Vertrauenswürdige Kommunikation in verteilten Systemen

Teil I

- Kryptographische Grundlagen
- Vertrauensmodelle
- Kerberos

Teil II

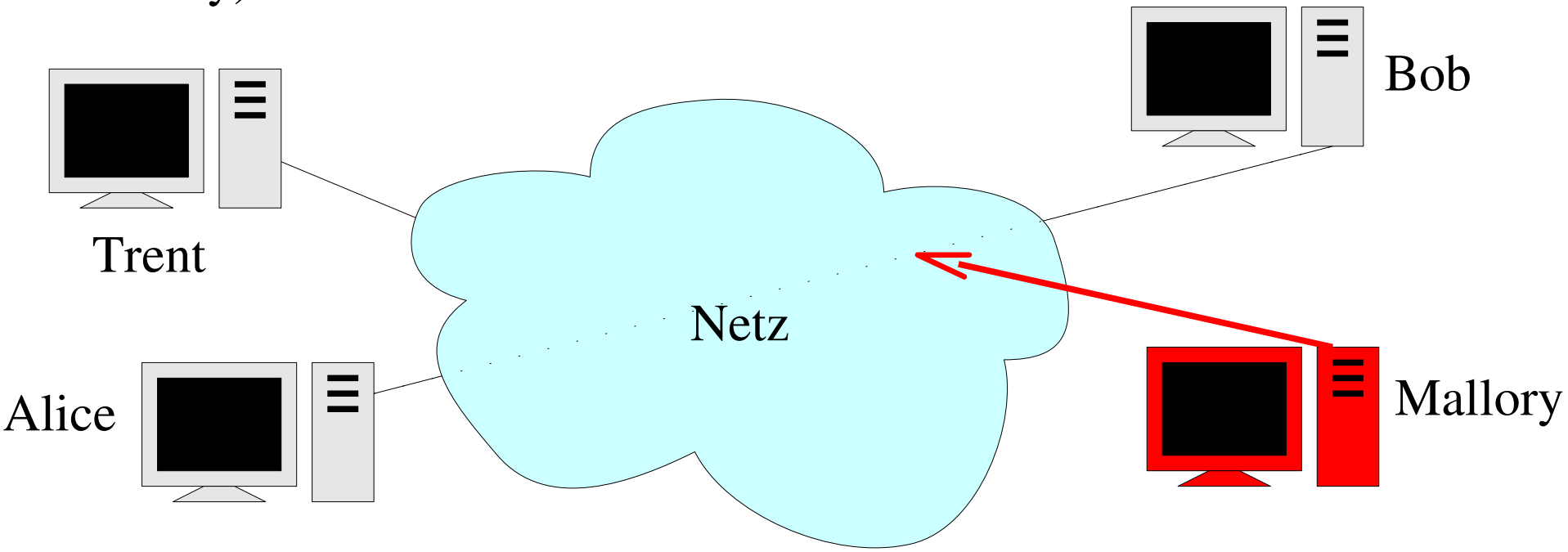
- IPSec
- AH/ESP
- IKE

Szenario

Alice möchte Bob „vertraulich“ eine Nachricht (typischerweise eine Anweisung) zukommen lassen

Mallory möchte die Nachricht lesen bzw. Bob eine Anweisung anstelle von Alice erteilen

manchmal gibt es Trent; er ist Alice und Bob (und manchmal auch Mallory) bekannt und kann vermitteln



Szenario

- Alice und Bob (und alle anderen) müssen nicht Menschen sein
  - evtl. Programme, die „im Auftrag“ von Menschen mehr oder weniger autonom kommunizieren
    - Alice: Texteditor; möchte Datei speichern
    - Bob: Fileserver; weiß dass Alice Datei schreiben darf
    - Mallory: Darf Inhalt der Datei nicht kennen und nicht verändern können

Szenario

Annahmen:

- Computer von Alice, Bob und Trent sind vertrauenswürdig
  - Mallory hat keine Kontrolle über einen dieser Rechner bekommt
- Mallory ist (was Kommunikation betrifft) allmächtig
  - kann beliebig viele Nachrichten in beliebiger Geschwindigkeit mit beliebigem Inhalt empfangen und versenden, von und zu jedem beliebigen Rechner

Welche der folgenden Komponenten sind **unentbehrlich...**

... damit Alice zu Bob eine „sichere“ Kommunikation aufbauen kann? **Raterunde!!**

- ein geswitchtes Netzwerk (im ggs. zu Hubs)
- ein Verschlüsselungsalgorithmus (z.B. AES)
- ein Prüfsummenalgorithmus (z.B. MD5)
- eine „Firewall“ (intern oder extern)
- ein Zufallszahlengenerator
- vorherige, bereits gesicherte Kommunikation

Welche der folgenden Komponenten sind **unentbehrlich...**

Die richtige Lösung: *überrascht?*

- ein geswitchtes Netzwerk (im ggs. zu Hubs)
- ein Verschlüsselungsalgorithmus (z.B. AES)
- ein Prüfsummenalgorithmus (z.B. MD5)
- eine „Firewall“ (intern oder extern)
- ein Zufallszahlengenerator
- vorherige, bereits gesicherte Kommunikation

Welche der folgenden Komponenten sind **unentbehrlich...**

Die richtige Lösung: *überrascht?*

- ein geswitchtes Netzwerk (im ggs. zu Hubs)
- ein Verschlüsselungsalgorithmus (z.B. AES)
- ein Prüfsummenalgorithmus (z.B. MD5)
- eine „Firewall“ (intern oder extern)
- ein Zufallszahlengenerator
- vorherige, bereits gesicherte Kommunikation*

*direkt miteinander, oder indirekt mit vertrauenswürdigem Drittem

Zentraler Merksatz

Verschlüsselung ist *nebensächlich-*

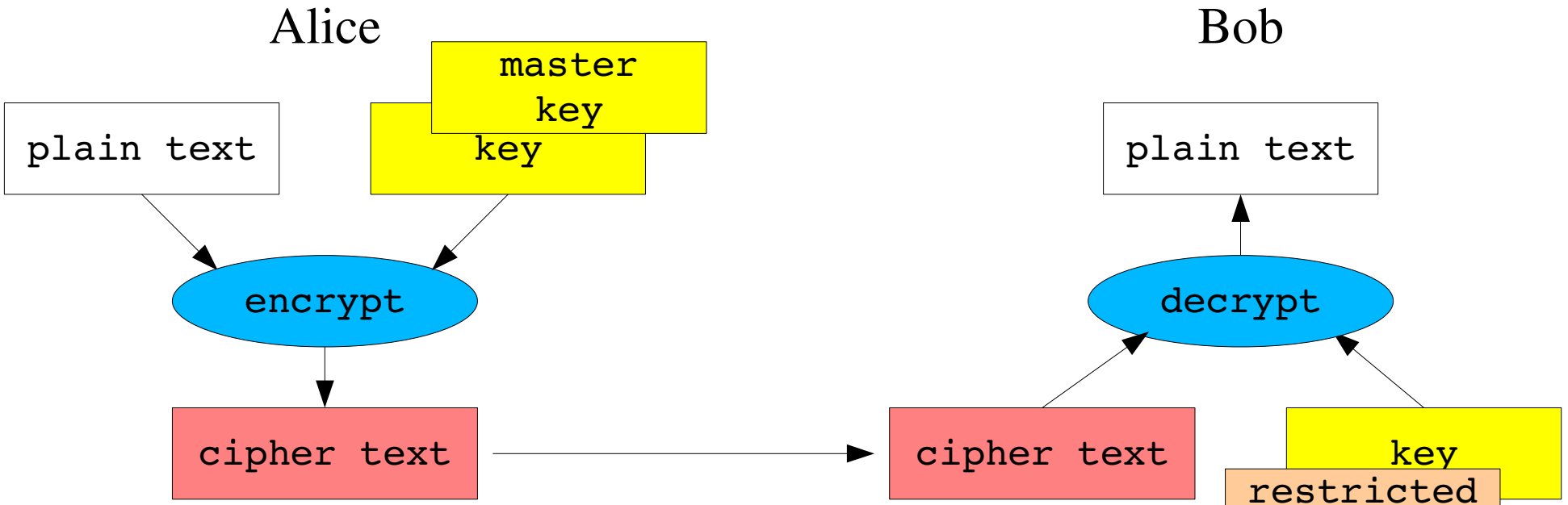
Authentisierung ist von entscheidender Bedeutung.

1 Kryptographische Grundlagen

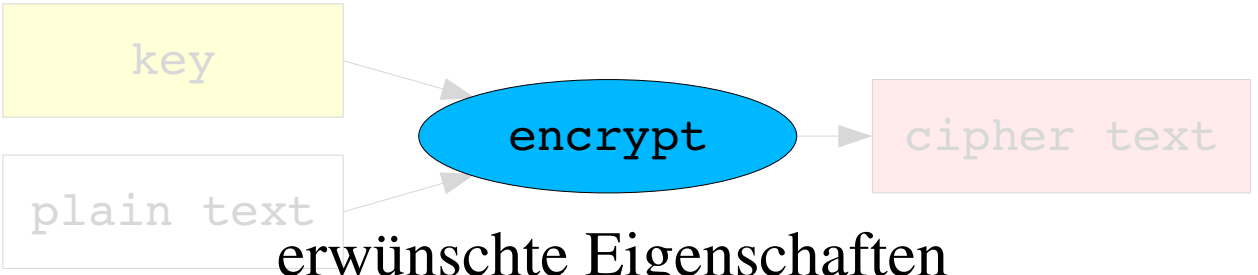
- Verschlüsselungsalgorithmen (Chiffren)
  - Blockchiffren
  - symmetrische, asymmetrische Chiffren
- Prüfsummenalgorithmen
  - Hashes
  - Authentikationscodes (MACs)
    - hash-basierte MACs (HMAC)
  - Signaturen
- Schlüsselaustausch

1 Kryptographische Grundlagen Verschlüsselungsalgorithmen

- Ziel: Nachricht für Lauscher "unverständlich" machen

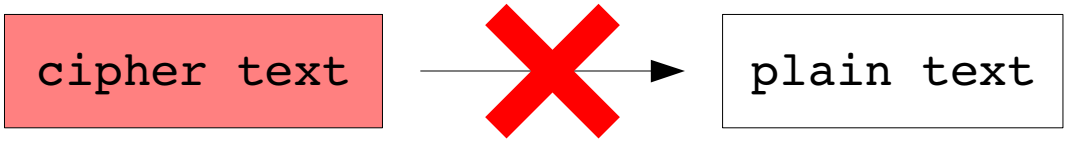


1 Kryptographische Grundlagen Verschlüsselungsalgorithmen

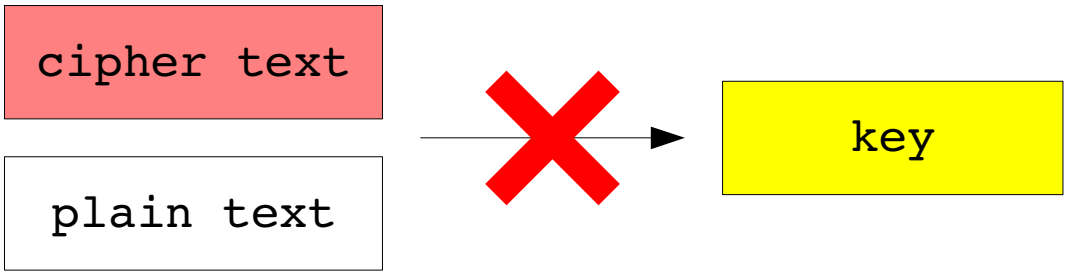


erwünschte Eigenschaften

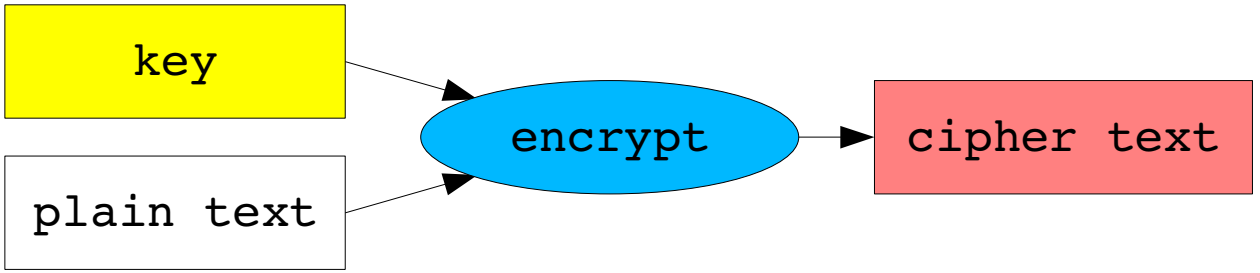
1. `"plain text"` aus `"cipher text"` ohne Kenntnis von `"key"` nicht rekontruierbar



2. `"key"` aus Kenntnis eines Paars von `"plain text"` und `"cipher text"` nicht rekontruierbar ("known plaintext attack")



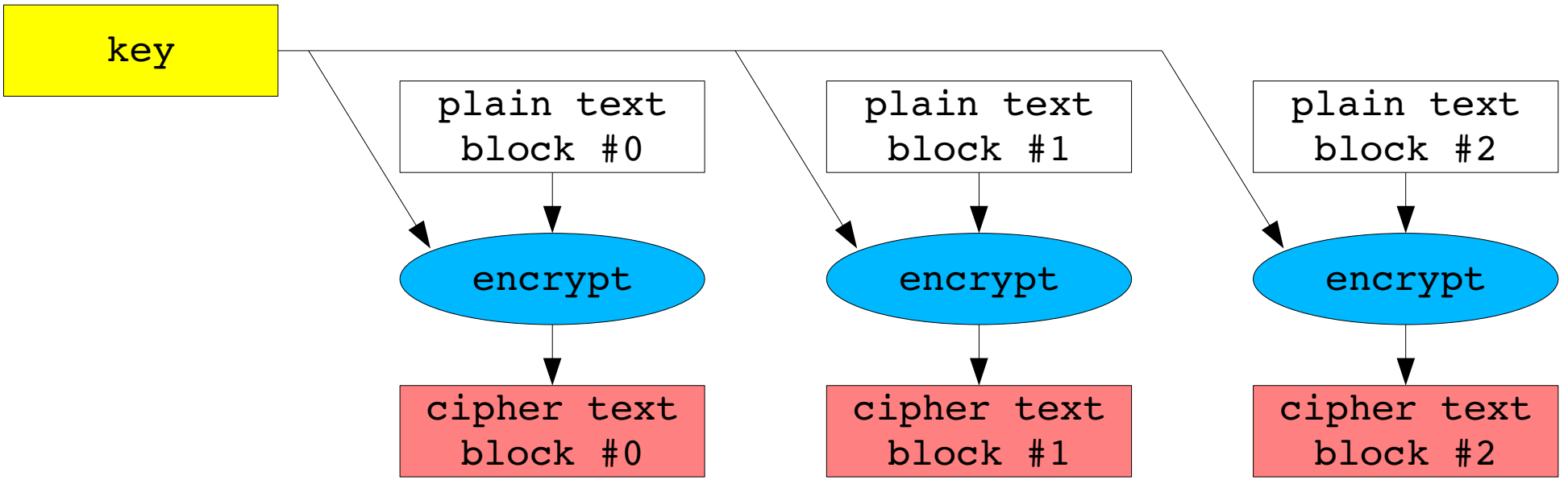
1 Kryptographische Grundlagen Blockchiffren



