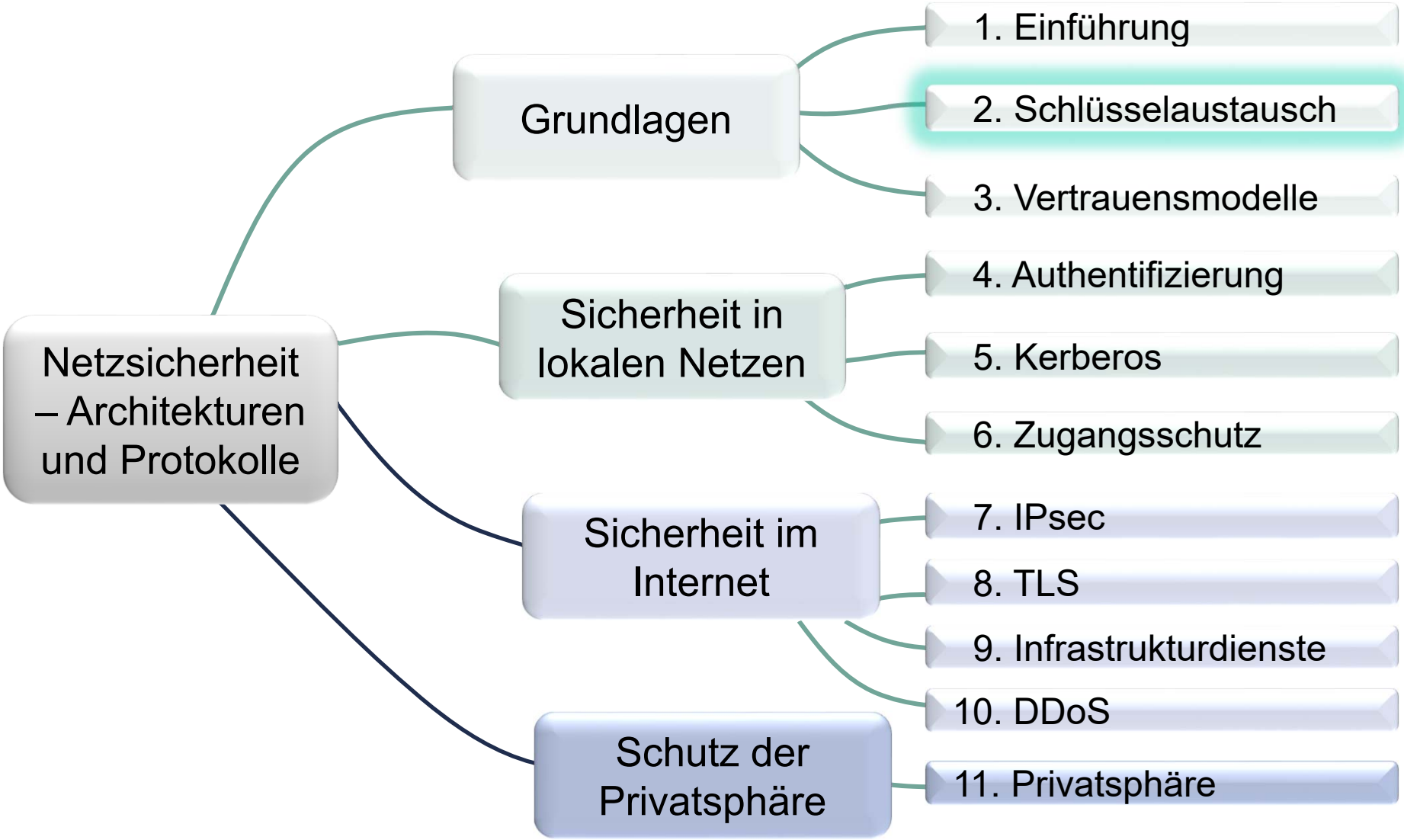
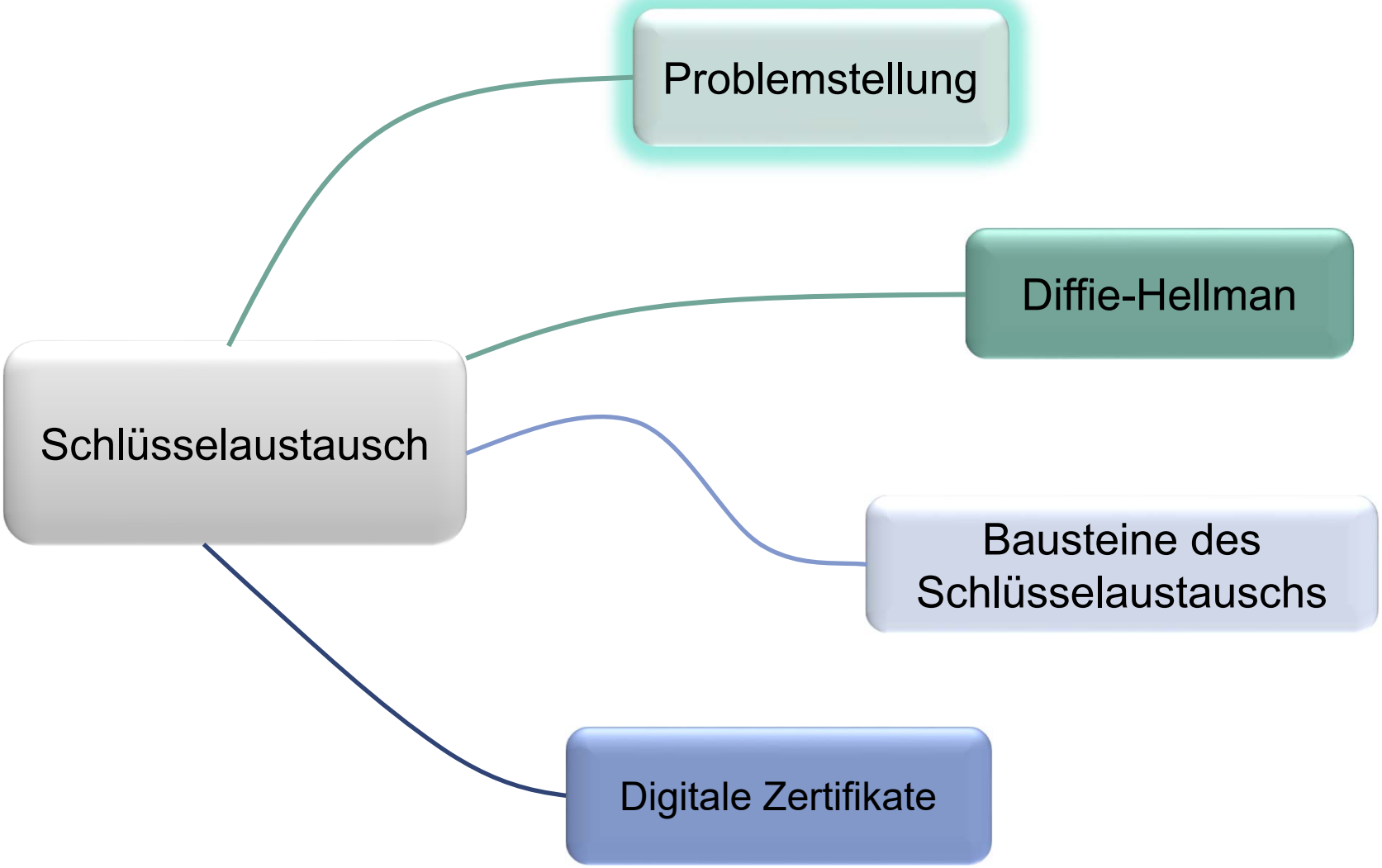
Vorlesung Netzsicherheit
Kapitel 2 – Schlüsselaustausch

PD Dr. Ingmar Baumgart, PD Dr. Roland Bless, Matthias Flittner, Prof. Dr. Martina Zitterbart
baumgart@fzi.de, [bless, flittner, zitterbart]@kit.edu

Inhalte der Vorlesung



Inhalte des Kapitels



Schlüsselaustauschverfahren

- Wie können Schlüssel sicher über einen ungesicherten Kanal ausgetauscht werden?
  - Austausch geheimer, symmetrischer Schlüssel

→ Schlüsselaustauschverfahren

- Aushandlung der Austausch-/Sicherungsverfahren zur gesicherten Kommunikation
- Authentizitätsüberprüfung des Kommunikationspartners
- Erzeugung gemeinsamer Schlüssel

Statischer Ansatz

- Einigung über das verwendete Verfahren und Austausch des Schlüsselmaterials durch persönliche Übergabe
  - Verschlüsselung/Authentifizierung der auszutauschenden Daten mit dem erhaltenen Schlüssel

- Sehr einfaches Verfahren
- Schlüssel ist automatisch authentifiziert

- „Persönliches Treffen“ notwendig
- Erneuerung der Schlüssel erfordert neues Treffen
- Schlechte Skalierbarkeit

