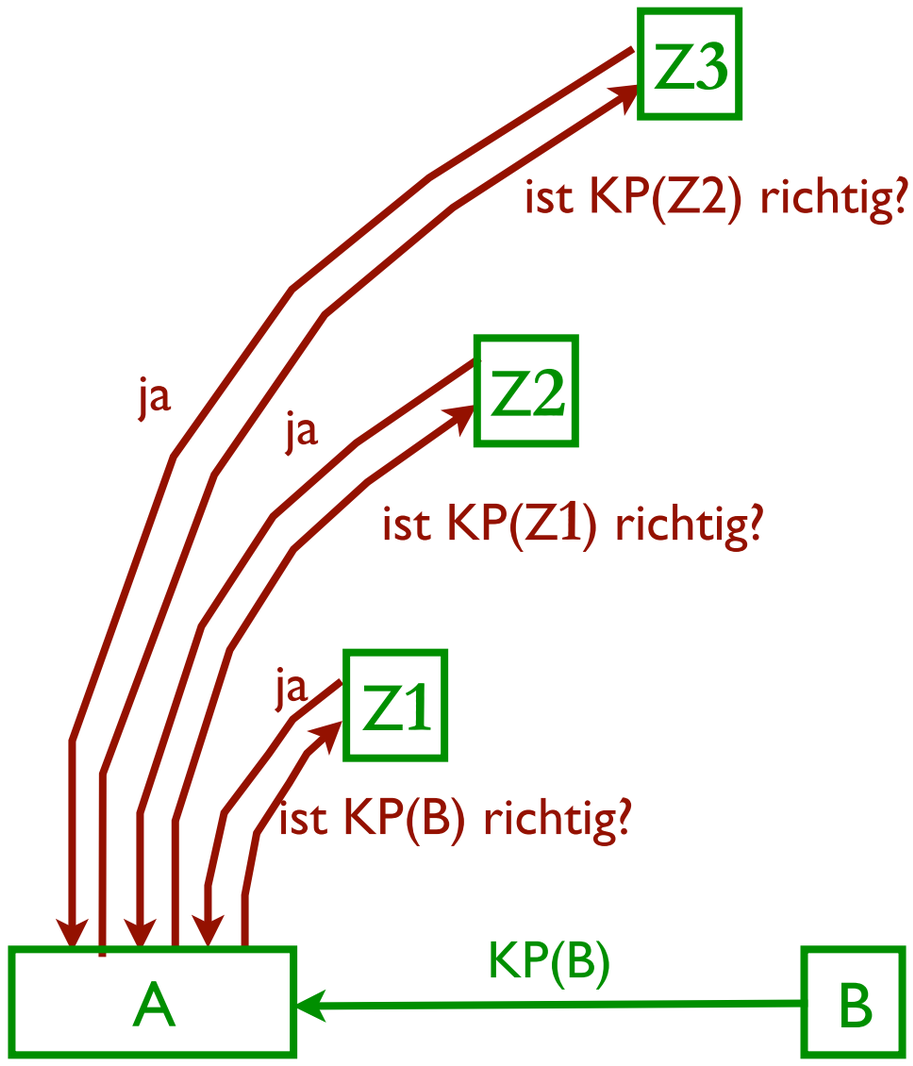
- Woher weiß man, dass öffentlicher Schlüssel von Zertifikatsstelle korrekt ist?

  ⇒ Schlüssel muss extrem öffentlich gemacht werden

  ⇒ bekannte Zertifikate (z.B. von Zertifizierungsstellen) werden mit entsprechender Software mitgeliefert (z.B. Browser)

  ⇒ Software aus vertrauenswürdiger Quelle nutzen...

- Verfeinerung: Hierarchie von Zertifizierungsstellen, die untergeordnete Zertifikate zertifizieren



- Zertifikate müssen nicht unbedingt explizit von den Zertifikatsstellen abgefragt werden

  ⇒ Können auch im Verzeichnis/Browser abgelegt worden sein (digital unterschrieben)

- Probleme Zertifikate:
  - Aufwändige Infrastruktur nötig (Zertifizierungsstellen...)
  - Nur sicher, wenn Zertifikatsketten vollständig und korrekt sind

- Probleme Zertifikate:
  - Aufwändige Infrastruktur nötig (Zertifizierungsstellen...)
  - Nur sicher, wenn Zertifikatsketten vollständig und korrekt sind

- Alternativ:
  - Bei unbekannten Public Keys Nutzer einbeziehen
  - Überprüfung über „andere Wege“