- `"plain text"`, `"cipher text"`, `"key"` fest vorgegebene Länge
  - typisch z.B. 256/256/256 Bit (AES) oder 128/128/64 Bit (DES)
- längere Nachrichten müssen in einzelne Blöcke aufgeteilt werden, Nachricht wird "blockweise" codiert

1 Kryptographische Grundlagen
Blockchiffren

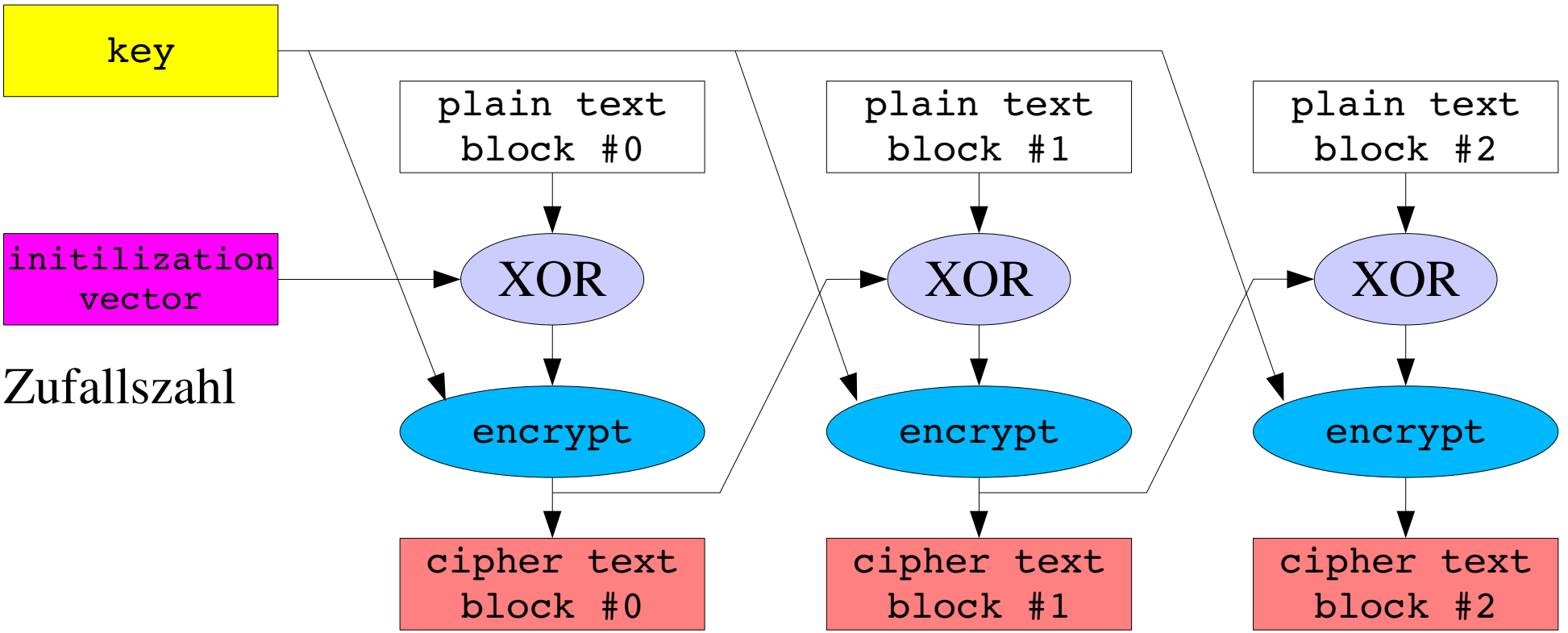
naiv:



wenn `"plain text block #0"="plain text block #2"`,
dann `"cipher text block #0"="cipher text block #2"`

1 Kryptographische Grundlagen
Blockchiffren

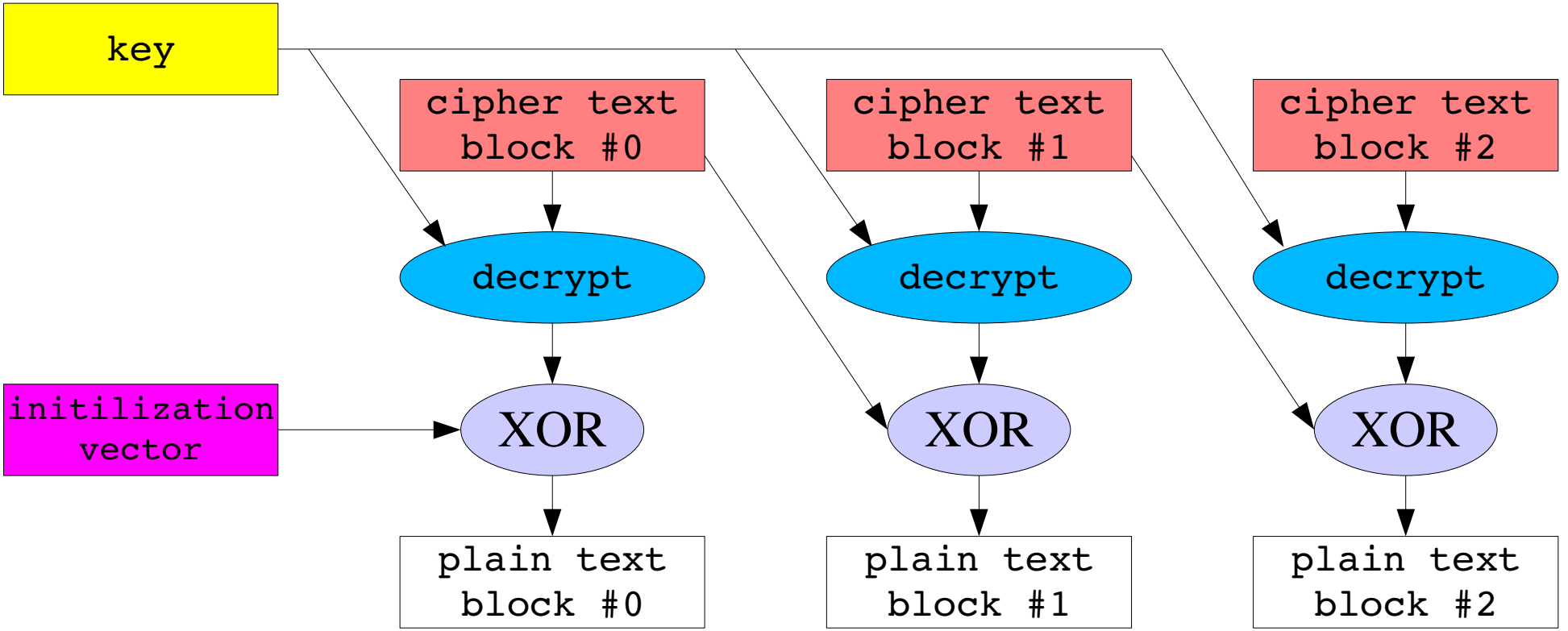
besser: CBC-mode ("chained block cipher"):



`initialization vector` auch zum Entschlüsseln wieder erforderlich, d.h. muss ebenfalls an Empfänger übertragen werden

1 Kryptographische Grundlagen
Blockchiffren

der Vollständigkeit halber: Entschlüsselung im CBC-mode

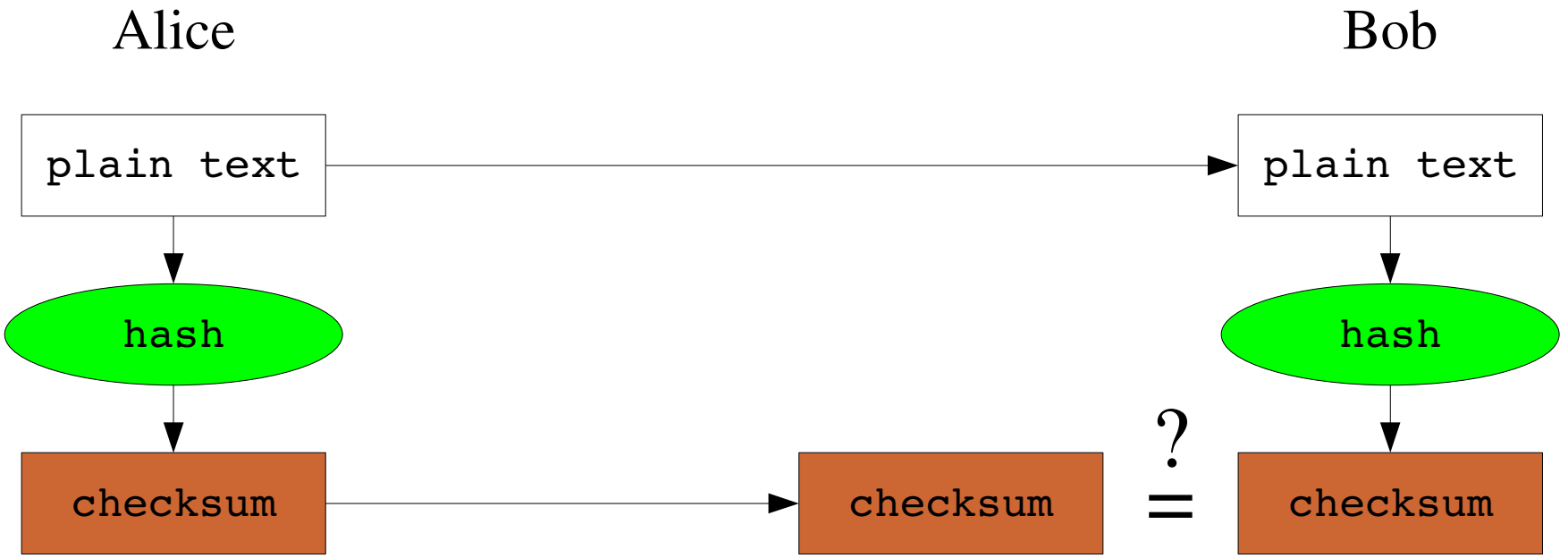


1 Kryptographische Grundlagen (a)symmetrische Chiffren

- symmetrische Chiffre: Schlüssel zum Ver- und Entschlüsseln identisch (z.B. AES, DES)
- asymmetrische Chiffre: Schlüssel zum Ver- und Entschlüsseln verschieden (z.B. RSA, ECC)
  - oftmals einer der beiden Schlüssel aus anderem berechenbar
- asymmetrische Verfahren *enorm* Rechenintensiv
  - für praktische Zwecke: symmetrische Verfahren bevorzugt verwenden wann immer möglich

1 Kryptographische Grundlagen
Prüfsummenalgorithmen

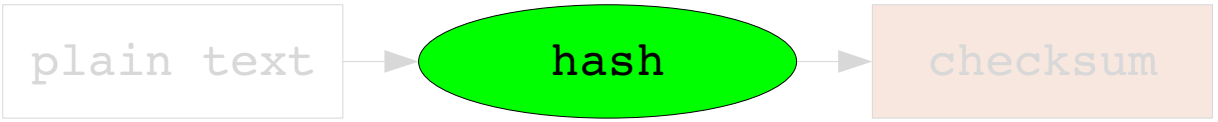
- Ziel: überprüfen, ob Nachricht verändert wurde



`plain text` kann beliebig lang sein; `checksum` hat typischerweise feste Länge (erheblich kürzer als `plain text`)

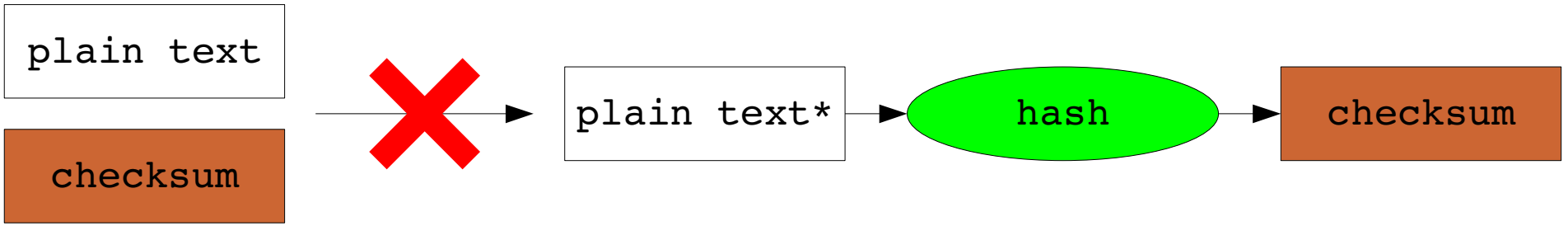
Schutz der Prüfsumme selbst vor Manipulation?

