Table of Contents

1. Grundlagen der Rechnernetze

1.1. Systeme

- räumlich **verteilt** (bzgl. Kontroll- und Implementierungsstruktur, keine globale Systemsicht / -steuerung)
- **offen** → Normen notwendig
- **diensteintegriert** (vereint verschiedenste Dienste)
- **digital**
- Rechnernetze (verteilte, offene Systeme)
  - > erster | zweiter Art (verbunden über Kommunikationsnetze (z.B. elektromechanische Vermittlungseinrichtung) | Rechner wesentlicher Bestandteil von Netz)
- Kommunikationsnetze
  - > Verteilnetze
  - > Vermittlungsnetze
  - > öffentliche Weitverkehrsnetze
    - diensteintegriertes Digitalnetz, Breitband erwünscht (Glasfaser, ISDN, B-ISDN)
    - Funknetze (für ortsbewegte Teilnehmer und als Ersatznetz)
  - > LAN (lokale Rechnernetze)
  - > CAN (Controller AREA Networks) verbindet Rechner mit Instrumenten im Auto

1.2. Schutzziele

- Schutz von Inhalt, Teilnehmern und Standorten
- Schutz der Kommunikationsinhalte | Schutz der Kommunikationsumstände
- Integrität → partielle Korrektheit (Nachrichten können auch nicht ankommen, aber wenn sie ankommen, sind sie authentifiziert)
- Integrität + Verfügbarkeit → totale Korrektheit (vorausgesetzt, ausreichend Betriebsmittel verfügbar)

a) Vertraulichkeit

- Unerwünschtes verhindern
- Schutz vor beobachtenden Angriffen
  - > [perfekt] unbeobachtbar, anonym, unverkettbar
  - > Schutz der Nachrichtenlänge
  - > Timing
- nur befugter Informationsgewinn → (mehrere) Konzelationssystem(e), Ende-zu-Ende-Verschlüsselung
- Anonymität
  - – Schutz von Sender und Empfänger → Schutz der Kommunikationsbeziehungen
    - – verschiedene öffentliche Anschlüsse
      → Nachteil: lokal beschränkt, schwer zu realisieren, unbequem, Rückschlüsse aus

Kommunikationsbeziehungen möglich (z.B. Wohnort)
    - Mix-Netz mit beidseitigen anonymen Rückadressen
- Schutz des Senders
    - DC-Netz
- Schutz des Empfängers
    - Verteilung, anonymes Abfragen
- Schutz der Interessensdaten
    - lokale Auswahl
    - öffentliche Anschlüsse
    - zeitlich entkoppelte Verarbeitung
    - sinnvoll bei unverschlüsselten Informationen oder wenn Netzbetreiber = Kommunikationspartner

- Verdecktheit (Nachrichtenexistenz nur Befugten bekannt) → Steganografie
- Unbeobachtbarkeit (Kommunikation verläuft komplett verdeckt) → Ressourcen & Dienste, die z.B. einem anderen Zweck dienen könnten

b) Integrität

- Erwünschtes leisten
- Schutz vor verändernden Angriffen
- keine unbefugte Modifikation von Informationen → (mehrfache) Authentifikation
- Zurechenbarkeit (Kommunikationsparter können sich authentifizieren) → Signatur
- Zeitstempel, Empfangsquittierungen
- Verbindlichkeit (Verträge, Verantwortung) → digitales Zahlungssystem

c) Verfügbarkeit & Erreichbarkeit

- Funktionalität wird nicht unbefugt beeinträchtigt → diversitäre Netze
- Inhalte und Funktionen stehen in benötigtem Maß zur Verfügung → faire Aufteilung von Betriebsmitteln

d) zu schützende Daten

- technisch: Nutzdaten, Vermittlungsdaten
- inhaltlich: Inhaltsdaten, Interessensdaten, Verkehrsdaten

e) Mehrseitige Sicherheit

- Einbeziehung der Schutzinteressen aller Beteiligten
- Austragen daraus resultierender Schutzkonflikte (geklärtes Kräfteverhältnis)

1.3. Angreifer

a) Angreifermodelle

- Zu beachten: Lebensdauer das Systems → Umstände können bestimmtes Angreifermodell verändern
- Naturgesetze / Naturgewalten
- Computerviren (lassen sich beschränken auf Trojanische Pferde, da physische Schutzmechanismen, principle of least privilege und Signaturen vor reinen CVs schützen – da das viel zu selten angewandt wird, funktionieren Trojanische Pferde)
- Personen
    > Außenstehende

> An der Entstehung & Wartung des Systems Beteiligte (im weitesten Sinn)

> Betreiber des Systems

> Nutzer des Systems

- Rechenkapazität

> komplexitätstheoretisch unbeschränkt = informationstheoretischer Angreifer

> komplexitätstheoretisch beschränkt = komplexitätstheoretischer theoretischer Angreifer

- Zeit-, Geldkapazitäten (Bestechung etc.), mit welcher Wahrscheinlichkeit soll er nicht in der Lage sein, erfolgreich anzugreifen, wie viel partielle Informationsgewinnung zählt als Angriff?
- beobachtend | verändernd

> wie wird der Angreifer in seiner aktuellen Rolle vom betroffenen IT-System bzw. wie wird das betroffene IT-System von außen wahrgenommen? (Erlaubnis der Handlung)

- Passiv
  - Konzelationssysteme:
    - **ciphertext-only attack**
    - **known-plaintext attack**
  - Authentifikationssysteme:
    - Angreifer beobachtet, bis er Nachricht zu fälschen versucht
- Aktiv (Kooperation des Opfers z.B. durch angeblichen Systemtest, Frage-Antwort, aber kein Angriff durch Bit-für-Bit-Kopie einer schon gesendeten Nachricht und Erhaltung deren Entschlüsselung, dagegen kann niemand was.)
  - Konzelationssysteme:
    - **chosen-plaintext attack** [sym.]
      - Angreifer wählt Nachrichten und erhält Verschlüsselung
    - **chosen-ciphertext attack**
      - Angreifer wählt Schlüsseltext und erhält Entschlüsselung
      - täuscht anderen Absender vor [sym.]
  - Authentifikationssysteme:
    - **chosen-signtext attack**
      - Angreifer wählt Nachricht und erhält MAC
    - **chosen-MAC attack** [sym.]
      - Angreifer wählt MAC und erhält Nachricht(en) (unrealistisch)
- Zusammenhang beobachtend/verändernd/passiv/aktiv!
- **Adaptivität** (Nachrichtenwahl nur zu Beginn oder je nach vorherigem Ergebnis wählbar?)

b) Beispiele

- Trojanisches Pferd (Prozess tut mehr, als er soll, oder nicht bzw. falsch, was er soll)

> transitiv (Rekursivität vergrößert potentiellen Angreiferkreis)

- Problem: Personenkreis ist unbekannter, schwerer / unakzeptabel zu kontrollieren

> universell: Handlungsanweisungen sind möglich

- (verdeckter) Eingabekanal
- Redundanz in der Eingabe, damit niemand sonst Handlungen auslösen kann

> unbefugter Informationsgewinn ((verdeckter) Ausgabekanal)

> unbefugte Modifikation von Informationen

> unbefugte Beeinträchtigung der Funktionalität

- Bedienfehler, zu leichte Passwörter

c) Angriffspunkte

– bei kryptografischen Systemen:
  – Definition eines kryptografischen Systems
  – Wahl des Sicherheitsparameters
  – Wahl der Schlüssel
  – Bearbeiten von Nachrichten

d) Angriffsziele

– bei kryptografischen Systemen:
  – **vollständig**es Brechen (Kenntnis des Schlüssels)
  – **universell**es Brechen (adäquates Verfahren zum Schlüssel)
  – nachrichtenbezogenes Brechen (Fälschen bzw. Entschlüsseln von bestimmter Nachr.)
    – **selektiv**es Brechen (selbst gewählte Nachricht)
    – **existentiell**es Brechen (irgendeine Nachricht) → nicht sinnvoll bei Konzelationssystemen
– zusätzliche Unterscheidung bei Konzelationssystemen:
  – **komplette Entschlüsselung**
  – **Partielle Information** → nur manche Eigenschaften (einzelne Bits, Quersumme)

1.4. Schutzmöglichkeiten

- Verbote / Gesetze
  > feste Definition von „befugt“ und „unbefugt“
  > Einhaltung muss überprüft, Strafverfolgung gesichtert und der ursprüngliche Zustand durch Schadensersatz wiederhergestellt werden → bei IT-Systemen nicht gegeben (Datendiebstahl, unberechtigte Weiterverarbeitung, Löschung von unberechtigten Daten)
- vorbeugende technische Schutzmaßnahmen bezogen auf
  > Entwurf, Prototypen, Endprodukt und Wartungstätigkeiten
    - verständliche Sprachen und Ergebnisse
    - Analyse mit unabhängigen Werkzeugen
    - Diversität: Entwurf durch mehrere unabhängige Entwerfer (verhindert Wkt. für covert channels)
  > Betreiber, Benutzer
    - Zugriff des auf Produkt physisch und logisch beschränken bzw. protokollieren
  > Außenstehende
    - Fernhaltung: physisch vom System, kryptografisch von Daten

a) physisch

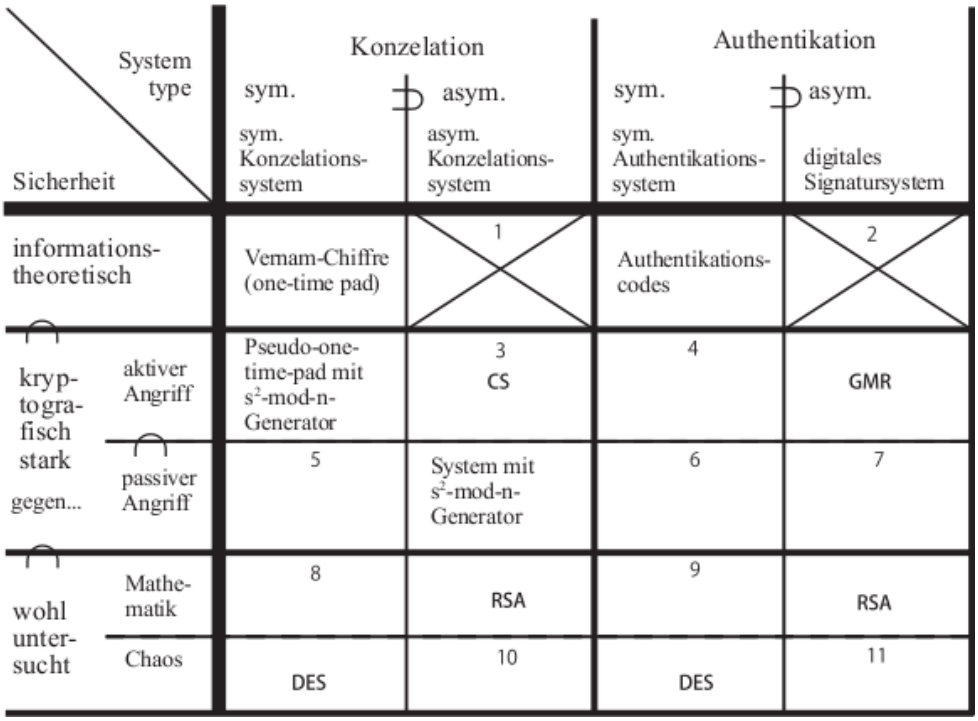
– es muss (mindestens) einen sicheren Bereich geben, in dem logische Schutzmaßnahmen umgesetzt werden können
– verteilte Systeme → Problem mit Vertraulichkeit und Integrität (wenn ein System verletzt wird), wirkt nur bei „perfektem“ physischen Schutz jeder Station
– **Schirmung** gegen Analyse elektromagnetische Strahlung
– **Erkennung** verändernder Eingriffe, deren **Bewertung**
– bei unbefugtem Eingriff **Verzögerung**, vor Ende der Verzögerung **Löschung** der Informationen
– schalenförmige Konzeption