Statischer Ansatz ohne persönliches Treffen

- Hinterlegung des Schlüsselmaterials bei einer vertrauenswürdigen Instanz (Trusted Third Party)
  - Vertrauenswürdige Verwaltung des Schlüsselmaterials
  - Herausgabe auf Anfrage der Kommunikationsteilnehmer

- Einfaches Verfahren
- Kein „persönliches Treffen“ notwendig
- Weniger Schlüssel bei den Kommunikationspartnern zu speichern

- Sicherer Kanal für Kommunikation zur vertrauenswürdigen Instanz erforderlich
- Schlüssel ist nur indirekt authentifiziert
- (Zentrale) Infrastruktur notwendig → Ausfall oder Kompromittierung betrifft alle Kommunikationsteilnehmer

Dynamischer Ansatz

- Nutzung asymmetrischer Verfahren (z.B. RSA)
  - Austausch eines geheimen, symmetrischen Sitzungsschlüssels über einen nicht vertrauenswürdigen Kanal
  - Verschlüsselung des Sitzungsschlüssels mit öffentlichem Schlüssel des Komm.-Partners

    - Kein „persönliches Treffen“ notwendig
    - Keine (zentrale) Infrastruktur notwendig
    - Sitzungsschlüssel sind authentifiziert (wenn öffentl. Schlüssel geprüft)
    - Dynamische Aushandlung von Schlüsselmaterial möglich

    - Sitzungsschlüssel an langlebiges Geheimnis gebunden, damit keine „Perfect Forward Secrecy“, d.h. nachträgliche Offenlegung der Kommunikation wenn asymmetrische Schlüssel kompromittiert
    - Rechenintensiv

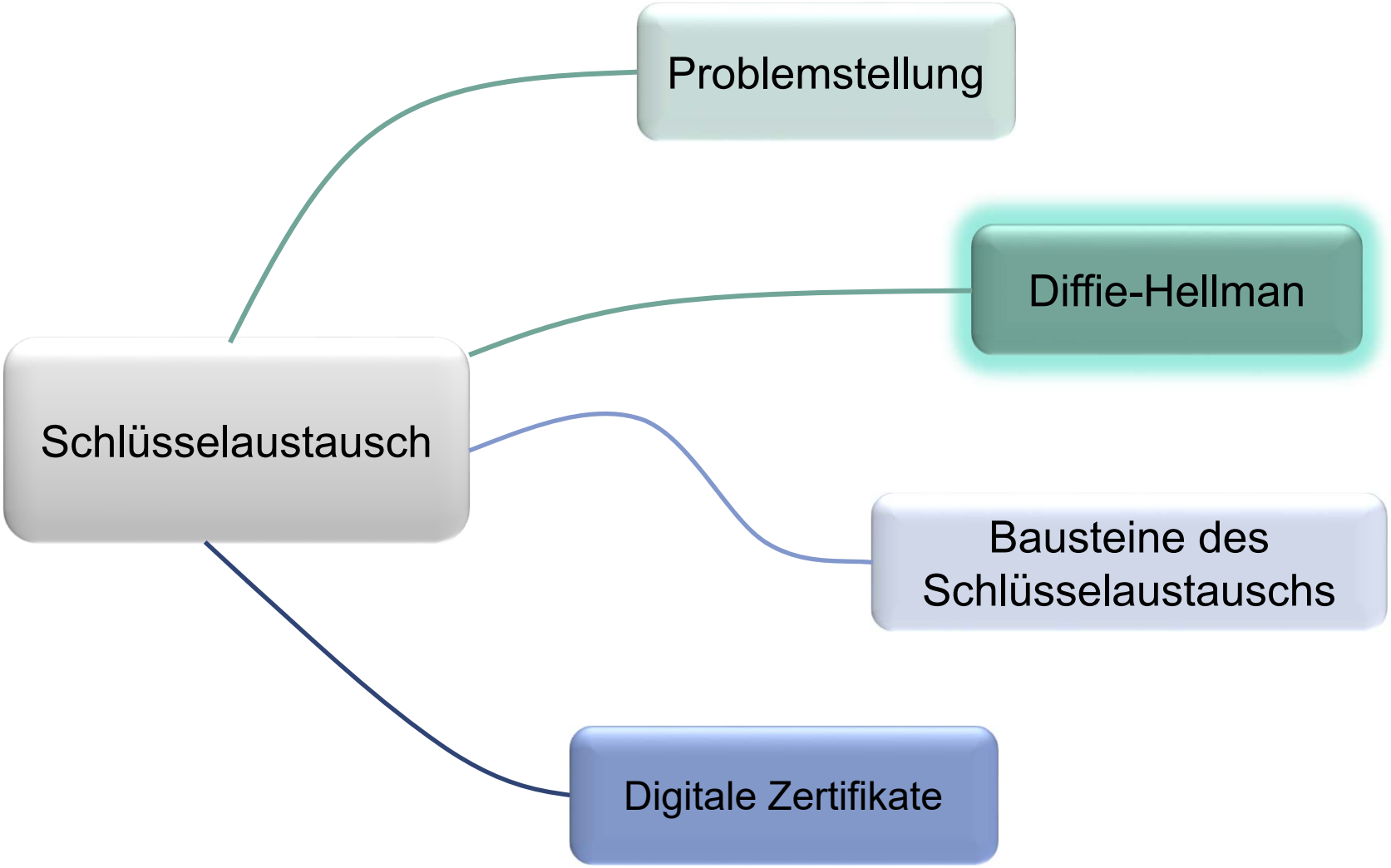
Dynamischer Ansatz

- Diffie-Hellman-Verfahren
  - Austausch eines geheimen, symmetrischen Sitzungsschlüssels über einen nicht vertrauenswürdigen Kanal

- Kein „persönliches Treffen“ notwendig
- Keine (zentrale) Infrastruktur notwendig
- Dynamische Aushandlung von Schlüsselmaterial möglich
  → ermöglicht „Perfect Forward Secrecy“ durch Generierung unabhängiger Sitzungsschlüssel

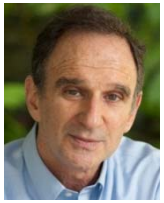
- Schlüssel sind nicht authentifiziert (sonst Zwischenschalten durch Angreifer möglich)
  → Zusätzliche Mechanismen notwendig
- Sehr rechenintensiv

Inhalte des Kapitels



Diffie-Hellman-Verfahren

- Problem
  - Wie können Schlüssel sicher über einen ungesicherten Kanal ausgetauscht werden?
- Diffie-Hellman-Verfahren
  - 1976 von Whitfield Diffie und Martin Hellman entwickelt

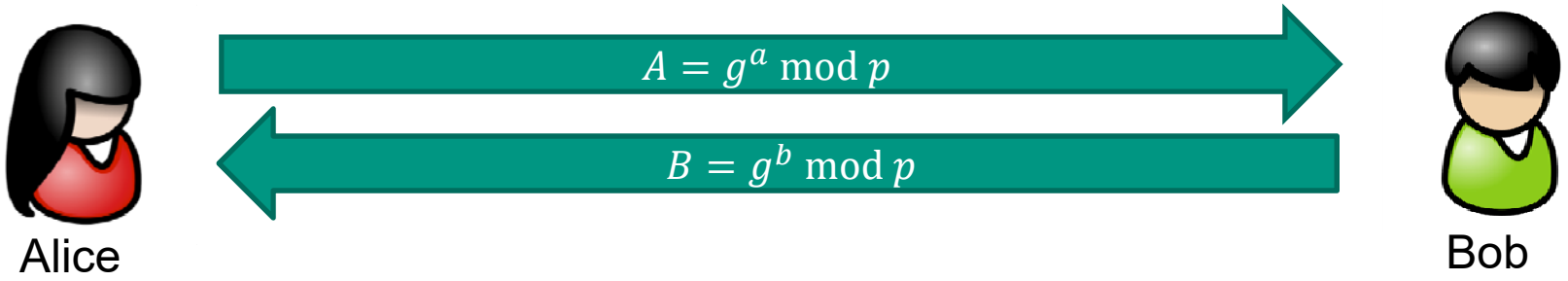
- Vorbereitung
  - Alice und Bob einigen sich auf eine große Primzahl p
  - Beide wählen gemeinsam Element g nach best. Rechenvorschrift
  - p und g sind öffentlich bekannt

- Schlüsselaustausch
  - Alice wählt Zufallszahl a mit $2 \leq a \leq p-2$ und berechnet
    - $\boldsymbol{A = g^a \bmod p}$ … öffentlicher Schlüssel von Alice
  - Bob wählt Zufallszahl b mit $2 \leq b \leq p-2$ und berechnet
    - $\boldsymbol{B = g^b \bmod p}$ … öffentlicher Schlüssel von Bob
  - a und b werden nicht bekannt gegeben!