1 Kryptographische Grundlagen
Prüfsummenalgorithmen

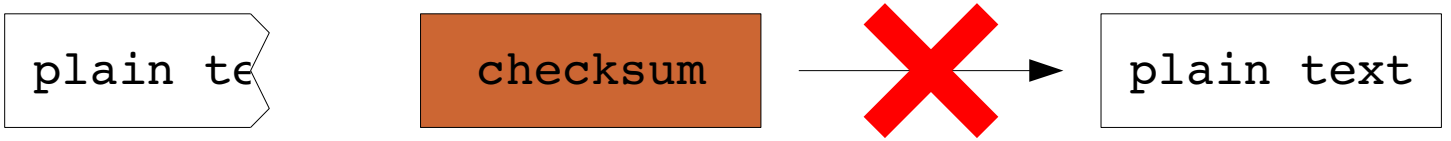


erwünschte Eigenschaften

1. kein manipulierter `"plain text*"` aus Kenntnis von `"checksum"` und `"plain text"`kontruierbar

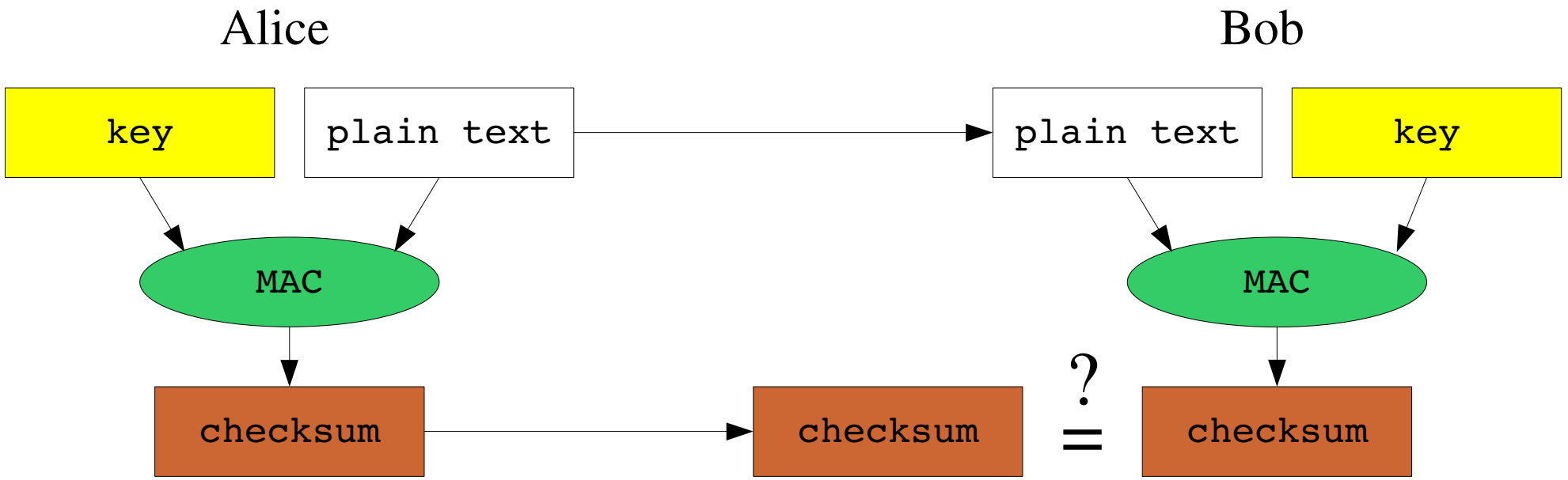


2. wenn nur Teil von `"plain text"` bekannt, dann aus Kenntnis von `"checksum"` nicht Rest von `"plain text"`kontruierbar



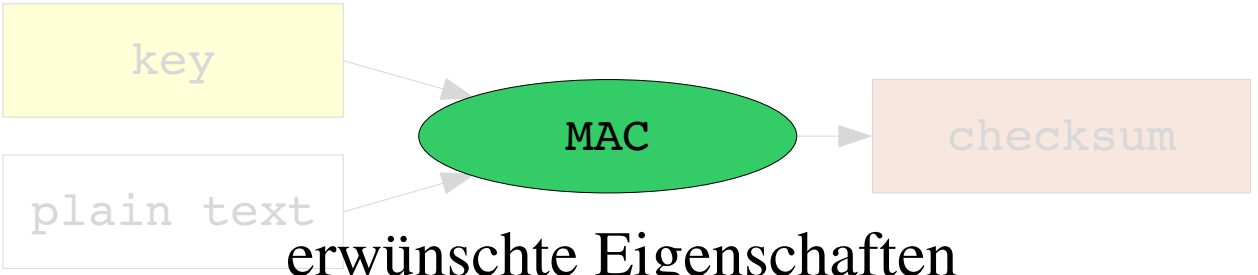
1 Kryptographische Grundlagen
Authentikationscodes (MACs)

- Ziel: Nachweis dass Nachricht unverändert von Absender stammt



ähnliches Verhalten wie hash, aber korrekte Prüfsumme nicht ohne Kenntnis von `key` berechenbar; Schutz der Prüfsumme daher nicht notwendig, da nur Alice korrekte Prüfsumme erzeugen kann

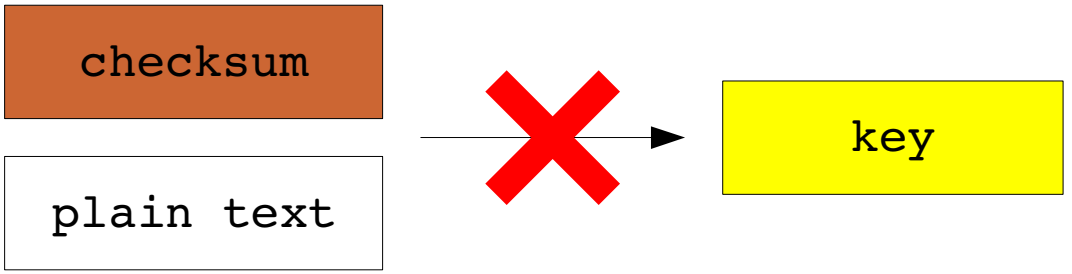
1 Kryptographische Grundlagen
Authentikationscodes (MACs)



erwünschte Eigenschaften

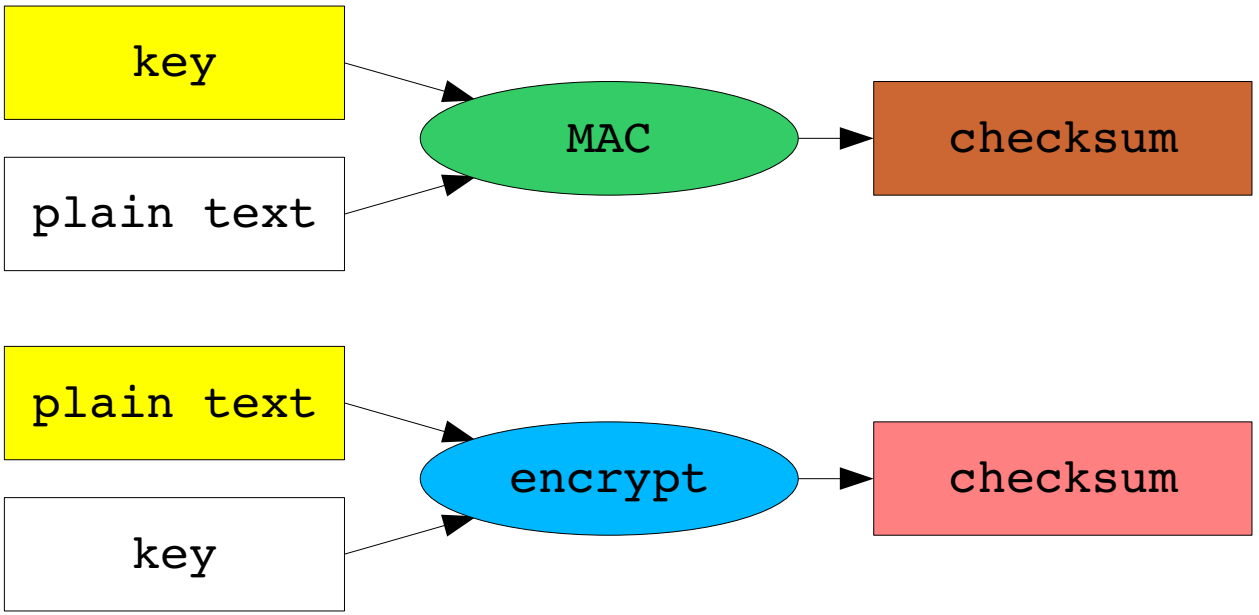
1. alle Eigenschaften einer hash-Funktion

2. `"key"` aus Kenntnis eines Paars von `"plain text"` und `"checksum"` nicht rekontruierbar



1 Kryptographische Grundlagen
Konstruktion von MACs

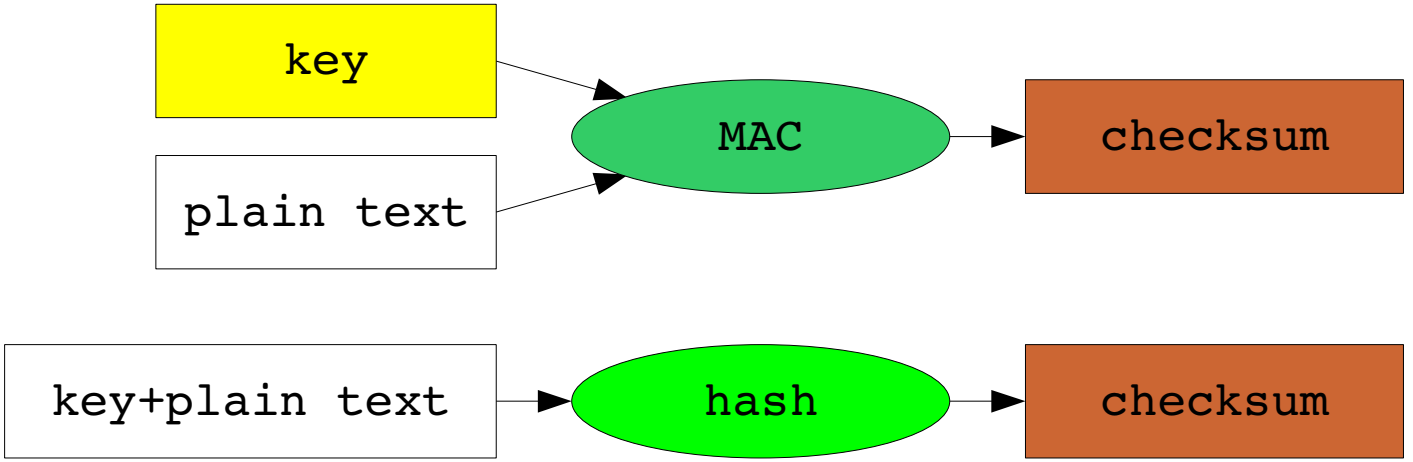
- jede Chiffre kann als MAC verwendet werden!



Blockchiffre: alle `plain text` Blöcke der Reihe nach als Schlüssel anwenden; der erste Block auf `key`, der zweite auf das Ergebnis der ersten Operation etc.

1 Kryptographische Grundlagen
Konstruktion von MACs

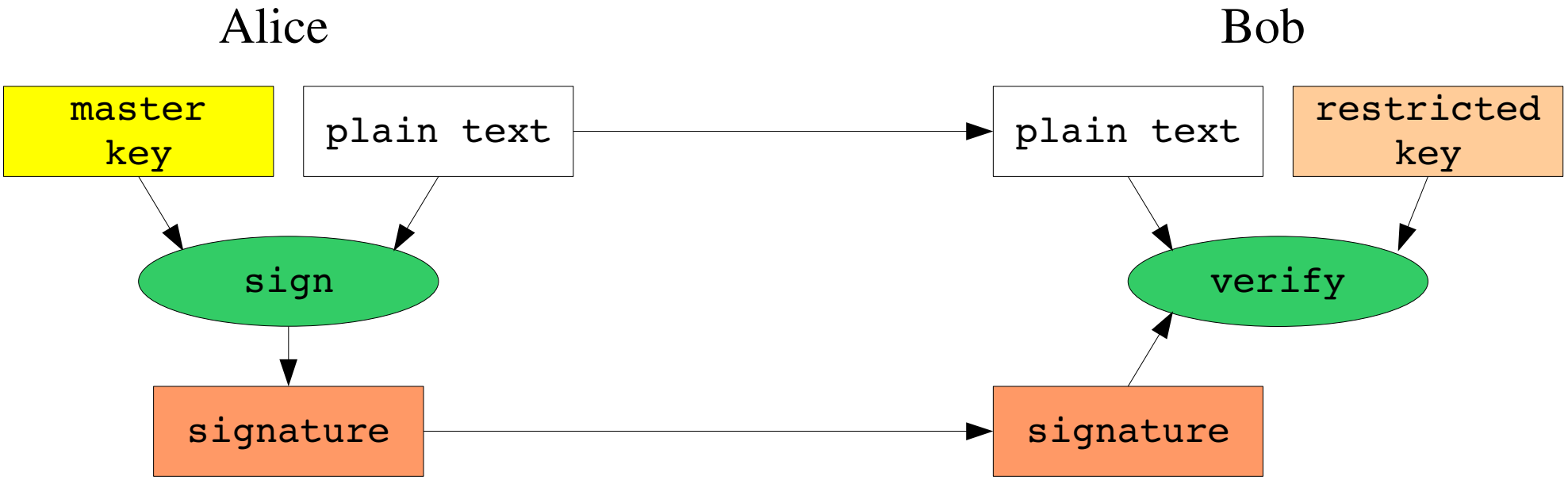
- jede Hash-Funktion kann als MAC verwenden werden (sog. H-MAC)!



hash-Funktionen i.A. leichter berechenbar als Chiffren
→ H-MAC gegenüber Chiffren-MACs bevorzugt

1 Kryptographische Grundlagen
Signaturen

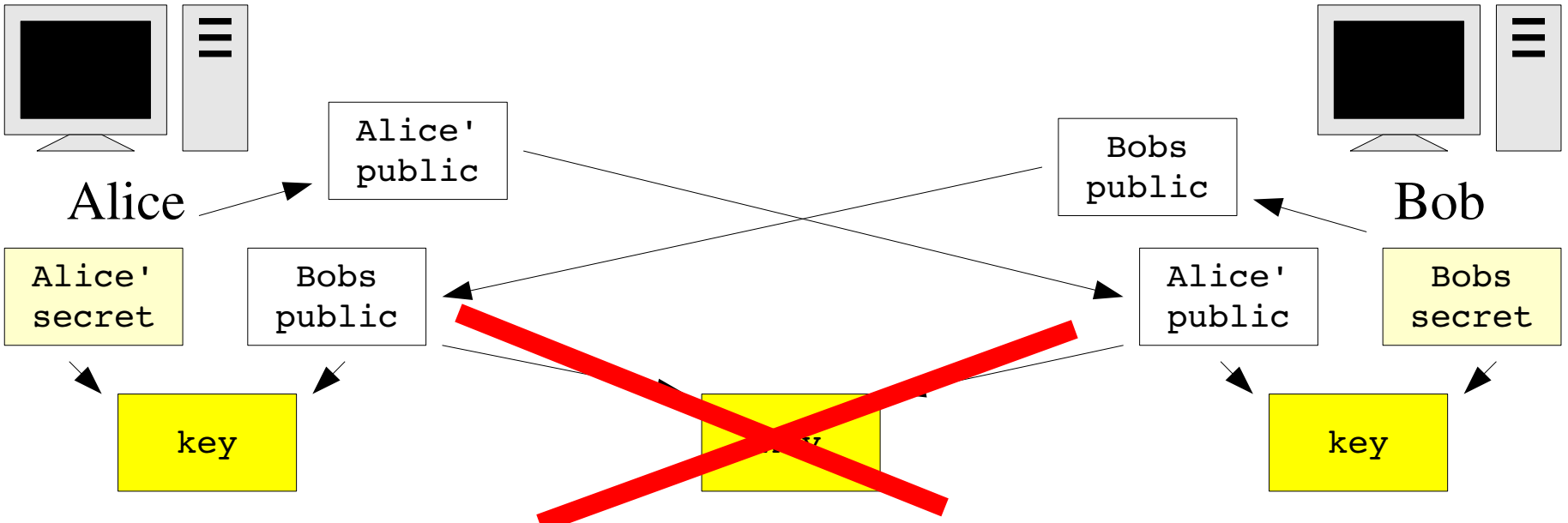
- analog zu Chiffren asymmetrische Varianten; werden als "Signaturen" bezeichnet



Bob kann verifizieren, dass `plain text`, `key#2` und `signature` zusammenpassen, aber er kann keine Signatur erzeugen