- Nachteil:
  - Validierung, keine genaue Kenntnis aktueller Technik
  - Preis, Aufwand bei Wartungsarbeiten
  - Beispiel Chipkarten

→ möglichst wenig physischer Schutz → möglichst viel Selbstkontrolle (organisatorischer Schutz)

b) logisch

- Identifikation
  - Typ
    - Mensch (Körpermerkmale & Verhalten, Besitz, Wissen)
    - IT-System (Aussehen & Verhalten, Wissen, Standort & Leitung (IP etc.))
  - Methode (zu Beginn, in Zeitintervallen oder kontinuierliche Identifikationsprüfung)
- Zugangskontrolle
- Zugriffskontrolle: Zugriffsmonitor
  - Autorisierung & Rechtebeschränkung bei Operationen
  - Protokollierung
  - Kontrolle des Ressourcenverbrauchs)
- Signatur
- Kryptografie
- Steganografie
- Organisation
  - principle of least privilege (bezüglich des Tuns und Lassens) → nie garantierbar richtig
- Wahnehmung von Schutzmöglichkeiten in verteilen Netzen (s. Kapitel 7)
- Probleme
  - Symmetrie – Austausch von persönlichen Daten mit ungewolltem Partner & vergeudete Zeit
  - Keine Entscheidungsfunktion für Viren möglich
  - Bekannte Viren sind nicht identifizierbar, da viele Programme sich selbst so modifizieren könnten, dass sie zu diesem Virus werden – zu viel Rechenaufwand, um das auszuschließen
  - Nicht feststellbar, was von Virus modifiziert wurde, da dieser seine Modifikationsfunktionen modifiziert haben könnte
  - Beweis der totalen Korrektheit eines Programms aus Komplexitätsgründen nicht machbar, Überprüfung des Schutzes nicht in ausreichendem Aufwand realisierbar



2. Kryptologie

- **Kryptografie**: Funktionsweise von Geheimschriften → die Guten
- **Kryptoanalyse**: Analyse, Entschlüsselung von Geheimschriften → die Bösen
  - Untersuchungen zu Buchstabenhäufigkeiten
    - jede Sprache hat charakteristische Verteilungskurven
  - Auswertung von Bi- und Trigrammhäufigkeiten (n-Grammhäufigkeiten)
  - zu beachten:
    - Text muss ausreichend lang für eine Analyse sein
    - Fachtexte mit höherem Anteil von Fremdwörtern können leicht andere Verteilungskurven haben als andere Texte

2.1. Zweck und nötige Schlüsselverteilung

a) Konzelationssysteme

- Vertraulichkeit der Kommunikationsinhalte
- Allgemeine Schutzmechanismen:
  - Länge der Nachrichten standardisieren → mit Nullen auffüllen
  - Nachricht in Nachrichtenstrom senden → schwere Unterscheidung zwischen Nonsense und Nachricht

symmetrisch

- Schlüssel k
- Schlüsselaustausch persönlich oder über (mehrere) Schlüsselverteilzentrale(n)
- n Teilnehmer → n*(n-1)/2 Schlüssel
- Vertreter:

- Vernam-Chiffre
- DES
- Diffie-Hellman Schlüsselaustausch

asymmetrisch

- öffentlicher Chiffrierschlüssel c (z.B. öffentliches Register, das den Schlüssel auch authentifizieren kann)
- geheimer Dechiffrierschlüssel d
- Klartext wird mit Zufallszahl kombiniert → Indeterminismus, Zufallszahl wird beim Entschlüsseln ignoriert
- n Teilnehmer → n Schlüsselpaare
- Vertreter:
  - RSA

b) Authentikationssysteme

- Integrität

symmetrisch

- Schlüssel k
- [message authentification code] MAC = k(x)
- Angreifer kennt alle möglichen MACs und kann sie ausprobieren, aber nicht testen
- Vertreter:
  - Authentikationscodes
  - DES

asymmetrisch → digitale Signatursysteme

- geheimer Signierschlüssel s
- öffentlicher Testschlüssel t
- Empfänger kann anderen beweisen, dass Signatur echt ist (bei symmetrischer Signatur hätte er sie auch selbst erzeugen können), wichtig z.B. bei Zahlungssystemen
- JEDER kann Echtheit einer Signatur testen
- **Zertifizierungsinstanz CA**
  - Überprüfung der Identität des Teilnehmers
  - Überprüfung des Antrags des Teilnehmers: A, t_A, s_A(A, t_A)
  - Prüfung der Eindeutigkeit des Schlüsselzertifikat-Namens des Teilnehmers → Konsistenz
  - Digitales Signieren von Namen des Teilnehmers und dessen öffentlichen Schlüssel (mithilfe des geheimen Schlüssels des CA, Beifügung des Testschlüssels des CA),
    Schlüsselzertifikat: A, t_A, s_A(A, t_A), S_R(A, t_A, s_A(A,t_A))
- Probleme
  - Länge des Schlüssels → kann nach bestimmter Zeit zu kurz sein und berechnet werden
    → Signatur wird von elektronischem Notar mit Zeitstempel versehen und signiert (notariell beurkundet)
- Möglichkeiten, auf denen die Geheimhaltung des geheimen Schlüssels trotz Kenntnis des öffentlichen Schlüssels beruht:
  - Faktorisierung von Produktion großer Primzahlen (1000-3000 Bit derzeit)
  - diskreter Logarithmus
- Vertreter:

- RSA
- GMR

c) hybride kryptografische Systeme

- asymmetrisches System zum Schlüsselaustausch → bequemes Schlüsselmanagement
- symmetrisches System zur weiteren Kommunikation → informationstheoretisch sicher, wenn erster Schritt nicht angegriffen wurde, effizienter, effizientere Nutzung des Übertragungskanals

2.2. Sicherheitsgrad

a) informationstheoretisch (absolut) sicher

- a priori Wissen = a posteriori Wissen: p(m|c) = p(m)
- bester Schutz, angestrebt
- symmetrische Systeme (z.B. one time pad)

Bedingungen bei Konzelationssystemen

- Schlüsselgeheimhaltung ist garantiert
- Nachricht und Schlüsseltexte stochastisch unabhängig
- |c| >= |m|, |k| >= |c|
- für jede Nachricht neuer, zufälliger Schlüssel
- $\forall s \in S \;\; \exists const \in \mathbb{N} \;\; \forall x \in X : \; |\{k \in K \mid k(x) = s\}| = const$
- $\forall s \in S \;\; \forall x \in X : \; W(x) = W(x|s)$ (äquivalent)

Bedingungen bei Authentifikationssystemen

- MAC-Fälscher kann ausschließlich raten →
  - $W(MAC) = \frac{1}{|MACs|} = 2^{-\sigma}$ für σ -bit MACs
  - $W(\; MAC' = code(k, y) \mid MAC = code(k, x) \;) \leqslant \epsilon$
    → $|K| \geqslant \frac{1}{\epsilon} \;\; (\forall k \in K) \;\; , \;\; |possible\, MAC's\ for\ y| = \frac{1}{\epsilon}$ → ...
- mögliches System:
  - Head | Tail – Nachricht, 2bit Schlüssel, 1bit MAC
  - Verbesserung: $\sigma - bit$ MAC, $2 \cdot \sigma$ Schlüssel → $\epsilon = 2^{-\sigma}$

b) kryptografisch / komplexitätstheoretisch sicher

- asymmetrische Systeme
- nicht zu umgehen bei digitalen Signatursystemen
- besser, wenn Risiko des Schlüsselschutzes größer ist als Schutz durch informationstheoretisch sicheres System
- leichter zu handhaben als informationstheoretisch sichere Systeme
- fast immer zu brechen mit höchstens exponentiellem Rechenaufwand: Durchprobieren der möglichen Schlüssel bzw. der möglichen MACs

kryptografisch stark

- Schlüsselgenerierung, Ver/Entschlüsseln: polynomial in l

- Brechen: exponentiell in l (schwerer geht nicht)
- Wkt., dass Algorithmus gebrochen wird, wird schnell kleiner, wenn Schlüssel länger wird
- keine worst-case-Komplexität, keine average-case-Komplexität: ueberzeugend viele Schlüssel sollen nicht berechenbar sein

wohluntersucht

- Algorithmus wurde von sehr vielen Menschen untersucht, Richtigkeit im Allgemeinen glaubhaft
- chaotisch: geringe Änderung des Inputs führt zu völlig unterschiedlichen Ergebnissen

Kerckhoffs'sches Prinzip

- nicht die Geheimhaltung des Kryptoalgorithmus, sondern die des Schlüssels soll ausschlaggebend für die Sicherheit sein
- je bekannter & länger bekannt ein Algorithmus ist, um so sicherer ist er (oder er wird gebrochen)

3. Funktionen / Substitutionen und deren Analyse

3.1. Bewertungskriterien

- Grad der **Vollständigkeit** (n Outputbits abhängig von k Inputbits) $\frac{k}{n}$
- **Avalanche** (Änderung eines Inputbits bewirkt Änderung von ungefähr der Hälfte der Outputbits)
  - strikt: jedes Inputbit ändert jedes Outputbit mit 50% Wkt.
- Nichtlinearität (Kein Outputbit hängt linear von einem Inputbit ab)
  - partiell linear: wenigstens ein Outputbit hängt linear von festen Inputbits ab
  - lineare Faktoren: z.B. erreichbar durch Aufstellung von Linearkombinationen zw. Output und Input
- Korrelationsimmunität (Kenntnis über Output bringt keine Informationen über Input und umgekehrt)
- Chaos: jedes Outputbit soll (von Runde zu Runde) möglichst viele Inputbits der nachfolgenden Substitution beeinflussen (wird errreicht durch geschickte Expansion)

3.2. MM-Substitution

- Vertreter: Tauschchiffre (umfasst Verschiebechiffren und Multiplikationschiffren)
- k … Multiplikationsfaktor, ggT(k,n) = 1 (→ sonst gäbe es kein multiplikatives Inverse und man könnte nicht entschlüsseln)
- s … Verschiebefaktor
- v(a) … Abbildung von a
- $t_s^k(a) = v^{-1}(k \cdot v(a) + s) \mod n$
- Buchstabenhäufigkeit: Wertebereich bleibt gleich, Definitionsbereich ändert sich (andere Anordnung gleicher Häufigkeiten)
- n-Grammhäufigkeit: Kurve bleibt gleich, n-Gramme ändern sich
- Kryptoanalyse mit Gleichungssystem z.B. mit den häufigsten beiden Buchstaben

3.3. PM-Substitution

- Vertreter: Vigenere-Chiffre
- fester Schlüssel fester Länge, der mit Text verknüpft wird (Addition mod n)
- Buchstabenhäufigkeit: Verteilung ist flacher, je länger der Schlüssel ist
- n-Grammhäufigkeit: Verteilung ist flacher, je länger der Schlüssel ist

a) Kasiski-Test

- Wiederholungen im Schlüsseltext (gleiche Klartextzeichen mit gleichen Schlüsselzeichen)
- deren Abstände sind Vielfaches des Schlüssels
- aus verschiedenen Wiederholungsabständen kann ggT gebildet werden
  → Schlüssellänge

b) Friedmann-Test

- Koinzidenzindex $I = \sum_{i=0}^{n-1} p_i^2$
- $I_{min} = \frac{1}{n} = \frac{1}{26} = 0.038$ bei $p_i = \frac{1}{n}$ und bei lateinischem Alphabet