[DiHe76]

Diffie-Hellman-Verfahren (II)

- Austausch der öffentlichen Schlüssel A und B



- Alice und Bob berechnen jeweils gemeinsamen geheimen Schlüssel K
  - Alice: $K = B^a \bmod p = \left(g^b\right)^a \bmod p = g^{ab} \bmod p$
  - Bob: $K = A^b \bmod p = (g^a)^b \bmod p = g^{ab} \bmod p$
  - Alice und Bob sind nun im Besitz eines gemeinsamen geheimen Schlüssels K

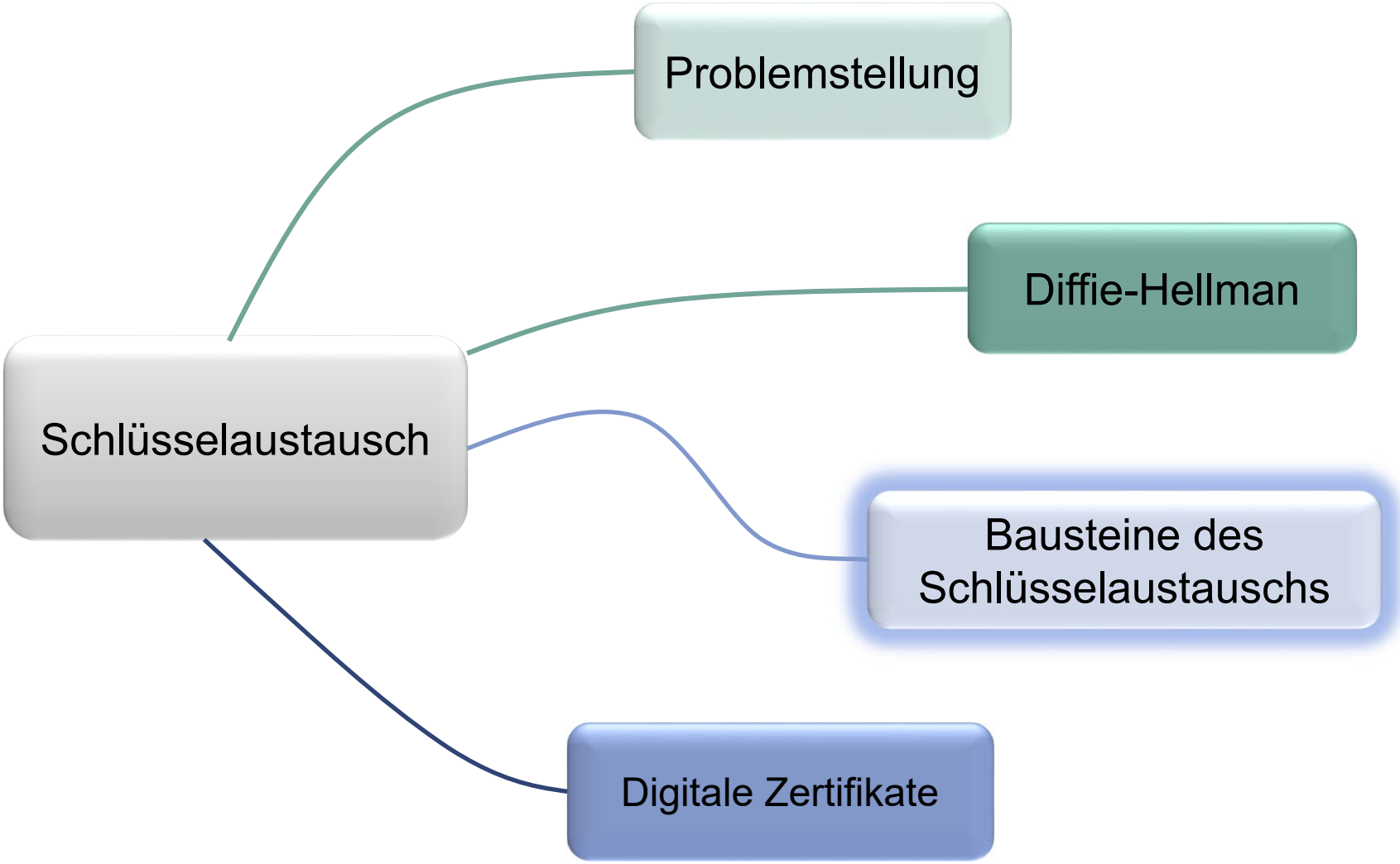
- Sicherheit des Verfahrens beruht darauf, dass es sehr schwierig ist, den diskreten Logarithmus in einem Primzahlkörper zu berechnen
  - d.h. es ist schwierig $b = \log g^b \bmod n$ zu berechnen

Diffie-Hellman-Verfahren (III)

- Vorteile
  - Gemeinsame Geheimnisse werden nur erzeugt wenn benötigt
  - Keine Infrastrukturunterstützung erforderlich
    - Lediglich Diffie-Hellmann-Gruppe muss bekannt sein
  - Geheime Zufallszahlen werden am Ende der Kommunikation gelöscht (→ Perfect Forward Secrecy)

- Nachteile
  - Anonymer Schlüsselaustausch
    - Keine Authentifizierung der Kommunikationspartner
  - Man-in-the-middle-Angriff möglich
  - Rechenintensiv
    - Also anfällig für DoS-Angriffe

Inhalte des Kapitels



Schlüsselaustausch-Protokolle: Bedrohungen

- Man-in-the-middle-Angriffe
  - Schlüsselaustausch wird unwissentlich mit dem Angreifer durchgeführt
- Wiedereinspielungsangriffe (Replay Attack)
  - Wiedereinspielen von zuvor aufgezeichneten Nachrichten
- Denial-of-Service-Angriffe (Denial of Service Attack)
  - Erschöpfen einer physischen (Speicher oder CPU-Zeit) oder einer virtuellen Ressource (Zustände)
- Downgrade Attack
  - Löschen von starken Algorithmen aus Liste der unterstützten Verfahren
- Schlüsselhinterlegung bei einer vertrauenswürdigen Organisation
  - Missbrauch des hinterlegten Schlüssels

Bausteine im Überblick

- Bausteine zum initialen Schlüsselaustausch
  - Perfect Forward Secrecy (PFS)
  - Schutz der Identitäten
  - Dynamische Wahl der Verfahren
  - Verhinderung von Downgrade-Angriffen
  - Einschränkung von DoS-Angriffen
- Bausteine für Schlüsselaustauschprotokolle nach dem initialen Austausch
  - Schlüsselerneuerung
  - Sitzungswiederaufnahme

Perfect Forward Secrecy (PFS)

- Ausgangslage
  - Dynamischer Austausch eines Sitzungsschlüssels
  - Authentifizierung mit Hilfe eines langlebigen Geheimnisses
    - Z.B. Nutzer-Passwort oder Zertifikate
  - Alice und Bob kommunizieren anschließend über gesicherten Kanal
    - Z.B. verschlüsselte Kommunikation unter Nutzung des Sitzungsschlüssels

- Definition Perfect Forward Secrecy (PFS)
  - Protokoll bietet PFS genau dann, wenn der Angreifer die Kommunikation nicht entschlüsseln kann, obwohl
    - Angreifer die komplette Kommunikation aufzeichnet
    - Danach in beide Endsysteme eindringt und das langlebige Geheimnis entwendet

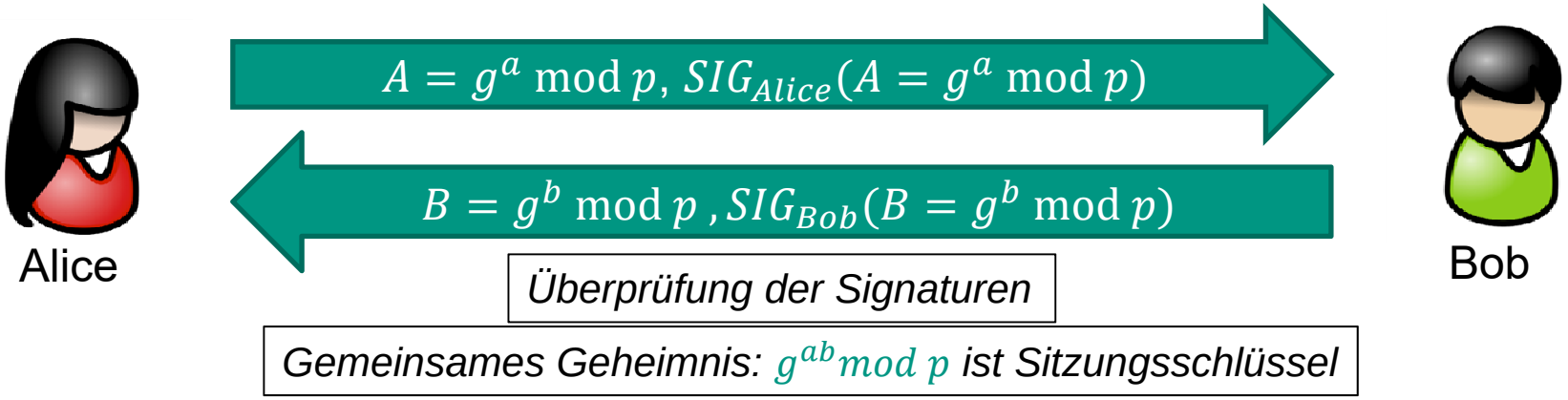
Perfect Forward Secrecy

- Maßnahmen um PFS zu erreichen
  - Sitzungsschlüssel muss unabhängig von langlebigem Geheimnis sein
  - Sitzungsschlüssel nicht aus vorigen Sitzungsschlüsseln berechnen
  - Vernichtung der Sitzungsschlüssel nach Ende der Kommunikation
  - Vernichtung jeglicher zur Berechnung genutzten Zustandsinformation
    - Zustand der Zufallszahlengeneratoren
  - Bei langlebiger Kommunikationsbeziehung periodische Schlüsselerneuerung
    - Erneuerung der Sitzungsschlüssel
- Beispiel
  - DH-Austausch mit Authentifizierung