1 Kryptographische Grundlagen
Schlüsselaustausch

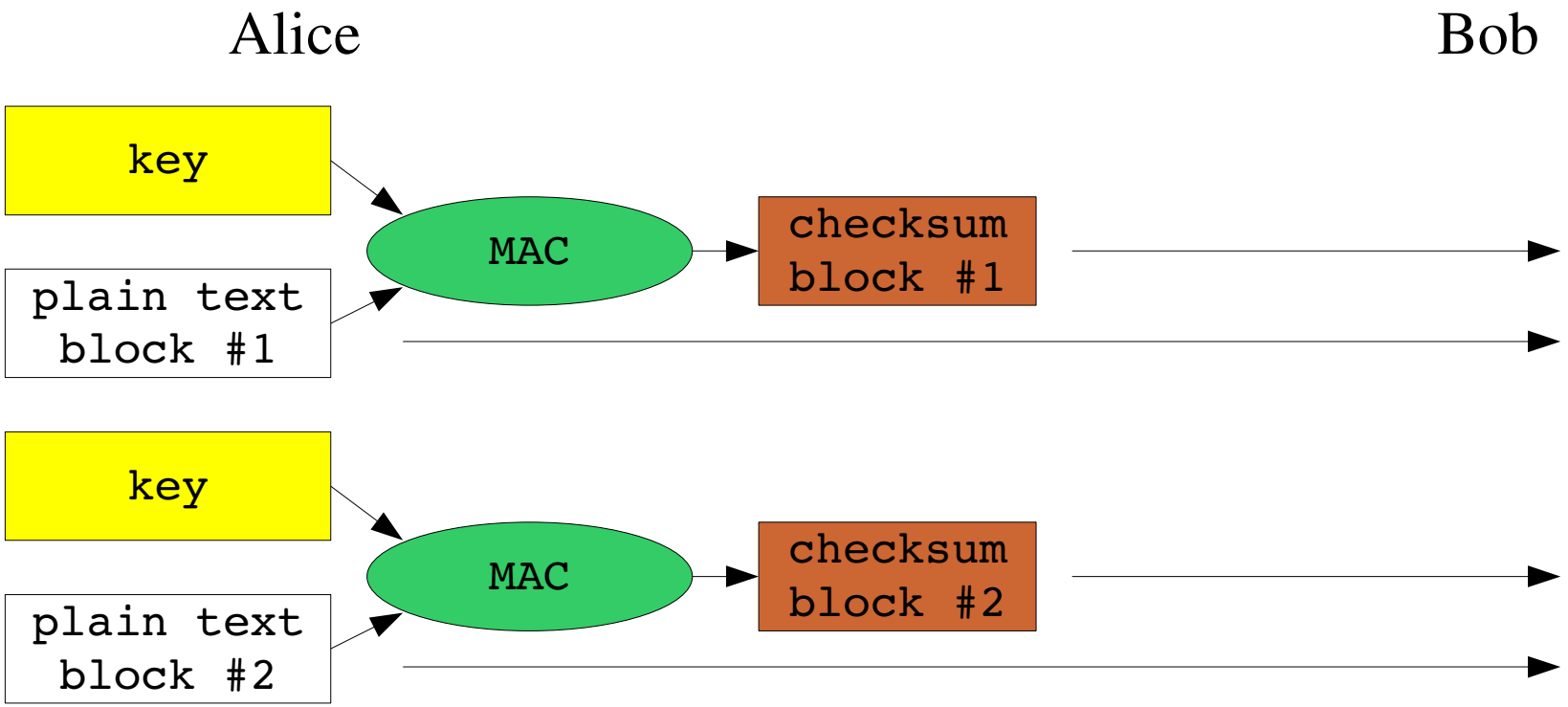
- Wie erhalten Alice und Bob einen gemeinsamen Schlüssel?
  - über *anderen* sicheren Kommunikationskanal ausgetauscht (z.B. "fest verdrahtet")
  - Diffie-Hellman Schlüsselaustausch-Protokoll



1.5 Kryptographische Grundlagen
Vertraulichkeit ohne Verschlüsselung

Alice und Bob haben gemeinsamen geheimen Schlüssel (durch vorhergehende sichere Kommunikation ausgetauscht)

Alice zerlegt Nachricht in Blöcke, generiert Prüfsumme, sendet beides an Bob (Prüfsummen-Algorithmus)

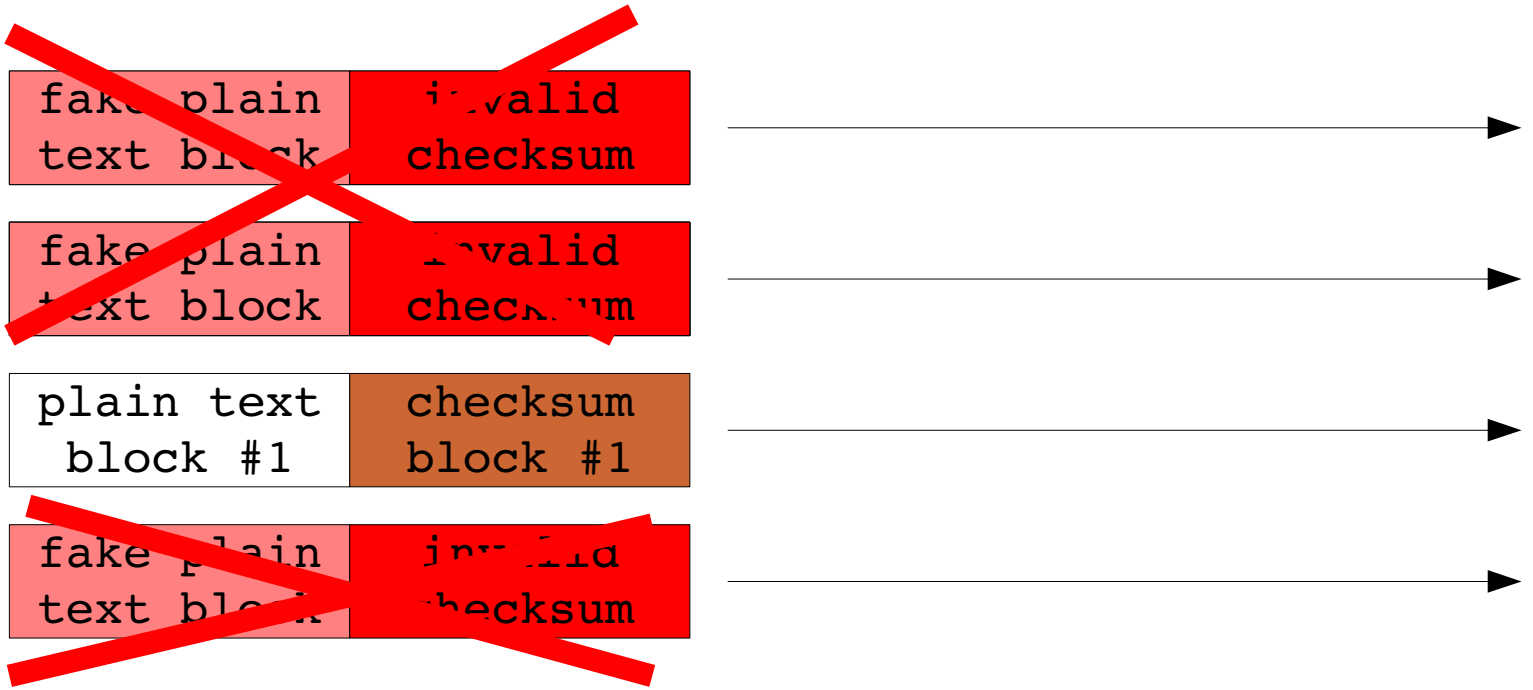


1.5 Kryptographische Grundlagen
Vertraulichkeit ohne Verschlüsselung

Alice erzeugt und sendet “falsche” plaintext-Blöcke (“Chaff”), erzeugt ungültige Prüfsumme für jeden dieser Blöcke (Zufallsgenerator)

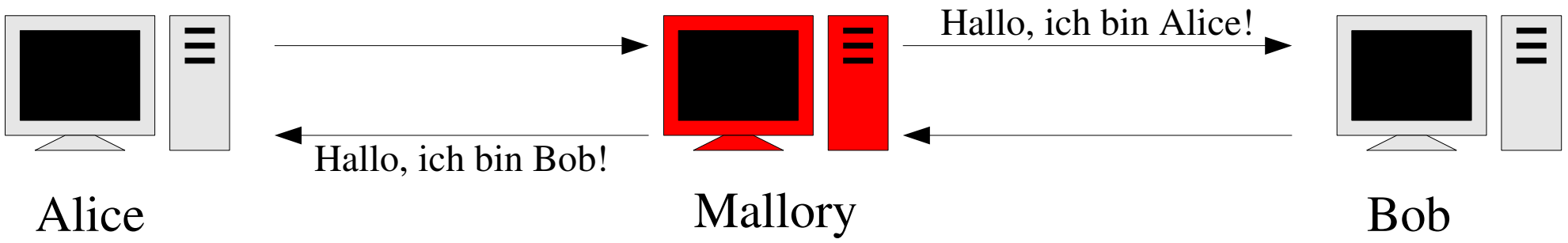
Bob sortiert plaintext-Blöcke anhand Prüfsumme aus (“winnow”)

Lauscher kann Prüfsumme nicht berechnen, da Schlüssel unbekannt



2 Vertrauensmodelle

- Warum können Alice und Bob nicht sicher kommunizieren, ohne vorher einmal kommuniziert zu haben?

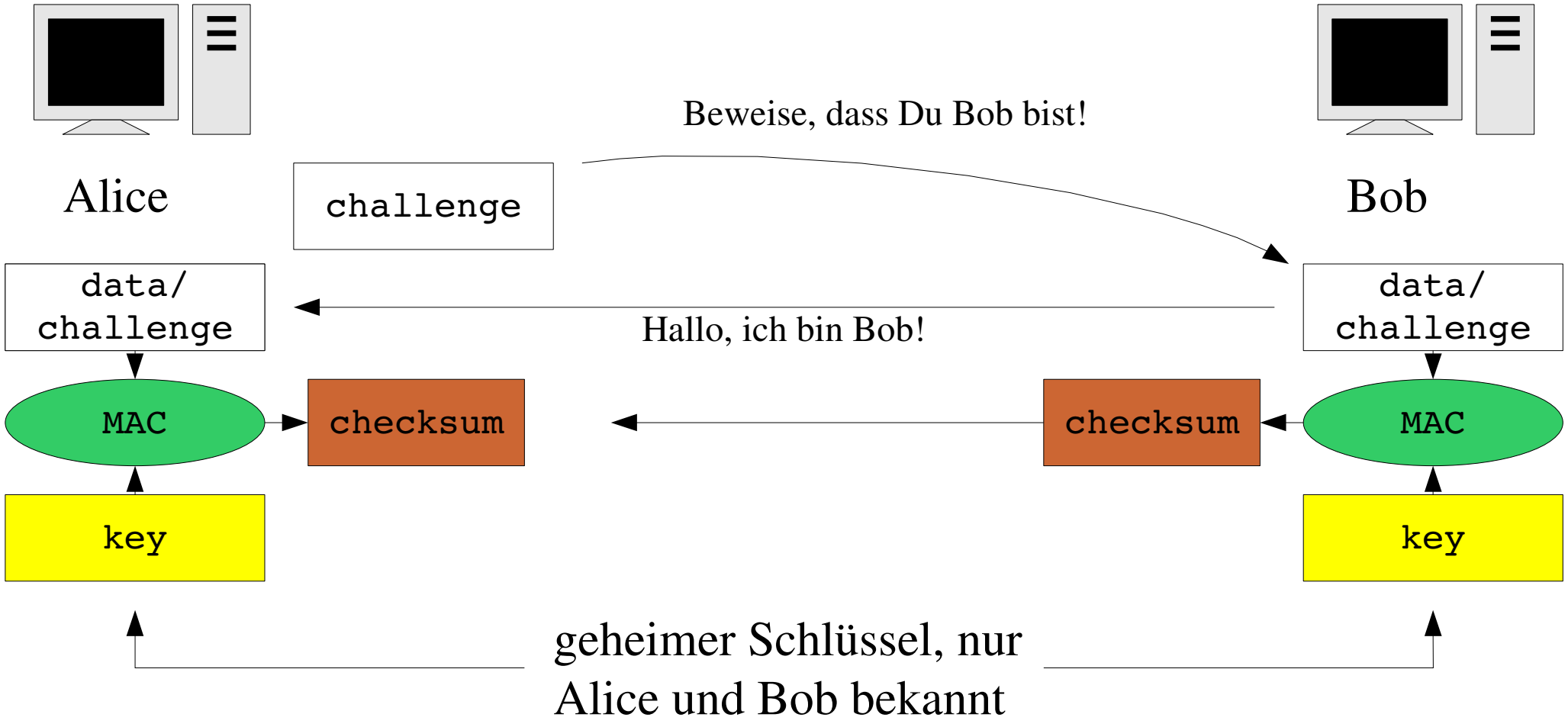


Alice muss "etwas" über Bob wissen, das Mallory nicht weiß. Andernfalls ist Mallory von Bob ununterscheidbar.

→Alice und Bob müssen vorher bereits vertraulich kommuniziert haben, ggf. mit einem vertrauenswürdigen Dritten.