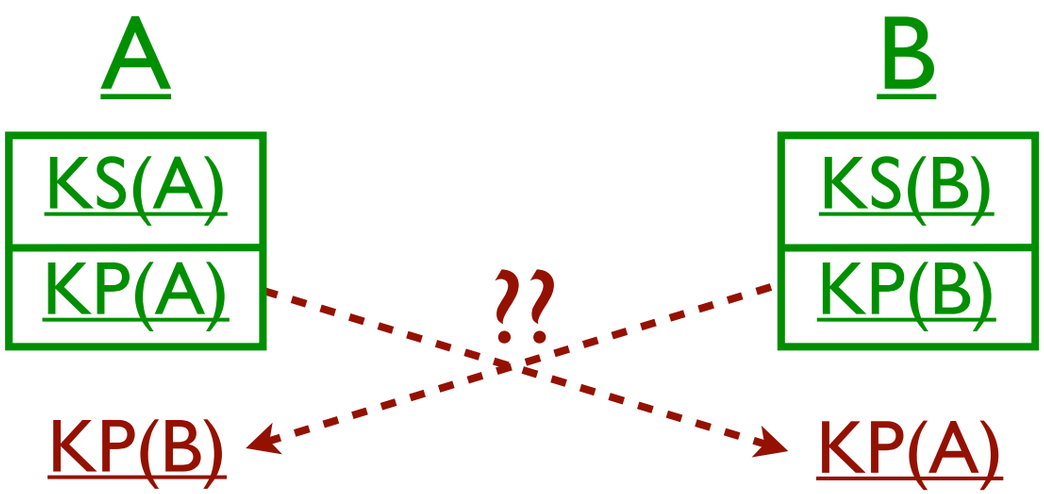
Zusammenfassung

- Symmetrische Verschlüsselung:

⇒ Problem: Geheimhaltung der Schlüsselverteilung

⇒ z.B. Diffie-Hellman, ...

- Asymmetrische Verschlüsselung:



⇒ Problem: Integrität der Schlüsselverteilung

⇒ z.B. Zertifikate, ...

Hybride Verfahren

- Nachteile symmetrische Verschlüsselung:
  - Bei Schlüsselverteilung Geheimhaltung sicherstellen
  - u.U. keine Initial-Authentisierung
- Nachteile asymmetrische Verschlüsselung:
  - Bei Schlüsselverteilung Integrität sicherstellen
  - Ver-/Entschlüsselung sehr komplex (Ressourcenbedarf, Dauer)
  - Angreifbarkeit der komplexen Schlüssel bei extensiver Nutzung

⇒ Kombination symmetrischer und asymmetrischer Verfahren:

- Austausch Sitzungsschlüssel mit Diffie-Hellman
- Authentisierung mit RSA/... (bzw. Server: RSA, Client: Passwort)
- Verschlüsselung mit AES

⇒ Ähnliche Kombinationen vielfach im Einsatz (z.B. https:....)

Fragen – Teil 3

- Charakterisiere *asymmetrische* Verschlüsselungsverfahren. Wie können sie zur Realisierung einer Vertraulichkeit eingesetzt werden? Warum werden häufig Mischformen von symmetrischen und asymmetrischen Verfahren eingesetzt?
- Was sind Zertifikate?
- Wie kann eine *digitale Unterschrift* erzeugt werden?

Zusammenfassung

- Angriffe, Sicherheitsziele, Sicherheitspolitik
- Lokale Zugangspolitik:
  - Authentisierung und Autorisierung
- Sicherheit in Rechnernetzen:
  - Symmetrische Verschlüsselung:
    - Schlüsselaustausch mit Diffie-Hellman
    - DES, AES
  - Asymmetrische Verschlüsselung:
    - Zertifikate
    - Digitale Signaturen
    - RSA
  - Hybride Verfahren

Informationssicherheit – Fragen

1. Nenne einige absichtliche und unabsichtliche Angriffe auf Hardware, Software und/oder Daten.
2. Welche grundsätzlichen *Sicherheitsziele* kann man unterscheiden?
3. Was ist eine *Sicherheitspolitik*?
4. Auf welche verschiedenen Arten kann sich ein Benutzer authentifizieren?
5. Welche Komponenten enthält eine Zugriffskontrollmatrix? Wie ordnen sich die Dateizugriffsrechte in Unix in dieses Schema ein?
6. Charakterisiere *symmetrische* und *asymmetrische* Verschlüsselungsverfahren. Wie können sie zur Realisierung einer Vertraulichkeit eingesetzt werden? Warum werden häufig Mischformen eingesetzt?
7. Was ist Diffie-Hellman?
8. Was sind Zertifikate?
9. Wie kann eine *digitale Unterschrift* erzeugt werden?