Perfect Forward Secrecy

- DH-Austausch mit Authentifizierung
  - Alice und Bob authentifizieren sich über Nutzung digitaler Signaturen
  - Von Alice signierte Nachricht m: $SIG_{Alice}(m)$

- Vorgehensweise



- Vernichten des Sitzungsschlüssels und a bzw. b nach Ende der Kommunikation

Schutz der Identitäten

- Problem
  - Passiver Angreifer kann Identitäten der Kommunikationspartner abhören
- Lösung
  - Zunächst anonymer Diffie-Hellman Austausch, Übertragung der Identität geschützt durch ausgetauschten Schlüssel
    - Anschließend optionale Authentifizierung
  - Danach Übertragung des signierten DH-Werts
    - Vorherige Kenntnis der öffentlichen Schlüssel notwendig
    - Alternativ: Nutzung von Zertifikaten

Schutz der Identitäten

- Anonymer Diffie-Hellman + nachgelagerte Authentifizierung

Alice

$K = g^{ab} \; mod \; p$

$A = g^a \bmod p$

$B = g^b \bmod p$

Gemeinsames Geheimnis K berechnet

Bob

$K = g^{ab} \; mod \; p$

$Enc_K("Alice", SIG_{Alice}(A))$

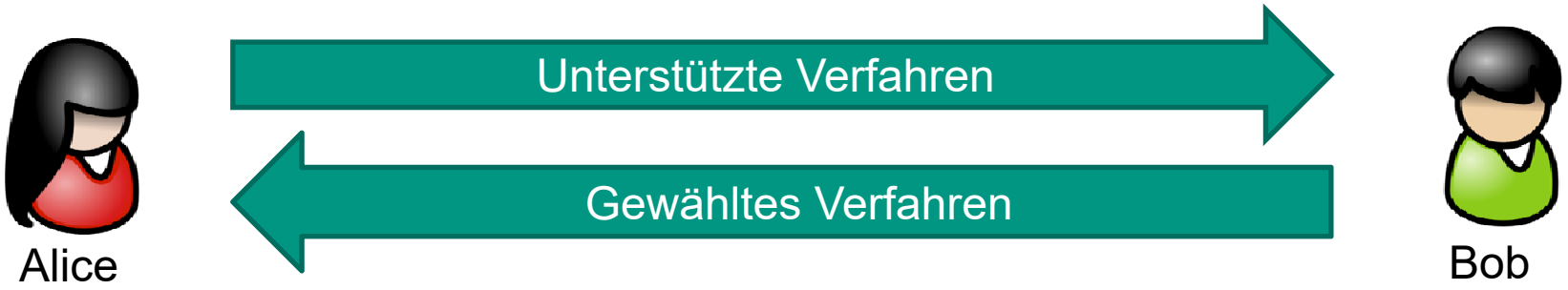
$Enc_K("Bob", SIG_{Bob}(B))$

Identitäten verschlüsselt übertragen

Dynamische Wahl der Verfahren

- Dynamische Wahl der genutzten Sicherungsmechanismen
  - Keine Festlegung durch Standardisierungsgremium notwendig
    - Verfahren für Interoperabilität
  - Einfache Migration zu kryptographisch stärkeren Verfahren
  - Ausschluss gebrochener Verfahren

- Einfacher Ansatz



…weiter mit Schlüsselaustausch..

Dynamische Wahl der Verfahren

- Problem 1: Komplexität des Protokolls
  - Wie werden Sicherungsmechanismen beschrieben?
  - Welche Kombinationen sind zulässig?

- Problem 2: Angreifer löscht die Verfahren, die er nicht berechnen kann
  - Downgrade-Angriff
  - Alice und Bob haben noch kein gemeinsames Geheimnis, um die Nachrichten zu schützen

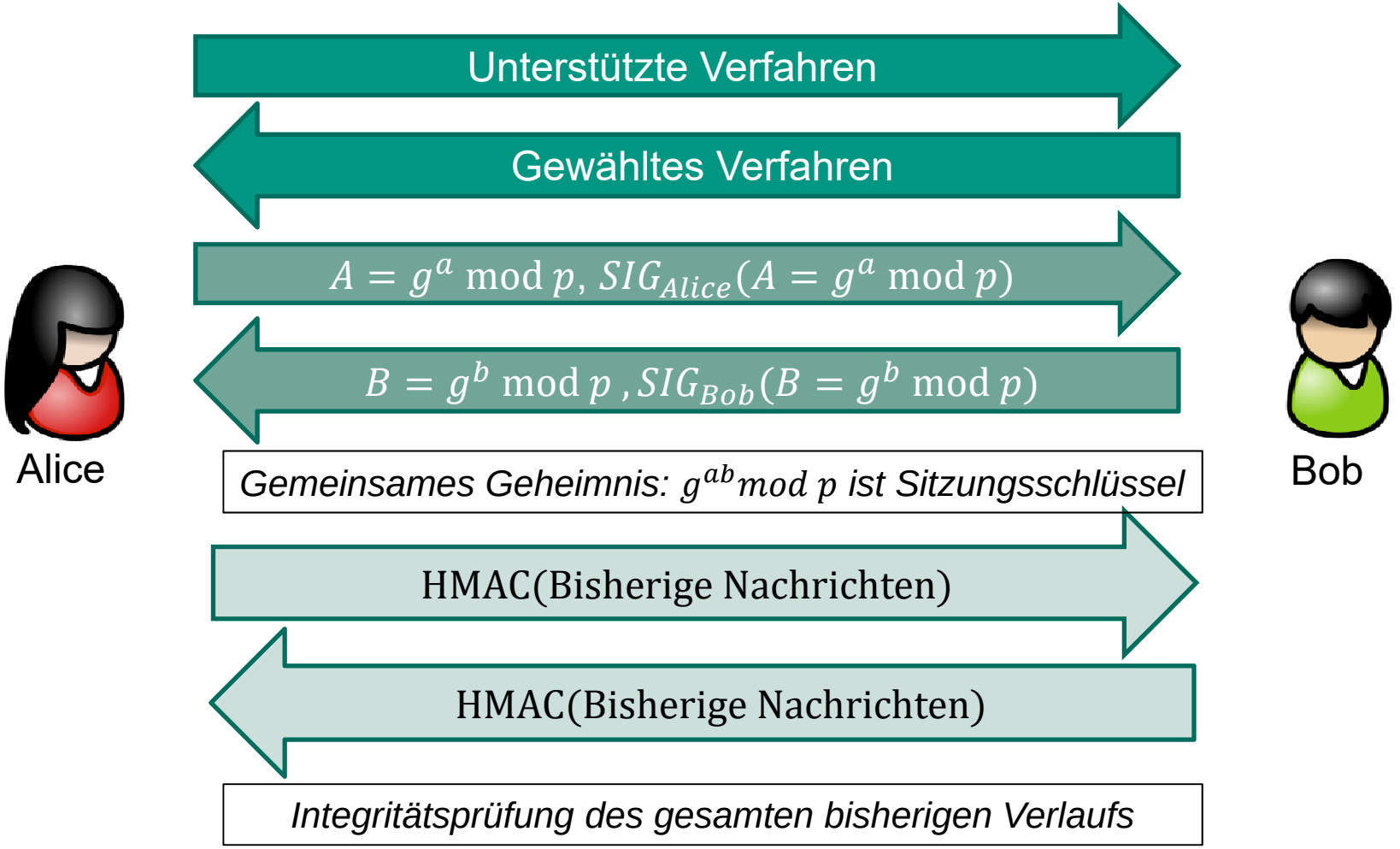
Verhinderung von Downgrade-Angriffen

- Ziel: Erkennung von Änderungen an Nachrichten
  - Z.B. Löschen kryptographisch starker Algorithmen

- Integritätsschutz über alle gesendeten Nachrichten und Felder
  - Anmerkung: bis jetzt noch keine vertraulichen Daten gesendet
  - Empfangene und berechnete Integritätsswerte stimmen nicht überein
    - Schlüsselmaterial ungültig machen
    - Verfahren abbrechen
  - Nutzung von digitalen Signaturen oder HMAC

Verhinderung von Downgrade-Angriffen

- Beispiel: Nutzung des HMAC-Verfahrens zur Überprüfung der Integrität



Einschränkung von DoS-Angriffen

- Ziel
  - Keine Durchführung rechenintensiver Operationen, so lange nicht klar ist, dass Absenderadresse nicht gefälscht ist, z.B.
    - Keine Zertifikatsüberprüfung
    - Keine Diffie-Hellman-Berechnung

