8 Digitale Unterschriften

- Authentisierung des Absenders
  - ◆ Bilden eines Hash-Wertes über die zu übermittelnde Nachricht
    - Hash-Wert ist ein Codewort fester Länge
    - es ist unmöglich oder nur mit hohem Aufwand möglich, für einen gegebenen Hash-Wert eine zugehörige Nachricht zu finden
  - ◆ Verschlüsseln des Hash-Wertes mit dem geheimen Schlüssel des Absenders (digitale Unterschrift, digitale Signatur)
  - ◆ Anhängen des verschlüsselten Hash-Wertes an die Nachricht
  - ◆ Empfänger kann den Hash-Wert mit dem öffentlichen Schlüssel des Absenders dechiffrieren und mit einem selbst berechneten Hash-Wert der Nachricht vergleichen
  - ◆ stimmen beide Werte überein muß die Nachricht vom Absender stammen, denn nur der besitzt den geheimen Schlüssel

8 Digitale Unterschriften (2)

- Kombination mit Verschlüsselung
  - ◆ erst signieren
  - ◆ dann mit dem öffentlichen Schlüssel des Adressaten verschlüsseln
- ▲ Reihenfolge wichtig
  - ◆ Man signiere nichts, was man nicht entschlüsseln kann
- Heute gängiges Hash-Verfahren
  - ◆ MD5
  - ◆ 128 Bit langer Hash-Wert

8 Digitale Unterschriften (3)

- Woher weiß ich, daß ein öffentlicher Schlüssel authentisch ist?
  - ◆ Ich bekomme den Schlüssel vom Eigentümer (persönlich, telefonisch).
    - Hash-Wert auf öffentlichen Schlüsseln, die leichter zu überprüfen sind (Finger prints)
  - ◆ Ich vertraue jemandem (Bürge), der zusichert, daß der Schlüssel authentisch ist.
    - Schlüssel werden von dem Bürgen signiert.
  - ◆ Möglichst weite Verbreitung von öffentlichen Schlüsseln ereichen (z.B. PGP: Webserver als Schlüsselserver)
- ▲ Mögliche Probleme von Public key-Verfahren
  - ◆ Geheimhaltung des geheimen Schlüssels (Time sharing-System, Backup; Schlüsselpaßwort / Pass phrase)
  - ◆ Vertrauen in die Programme (z.B. PGP)
  - ◆ Ausspähung während des Ver- und Entschlüsselungsvorgangs

I.7 Authentisierung im Netzwerk

- Viele Klienten, die viele Dienste in Anspruch nehmen wollen
  - ◆ Dienste *(Server)* wollen wissen welcher Benutzer *(Principal)*, den Dienst in Anspruch nehmen will (z.B. zum Accounting, Zugriffsschutz, etc.)
  - ◆ Im lokalen System reicht die (durch das Betriebssystem) geschützte Benutzerkennung (z.B. UNIX UID) als Ausweis
  - ◆ Im Netzwerk können Pakete abgefangen, verfälscht und gefälscht werden (einfache Übertragung einer Benutzerkennung nicht ausreichend sicher)
- Public key-Verfahren
  - ◆ Authentisierung durch digitale Unterschrift (mit geheimen Schlüssel des Senders) und Verschlüsseln (mit öffentlichem Schlüssel des Empfängers)
  - ◆ Nachteile
    - jeder Dienst benötigt sicheren Zugang zu allen öffentlichen Schlüsseln
    - Verschlüsseln und Signieren mit RSA ist sehr teuer

I.7 Authentisierung im Netzwerk (2)

★ Einsatz von Authentisierungsdiensten

- ◆ zentraler Server, der alle Benutzer kennt
- ◆ Authentisierungsdienst garantiert einem Netzwerkdienst, daß ein Benutzer auch der ist, der er vorgibt zu sein