- $I_{max} = I(Klartext) = 0.0762$ bei deutscher Sprache
- r = length(key)
- Tabelle Schlüssel k_0, k_1, …, k_(r-1)
  Schlüsseltext c_0, c_1, …, c_(r-1)
  c_r, c_r+1, …, c_(2r-1)
  …
- (Paar aus verschiedenen Spalten: Wkt = I_min)
- Paar aus gleicher Spalte: Wkt I_max
  → bei unterschiedlicher Wahl von r kann man den Koinzidenzindex der Spalten berechnen und mit I_max vergleichen, bei richtiger Wahl sind beide Werte gleich
- schrittweise Bestimmung von k (wahrscheinlich, indem man für jeden Buchstaben des Alphabets beobachtet, ob sich der Koinzidenzindex der Spalte erhöht?)
- unterscheidet PM-Substitution sofort von den anderen beiden Chiffrierungen

c) Transposition

- Vertreter: einfache Matrixtransposition
- Text wird in Blöcke geteilt (Matrixform), die Zeilen werden von oben nach unten gefüllt, im Anschluss Spaltenpermutation
- Buchstabenhäufigkeit: gleiche Verteilung wie die des Klartextes
- n-Grammhäufigkeit: Verteilung ist flacher, je größer n ist (durch Umordnung)
- Kryptoanalyse:
  - Bestimmung der Blockgröße
    - anhand von Vokalhäufigkeiten
    - Auswertung mit eventuell vorkommenden Wörtern
  - Bestimmung der Permutation anhand besonders markanten Eigenschaften der deutschen Sprache
- aktiver Angriff:
  - jede permutierende Blockchiffre der Länge b ist mit einem Aufwand von $2^b - 1$ brechbar durch Verschlüsseln und Entschlüsseln lassen aller möglichen Blöcke gleicher Länge außer dem von Interesse

4. Klassifikation von Chiffren

- n … Länge des Alphabetes

4.1. Blockchiffre

- Eingabe: m Klartextblock fester Länge, k Schlüssel
- Ausgabe: c Chiffretextblock fester Länge
- Zugeordnet im binären Alphabet: DES, RSA
- Expansionsfaktor: $\frac{|k|}{|m|}$
- KP:
  - Verschlüsselung $E: K x M \rightarrow C$ rechtseindeutig (nicht zwingend bijektiv!)
  - Beispiel: Passwörter → nur Chiffre wird gespeichert und verglichen
  - möglichst komplexe Abbildung:
    - Realisierung einer Tabelle → extrem hoher Speicherbedarf
    - Produktchiffre aus Substitutionen und Permutationen

4.2. Stromchiffre

- Ver- / Entschlüsselung von Zeichenketten variabler Länge
- Zugeordnet im binären Alphabet: GMR, Vernam-Chiffre, s2-mod-n-Generator
- **synchron**:
  - Verschlüsselung eines Zeichens hängt von allen vorherigen / seiner Position ab
  - bei nicht-synchroner Übertragung (Fehler in einem Bit) kann komplette Nachricht zerstört werden
- **selbstsynchronisierend**:
  - Verschlüsselung eines Zeichens hängt nur von beschränkter Zahl direkt vorgegebener Zeichen ab
  - → wahlfreier Zugriff bei Entschlüsselung
- **symmetrisch**
  - im Blockchiffre wird nur Verschlüsselungsfunktion angewandt
  - deterministisch
  - beliebige Länge der Eingaben
  - Blockchiffre für mindestens einen Block vorausberechenbar
- **asymmetrisch → asymmetrisch**
  - aus asymmetrischen Blockchiffren wird asymmetrische Stromchiffre
  - indeterministisch möglich
  - Eingabelänge durch Blocklänge bestimmt
  - Blockchiffre nicht vorausberechenbar
- geeignete Initialisierung bzw. bei Authentikation zusätzlich gleiche Initialisierung auf beiden Seiten

4.3. Betriebsarten

Konstruktionen von selbstsynchronisierenden oder synchronen Stromchiffren aus Blockchiffren

- Ergänzung von Bits zur Rückkopplungseinheit r wenig sinnvoll wegen Verkleinerung der möglichen Inhalte des Schieberegisters

a) selbstsynchronisierend

- **ECB** (Elektronisches Codebuch)
  - Aufspaltung des Klartextes in Blöcke
    → mögliche Mustererkennung des Angreifers (wenn kontextunabhängig, deterministisch)
  - Fehlererweiterung: nur innerhalb des betroffenen Blockes
  - asymmetrisch → asymmetrisch
  - parallelisierbar, wahlfreier Zugriff
- **CBC** (cipher block chaining – Blockchiffre mit Blockverkettung)
  - Verschlüsselung: Klartextblock wird mit Verschlüsselung des vorherigen Blockes modular addiert
  - Entschlüsselung: Verschlüsselung des vorherigen Blockes wird von Entschlüsselung des aktuellen Blockes modular subtrahiert
  - asymmetrisch → asymmetrisch
  - indeterministische Blockchiffren: Nutzung der Auswahl von längeren Schlüsseltextblöcken
  - Authenfikation: (deterministische) Verschlüsselung des aktuellen Blockes mit Schlüsseltext des vorherigen Blockes dient als MAC, die der Empfänger durch entsprechende Entschlüsselung testen kann
  - begrenzte Fehlererweiterung: 2 Blöcke
  - Problem: gleiche Schlüsseltextanfänge bei gleichem Klartextanfang
    Lösung: erster Schlüsseltextblock ist zufälliger Schlüssel für ersten Klartextblock
- **CFB** (cipher feedback – Schlüsseltextrückführung)
  - Schieberegister: Rückkopplungseinheit r aus Auswahl von Schlüsseltext des vorherigen Blocks
  - Verschlüsselung des Schieberegisters mit Schlüssel
  - modulare Addition einer Auswahl davon mit Klartextstrom
  - Entschlüsselung mit gleich geführtem Schieberegister und modularer Subtraktion
  - symmetrisch
  - Authentikation: realisierbar, Auswahl problematisch (da nicht nachvollziehbar)
  - Fehererweiterung: nur so lang, wie Fehler im Register ist

b) synchron

- **OFB** (output feedback – Ergebnisrückführung)
  - Beispiel: Pseudo-onetimepad
  - verschlüsselter Inhalt des Schieberegisters / Speichers erzeugt neue Blockverschlüsselung, die wieder ins Schieberegister kommt und zum Klartext modular addiert wird
  - symmetrisch
  - keine Fehlererweiterung durch Verfälschung des Schlüsseltextstromes
  - Fehlererweiterung bis zur Wiederherstellung der Synchronisation durch Verfälschung eines Schlüssels
- **PCBC** (plain cipher block chaining – Blockchiffre mit Blockverkettung über Schlüssel- und Klartext)
  - Klartext wird modular addiert mit Verknüpfung über Funktion h von vorherigem Schlüsseltext- und Klartextblock und anschließend verschlüsselt
  - Klartext wird entschlüsselt und modular subtrahiert mit vorherigem Schlüsseltext- und Klartextblock
  - asymmetrisch → asymmetrisch
  - gleichzeitige Konzelation und Authentikation: Klartext wird erweitert um Prüfblock (z.B. nur Nullen)
    → Einsparung der Hälfte des Aufwands
  - Fehlererweiterung: potentiell unbegrenzt
  - Speicherplatz sparen: nur Speicherung des Ergebnisses von h(Klartext, Schlüsseltext)

- **OCFB** (output cipher feedback - Schlüsseltext- und Ergebnisrückführung)
  - Schieberegister: Auswahl von h(vorheriger Schlüsseltext, vorheriges Ergebnis) wird verschlüsselt → Ergebnis, modulare Addition mit Klartext → Schlüsseltext
  - symmetrisch
  - Fehlererweiterung: potentiell unbegrenzt

c) Kollisionsresistente Hashfunktion aus Blockchiffre

- Hashfunktion: bildet beliebig lange Argumente auf Werte fester Länge ab
- Klartext wird in Blöcke der Länge der Schlüssel geteilt, jeder einzelne verschlüsselt Ergebnis der vorherigen Runde,
  Verschlüsselung wird modular verknüpft mit dem Ergebnis der vorherigen Runde → Ausgabe
- Klartext muss postfixfrei codiert sein oder festen Initialisierungswert haben!
  Behebung des Problems: Nachrichtenlänge als Block ans Ende der Nachricht hängen

5. Kryptografische Systeme

5.1. Feistel-Chiffre (Beispiel DES)

- Expansionsfaktor 1 → längentreu, |m|, |k| gerade
- gleiche Funktion für Ver- und Entschlüsselung
- DES = Data Encryption Standard
- symmetrisches Konzelationssystem
- k = Schlüssel, |k| = 64 Bit, aber nur 56 werden genutzt, m = Klartext-Block, |m| = 64 Bit
- Schlüsselerzeugung von 16 Schlüsseln k_i aus k:
  - PC - 1 (permutierte Auswahlfunktion: aus 64-Bit-Schlüssel wähle 56 Bit)
  - Teilung in $C_{0,} D_0$
  - 16 x Iteration:
    - zyklische Linksrotation um 1 (C) bzw. um 2 (D)
    - Teilschlüsselausgabe C konkateniert mit D = PC – 2 → k mit 48 Bit

a) Verschlüsselung

- Eingangspermutation IP : Eingabeblock → Teilung in L_0 und R_0
- 16x Iterationsrunde:
  - Eingabe: L_{i-1}, R_{i-1}
  - Ausgabe: $L_i = R_{i-1}$, $R_i = L_{i-1}\ XOR\ f(R_{i-1}, k_i)$
  - Verschlüsselungsfunktion f:
    - Expansion von R_{i-1} (32 Bit) auf 48 Bit
    - XOR-Verknüpfung mit k_i (48 Bit)
    - je 6 Bit in eine Substitutionsbox (ingesamt 8 Boxen), Zuordnung eines 4-Bit-Wertes → 6*4 = 32 Bit Block
    - Mischung durch Permutation P auf 32-Bit-Block → Ausgabe (32-Bit-Block)
      ($8 \cdot 2^6 \cdot 4\,Bit$ statt $1 \cdot 2^{48} \cdot 32\,Bit$)
- Tauschen von L_{16} und R_{16}
- Ausgangspermutiation IP^{-1}

b) Entschlüsselung

– Durch XOR-Verknüpfung kann Iterationsrunde umgedreht werden, indem Links- und Rechts-Blöcke umgetauscht werden und die gleiche Iteration angewandt wird, von mit umgekehrter Reihenfolge der Schlüssel

c) Komplementaritätseigenschaft

– $DES(\bar{k}, \bar{x}) = \overline{DES(k, x)}$
Begründung: Permutation / Kürzung / Teilung einer Bitkette unabhängig von Werten der Bits, vor Substitution wird Komplement aufgehoben (da $\bar{k}\ XOR \bar{x} = k\ XOR\ x$) und nach der Substitution durch Addition mit L_{i-1} wiederhergestellt

d) Sicherheitsgrad

– Aktiver Angriff:

  – maximale Sicherheit: Brechen nur mittels $2^{|m|}$ Klartext/Schlüsseltext-Blöcken möglich

  – geg.: $x, DES(k, x)$. Senden zum Entschlüsseln: $\bar{x}$ → $DES(k, \bar{x}) = \overline{DES(\bar{k}, x)}$
  Gewinnung zwei komplementärer Schlüsseltexte mit zugehörigem Klartext halbiert Suche im Schlüsselraum