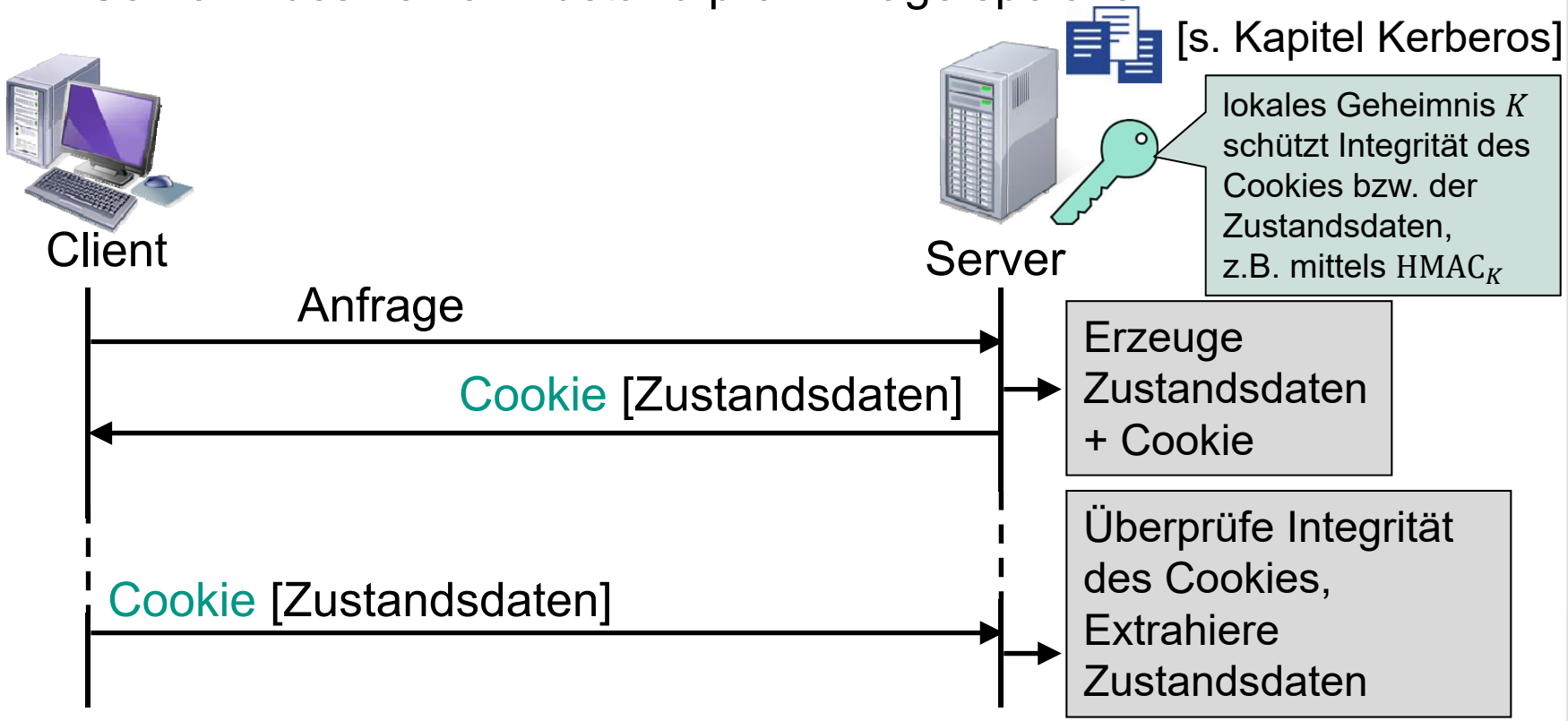
- Cookie 
  - Enthält zufällige Information des Angefragten
  - Wird Cookie zurück gesendet, ist mit hoher Wahrscheinlichkeit die Absenderadresse nicht gefälscht (kein IP-Spoofing)

- Puzzle 
  - Stellen einer rechenintensiven Aufgabe an den Anfragenden
  - Keine lokale Zustandshaltung beim Angefragten

Konzept: Tokens/Cookies

- Zustand vom Server durch Cookie auf Client auslagern

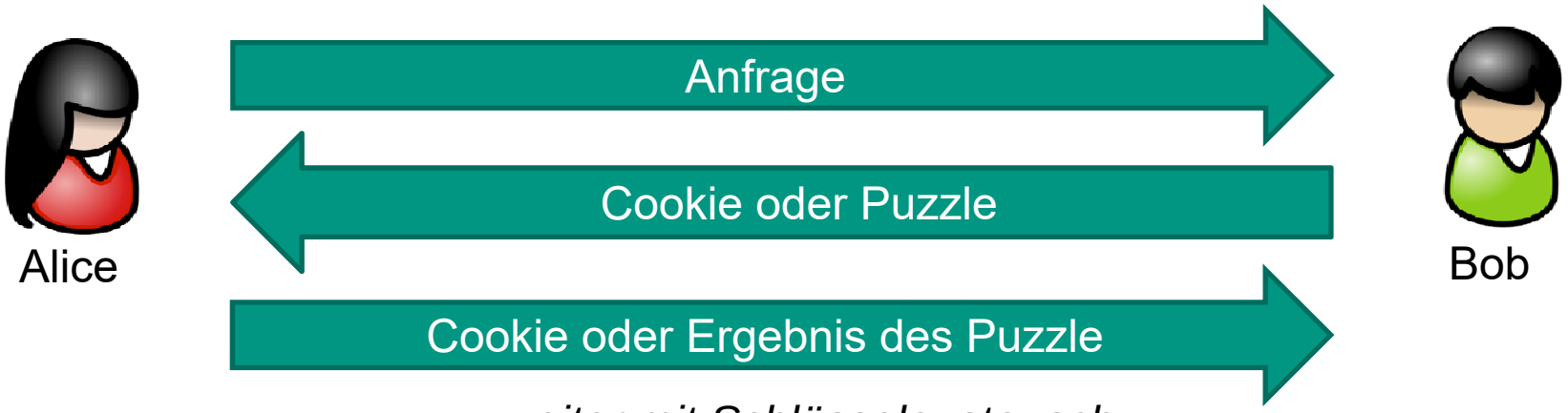
  → Server muss keinen Zustand pro Anfrage speichern



- lokales Geheimnis K sollte regelmäßig geändert werden

Einschränkung von DoS-Angriffen

- Hierzu: Verwendung von
  - Cookies oder Puzzles



…weiter mit Schlüsselaustausch…

Anforderungen an Cookies

- Aktualität (Frische!) und Eindeutigkeit
  - Cookie muss sich mit jeder Anfrage ändern, um Wiedereinspielungsangriffe zu verhindern
- Unvorhersagbarkeit
  - Cookie sollte nicht zu erraten sein, z.B. durch Sequenznummern oder Zeitstempel
- Einfache Erzeugung
  - hoher Aufwand würde Denial-of-Service begünstigen
- Leichte Verifikation
  - schnelles Verwerfen gefälschter/falscher Cookies

Ein Backrezept für Cookies!

- Cookie:= (Nonce n | Key-ID i | State s | $\mathrm{HMAC}_{K[i]}(n, i, s)$)
  - Nonce sorgt für Frische des Cookies und Schutz vor Wiedereinspielen [RFC7296]
  - lokales (Server-)Geheimnis ist Schlüssel $K[i]$
    - verlässt das System niemals
    - sollte für mehrere/viele Kommunikationsbeziehungen genutzt werden (kein Zustand pro Verbindung!)
  - Überprüfung des Cookies mit HMAC sollte nur stattfinden, wenn Key-ID i gültig, z.B. $\mathrm{i} \in [k \ldots k + l]$ aus 64-bit Raum, l klein gewählt, z.B. $l \leq 10$ → mildert Angriffe von blinden Angreifern (die Cookie nicht mitlesen können)
    - Key-ID ermöglicht einfachen Key-Rollover zwischen Anfragen
    - Server kann Schlüssel häufiger ändern, wenn Angriff vermutet wird, d.h., die Überprüfung der Cookies oft fehlschlägt
  - State s kann im Klartext oder verschlüsselt (mit lokalem Geheimnis) übertragen werden, sollte ggf. Peer-ID enthalten

Schlüsselerneuerung

- Aushandlung neuer Schlüssel notwendig, wenn
  - Lebenszeit des Schlüsselmaterials abgelaufen ist
  - Maximal zu schützende Datenmenge gesichert wurde
  - Sequenznummer überläuft

- Neuaushandlung der Schlüssel
  - Schlüsselerneuerung für Kontrollpfad (Schlüsselaustausch)
  - Schlüsselerneuerung für Datenpfad (gesicherte Kommunikation)
  - DH-Austausch, wenn PFS gefordert ist
    - Unabhängigkeit von langlebigem und aktuellem Schlüssel