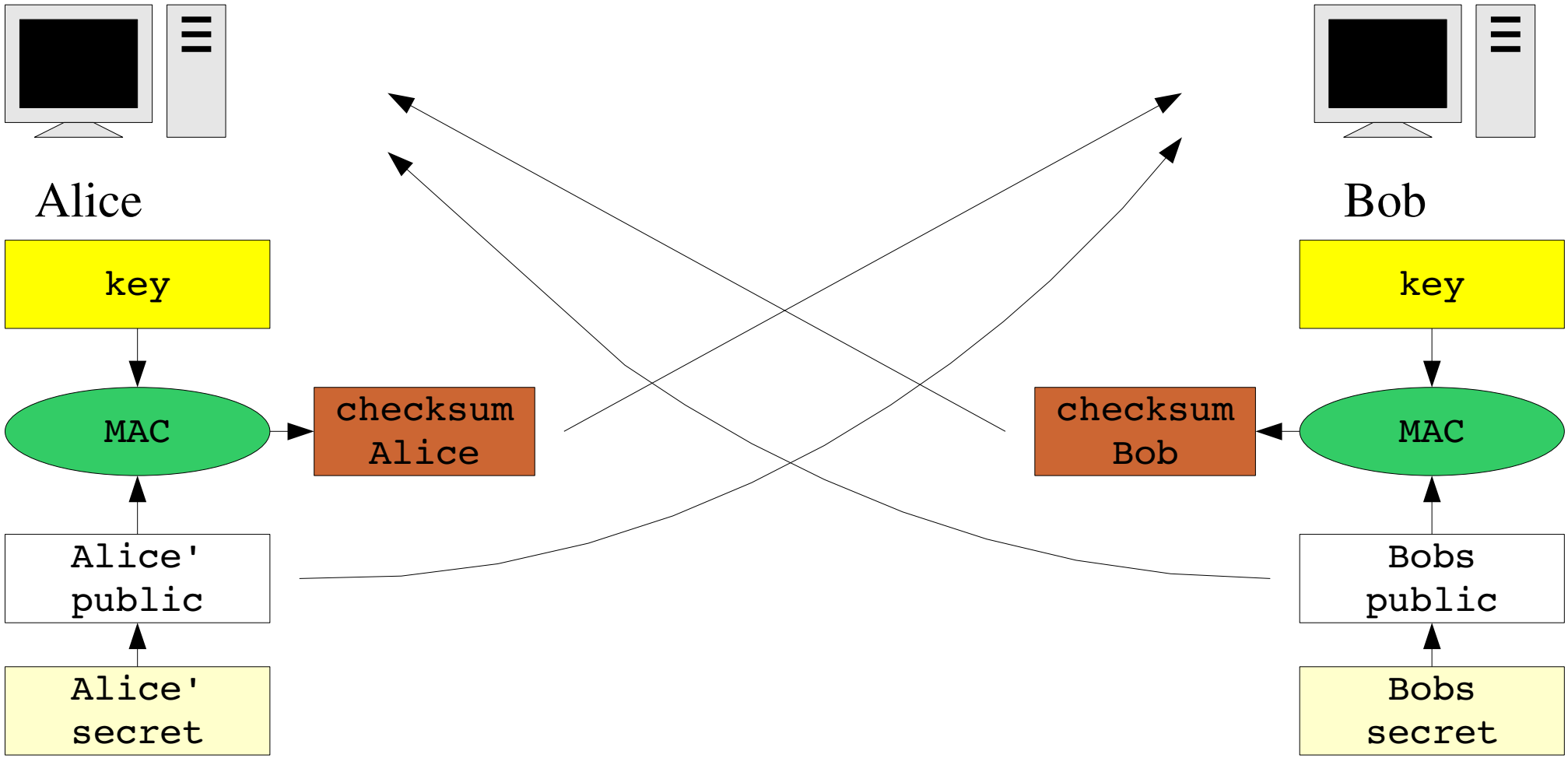
2 Vertrauensmodelle

"Wissen" zur sicheren Identifikation des Kommunkationspartners kann durch kryptographische Schlüssel ausgedrückt werden.