– brute force (hat schon Erfolg)

– Problem der schwachen Schlüssel (symmetrische Folge der Teilschlüssel)

– mittlerweile kann Sicherheit ohne speichertechnische Probleme erreicht werden, indem Teilschlüssel einzeln zufällig unabhängig gewählt werden → |k| = 16*48 Bit

5.2. Vernam-Chiffre

– zu jedem Klartext gibt es (min.) einen Schlüssel, so dass k(x) = S

– $\forall k \in K, s \in S, x \in X$: |K| >= |S| >= |X|

– |k| = |s| =|x|

– $s_i = x_i + k_i$, $x_i = s_i - k_i$

– bitweise XOR-Verknüpfung

– jeder Schlüsseltext sollte allen möglichen Klartexten entsprechen können

a) aktiver Angriff

– irgendeinen Schlüsseltext an Empfänger schicken und aus Entschlüsselung aktuelles k berechnen

– nächsten Schlüsseltext des eigentlichen Senders abfangen und mit k entschlüsseln

5.3. Authentikationscodes

– informationstheoretisch sicheres, symmetrisches Authentikationssystem

– sicher heißt hier: Wkt(Angreifer rät richtig) <= $\epsilon = (\frac{1}{2})^{|MACs\ pro\ Zeichen|}$

– |k| >= |Alphabet| * |MACs pro Zeichen| * |Zeichen in Nachricht|

5.4. RSA als Konzelationssystem

– p, q, n, $c \in \mathbb{Z}^*_{\phi(n)}, c \neq 1$, $d \equiv c^{-1} \pmod{\phi(n)}$ (dafür wird p und q - nur bei Schlüsselgenerierung - benötigt)

a) naiv (deterministisch)

- c, n öffentlich, d geheim
- Verschlüsselung: $m^c \pmod n$
- Entschlüsselung: $(m^c)^d \pmod n$ → funktioniert aufgrund des Satzes von Fermat)
- Exponentation injektiv und surjektiv → Permutation
- Unterteilung des Klartextes in Blöcke mit $l \leqslant |n| - 1\ Bits$
- passive und aktive **Angriffe bei Betriebsart ECB**:
  - bei zu kurzer Blockgröße: mögliche Klartexte bei bekanntem Alphabet verschlüsseln, Vergleich der Schlüsseltexte, hilfreich sind oft benutzte Klartexte (Anreden, Währungen)
  - Jakobi-Symbol von Schlüsseltext und Klartext sind gleich:
    $$\left(\frac{m^c (mod\ n)}{n}\right) = \left(\frac{m^c}{n}\right) = \left(\frac{m}{n}\right)^c = \left(\frac{m}{n}\right)$$ (da c ungerade)
- **aktiver (nicht adaptiver) Angriff** → selektives Brechen:
  - Angriffspunkt: multiplikative Struktur des RSA (Angriff nach DENNING):
    frei wählbarer Schlüsseltext S_3 (der entschlüsselt werden soll),
    $$r \in \mathbb{Z}_n^* : \quad S_2 := S_3 \cdot r^c \rightarrow S_2{}^d = (m_3 \cdot r^c)^d = S_3{}^d \cdot r^{c \cdot d} = S_3{}^d \cdot r$$
    $$\rightarrow \quad m_3 = S_3{}^d \cdot r^{-1}$$

b) Verbesserung (indeterministisch)

- kollisionsresistente Hashfunktion $f_m(x) = y$, kurz $h(m)$, neutralisiert multiplikative Struktur von RSA
- (Zufallszahl z)
- Verschlüsselung: $(z, m, h(z, m))^c$
- Entschlüsselung: $((z, m, y)^c)^d = z, m, y$, Test ob $y = h(z, m)$

c) Implementierung

- c möglichst kurz
  - Problem: bei c = 3 Johan Håstad Angriff: bei gleichem m, gleichem c und gleicher Zufallszahl, die c-mal mit verschiedenen n's verschlüsselt werden, lässt sich m mit Hilfe des CRA berechnen
  - beste Wahl: $c = 2^{16} + 1$
- d_p, d_q berechnen und Schlüsseltext mod p, q rechnen, Ergebnis per CRA verknüpfen

d) Sicherheitsgrad

- nicht schwerer als modulare Faktorisierung, Gleichheit der Schwierigkeit nicht bewiesen

5.5. RSA als Authentikationssystem

a) naiv

- wie bei asymmetrischem Konzelationssystem, c → t (testen), d → s (signieren)
- passiver Angriff → existentielles Brechen:
  Signatur mit t exponenzieren → passender Klartext mit dazugehöriger Signatur
- aktiver (nicht adaptiver) Angriff → selektives Brechen:
  - Angriffspunkt: multiplikative Struktur des RSA:

frei wählbare Nachricht m_3

$m_3 := m_1 \cdot m_2 \pmod n \rightarrow {m_3}^s := {m_1}^s \cdot {m_2}^s \pmod n$ oder

$r \in \mathbb{Z}_n^* : \quad m_2 := m_3 \cdot r^t \rightarrow {m_2}^s = (m_3 \cdot r^t)^s = {m_3}^s \cdot r^{t \cdot s} = {m_3}^s \cdot r$

b) Verbesserung

- kollisionsresistente Hashfunktion
- Signatur: $m, (h(m))^s$
- Test: $(y^s)^t$, Test ob $y = h(m)$
- möglicherweise schneller, weil Länge der Signatur egal ist

c) Sicherheitsgrad

- effizienter programmierbar als GMR, allerdings ist GMR bewiesen kryptografisch stark

5.6. GMR

- Digitales Signatursystem
- Geheim: p, q prim, $p \equiv 3 \pmod 8, q \equiv 7 \pmod 8$
- Öffentlich: n (=p*q), R (authentisierte Einweg-Referenz)
- kollisionsresistente Permutationenpaare mit Geheimnis (Kollision kann kryptographisch nicht gefunden werden, wenn man Geheimnis nicht kennt):

  Definitionsbereich $D_n := \left\{ x \in \mathbb{Z}_n^* \mid \left(\frac{x}{n}\right) = 1, \quad x < n/2 \right\}$,

  $$f_0 = \begin{cases} x^2 \pmod n & falls\ (x^2 \bmod n) < (n/2) \\ -x^2 \pmod n & sonst \end{cases}$$

  $$f_1 = \begin{cases} 4 \cdot x^2 \pmod n & falls\ (4 \cdot x^2 \bmod n) < (n/2) \\ -4 \cdot x^2 \pmod n & sonst \end{cases}$$

- Signatur $s(0) := f_0^{-1}(R)$, $s(1) := f_1^{-1}(R)$, verschachtelt für $m = b_0 b_1 \ldots b_m$:

  $s(m) = f_{b_0 b_1 \ldots b_m}^{-1}(R) = f_{b_0}^{-1}(f_{b_1}^{-1}(\ldots f_{b_m}^{-1}(R)\ldots))$ (**DAMGARD-Hashfunktion**)

  - für $f^{-1}(x)$ gilt:

    1. Test, ob $x \in QR_n$, wenn nicht, setzte x = -x (mod n)
    2. Wurzeln in p und q berechnen, danach mit 2^{-1} multiplizieren, wenn f_1
    3. Ergebnisse $y_p \in QR_p$, $y_q \in QR_q$ per CRA verknüpfen → $y \in QR_n$
    4. wenn y>n/2, y = -y (mod n) → $y \in D_n$

- Test: $R = f_{b_0 b_1 \ldots b_m}(s(m)) = f_{b_m}(\ldots f_{b_1}(f_{b_0}(s(m)))\ldots)$
- Präfixfreie Abbildung: Angabe der Nachrichtenlänge zu Beginn der Nachricht (sonst könnte ein passiver Angreifer einfach Bits am Ende löschen → Verfälschung der Nachricht)
- Authentisierung vieler Referenzen:
  - öffentliche, verbrauchbare Referenzenliste
  - Referenzenbaum, öffentliche Wurzelreferenz, jeder Knoten authentisiert 2 Nachfolger und eine Nachricht, rekursiv von Blatt zu Wurzel berechenbar zum Testen der Signatur

5.7. Weitere Systeme

a) Diffie-Hellman Schlüsselaustausch

- Kryptografisch stark (kryptografische Schwierigkeit der Berechnung von diskreten Logarithmen)
- öffentlich: Primzahl p, Generator g
- jeder Teilnehmer wählt geheim: $x \in \mathbb{Z}_p^*$ → Veröffentlichung von $g^x \pmod p$
- gemeinsamer geheimer Schlüssel von zwei Teilnehmern mit eigenen geheimen Schlüsseln x und y:
  $k = g^{x \cdot y} = (g^x)^y = (g^y)^x \pmod p$
- Anonymität z.B. über Synonyme
- Angriffe:
  - Man-in-the-Middle-Angriff: Angreifer gibt sich bei beiden als der jeweils andere Kommunikationspartner aus → Abhörung und Manipulation
    Lösung: Authentikation
  - Berechnung des diskreten Logarithmus mit BabyStep-GiantStep-Verfahren:
    - $t = \sqrt{p-1}$ (abgrerundet) → 0 <= {u, v} <= t
    - $y \equiv g^x \equiv g^{u \cdot t + v} \pmod p \Leftrightarrow y \cdot g^{-v} \equiv g^{u \cdot t} \mod p$
    - Babystep-Liste → v-Seite, Giantstep-Liste → u-Seite, Listen werden verglichen
- als digitales Signatursystem implementierbar: El-Gamal-Signatur → DSS (Digital Signature Standard)

b) unbedingte Sicherheit für den Unterzeichner

- bei digitalen Signatursystemen ist bei erfolgreichem Angriff immer der Signierer das arme Schwein, weil gerichtlich beweisbar ist, dass er die Nachricht unterzeichnet hat (da aber nur komplexitätstheoretische Sicherheit bei Asymmetrie existiert, bleibt ein Restrisiko)
- Lösung:
  - es gibt sehr viele geheime Signaturschlüssel, Signierer kennt nur einen, kann damit zeigen, dass eine andere, aber richtige Signatur nicht von ihm sein kann, da er den Schlüssel dazu nicht kennt
  - → nur kryptografische Sicherheit für den Empfänger
  - → Fail-Stop-Signatursystem (Beweis für gebrochenes Signatursystem)

c) Undeniable Signatures

- Nur der Ersteller einer Signatur kann beweisen, dass die Signatur echt ist (Testprotokoll verlangt Interaktion mit Ersteller der Signatur)
- der Ersteller erfährt so, wer seine Signatur nutzen / prüfen will
- Ersteller kann nicht ablehnen, Signatur zu prüfen, sonst wird Signatur als gültig anerkannt

d) Blind Signatures

- Empfänger der Signatur blendet zu signierenden Text mit Zufallszahl und sendet diese an Signierer
- Signierer signiert blind
- Empfänger kann entblenden und erhält korrekte Signatur zu seinem Klartext

e) Schwellwertschema

- Schlüsseltexte werden in Teile zerlegt, bis zu einem bestimmten Schwellwert der Anzahl der abgefangenen Teile kann der Angreifer daraus keine Informationen schöpfen
- Polynom k-1'ten Grades: nur eindeutig lösbar (q(0) berechenbar), wenn k Punkte (k, p(k)) bekannt sind (k = 1, … ,k)

f) Zero-Knowledge-Verfahren

- Beweiser – mit Geheimnis
- Verifizierer – überzeugt sich von Kenntnis des Beweisers über das Geheimnis, erfährt aber selbst nichts davon
- Commitment-Verfahren:
  - Festlegungsphase
    - Sender legt Wert m fest
    - Sendung des Ergebnisses einer kollisionsresistentren Hashfuntion h(m) an Empfänger
  - Öffnungsphase: Sender teilt Empfänger geheimes m mit
  - Geheimhaltung: durch fehlende effiziente Umkehrfunktion der Hashfunktion gegeben
  - Eindeutigkeit: durch komplexitätstheoretisch schwer zu berechnende Kollision gegeben
- interaktive Beweissysteme
  - vollständig (ehrliche Beweiser werden vom Verifizierer akzeptiert)
  - korrekt (unehrliche Beweiser werden vom Verifizierer nicht akzeptiert)
  - Zero-Knowledge-Eigenschaft: im Protokoll wird kein Wissen übertragen
    Beweisskizze: Verifizierer hätte komplette Konversation auch selbst erzeugen können
    - Simulator, der in polynomialer Zeit ein Kommunikationsprotokoll täuschungsecht erzeugen kann
      - perfekt (informationstheoretisch nicht unterscheidbar)
      - statistisch (Unterschied zwischen Wkt.verteilungen ist verschwindend gering)
      - berechenbar (komplexitätstheoretisch nicht unterscheidbar)
  - P→V: Commitment a » V→P: Challenge c » P→V: Response z
  - Beispiele:
    - Beweissystem für Graphisomorphismus
    - Beweissystem für quadratische Reste (Fiat Shamir Protokoll)
    - Beweissystem für den diskreten Logarithmus:

      $x = g^w \pmod p$, g *Generator von* G_q , q | (p-1)

      x öffentlich, w geheim

      P→V: a, $a = g^s$ (s zufällig in $\mathbb{Z}_q$)

      V→P: c, $c \in \{0,1\}$

      P→V: z, $z = s + w^c$

      Test V: $g^z = a \cdot x^c$?

      Simulator rät c, setzt: $c = 0 \rightarrow a = g^s$, $c = 1 \rightarrow a = g^s \cdot x^{-1}$
    - nicht interaktives Beweissystem (Fiat Shamir Heuristik)

      Beweiser ermittelt c selbst → Ergebnisbitfolge bekannter Hashfunktion mit $a_1, \ldots, a_k$

      Veröffentlichung von $a_1, \ldots, a_k$ und $z_1, \ldots, z_k$

      → Verwendung als digitales Signatursystem möglich: Nachricht wird mitgehasht
  - parallele Komposition: Verifizierer stellt Beweiser mehrere Challenges auf einmal
    → Zero-Knowledge-Eigenschaft nicht gegeben bei unehrlichem Verifizierer
- Angriff: Man-in-the-Middle.
  - Lösung: OR-Protokoll: Beweiser kann zeigen, dass er von zwei öffentlichen Werten ein zugehöriges Geheimnis kennt, es wird nicht gezeigt, welches

6. Steganografie

- Konzelation: Verbergen der Existenz einer Nachricht
  → Vertraulichkeit der Nachricht, Ausnutzung von Toleranzbereichen
  → Cover kann auch synthetisch erzeugt werden
- Authentikation: Wasserzeichen, Fingerabdruck (Urheberrecht)
  → muss robust gegenüber Veränderungen des Bildes sein
  → soll Bildinhalt nicht stören
- Hülle (Cover), Stegotext
- Einbetten und Extrahieren
- symmetrisch mit Stegoschlüssel k
- hybrid mit öffentlichen Schlüsseln:
  - asymmetrischer Diffie-Hellman-Schlüsselaustausch → Generierung symmetrischen Schlüssels
- Bild b Matrix mit M Spalten und N Zeilen (x,y) = (Spalte, Zeile)

6.1. EInbettung

- LSB-Methode
  - Codierung der Nachricht im least significant bit
  - schlüsselgesteuert: Schlüssel gibt Abstände zwischen mit Nachrichtenbits behafteten Pixeln an
- Paritäts-Methode
  - n Pixel werden zusammengefasst, deren Parität gibt Nachrichtenbit an
  - muss die Parität des Blocks beim Einbetten geändert werden, ist beliebig, welches Pixel im Block geändert wird
- Inkrementieren
  - ähnlich wie LSB, nur wird Grauwert immer erhöht
  - speziell zu behandeln: Wert 255 im Coverbild

6.2. Effizienz

- Einbettungskapazität (benutzbare Pixel)
- Einbettungsrate (tatsächlich benutzte Pixel) <= Einbettungskapazität
- Schwundrate (z.B. bei Inkrementieren, wenn höchster Grauwert nicht speziell behandelt wird)
- Änderungsrate

6.3. Angriffe

- Wahrnehmen der Modifikation
- natürliches Rauschen hat Strukturen
- statistische Tests (automatisierbar)
- Vergleich mit empirisch ermittelten Schwellwerten (automatisierbar)
- absolutes Differenzbild, binäres (invertiertes) Differenzbild (weiß = gleich, schwarz = verschieden)
- LSB: Angleichung der Häufigkeiten der Grauwerte (ungerade wird kleiner, gerade größer) ??
  → Chitest – berechnet Verteilung der Grauwerte und bestimmt, bis zu welchem Pixel eine Nachricht eingebettet ist
- ABER jeder Angriff führt zur Verbesserung des Stegosystems

7. Kommunikationsnetze

7.1. Einsatz von Verschlüsselungen

a) Verbindungs-Verschlüsselung

- Verschlüsselung zwischen benachbarten Netzknoten
- gleichmäßiger Zeichenstrom → sichere, selbstsynchronisierende Stromchiffre → perfekt unbeobachtbar
- Nachteile:
  - Klartext in Vermittlungszentralen
  - zueinander passende kryptografische Systeme und Verschlüsselungsaufwand (Sparmöglichkeit: HDTV: nur Verschlüsselung der Interessensdaten, nicht des Inhalts)

b) Ende-zu-Ende-Verschlüsselung

- verschlüsselte Übertragung zwischen Teilnehmerstationen
- Nachteile:
  - nur Schutz der Nutzdaten, nicht der Vermittlungs- bzw. Verkehrsdaten (bei vorheriger Kenntnis der Nutzdaten erhält man Interessensdaten)
  - Umsetzung eines offenen, asymmetrischen Systems mit EzE-Verschlüsselung → formelle Normung
  - ohne Normung Vertraulichkeit nur bei geschlossenen, in sich passenden Systemen erreichbar
  - Verschlüsselungsaufwand

7.2. Schutzmaßnahmen

a) technische Sicht

- Schutz der Verkehrsdaten vor Vermittlungszentralen, Schutz der Identität vor Kommunikationspartner
- Kombination: Schutz der Inhaltsdaten durch Ende-zu-Ende-Verschlüsselung, Verbindungsverschlüsselung zwischen Netzabschlüssen und Verbindungszentralen

b) Verteilung

→ perfekte informationstheoretische Anonymität, Unbeobachtbarkeit und Unverkettbarkeit für Empfänger

- verändernder Angriff
  - Diensterbringung verhindern
  - Verteilung stören und durch Reaktion der Teilnehmer diese deanonymisieren
  - → analoge Maßnahmen (nichtzellulare Funk- und Satellitennetze)
    → digitale Maßnahmen (Erkennung inkonsistenter Verteilung)

Adressierung

- Öffentliche Adressen | Private Adressen (Telefonbuch | in vorheriger Nachricht mitgeschickte Adr.)
- implizite | explizite Adressierung (Adresse lässt (keine) Rückschlüsse auf Adressat bzw. dessen Aufenthaltsort zu)
- implizierte, verdeckte Adressierung
  - Redundanz innerhalb des Nachrichteninhalts
  - asymmetrisches Konzelationssystem
    - Nachricht wird mit öffentlichem Schlüssel des Empfängers verschlüsselt, jeder entschlüsselt

Nachricht mit eigenen Dechiffrierschlüsseln → bei richtiger Redundanz ist er der Empfänger

    - wenige / ein Dechiffrierschlüssel nötig
    - höherer Implementierungsaufwand
  - symmetrisches Konzelationssystem
    - niedriger Implementierungsaufwand, wenn in Nachricht gegeben ist, wie sie verschüsselt wurde
    - viele Schlüssel nötig, alle müssen bei jeder Nachricht probiert werden
  - symmetrisches Authentikationssystem
    - Nachricht wird mit MAC versendet, Empfänger prüft, ob der MAC mit seinem Schlüssel zur Nachricht passt
  - aufwendig, resistenter gegen Übertragungsfehler, nötig für Kontaktaufnahme
- implizite, offene Adressierung
  - Empfänger wählt immer neue Zahl als (optimal private) Adresse (~Stromchiffre), Nachricht hat Adressfeld → Vergleich mittels Assoziativspeicher
  - Verkettungen leicht zu finden → leichte Adresserkennung
  - effizient, aber leicht zu stören

Fehlertoleranz

- bei fehlerhaften Nutzdaten muss Empfänger auf erneute Sendung bestehen, dazu müssen Fehler erkannt und fehlerfreie Versionen beschafft werden können und der Empfänger in der Lage sein, mit der Sendestation zu kommunizieren
- implizite Adressierung: vorangehende Informationseinheiten dürfen Adressierung nicht verfälschen → Blockchiffren. Bei immer neuer Adressangabe in Nachricht ist neue Synchronisation nötig

c) Abfragen und Überlagern

- Schutz des Empfängers
- Nachrichtenservice (jede Nachricht kann angefordert werden)
- m Server mit je n Speicherzellen für n Nachrichten (gleiche Nachrichten, gleiche Reihenfolge)
- verschlüsselte Kommunikation:
  - Anfrage des Teilnehmers an s Server: 1 < s <= m,
    Senden von s-1 Abfragevektoren mit je n zufälligen Bits, s'te Vektor = XOR aller s-1 – Vektoren
  - Antwort: XOR der betreffenden Nachrichten (je nach gesetztem Bit im Anfragevektor mit gekipptem Bit bei gewünschter Nachricht)
- auch wenn s-1 Server kooperieren, erfahren sie nicht die den Teilnehmer interessierende Nachricht
- Empfänger muss benachrichtigt werden, wenn Nachricht vorliegt

d) dummy traffic

- (schwacher) Schutz des Senders und Empfängers
- bedeutungslose Nachrichten (Sendung z.B. an nicht existierende Adressen)
- nur für Unbeteiligte unbeobachtbar und unverkettbar (kein anonymes Senden von bedeutungsvollen Nachrichten aus Sicht des Empfängers)
- hoher Aufwand, statische Nachrichtenlänge

e) digitale Signalregenerierung

- Schutz des Senders
- empfangene Signale werden vor dem Weiterleiten so regeneriert, dass sie sich nicht unterscheiden lassen von von der Station generierten Signalen → aus Signalform kann keine Information über Sender gewonnen werden. Regenerierte Signale verschiedener Stationen sind aber unterscheidbar