Sitzungswiederaufnahme

- Problem: Initialer Schlüsselaustausch häufig teuer
  - Hoher Rechenaufwand durch
    - DH-Austausch
    - Überprüfung von Zertifikaten bzw. Signaturen
  - Hoher Kommunikationsaufwand
    - Typischerweise 2 RTTs oder mehr
    - Bei modularer Authentifizierung noch mehr
- Verbindungsabbruch bei einem der Kommunikationspartner bedingt erneuten Schlüsselaustausch

→ Sitzungswiederaufnahme (Session Resumption) ohne kompletten Schlüsselaustausch wünschenswert

Sitzungswiederaufnahme

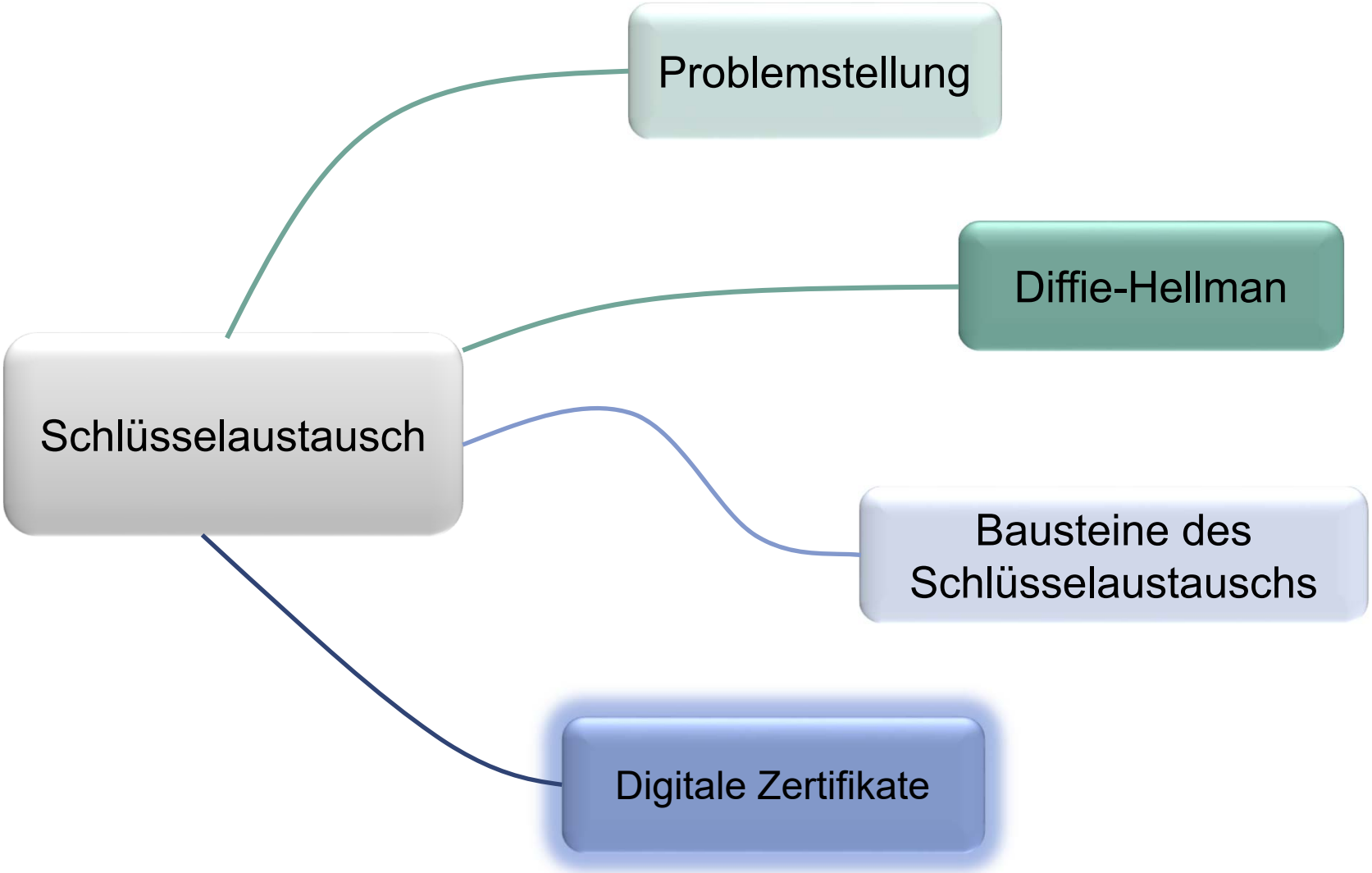
- Ansatz 1: zustandsbehaftet
  - Zustand nach Schlüsselaustausch wird bei einem Kommunikationspartner gespeichert
    - Unterscheidung verschiedener Zustände über Identifikatoren (IDs)
  - Wiederaufnahme
    - Vorlegen der ID um Zustand wieder neu zu etablieren
  - → Vergleiche TLS Session Resumption

[s. Kapitel TLS]

- Ansatz 2: zustandslos (für Bob)
  - Zustand wird komplett in geschütztes Cookie o.ä. codiert
  - Cookie wird ausgelagert und zur Wiederaufnahme von Alice vorgelegt
  - → Vergleiche Tickets bei Kerberos

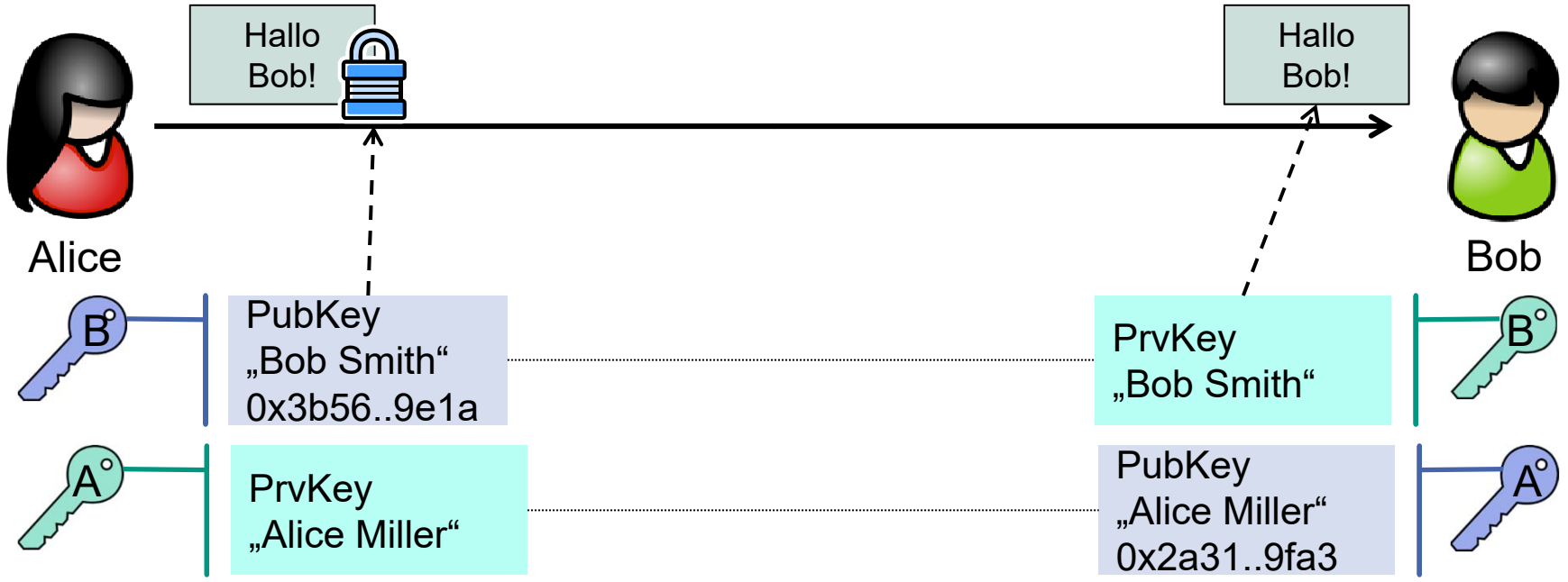
[s. Kapitel Kerberos]