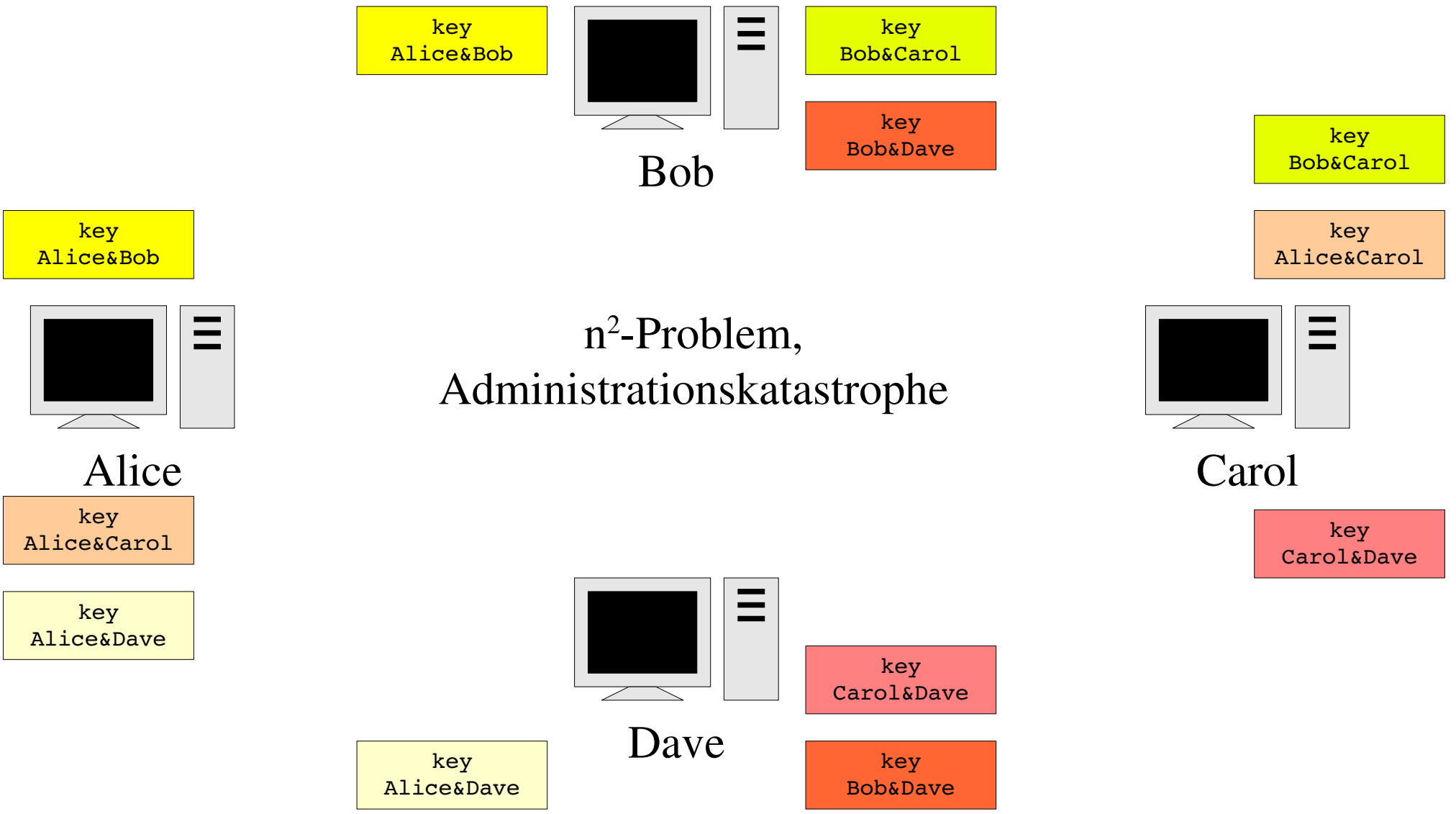


2 Vertrauensmodelle

z.B. authentisierter DH-Schlüsselaustausch



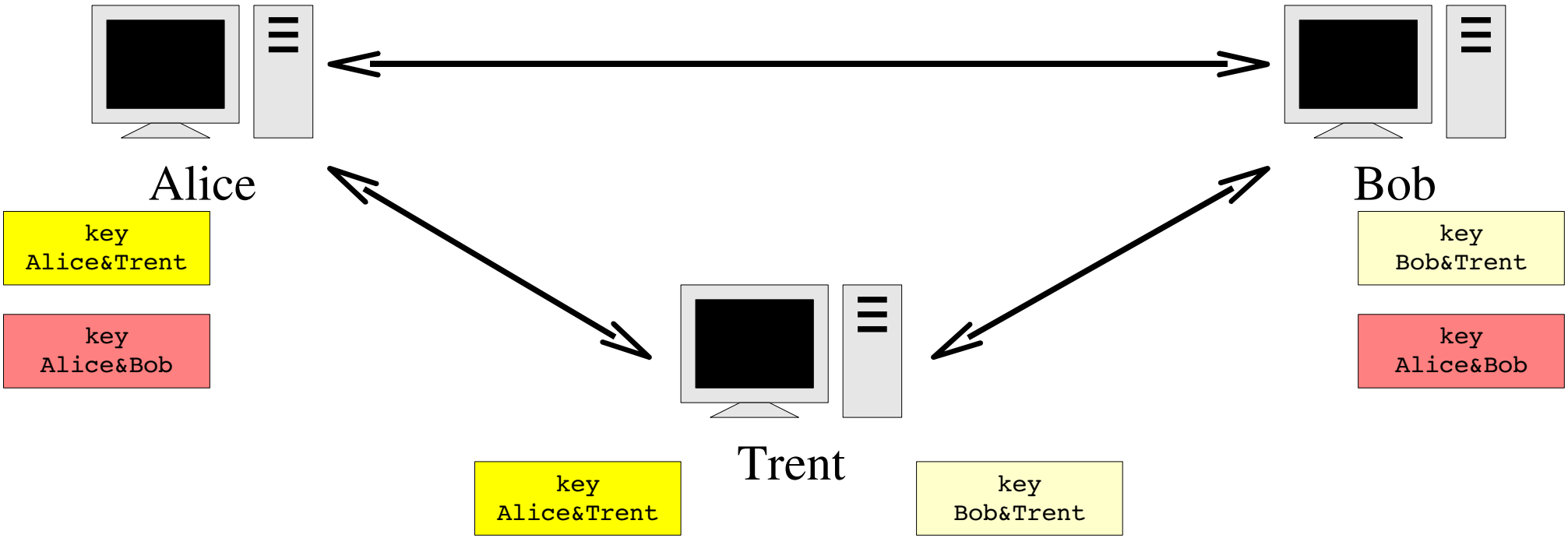
2 Vertrauensmodelle



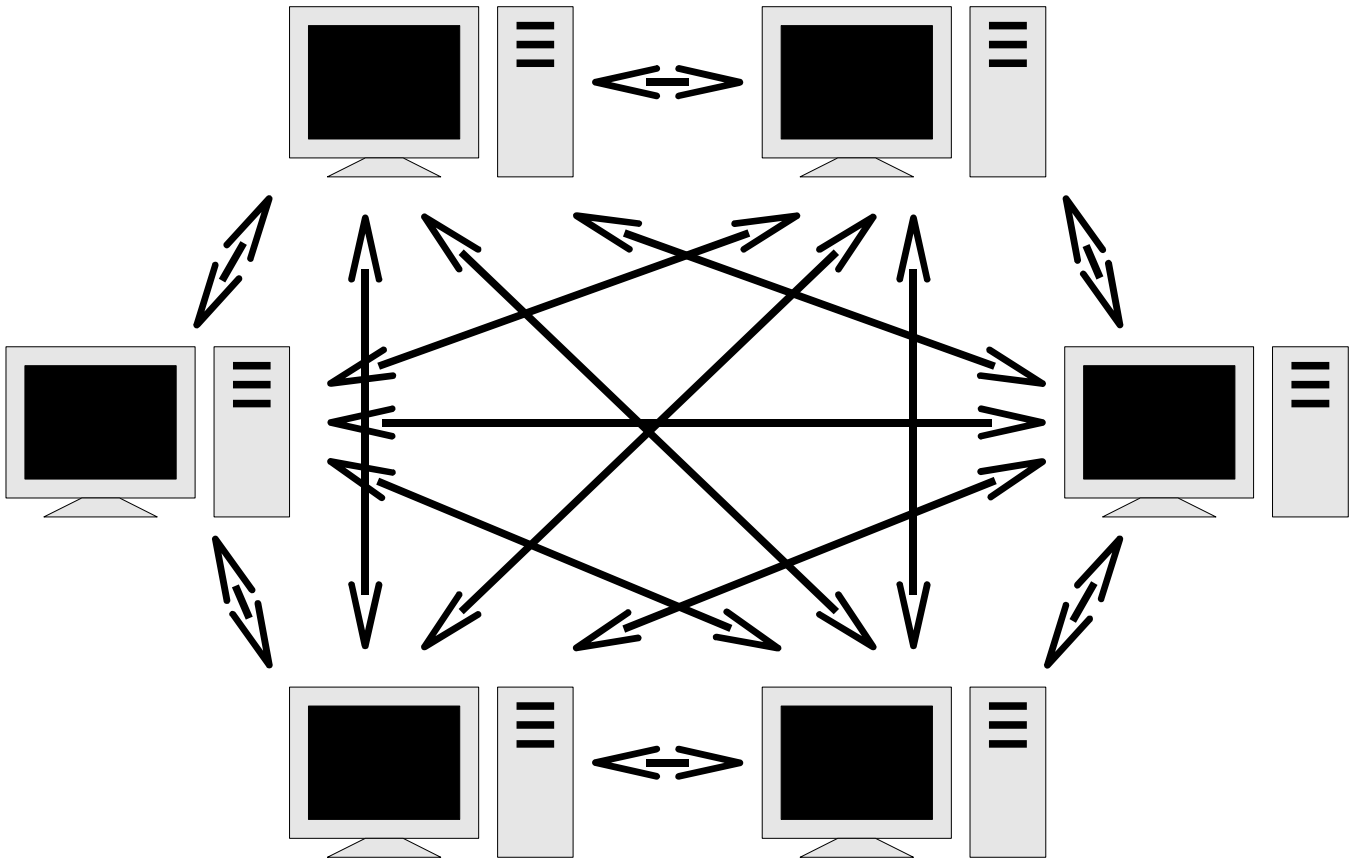
2 Vertrauensmodelle

2 Vertrauensmodelle

Trent kann vermitteln
man nennt dies *transitives Vertrauen*

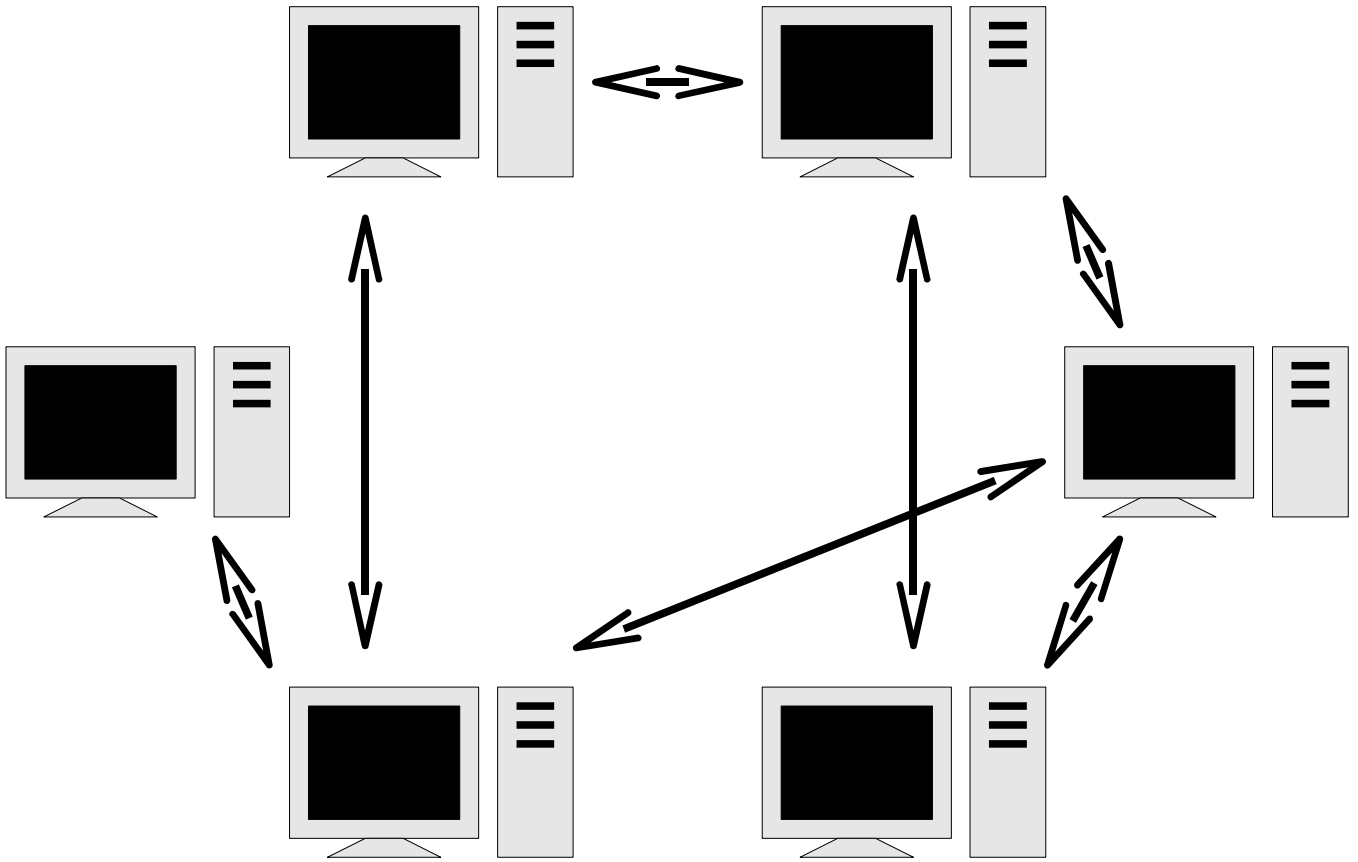


2 Vertrauensmodelle



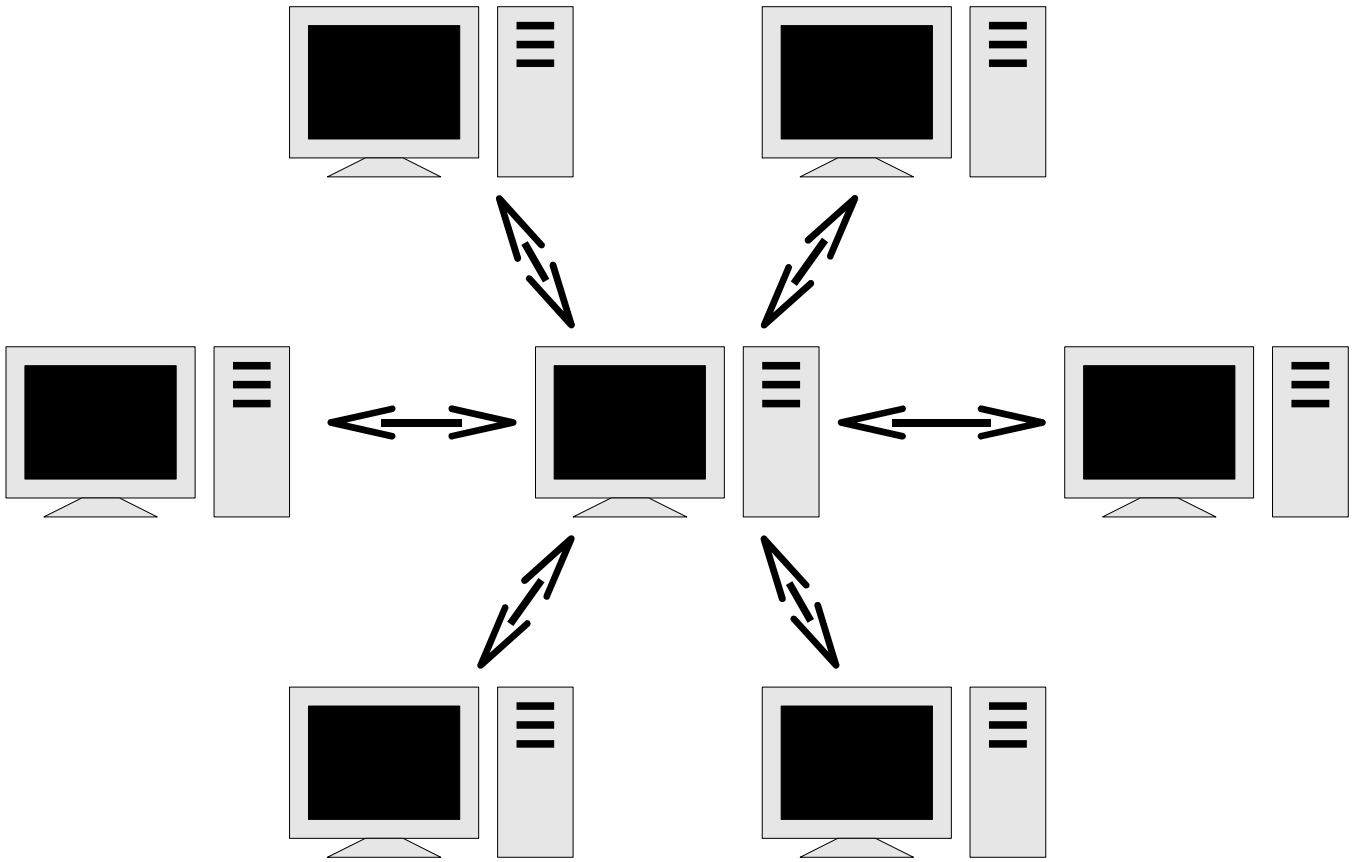
peer-to-peer, vollvermascht

2 Vertrauensmodelle



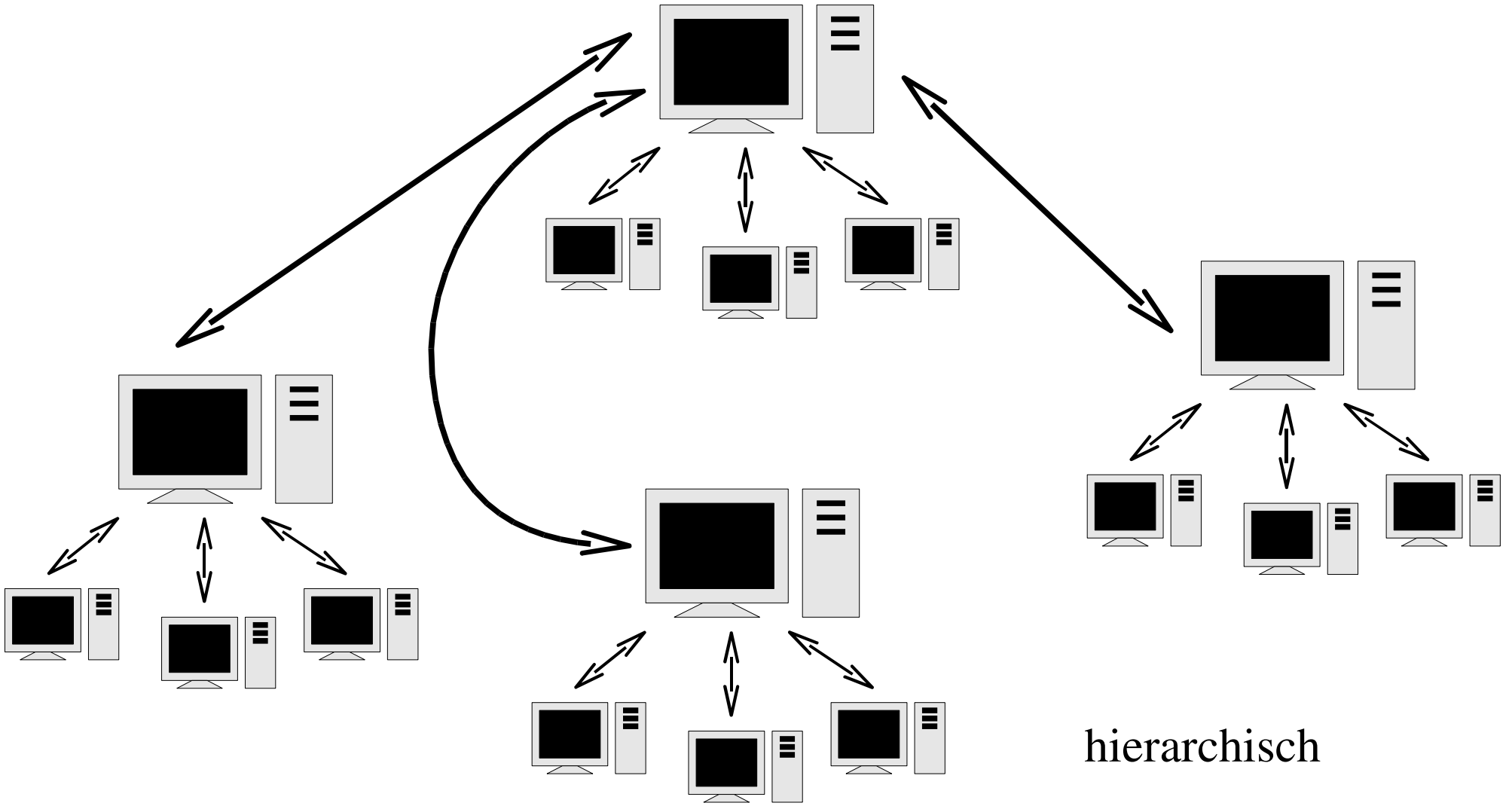
web of trust

2 Vertrauensmodelle



zentralisiert

2 Vertrauensmodelle

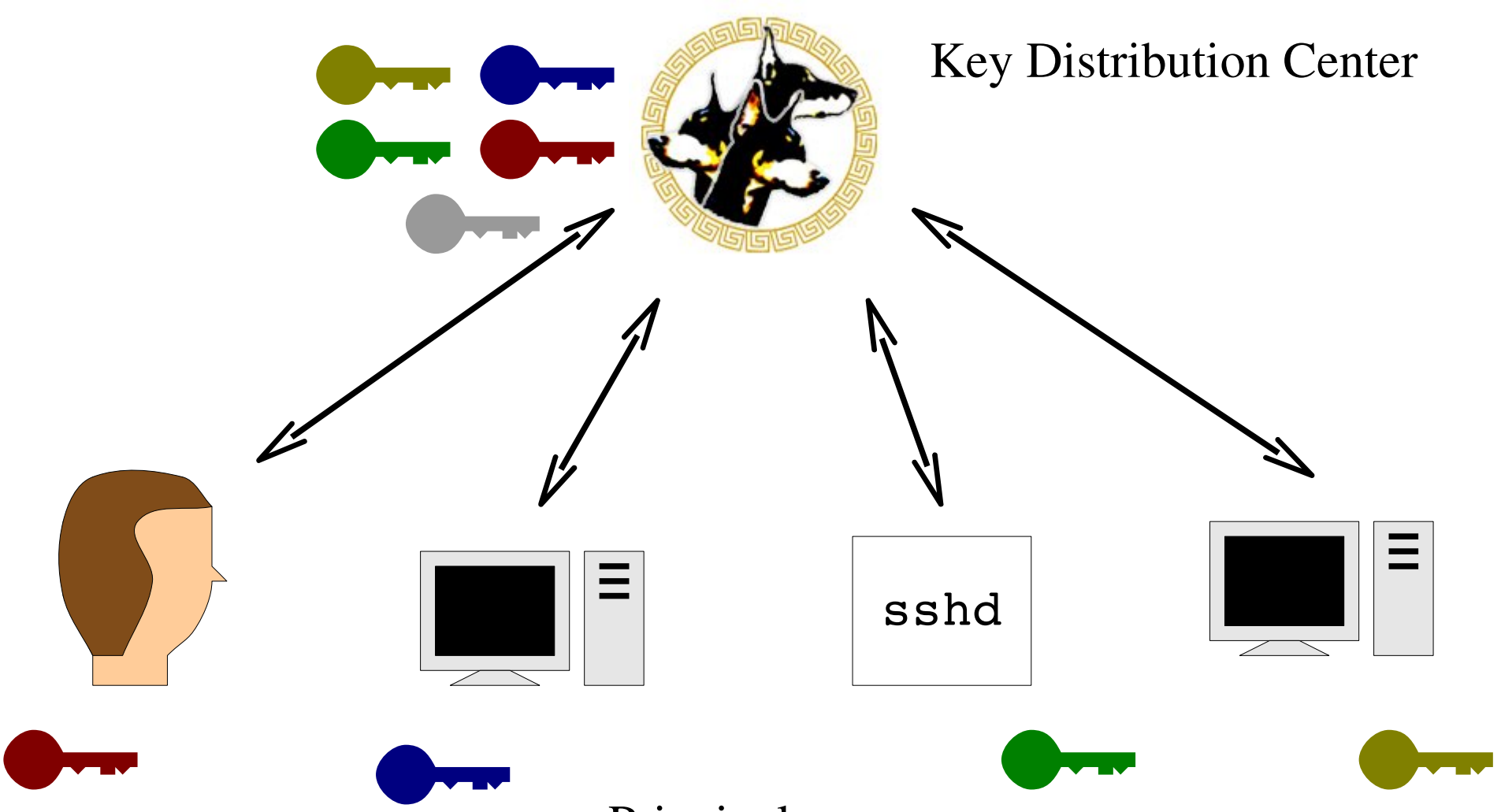


hierarchisch

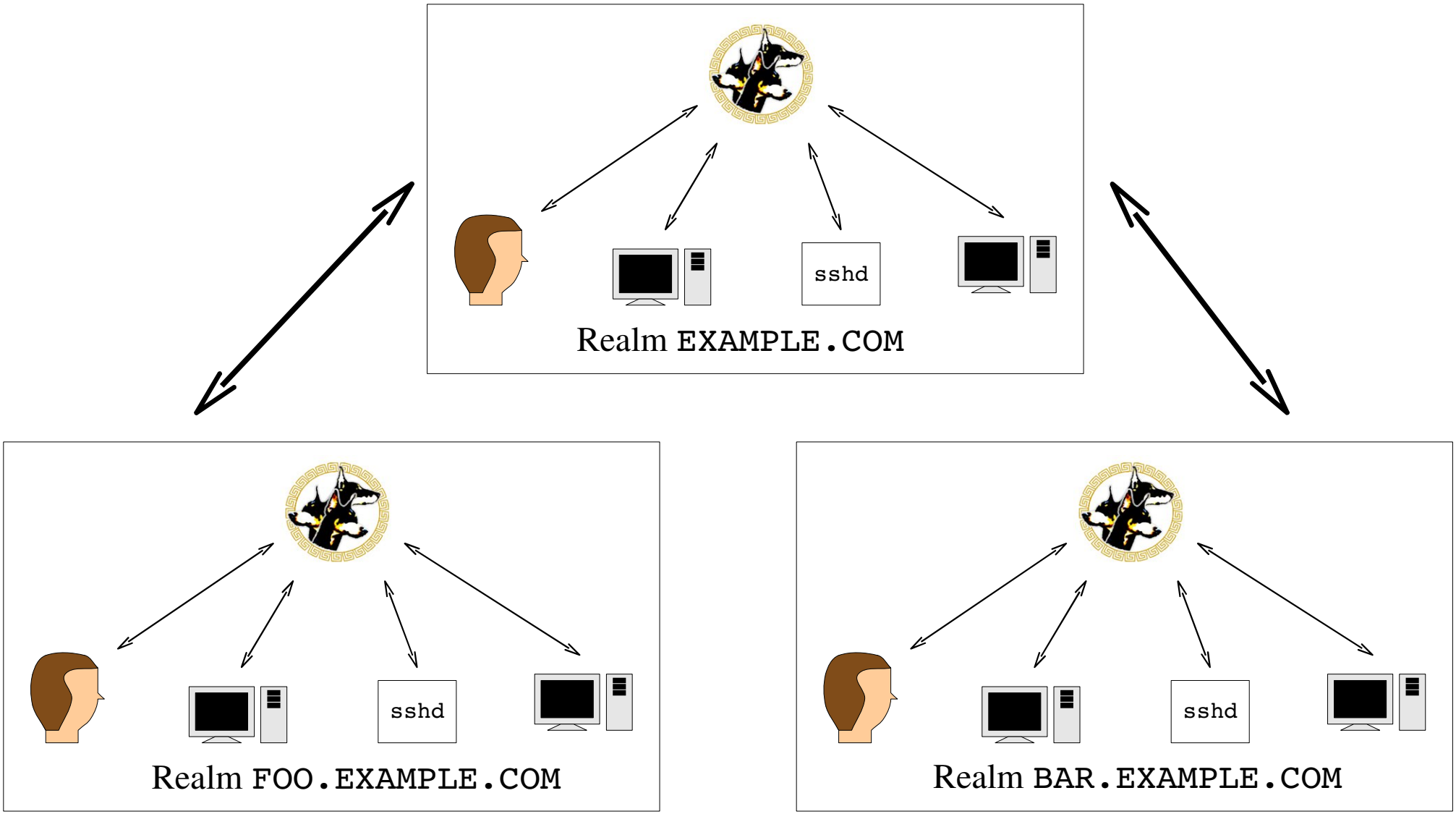
2 Vertrauensmodelle

- typischerweise "hybride Modelle"