7.3. Physische Netzarten

- Voraussetzung: Abhören der Leitungen gleicher Aufwand für Angreifer wie Beobachten der Stationen
- beobachtete Nachrichten an Stationen sollen von möglichst vielen nicht kontrollierten Teilnehmerstationen kommen und zu möglichst vielen möglichen unkontrollierten Teilnehmerstationen gehen, Angreifer ist nicht Sender oder Empfänger

a) RING-Netz

- jede Station empfängt jede Nachricht mindestens einmal
- Nachricht durchläuft Ring, bis sie wieder beim Sender ist, der überschreibt sie
- Verbindungsverschlüsselung, digitale Signalregenerierung
- Umsetzung von Zeitstempeln: Empfänger ersetzt Nachricht mit Antwort für Sender
- effizientes Ringzugriffsverfahren (verteiltes und anonymes Abfragen)
  - n-anonym: Angreifer kreist n (nicht kontrollierte) Stationen ein
    → keine Identifizierung von Sender- oder Empfängerstation in n möglich bei n > 1
- Fehlertoleranz:
  - transiente Fehler
    Neustart des Zugriffsverfahrens
    Nachricht umkreist RING mehrfach
  - permanente Fehler → Ringrekonfigurierung
    - By-Pass-Einrichtung: Ring kann bei erkanntem (!) Fehler innerhalb (!) der Station geschlossen werden. Angreifer merkt, wenn Nachbar By-Pass aktiviert (wegen Eigenschaft digitaler Signalregeneration)
    - Ring-Verkabelungs-Konzetratoren: ideales Ziel für Angreifer
    - mehrere parallele Ringe
    - geflochtener Ring: Ausfall von..
      ..Station: äußerer Ring nutzt Verbindung des inneren zwischen den Nachbarn
      ..innerer Leitung: ausschließliche Nutzung des äußeren Rings
      ..äußere Leitung: äußere Leitung nutzt zwei Verbindungen der inneren und teilt Datenstrom → kann Anonymität verletzen
      Stationen signalisieren, wenn sie auf einem Kanal nichts mehr empfangen → alternative Leitungen werden benutzt, bis fehlerhafte Station signalisiert, dass sie wieder funktioniert
    - asymmetrische Protokolle: nicht anonyme Stationen zur gezielten Fehlertoleranz

b) BAUM-Netz

- innere Knoten: Kollisionen verhindernde Schalter
- Blätter: Teilnehmerstationen
- Schaltung nach oben: zufällig, nur wenn Leitung frei ist
- Informationsstrom von der Wurzel wird an alle Blätter weitergeleitet
- geringere Senderanonymität als bei RING (nur eine Leitung zu beobachten)
- hervorragendes Leistungsverhalten
- BAUM-Netz mit überlagerndem Senden: Knoten als XOR-Verknüpfer
  (min. genauso effektiv wie mit Schaltern)

7.4. DC-Netz

a) Funktionsweise

- **verallgemeinertes überlagerndes Senden**
  - Schlüsselgraph: verschlüsselte Sendung von Schlüsseln an andere Stationen, minimal: zyklenfrei

- lokale Überlagerung:
  Addition aller selbst erzeugten Schlüsselzeichen
  Subtraktion aller erhaltenen Schlüsselzeichen,
  Addition des zu Senden gewünschten Nutzzeichens (Ende-zu-Ende-Verschlüsselung)
- globale Überlagerung: Summe aller lokalen Überlagerungen
- Verteilung der globalen Überlagerung
- binäres überlagerndes Senden → egal, ob Addition oder Subtraktion
- Überlagerungs-Kollisionen: wohldefinierte Summe des gleichzeitig Gesendeten
  → Lösung: Zugriffsverfahren, Übertragungsrahmen, in denen immer nur einer sendet

- **überlagerndes Empfangen**
  - global: Überlagerung von n Nachrichten → einzelne Sendung von n-1 Nachrichten
  - paarweise: exklusives Senderecht, Berechnung der anderen Nachricht durch eigene
    → doppelt so gut genutzte Bandbreite, geeignet für anonymen Schlüsselaustausch
- **Kollisionsauflösung**
  - Mittelwertvergleich:
    - s Stationen, max. m gleichzeitige Nachrichten pro Station von max. Größe G, Addition mod g
      → $g > s \cdot m \cdot G$
    - Mittelwert berechnet sich abgerundet aus globaler Überlagerung (garantiert kleiner g) und Summe der sendenden Stationen (letztes Bit in Nachricht gesetzt) → wessen Nachricht darunter liegt, der sendet erneut (bei gleichen Nachrichten senden alle oder keiner) → Baumstruktur
    - echter Determinismus: garantierte obere Sendezeitschranke, realer Durchsatz von nahezu 100%
    - Möglichkeiten zur Verwendung von kleinerem g existieren, büßen aber Determinismus ein
    - kombiniert mit überlagerndem Empfangen optimales Mehrfachzugriffsverfahren im DC-Netz
- **Reservierungsschema**
  - Bitleiste: feste Anzahl von Bits pro Informationseinheit, wer senden will, setzt 1
  - Vorsicht, wenn mehr Reservierungen gesetzt werden als Speicherplatz für Reservierung groß ist

b) Sicherheit

- Überlagerndes Senden → perfekte informationstheoretische Anonymität und Unverkettbarkeit des Senders gegen passive beobachtende und verändernde Angriffe durch Garantie von
  - Gleichbehandlung aller Stationen → Anonymität
  - Erlaubnis für jede Station, die gesamte Bandbreite zu nutzen → Unverkettbarkeit
- Überlagerndes Empfangen → perfekte informationstheoretische Konzelation
  → durch entsprechende Algorithmen ist perfekte Anonymität und und Integrität möglich

c) Optimierung

- Ziele:
  - geringe Verzögerungszeit = Zeit zwischen Sendung und Rückmeldung (Kollisionserkennung)
  - ausreichend große Alphabetgröße
  - zur Verfügung stehende hohe Bitrate
  - Realisierung mit geringem Aufwand
- Verwendung von Pseudozufallsbitfolgengeneratoren zur Reduzierung der auszutauschenden Schlüssellänge
  - nur noch komplexitätstheoretische Sicherheit
  - gute Generatoren sind langsam → Kombination mehrerer Generatoren zur schnelleren Generierung der Folgen
  - wenn Stationen unterschiedlich starke Generatoren nutzen, ist ggf. Anonymität verletzt

- Implementierung als RING-Netz: Nachrichten kreisen zuerst zum Senden, dann zum Empfangen um den Ring → Angreifer kann nicht einzelne gesendete Schlüsselnachrichten abfangen, deswegen können auch schwächere Zufallsbitfolgengeneratoren genutzt werden, 4-facher Übertragungsaufwand, Verzögerungszeit proportional zu |Stationen|
- Implementierung als BAUM- Netz (dezentral) oder STERN-Netz (zentral): n-facher Übertragungsaufwand, aber kürzere Verzögerungszeit (proportional zu log(|Stationen|))

d) Fehlertoleranz

- mehrere unabhängige DC-Netze (statisch erzeugte Parallel-Redundanz, (statisch oder) dynamisch aktivierbar)
- mehrere senderpartitionierte DC-Netze (Stationen haben nicht in jedem Netz Schreibrecht)
  → geringere Senderanonymität
  → vollständige Fehlerüberdeckung
- Fehlererkennung und -lokalisierung
  - **Knack-Schnapp-Verteilung**
    - Station i sendet empfangene globale Überlagerung signiert an alle anderen Stationen j | j>i
    - Unterschied zwischen eigener globaler Überlagerung und den empfangenen, eine Station sendet nicht → Nachrichtenüberlagerung gestoppt → Wahl von zufälligem, gleichverteiltem Ausgabezeichen
    - Sicherheit der Authentikation bestimmt Sicherheit des resultierenden Schemas
    - Deterministische Knack-Schnapp-Schlüsselgenerierung
      - vollständige (nicht ausschließliche) Abhängigkeit zwischen auszutauschenden Schlüsseln zwischen den Stationen und vorherigen globalen Überlagerungsergebnissen
        (z.B.: zwei vertraulich und authentisch ausgetauschte Bitfolgen:
        Multiplikation der einen mit jeweiligem globalen Ü-Ergebnis der früheren Runde, Addition mit der anderen Folge)
      - zyklische Verwendung von Zufallsfolgen möglich, wenn in festen Abständen von jeder Station ein Zeichen gesendet wird und das globale Ü-Ergebnis mit dem Zeichen übereinstimmt
      - Inkonsistenz von globalen Ü-Ergebnissen → ungleiche Schlüssel → Übertragungsstop
      - effizienter (kein Autausch und keine Testung von signierten globalen Überlagerungen)
    - probabilistisch:
      - Störung nicht zu 100% und nicht bis in alle Ewigkeit → Effizienz
  - allen bekanntes **lineares Prüfzeichen** am Ende der verschlüsselten Informationseinheit (durch Linearität bleibt es auch bei Überlagerung erhalten)
  - **Fehlerlokalisierungs und -behebungs – Protokoll**:
    - A-Modus (Anonymität ist gewahrt) | F-Modus (Fehler wird lokalisiert)
    - mögliche Fehlerstellen:
      - lokal innerhalb einer Station (PZG, …)
      - Synchronisation der Schlüssel zwischen mindestens 2 Stationen nicht mehr gegeben
      - Kommunikationsnetz oder globale Überlagerung fehlerhaft
    - Erstellung eines Rücksetzpunktes
    - Selbstkontrolle der Station (PZG etc.)
      bei Fehler: Auskopplung aus System (Nachricht an andere Stationen, die zur Station gehörige Schlüssel verwerfen und wieder in A-Modus wechseln) und Fehlerbehebung
    - Anzahl der ausgetauschten Schlüsselpaare halbiert und eine Hälfte testet mit 100 neuen, einmalig zu verwendenden Schlüsseln
    - bei verbleibenden zwei Stationen wird Synchronisation getestet, dann neu initialisiert

7.5. MIX-Netz

a) Funktionsweise

- unabhängige Zwischenstationen, die Schübe von Nachrichten gleicher Länge puffern, umcodieren und umsortieren
- Mixreihenfolge: vom Teilnehmer selbst bestimmt | Mixkaskade
- Schlüsselverteilung
  - Sender spezifiziert die entschlüsselnde Umcodierung
  - Empfänger spezifiziert die verschlüsselnde Umcodierung
  - mittlerer MIX darf aus Schlüsseln, die er erhält, nicht auf Sender oder Empfänger schließen können
    - paarweise überlagerndes Empfangen → informationstheoretisch sicher, aber mit zu hohem Aufwand verbunden → asymmetrische Konzelationssysteme
- **Pufferung**
  - Batchmodus | Poolmodus (nicht deterministisch!)
  - keine spezielle Umcodierung darf mit gleichen Schlüsseln mehrfach ausgeführt werden
    → betrifft Nachrichteneinheit (Senderanonymität) | Rückadresseneinheit (Empfängeranonymität)
    → Reduzierung von zu speichernder Vergangenheit:
    - häufigeres Wechseln der öffentlichen Schlüssel des Mixes
    - Einsatz von Zeitstempeln (schlecht geeignet für anonyme Rückadressen)
    - Verkürzung der Nachricht mittels Hashfunktion oder fester Auswahl
- **Umcodierung**
  - **direktes Umcodierungsschema zum Schutz der Senderanonymität**

$MIX_1, \ldots, MIX_n$	Mixe, von denen die Nachricht weitergegeben werden soll
$A_1, \ldots, A_n$	Folge der Adressen der Mixe
$c_1, \ldots, c_n$	Folge der öffentlich bekannten Chiffrierschlüssel der Mixe
$MIX_{n+1}, A_{n+1}, c_{n+1}$	Empfänger, seine Adresse, sein Chiffrierschlüssel
$z_1, \ldots, z_n$	zufällige Bitketten

$$N_{n+1} = c_{n+1}(N) \quad , \quad N_i = c_i(z_i, A_{i+1}, N_{i+1})$$

  - Sender sendet N_1 an MIX_1
  - Zufallsbitfolge wird im Mix mit entschlüsselt, aber nicht ausgegeben, ohne Mitverschlüsselung der zufälligen Bitketten könnte Angreifer Ausgabe des Mixes mit öffentlichem Chiffrierschlüssel verschlüsseln und mit Eingabe vergleichen
  - **indirektes Umcodierungsschema zum Schutz der Empfängeranonymität**