Inhalte des Kapitels

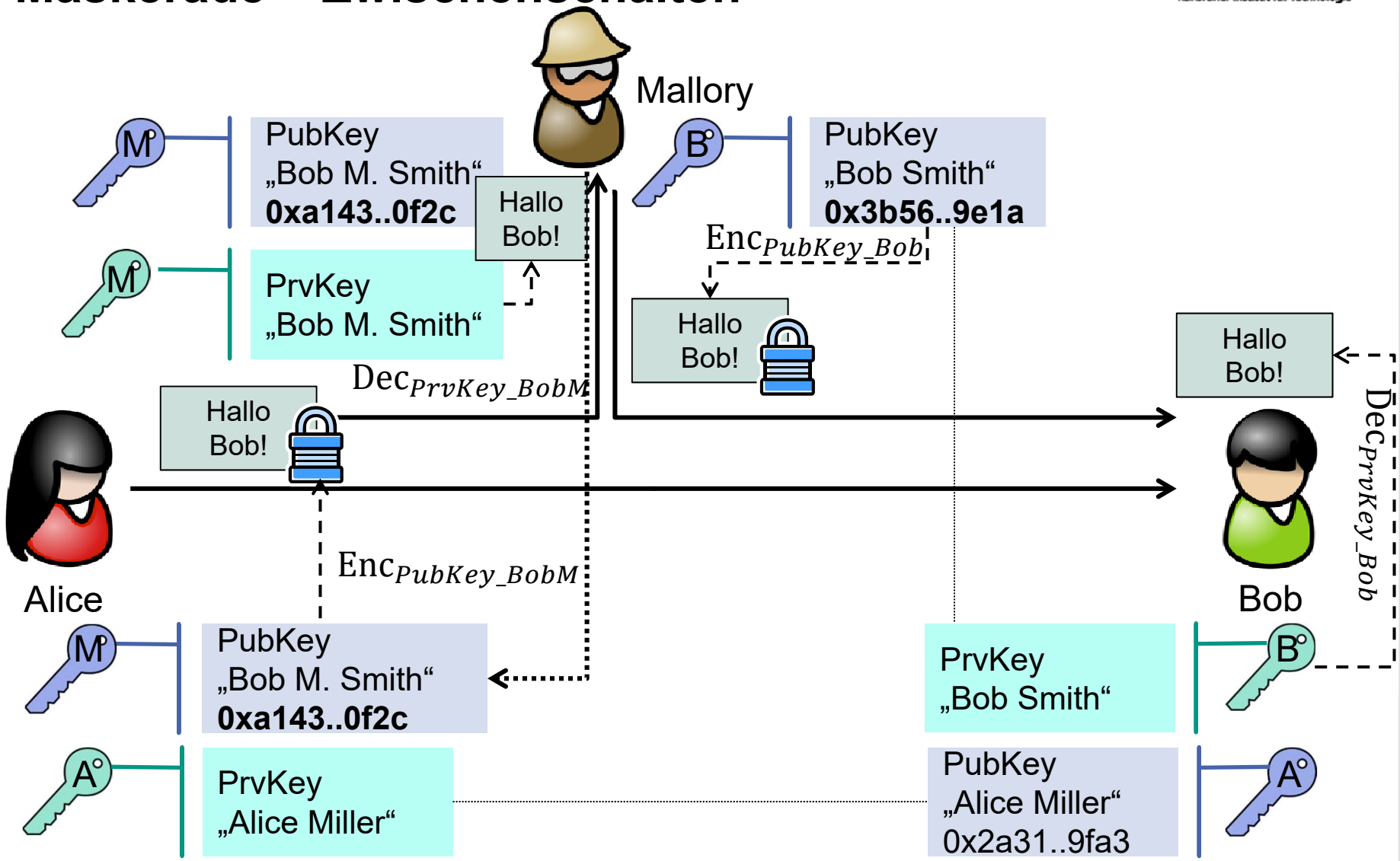


Schutzziel Authentifizierung

- Alice möchte mit Bob vertraulich kommunizieren
- Alice verwendet ein asymmetrisches Verschlüsselungsverfahren und benötigt Bobs öffentlichen Schlüssel
- Aber: woher weiß Alice, dass sie tatsächlich mit Bob kommuniziert?

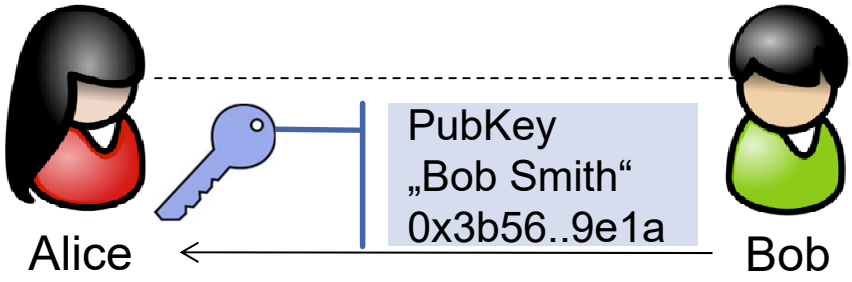


Maskerade + Zwischenschalten



Beziehen öffentlicher Schlüssel

- Alice benötigt öffentlichen Schlüssel von Bob
- Frage: woher bezieht man den öffentlichen Schlüssel?
  - Persönlicher Austausch?
    - Alice identifiziert Bob als Person
    - Bob muss versichern, dass 0x3b56..9e1a sein Public Key ist
    - Keine elegante Lösung für das Schlüsselverteilproblem



  - Via E-Mail oder Web-Site? → Maskerade ggf. ebenfalls möglich

→ Sichere Zuordnung benötigt:
Öffentlicher Schlüssel ⇔ Identität Kommunikationspartner

Lösungsvorschlag: Öffentliches Verzeichnis?

- Zuordnung: Name zu öffentlichem Schlüssel
  - ähnlich Telefonbuch
- Antworten auf Anfragen können symmetrisch geschützt werden
- Probleme des Ansatzes?
  - Authentizität der Zuordnung nicht transportierbar
  - Vertrauen in Verzeichnisdienstanbieter nötig (Bemerkung: Auch bei Verwendung einer PKI ist Vertrauen nötig, hier ist aber ein flexibleres Vorgehen möglich)
  - Verzeichnisdienstanbieter erfährt, wer mit wem kommunizieren will
  - Single Point of Failure: Bei Ausfall des Verzeichnisses keine Schlüssel-Anfrage mehr möglich
  - Schlechte Skalierbarkeit: Verzeichnis bei jeder Transaktion beteiligt

Digitale Zertifikate

Problemstellung

- Authentifizierung eines Sachverhaltes, den man nicht selbst überprüfen kann

→ man verlässt sich auf vertrauenswürdige Dritte, die ihn schon kontrolliert haben

Frage: was ist ein digitales Zertifikat?

- ein digitales Dokument, in dem eine Instanz einen bestimmten Sachverhalt mittels digitaler Signatur bestätigt

→ erzeugt Vertrauen in den Sachverhalt z.B. E-Mailadresse alice@wonderland.org → PubKey Alice

Frage: wer erstellt die Zertifikate?

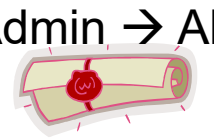
- eine vertrauenswürdige Instanz: Certification Authority (CA)
  - Beispiele für CA: Institution, Behörde, eine einzelne Person

Klassifizierung von Zertifikaten (I)

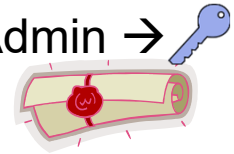
- ID-Zertifikate
  - öffentlicher Schlüssel → eindeutiger Name (Identität)
  - Authentifizierung von öffentlichen Schlüsseln
  - Realweltbeispiel für ID-Zertifikat: Reisepass
    - Ausstellung: Behörde (Land), Person+Urkunde → Ausweis
    - Überprüfen: Ausweis m. Foto/Fingerabdruck → Person → Name (Identität)
  - Haupteinsatz Internet: SSL/TLS f. Serveridentitäten, S/MIME f. E-Mail-Nutzer

Klassifizierung von Zertifikaten (II)

- Attributzertifikate
  - Attributwerte → Identität
  - bindet Attribute/Eigenschaft (z.B. Privilegien) an Identität
  - statt öffentlicher Schlüssel werden Attribute beglaubigt
  - Realweltbeispiel: Graduierungsurkunde, Führerschein (inkl. ID-Zert.)



- Autorisierungszertifikate
  - binden Privilegien → öffentlichen Schlüssel
  - Autorisation zur Nutzung von Diensten
  - Realweltbeispiel: Fahrkarte, Eintrittskarte zu Konzert