■ Benutzerausweis

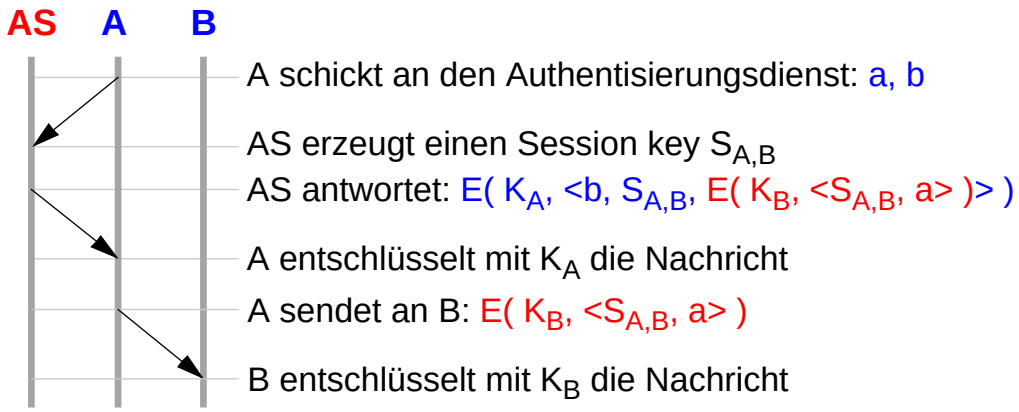
- ◆ der Authentisierungsdienst erkennt den Benutzer anhand eines geheimen Schlüssels oder Paßworts
- ◆ der Schlüssel ist nur dem Authentisierungsdienst und dem Benutzer bekannt

■ Vorgang

- ◆ Benutzer *(Principal)* will mit einem Programm *(Client)* einen Dienst *(Server)* in Anspruch nehmen
- ◆ durch geeignetes Protokoll erhalten Client und Server jeweils einen nur ihnen bekannten Schlüssel, mit dem sie ihre Kommunikation verschlüsseln können *(Session key)*

1 Einfacher Authentisierungsdienst

■ A will den Dienst B in Anspruch nehmen (Nach Needham-Schröder):



- ◆ K_X ist der geheime Schlüssel, den nur Authentisierungsdienst und X kennen
- ◆ nach dem Protokollablauf kennen sowohl A und B den Session key
- ◆ A weiß, daß nur B den Session key kennt
- ◆ B weiß, daß nur A den Session key kennt

1 Einfacher Authentisierungsdienst (2)

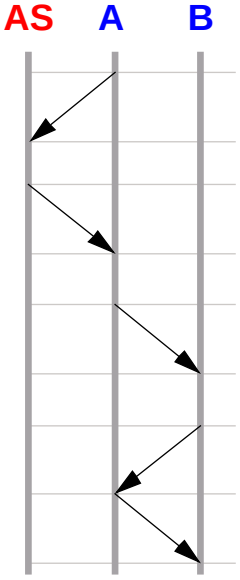
▲ Problem

- ◆ letzte Nachricht von A an B könnte aufgefangen und später erneut ins Netz gegeben werden *(Replay attack)*
- ◆ Folge: Kommunikation zwischen A und B kann gestört werden

★ Korrektur durch zusätzliches Versenden einer Verbindungsbestätigung durch B und A

2 Authorisierungsdienst mit Bestätigung

■ Bestätigung enthält Einmalinformation *(Nonce)*



A schickt an den Authentisierungsdienst: a, b, i_a

AS erzeugt einen Session key $S_{A,B}$

AS antwortet: $E(K_A, <i_a, b, S_{A,B}, E(K_B, <S_{A,B}, a>)>)$

A entschlüsselt mit K_A die Nachricht

A sendet an B: $E(K_B, <S_{A,B}, a>)$

B entschlüsselt mit K_B die Nachricht

B schickt an A: $E(S_{A,B}, i_b)$

A antwortet an B: $E(S_{A,B}, i_b+1)$

- ◆ ein Wiedereinspielen der Nachricht $E(K_B, <S_{A,B}, a>)$ oder $E(S_{A,B}, i_b+1)$ wird erkannt und kann ignoriert werden

2 Authorisierungsdienst mit Bestätigung

▲ Problem

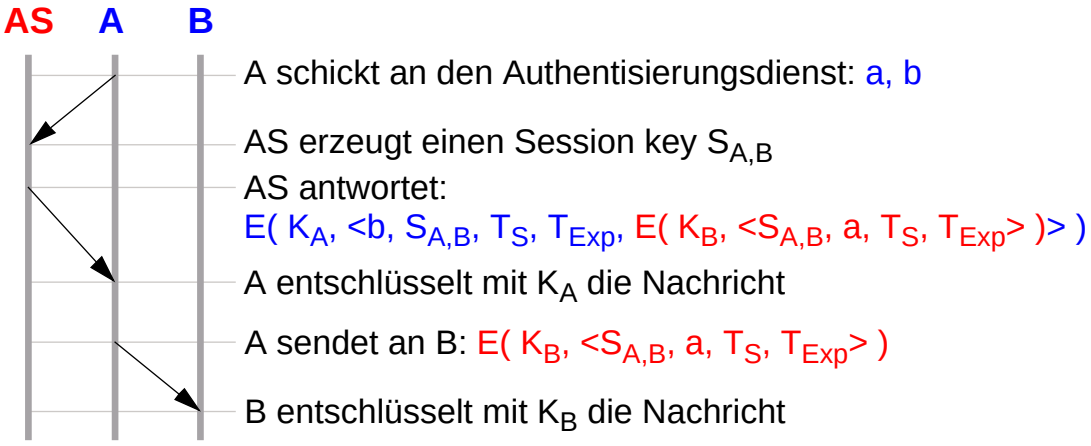
- ◆ Aufzeichnen von E(K_B, <$S_{A,B}$, a>) und
- ◆ Brechen von $S_{A,B}$ erlaubt das Aufbauen einer Verbindung.
- ◆ ein Dritter kann dann die erste Bestätigung abfangen und die zweite Bestätigung verschicken