$MIX_1, \ldots, MIX_m$	Mixe, von denen die Nachricht weitergegeben werden soll
MIX_0	Sender
MIX_{m+1}	Empfänger
(k_0, A_1, R_1)	anonyme Rückadresse, k_j ... symmetrische Schlüssel

$$R_{m+1} = e \quad , \quad R_j = c_j(k_j, A_{j+1}, R_{j+1})$$ (Rückadressteil. e ist irgendwas Festes)

  - Sender sendet $(k_0(N), R_1)$ an MIX_1
  - **indirektes längengetreues Umcodierungsschema**
    - Vermeidung von abnehmenden Nachrichtenlängen, wenn Nachrichten Mixe in unterschiedlicher Reihenfolge passieren
    - b Blöcke fester Größe, Z Zufallsblock

- $N_1 = R_1 I_1$, $N_j = R_j I_j$
- $I_1 = k_0(Klartext)$, $I_j = [Z_{j-1}]\ k_{j-1}(I_{j-1})$ (Informationsteil)
- $R_{m+1} = [e]$, $R_i = [c_j(k_j, A_{j+1}), k_j(R_{j+1})]$ (Rückadressteil)
  - erster Block wird nicht weitergeleitet, dafür wird zwischen Rückadressteil und Nachrichtenteil ein Block mit zufälligen Bits gesetzt
  - Entschlüsselung mit k_j (Rückadressteil und Zufallsblöcke)
    Ent- / Verschlüsselung (Sender- / Empfängeranonymität) (Nachrichtenteil)
    (diese Information ist Teil von k)
  - damit MIX nicht zu viel über seine Position in der Mixkette erfährt, braucht er nur die Nachrichtenlänge zu wissen, um die Platzierung für den Zufallsblock zu kennen, er muss nicht wissen, wo die Grenze zwischen Rückadressteil und Zufallsblöcken ist
  - wenn man mit k genauso ver- wie entschlüsselt, muss nicht zwischen Sender- und Empfängeranonymität unterschieden werden und die Länge der Informationsblöcke nicht bekannt sein, da komplette Nachricht außer erstem Block mit k verschlüsselt werden kann
    → Zufallsblöcke am Ende
- Umsortierung
  - z.B. alphabetisch (besser als zufällig, da unabhängig kontrollierbar → schließt covert channel)
  - Nachrichten müssen Mixe gleichzeitig verlassen

b) Optimierung

- Kanalnutzung – hybride Verschlüsselung
  - Bandbreite teilt sich in Signalisierungsanteil (Kanalaufbau) und Datenanteil (Datenvermittlung)
  - asymmetrisch im Signalisierungskanal: Kanalaufbaunachricht sendet an jeden teilnehmenden MIX einen symmetrischen Schlüssel, Nachricht wird am Ende nicht verteilt
  - Speicherung von Ein- und Ausgabeschüben der Kanalaufbaunachricht in den Mixen
  - Datenkanal: symmetrische Stromchiffre: Nachricht wird mit symmetrischen Schlüsseln verschlüsselt und auf dem selben Weg wie Aufbaunachricht weitergegeben
  - Sendekanal: Sender baut Kanal auf, verschlüsselt N und sendet N, Mixe entschlüsseln N sukzessive
  - Empfangskanal: Empfänger baut Kanal auf, Sender schickt Ende-zu-Ende-verschlüsselte Nachricht an letzten Mix, Mixe verschlüsseln N sukzessive mit symmetrischen Schlüsseln
  - MIX-Kanal:
    - Kanalwunschnachricht: Rufender übermittelt Kanalkennzeichen, Schlüssel für Ende-zu-Ende-Verschlüsselung, Angabe, ob er der Sender ist, …) → werden verteilt
    - Sender baut Sendekanal bis MIX_m , Empfänger baut Empfangskanal ab MIX_m auf
    - Verknüpfung zugehöriger Kanäle über bei MIX_m ankommenden Klartext aus der Kanalaufbaunachricht: das Kanalkennzeichen
    - jede Station unterhält gleich viele Sende- / Empfangskanäle zu jeder Zeit, notfalls Scheinkanäle
      → Unbeobachtbarkeit des Sendens / Empfangens

c) Sicherheit

- Unbeobachtbarkeit, Unverkettbarkeit der Kommunikationsbeziehung:
  - perfekt komplexitätstheoretisch (→ Schlüsselverteilung) realisierbar, wenn alle Teilnehmerstationen in festem Zeitintervall konstante Anzahl von Nachrichten jeder Länge senden und empfangen (dummy traffic)
- Anonymität der Kommunikationspartner:
  - Verbindungsverschlüsselungen → Angreifer muss MIX | Station okkupieren
  - implizite, private Adressen

- Verteilung
- nichtanonyme Instanz X (MIX):
  - längengetreues indirektes Umcodierungsschema und nicht-anonymes Adressverzeichnis mit anonymen Rückadressen
  - anonymes Abfragen (Sender sendet Nachricht anonym an X, Empfänger sendet anonym Rückadresse an X, Empfänger fragt Nachricht anonym von X ab (zeitlich nicht abhängig)

- Angriff
  - Alle Sender und Empfänger außer das Opfer oder alle Stationen müssen zusammenarbeiten
  - aktiv:
    Eingabe: M^c (soll verfolgt werden), $M^c \cdot f^c$ (wird generiert)
    Ausgabe: $Y = z \cdot 2^B + N$, $Y^* = z^* \cdot 2^B + N^*$, B = |N|, b = |z|
    → $z^* \cdot 2^B + N^* \equiv f \cdot (z \cdot 2^B + N)$ → $f \cdot z - z^* \equiv (N^* - f \cdot N) \cdot 2^{-B}$
    → letzteres kann berechnet werden, muss im Wertebereich der Zufallszahl liegen:
    $(N^* - f \cdot N) \cdot 2^{-B} \in \{(-2^b + 1) \bmod m, \ldots, (f \cdot (2^b - 1) \bmod m)\}$
    Lösungen: Streuung von zufälligen Bitketten in die Nachricht
    Verschlüsselung in einem ganz anderen (symmetrischen) Konzelationssystem
  - passiv:
    - beliebig wählbare Mixreihenfolge und – länge
    - Angreifer hat alle Mixe außer einem unter Kontrolle
    - → Angreifer addiert schon durchlaufende Mixe einer Nachricht mit denen, die die ausgehenden Nachrichten des vertrauenswürdigen Mixes noch passieren → korrekte Summe zeigt Mixweg

d) Fehlertoleranz

- Wahrnehmung möglich durch zeit-periodisches Signal der Mixe
- transiente Fehler im Mix:
  Vermerk im Ende-zu-Ende-Protokoll → Wiederholung der Sendung
- permanente Fehler im Mix:
  - Fehlerbehebung durch Koordination der Mixe untereinander
  - zwei-Phasen-Konzept:
    Phase 1: Informationsübertragung
    Phase 2: Fehlertolerierung
    gegenseitige anonyme Rückadressen: Aktualisierung der nicht-anonymen Stellen zur Adressverteilung und Wahl neuer anonymer Rückadressen
    Verteilung, anonymer Abruf: Mixe im Senderanonymitätsschema aktualisieren
  - Austausch von Mixen:
    - disjunkte Mix-Folgen (langsam, problematisch bei Empfängeranonymität)
    - parallele Reserve-Mixe, die Geheimnis des ausgefallenen Mixes kennen (z.B. Schwellwertschema)
    - ineffektives Koordinationsprotokoll: jeder Mix speichert Protokoll aller Mixe, die er potentiell ersetzen könnte, zusätzlich zu seinem eigenen
    - effektives Koordinationsprotokoll: Mix sendet EingabeInformationseinheiten erst bei deren Ausgabe an Mehrheit aller anderen Mixe
  - Auslassen von Mixen:
    - $N_{n+1} = c_{n+1}(N)$
      $N_n = c_n(k_n, A_{n+1}), k_n(N_{n+1})$
      $N_i = c_i(k_i, A_{i+1}, k_{i+1}, A_{i+2}), k_i(N_{i+1})$

7.6. Angriffe

- verändernd: permanentes Stören, Ziel: denial of service

    - bei Verdacht: Aufdeckung aller Aktivitäten einer Station, Offenbarung:
        - MIX: alle ein- und auslaufenden Informationseinheiten (Zusammenhänge nur wenn gefragt)
        - DC: alle Schlüsselzeichen und gesendeten Zeichen
        - RING / BAUM: alle gesendeten Zeichen
    - Aufdeckverfahren:
        - potentielle Möglichkeit zur Aufdeckung aller Zeichen
        - keine Zeichen werden aufgedeckt, die die Senderanonymität untergraben
        - kombiniert mit Reservierung: jeder Mix reserviert einen Übertragungsrahmen, man sieht, wer nicht mitmacht, alle anderen bleiben anonym
- aktiv: Verkettungsangriff über Betriebsmittelknappheit
    - Ziel: Assoziation zwischen zwei impliziten Adressen
    - knappe Betriebsmittel: Rechenleistung, Sende- und Empfangszeit
    - Bearbeitungszeit einer Station bedingt durch lokale Rechenleistung oder durch vom Kommunikationsnetz vorgegebene Bandbreite → wird geraten oder ist bekannt
    - Angreifertyp: viele kooperierende Angreiferstationen oder wenige, die gegen faires Zugriffsprotokoll verstoßen
    - Erkennung: leicht (keine Knappheit, kein Angriff)
    - Verhinderung:
        - statische Aufteilung aller Betriebsmittel einer Station auf vorhandene Adressen
        - dynamische Aufteilung der Betriebsmittel: pseudozufällige Verteilung, temporäre Nutzung der gesamten Bandbreite (wenn es das Netz erlaubt)

7.7. Einordnung ins Schichtenmodell

- Schicht 0: RING, BAUM (speziell zur Verteilung entwickelte Kommunikationsnetze)
- Schicht 1: Verbindungs-Verschlüsselung (Angreifer erfährt nichts unnötig in höheren Protokollen), Verteilung über Kanalselektion, DC-Zeichenübertragung und -überlagerung, digitale RING-BAUM-Signalgenerierung
- Schicht 2: anonymes Mehrfachzugriffsverfahren
- Schicht 3: Verteilung (z.B. Überflutung), MIX: Pufferung und Umcodierung
- Schicht 4: Ende-zu-Ende-Verschlüsselung (direkte Arbeit zwischen Teilnehmerstationen), Generierung und Auswertung von impliziten Adressen
- Schicht 5-7 müssen nur dafür sorgen, dass Anonymität und Unverkettbarkeit gewahrt bleiben, sie nutzen ansonsten die Dienste der unteren Schichten und haben mit Kommunikationsbeziehungen nichts zu tun

8. Zufallszahlenerzeugung

8.1. Allgemeines

- nicht gut: Datum, random-Funktion einer Standard-Programmiersprache
- physikalische Phänomene: Rauschdioden, radioaktive Prozesse, Turbulenzen im Lüfter eines normalen Rechners
- Zufallszahlenkombination (XOR)
  - zufällig und geheim bei einer zufällig und geheimen Zahl
  - gut, wenn andere Instanz garantieren will, dass man eine zufällige Zahl wählt