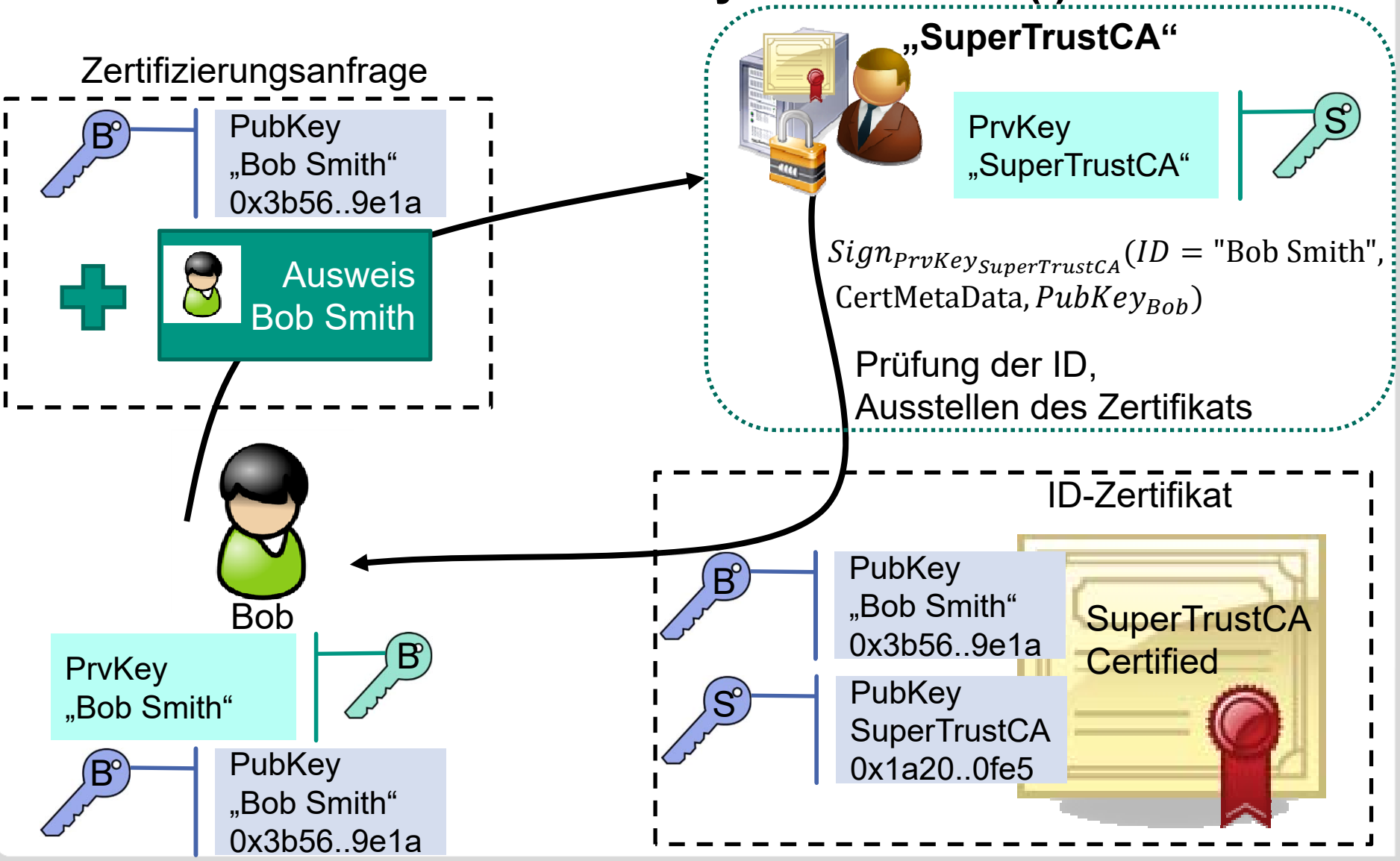
Digitale ID-Zertifikate

- ID-Zertifikat Bestandteile u.a.
  - Name des Subjekts
    - z.B. einer Person (Realname, E-Mailadresse), eines Servers (IP-Adresse, DNS-Name)
  - Öffentlicher Schlüssel des Subjekts
  - Seriennummer und Ausstellungszeitpunkt
  - Ablaufdatum
  - Signatur der CA
  - Name der CA
  - Öffentlicher Schlüssel der CA
  - Angaben zu den eingesetzten kryptographischen Verfahren

Zertifikate – Eigenschaften

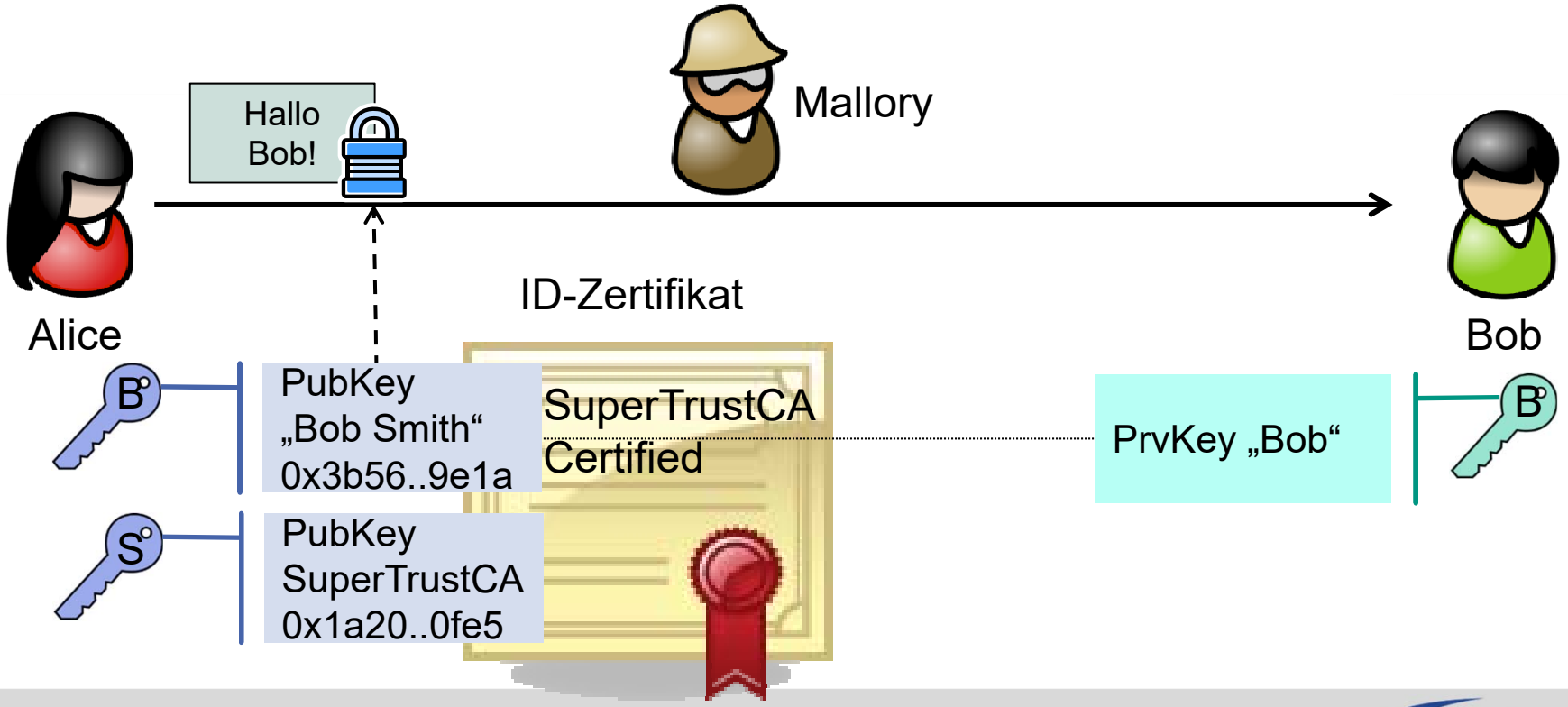
- Zertifikate ersetzen zentrale globale Listen
- „öffentliches Verzeichnis“ mit (ID, öffentlicher Schlüssel) durch ID-Zertifikat
- Theoretisch „offline“ und dezentral bzw. autonom überprüfbar, da digitale Signatur der CA überprüft werden kann (öffentlicher Schlüssel der CA bzw. Zertifikat der CA erforderlich)
- Transport des Zertifikats kann über ungeschützte Kommunikationsverbindungen erfolgen (Integrität durch Signatur gesichert!)
- Unterschiedliche Zertifikattypen für unterschiedliche Zwecke
  - Ausstellen von Zertifikaten
  - Signieren von Software
  - Nutzeridentitäten
  - Server-Identitäten

Alice und Bob – mit Public Key Zertifikaten (I)



Alice und Bob – mit Public Key Zertifikaten (II)

- Alice beschafft sich Bobs Zertifikat, z.B.
  - von Bob
  - aus einem öffentlichen Verzeichnis
  - von anderer Quelle…
- Alice prüft Signatur des ID-Zertifikats und Gültigkeit



Zusammenfassung

- Erzeugung und Austausch gemeinsamer Schlüssel
  - Statischer Ansatz: persönliches Treffen, Trusted Third Party
  - Dynamischer Ansatz: Diffie-Hellmann
- Aushandlung der Austausch-/Sicherungsverfahren zur gesicherten Kommunikation
  - Gefahr u.a. durch Downgrade-Angriffe
- Authentizitätsüberprüfung des Kommunikationspartners
  - Diffie-Hellmann bietet keinen Schutz vor Man-in-the-Middle-Angriffen
  - Zusätzliche Authentizitätsprüfung z.B. auf Basis digitaler Zertifikate

Literatur

[DiHe76] W. Diffie, M. E. Hellman; New Directions in Cryptography. In: IEEE Transactions on Information Theory. 22, Nr. 6, 1976, S. 644–654

[DiOW92] W. Diffie, P. C. van Oorschot, and M. J. Wiener; Authentication and Authenticated Key Exchanges. In: Designs, Codes and Cryptography 2 (2): , 1992, S. 107–125

[RFC7296] C. Kaufman, P. Hoffman, Y. Nir, P. Eronen, T. Kivinen; Internet Key Exchange (IKEv2) Protocol; Oct 2014