- Vertrauensbeziehungen nicht unbedingt gleichwertig
- Regeln definieren Transitivität
  - z.B. begrenzte Länge des Pfades
  - z.B. nur bestimmte Arten von Vertrauensbeziehungen innerhalb eines Pfades
- Vertrauensmanagement ungelöstes Problem

3 Kerberos Grundbegriffe



3 Kerberos Realms

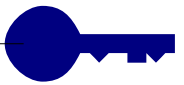


3 Kerberos Principals

alice@EXAMPLE.COM

host/my.computer.org@EXAMPLE.COM

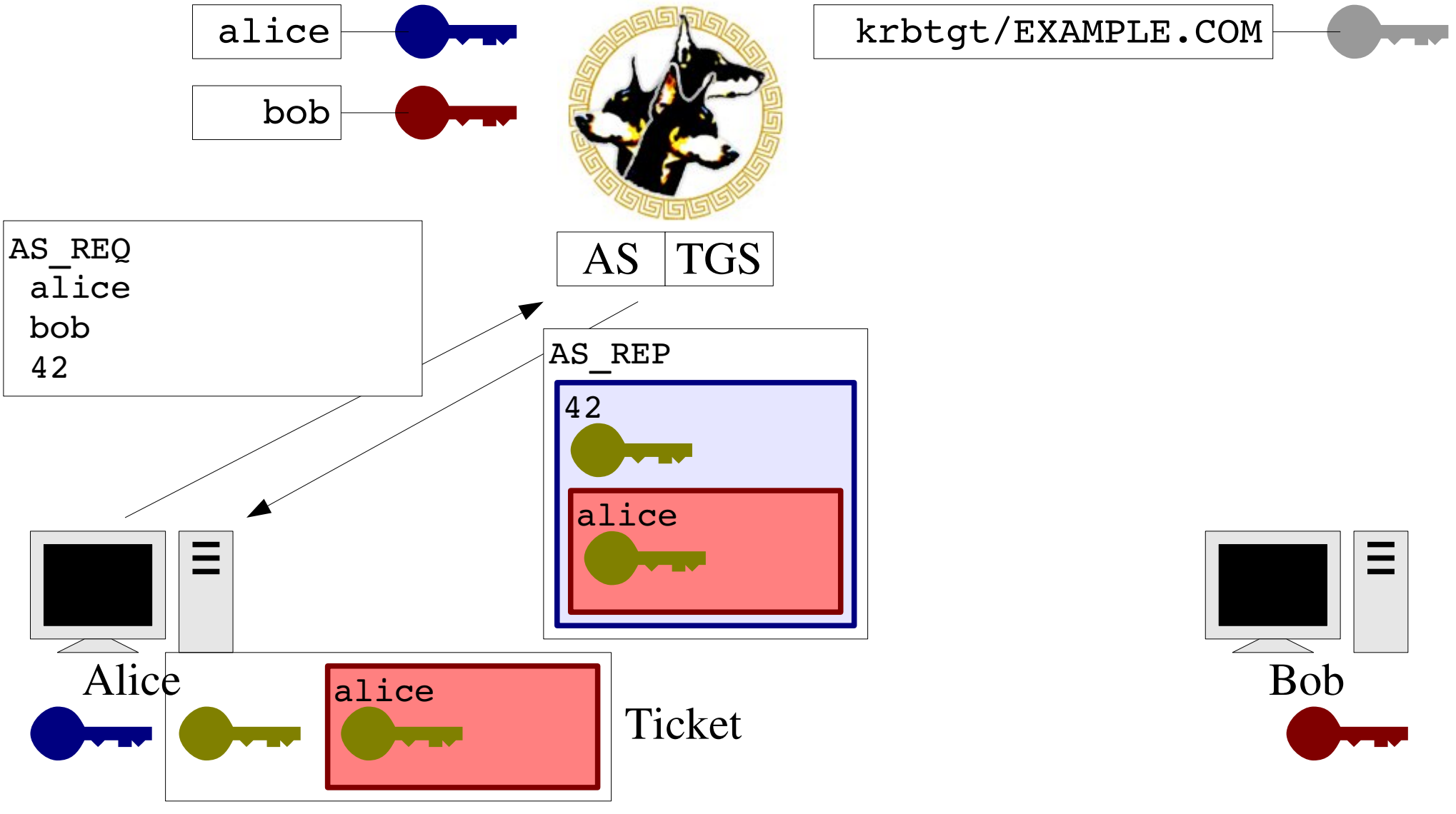
ssh/my.computer.org@EXAMPLE.COM



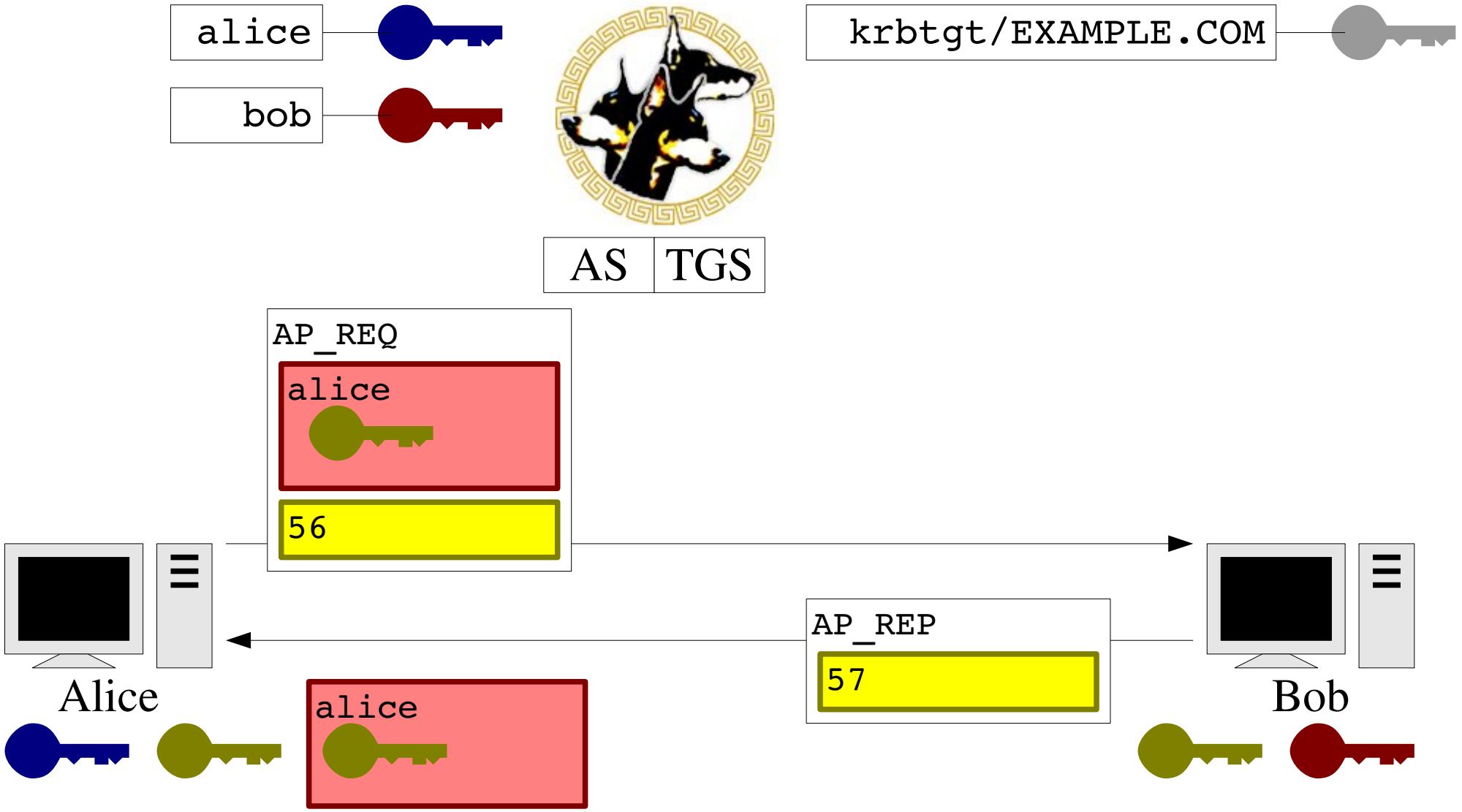
host/dodo.example.com@EXAMPLE.COM

krbtgt/EXAMPLE.COM@EXAMPLE.COM

3 Kerberos
Auth. Service (Schroeder-Needham)



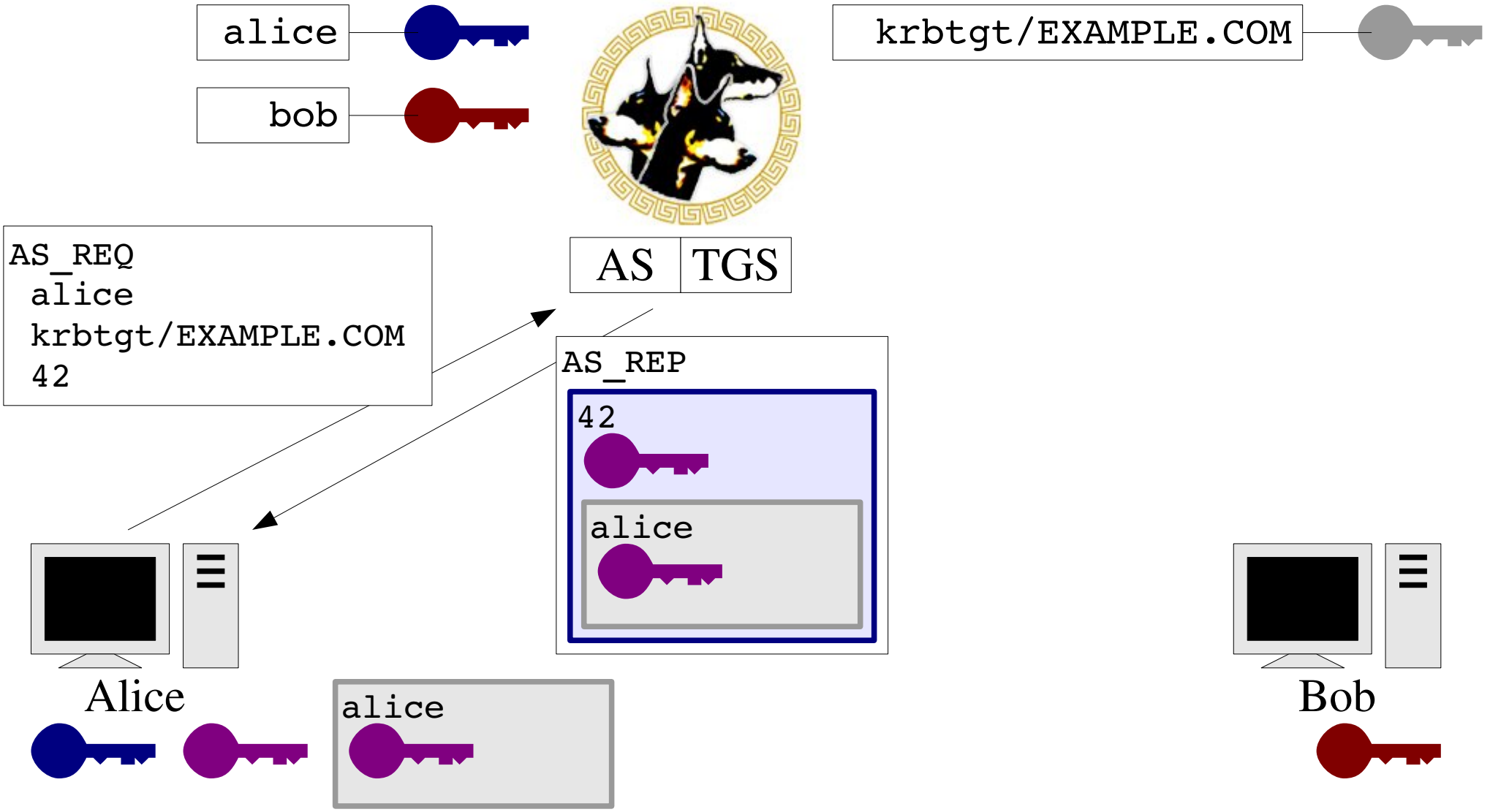
3 Kerberos
Auth. Service (Schroeder-Needham)