★ Lösung

- ◆ Einführung von Zeitstempeln *(Time stamp)* und Angaben zur Lebensdauer *(Expiration time)*

3 Authorisierungsdienst mit Zeitstempeln

■ Authentisierungsdienst versieht seine Nachrichten mit Zeitstempeln



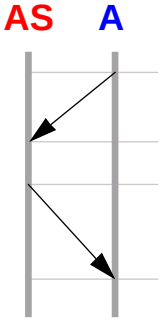
- ◆ T_S = Zeitstempel der Nachrichtenerzeugung
- ◆ T_{Exp} = maximale Lebensdauer der Nachricht
- ◆ aufgezeichnete Nachricht kann nach kurzer Zeit (z.B. 5min) nicht nocheinmal zum Aufbau einer Verbindung verwendet werden

4 Beispiel: Kerberos

- Kerberos V5
  - Softwaresystem implementiert Weiterentwicklung des Needham-Schröder-Protokolls
  - entwickelt am MIT seit 1986
- Ziel
  - Authentisierung und Erzeugung eines gemeinsamen Schlüssels durch den vertrauenswürdigen Kerberos-Server
- Idee
  - Trennung von Authentisierungsdienst und Schlüsselerzeugung
  - reduziert die nötige Übertragung einer Identifikation oder eines Paßworts zum Kerberos-Server

4 Beispiel: Kerberos (2)

- Benutzer holt sich zunächst ein Ticket vom Authentisierungsdienst



A schickt an den Authentisierungsdienst: a, tgs, T_{Exp}, i_a

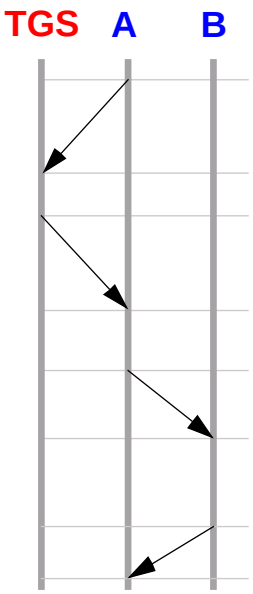
AS erzeugt ein Ticket für den Ticket Server tgs
AS antwortet:
$E(K_A, <S_{A,TGS}, tgs, T_{Exp}, i_a, E(K_{TGS}, <S_{A,TGS}, a, T_{Exp}>)>)$

A entschlüsselt mit K_A die Nachricht

  - das Ticket besteht aus $<S_{A,TGS}, a, T_{Exp}>$
  - es enthält einen Session key für die Kommunikation mit einem Ticket granting server, der dann die Verbindung zu einem Netzwerkdienst bereitstellen kann

4 Beispiel: Kerberos (3)

- Authentisierungsdienst versieht seine Nachrichten mit Zeitstempeln



A schickt an den Ticket granting server:
$E(S_{A,TGS}, T_s), E(K_{TGS}, <S_{A,TGS}, a, T_{Exp}>), b, T_{Exp}, i_a'$

TGS erzeugt einen Session key $S_{A,B}$

TGS antwortet:
$E(S_{A,TGS}, <S_{A,B}, b, T_{Exp}, i_a'>), E(K_B, <S_{A,B}, a, T_{Exp}>)$

A entschlüsselt mit $S_{A,TGS}$ die Nachricht

A sendet an B: $E(S_{A,B}, <T_s, C, S_{sub}>), E(K_B, <S_{A,B}, a, T_{Exp}>)$

B entschlüsselt mit K_B den zweiten Teil der Nachricht (Ticket) und mit $S_{A,B}$ den ersten Teil

B antwortet (optional): $E(S_{A,B}, T_s)$

- C = Checksumme zur Überprüfung der richtigen Entschlüsselung
- A kann mehrere Verbindungen mit seinem Ticket öffnen