8.2. Pseudozufallsbitfolgengenerator (PBG)

a) Grundlagen

- p, q unabhängig und zufällig gewählte Primzahlen
- $n = p \cdot q$
- $p \equiv q \equiv 3 \pmod 4$
- Sicherheitsparameter: Länge l der Faktoren
- Annahme für polynomialen Algorithmus F, Polynom Q, $\exists L \ \forall l \geq L$:
  $W(F(n) = (p;q)) \leq \frac{1}{Q(l)}$
- p+1, p-1, q+1, p-1 müssen jeweils (min.) einen großen Primfaktor haben (min. 100 Bit)
  → ist hinreichend wahrscheinlich für alle Primzahlen mit $l \gg 500\ Bit$

b) Anforderungen

- Startwertgenerieralgorithmus → Sicherheitsparameter l: Schlüssel n, Startwert s (seed) der Länge l
- Bitfolgengenerieralgorithmus → (beliebig lange) Bitfolge $b_0 b_1 b_2 \ldots$
- **deterministisch**
- **effizient** (in polynomialer Zeit berechenbar)
- **sicher**:
  - nicht mit signifikanter Wkt. von echten Zufallsfolgen unterscheidbar durch probabilistischen Test polynomialer Laufzeit
  - Test T ordnet z.B. Bitfolge reeller Zahl zwischen 0 und 1 zu, Mittel der Ergebnisse bei PBG und echten Zufallsfolgen wird verglichen:
    - α_m durchschn. Zuordnung durch T von echten m-Bit-Folgen
    - $\beta_{l,m}$ durchschn. Zuordnung durch T von PBG-generierten m-Bit-Folge der Länge l

c) Verwendung

- **Pseudo-one-time-pad (Vigenere-Chiffre)**
  - symmetrisch: Schlüssel n, Startwert s ist geheim und Algorithmus zum Erzeugen der Bitfolge Sender und Empfänger bekannt
- **s²-mod-n-Generator**:
  - p, q (ungefähr gleiche Bitlänge, Anforderungen wie oben)
  - Startwert s zufällig aus $\mathbb{Z}_n^*$

- Generierung der Folge:
  - $s_0 = s^2 \pmod n$, $s_{i+1} = s_i^2 \pmod n$, $b_i = s_i \pmod 2$ (letztes Bit)
  - Ausgabe: $b_0 b_1 b_2 \ldots$
- asymmetrisch (= **BIG-System**):
  - geheim: p,q Faktoren von n
  - öffentlich: n
  - Sender wählt zufälligen Startwert und verknüpft Klartext mit aus Startwert resultierender Folge, hängt ans Ende s_{k+1} (letztes nicht gebrauchtes Quadrat s)
  - Empfänger zieht Wurzeln aus s_{k+1} mit p und q
  - sicher gegen passive Angriffe
  - aktiver Angriff:
    - chosen-ciphertext attack: Angreifer wählt irgendeine Bitfolge, fügt das s_{k+1} der ihn interessierenden Nachricht an, erhält Entschlüsselung und kann daraus Pseudozufallsbitfolge berechnen und die gewünschte Nachricht entschlüsseln

d) Primzahlenerzeugung

- **Primzahlsatz**:
  - Anzahl $\pi(x)$ vom Primzahlen bis Maximum x: $\frac{\pi(x)}{x} \approx \frac{1}{\ln(x)}$

    → für Länge l Bit: $\pi(2^l) \approx \frac{2^l}{\ln(2^l)} \approx \frac{2^l}{l \cdot \ln(2)}$ → jede $l \cdot \ln(2)$ 'te Zahl ist prim
  - im Mittel ist ½ aller Primzahlen $\equiv 3 \pmod 4$ und ¼ $\equiv 3 bzw. 7 \pmod 8$
- Erzeugung zufälliger Primzahlen p:
  - wähle zufällig Zahl der Länge l Bits → nach ca. $l \cdot \ln(2)$ Versuchen Erfolg
- **RABIN MILLER-Test** (ist p prim?):
  - in Praxis zunächst Test durch Division mit kleineren Primzahlen (höchstens Rechner-Wortlänge)
  - wenn p prim → $\forall a \in \mathbb{Z}_p^*: \; a^{\frac{p-1}{2}} \equiv \pm 1 \pmod p$ (→ kleiner Satz von Fermat)
  - wenn p nicht prim, gilt Formel nur in ¼ Fällen
  - Teste σ -mal → wenn Formel immer gilt, ist p mit $1 - 4^{\sigma}$ Wkt. Prim

    → Pseudoprimzahlen (sind nicht mit 100% Wkt prim)

9. Rechnen mod n

- Teilbarkeit, Kongruenz, mod-Operator, div-Operator
- Additives Inverse | Multiplikatives Inverse
- abelsche Gruppe: Kommutativität gilt
- Kürzungsregel: $x \cdot a \equiv x \cdot b \pmod{x \cdot n}$ → x lässt sich kürzen
- $a_{0,} a_{1,} \ldots, a_{n-1}$ vollständiges (betragskleinstes) Restsystem mod n
- $\mathbb{Z}_n$ … Restklassenring
- Generator g von G in multiplikativer Gruppe G: Potenzen von g erzeugen alle Elemente aus G
- zyklische Gruppe: Gruppe mit mindestens einem Generator
- diskreter Logarithmus von h zur Basis g modulo p: $x = \log_g(h) \bmod(p)$, $g^x = h \pmod{p}$
- $\mathbb{Z}_n^*$ … primäres oder reduziertes Restsystem mod n, multiplikative Gruppe von $\mathbb{Z}_n$: alle Elemente a mit Inversem → ggT(a, n) = 1,

  Beweis:

  für $a, b, c, d, e, n \in \mathbb{N}$ und $ggT(a,n) = x$ gilt:

  $a = c \cdot x$, $n = d \cdot x$, $b = a^{-1}$: $b \cdot cx = e \cdot dx + 1$ → $1 = (b \cdot c - e \cdot d) \cdot x$ → $x = 1$

Eulersche ϕ - Funktion

- $n = p_1^{r_1} \cdot \ldots \cdot p_m^{r_m}$: $\phi(n) := \left|(\mathbb{Z}_n^*)\right| = (p_1 - 1) \cdot p_1^{r_1 - 1} \cdot \ldots \cdot (p_m - 1) \cdot p_m^{r_m - 1}$

Satz von Euler

- $n \in \mathbb{N}, a \in \mathbb{Z}, ggT(n,a) = 1$: $a^{\phi(n)} \equiv 1 \pmod n$ → $a^b \equiv a^{b \bmod \phi(n)} \pmod n$

  → p prim: $a^{p-1} \equiv 1 \pmod p$ (**kleiner Satz von FERMAT**)

SMA (square and multiply)

- $a^b \pmod n$
- $b = (b_{l-1} \ \ldots \ b_0)_2$
- $a^{(b_{l-1} \,..\, b_{l-i} \ 0)_2} = a^{2 \cdot (b_{l-1} \,..\, b_{l-i})_2} = \left(a^{(b_{l-1} \,..\, b_{l-i})_2}\right)^2$
- $a^{(b_{l-1} \,..\, b_{l-i} \ 1)_2} = a^{2 \cdot (b_{l-1} \,..\, b_{l-i})_2 + 1} = \left(a^{(b_{l-1} \,..\, b_{l-i})_2}\right)^2 \cdot a$

Chinesischer Restsatz

- $a \equiv b \pmod n \Leftrightarrow a \equiv b \pmod p \wedge a \equiv b \pmod q$ (p, q, … paarweise teilerfremd)
- **CRA** (Chinesischer Restealgorithmus):
  - $x = r_i \pmod{m_i}$:
  - $M := m_1 ; x = r_1 \pmod M$
  - for i = 2...k: $x := x + M \cdot ((r_i - x) \cdot M^{-1} \pmod{m_i}) \pmod{\prod m_i}$

    $M := M \cdot m_i$

Quadrate & Wurzeln

- $QR_n := \{\ x \in \mathbb{Z}_n^* \mid \exists\, y \in \mathbb{Z}_n^* :\ y^2 = x \pmod n\ \}$ (**quadratische Reste**)
- bilden multiplikative Gruppe
- **in** $\mathbb{Z}_p$ **:**
  - Körper (jedes Element besitzt multiplikatives Inverses)
  - für p>2 haben alle quadratischen Reste x genau 2 Wurzeln (y und -y)

    → $|QR_n| = \frac{p-1}{2}$
  - **Legendre-Symbol**: $\frac{x}{p} := \begin{cases} +1 & falls\ x \in QR_p \\ 0 & falls\ p|x \\ -1 & falls\ x \in QNR_p \end{cases}$ (multiplikativ)
  - **Euler-Kriterium**: $\frac{x}{p} = x^{\frac{p-1}{2}} \pmod p \qquad \in \{\pm 1\}, weil\ (x^{\frac{p-1}{2}})^2 \equiv 1$

    (nach Satz von Fermat, iterativ, Beispiel (-1))
  - für $p \equiv 3 \pmod 4$ gilt: $y = x^{\frac{p+1}{4}} \pmod p$

    Beweis:

    $$y^2 = x = 1 \cdot x = x^{\frac{p-1}{2}} \cdot x = x^{\frac{p-1}{2}+1} = x^{\frac{p+1}{2}} = (x^{\frac{p+1}{4}})^2$$
  - $-1 \in QNR_p$, $x \in QR_p$ → $-x \in QNR_p$
- **in** $\mathbb{Z}_n$ **:**
  - $n = p_1^{r_1} \cdot \ldots \cdot p_m^{r_m}$
  - **Jakobi-Symbol**: $\left(\frac{q}{n}\right) = \left(\frac{q}{p_1}\right)^{r_1} \cdot \ldots \cdot \left(\frac{q}{p_m}\right)^{r_m}$
  - Bestimmen der wesentlich verschiedenen Wurzeln:
    - Berechnungen der Wurzeln mod der Primfaktoren
    - Verknüpfung per CRA:

      $$r_0 = CRA\begin{pmatrix} x \equiv r_q \pmod q \\ x \equiv r_p \pmod p \end{pmatrix} \ ,\ r_1 = CRA\begin{pmatrix} x \equiv r_q \pmod q \\ x \equiv -r_p \pmod p \end{pmatrix}$$
    - wer Wurzeln ziehen kann, kann auch faktorisieren:

      w, w' wesentlich verschiedene Wurzeln von x (mod n)

      → $w^2 - w'^2 \equiv 0 \pmod n$ → $(w-w')\cdot(w+w') \equiv 0 \pmod n$

      aber $w \neq \pm w'$ → ein Faktor: $ggT(n, w+w')$

BLUM-Zahlen

- $BLUM_k = \{n = p \cdot q \mid p, q\ prim,\ \ p, q \equiv 3 \pmod 4,\ |p| = |q| = k\}$

10. Komplexitätstheorie

- Menge aller deterministisch polynomial lösbaren Probleme **P**
  - es gibt einen Algorithmus, der Lösung systematisch, deterministisch, polynomial berechnet
- Menge aller indeterministisch polynomial lösbaren Probleme **NP**
  - man rät (indeterministisch) Lösungen und testet sie mit deterministisch polynomialem Algorithmus
  - Beispiel: QRA
- $P \subseteq NP$
- jedes andere Problem in NP lässt sich in polynomialem Zeitaufwand auf ein **NP-vollständig**es Problem reduzieren
- Addition / Subtraktion / Multiplikation: $O(l^2)$, Exponentation: $O(l^3)$ ($l = |n|$)

11. OSI/ISO Schichtenmodell

- Schicht 0: Medium
- Schicht 1: Bitübertragung – digitale Signalgenerierung
- Schicht 2: Datensicherung – Blockeinteilung, Bildung von Prüfsummen, Übertragungsfehler, Mehrfachzugriff
- Schicht 3: Network – Weiterleiten, Finden von Wegen, Router
- Schicht 4: Transport – Transport- und Ende-zu-Ende-Kontrolle, Segmentierung, Stauvermeidung, Multiplexmechanismen, Fehlersicherung und -behebung

°)° → '}'