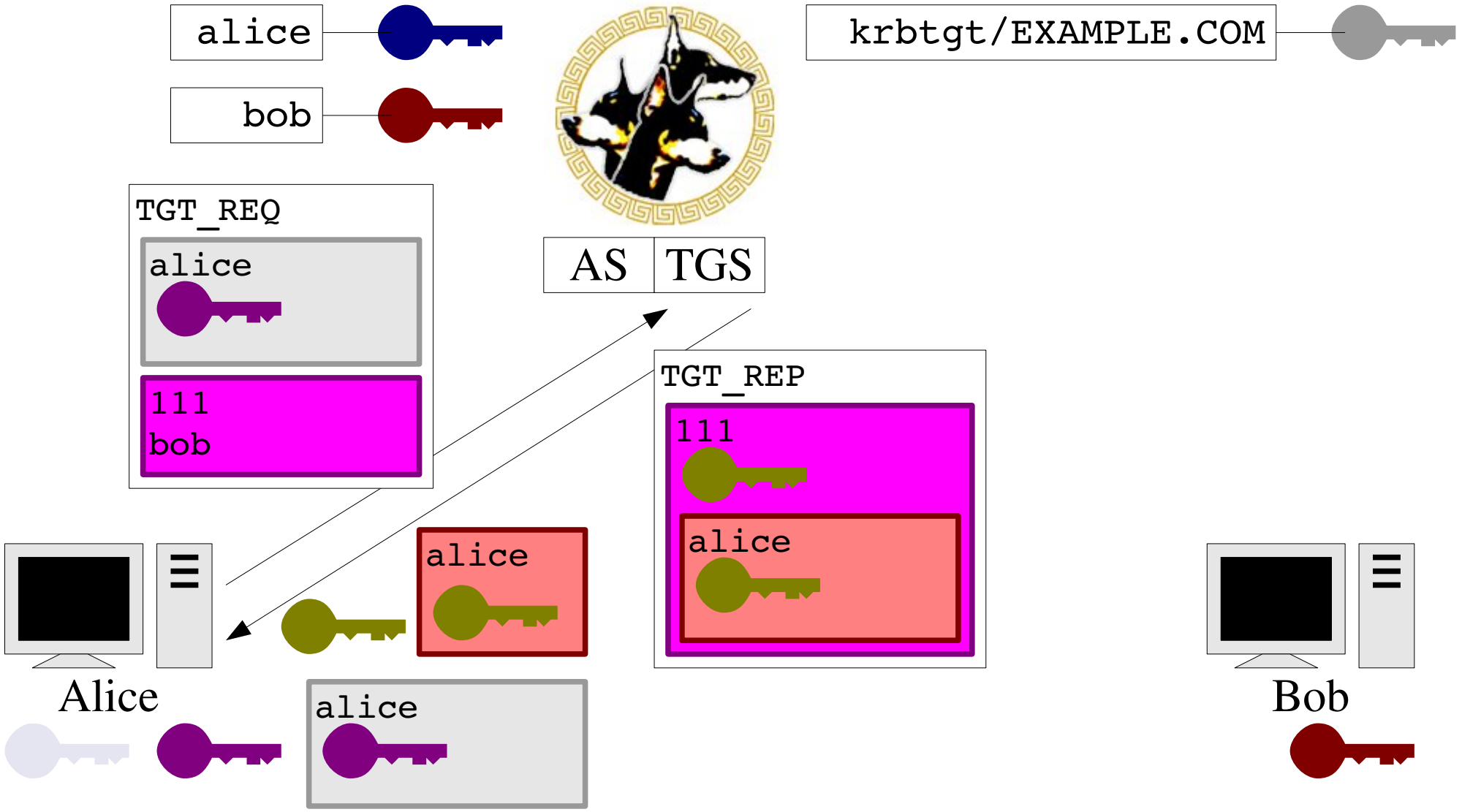
3 Kerberos

- vorgehensweise wie zuvor beschrieben ist möglich, aber eigentlich unüblich
  - Alice muss eigenen Schlüssel zur Verfügung haben, wenn Kontakt zu einem neuen Kommunikationspartner aufzunehmen
  - Schlüssel kann aus Passwort abgeleitet sein:
    - entweder ständig erneut eingeben (schlecht)
    - oder abspeichern (auch schlecht)

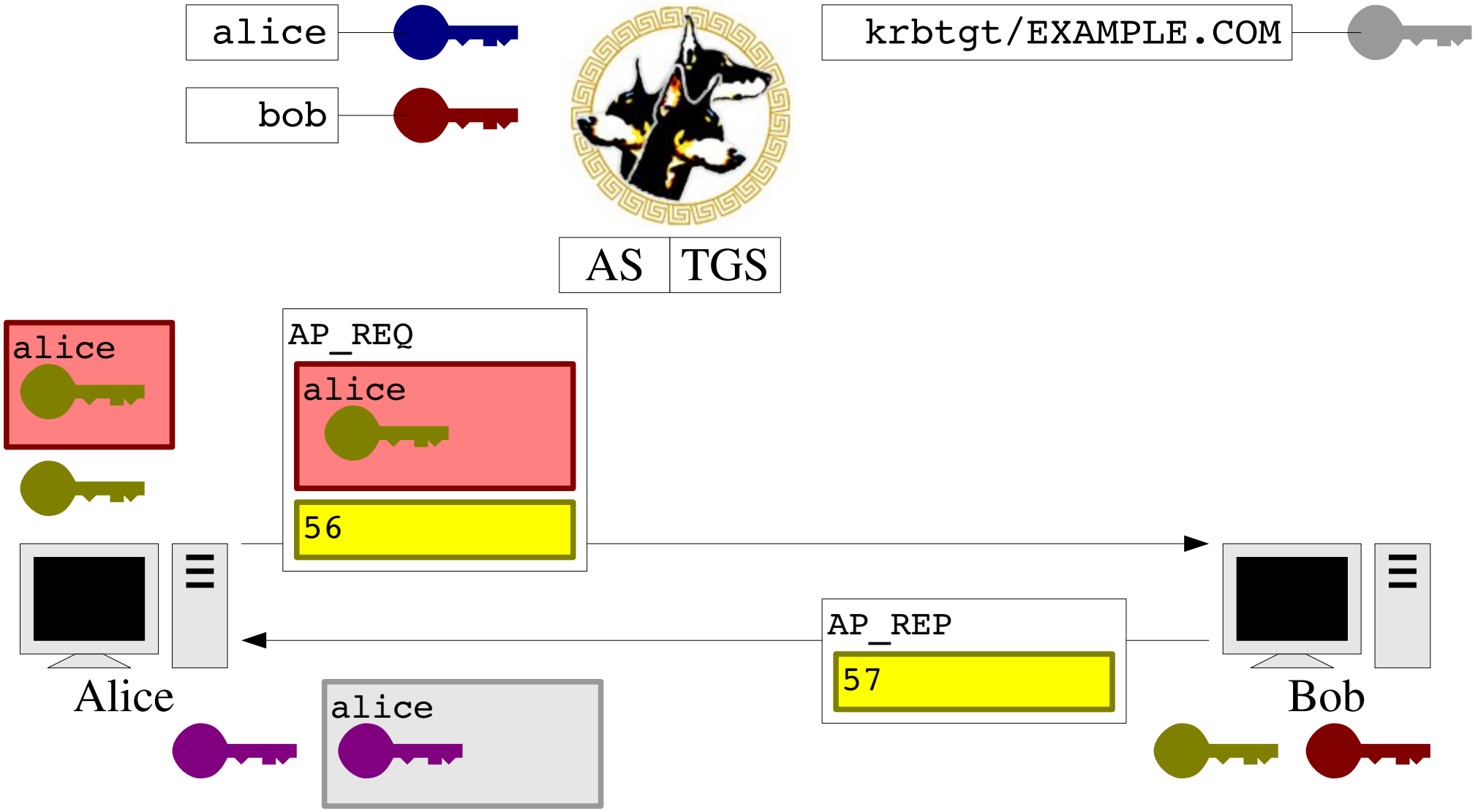
3 Kerberos
Ticket Granting Ticket



3 Kerberos
Ticket Granting Service



3 Kerberos
Ticket Granting Service

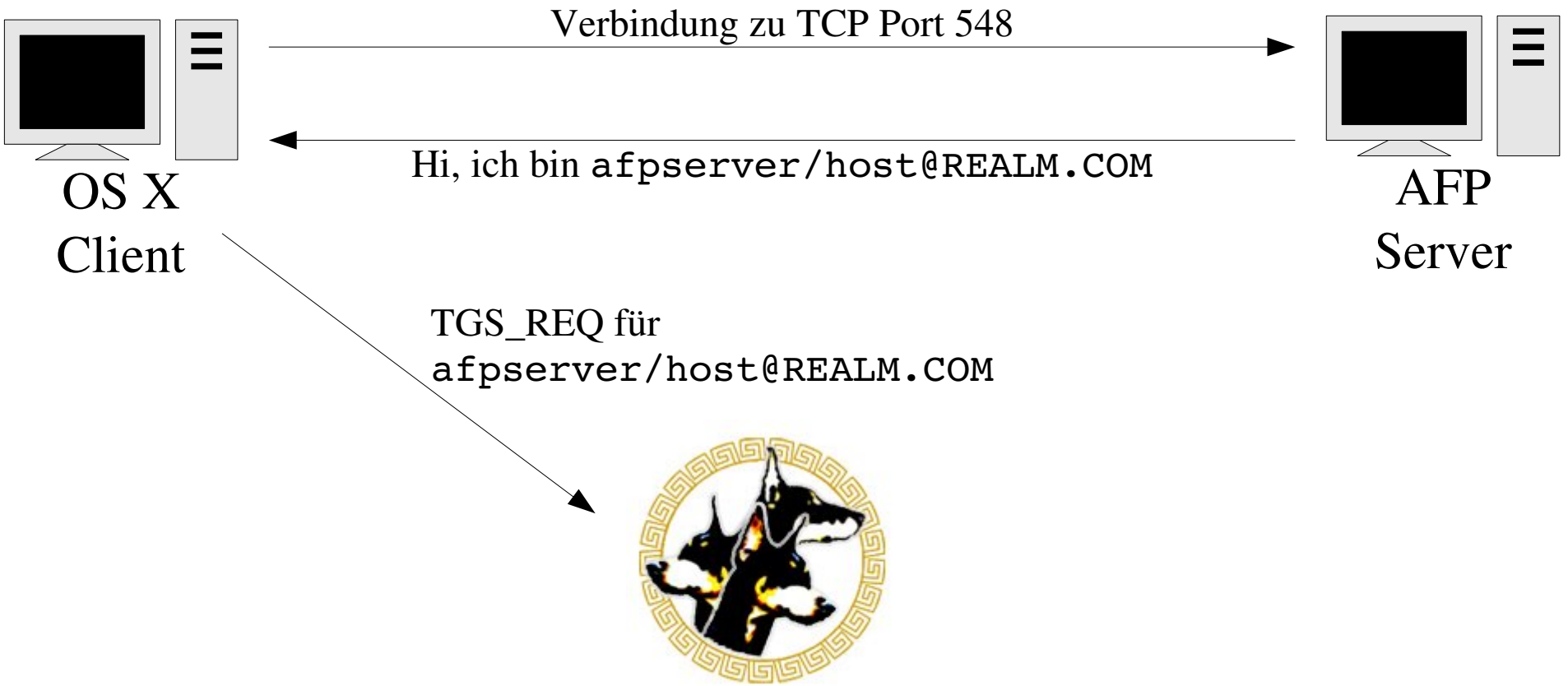


3 Kerberos
the good ...

- weitere Aspekte
  - alle Tickets besitzen Verfallsdatum, replay cache
  - preauthentication
  - cross-realm-Authentisierung
- Implementierungen, die Kerberos unterstützen (Auswahl):
  OpenLDAP, OpenSSH, Linux NFS, PostgreSQL, Samba, Microsoft Active Directory, Microsoft CIFS ...
- nicht immer ist Datenaustausch verschlüsselt!

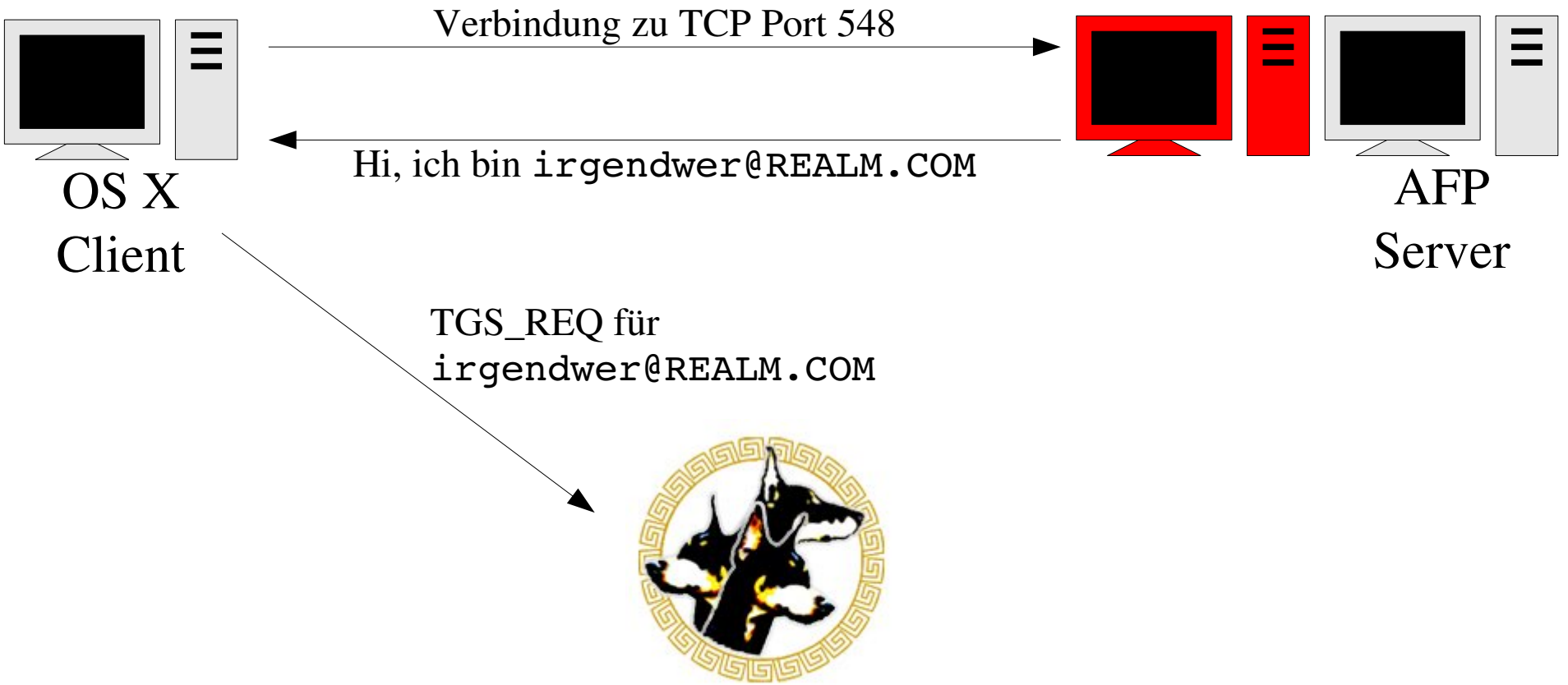
3 Kerberos
... the bad ...

- Unterstützung "dem Wortlaut nach"...



3 Kerberos
... getting ugly

- Einladung zu "Man-in-the-Middle"



Vielen Dank für
die Aufmerksamkeit!