Data Security H2: Cryptografie

Inhoud

[Cryptografie 3](#_Toc466206220)

[Doelstellingen 3](#_Toc466206221)

[Symmetrische encryptie 3](#_Toc466206222)

[Asymmetrische encryptie 4](#_Toc466206223)

[Hashing 5](#_Toc466206224)

[Veiligheid 6](#_Toc466206225)

[Toepassingen 6](#_Toc466206226)

# Cryptografie

* Encryptie
  + Doel: data coderen in onherkenbaar formaat
* Bruteforce attack
  + Alle mogelijke sleutels proberen
* Veilige sleutel
  + Von Neumann - Landauer: fysische limiet
    - Niet-reversibele computer
* Cryptoanalyse
  + Zwakheden algoritme of kennis over plaintext
  + Frequentietabellen, statistische analysie,...
  + Mogelijke countermeasure: de hoeveelheid data die op dezelfde manier gecodeerd wordt, beperken
  + VB: een bericht onderverdelen in stukken van verschillende keys
    - MAAR: complexere sleuteldistributie
* Side channel attacks
  + Gebruik maken van nevenverschijnselen
  + VB: iPhone naast laptop registreert toetsaanslagen
* Algoritme voor encryptie
  + Goed algoritme beschermt data op basis van (geheime) sleutel & is publiek beschikbaar

# Doelstellingen

* Confidentiality: informatie geheim houden
* Integrity: garantie op correctheid van de informatie
* Authenticatie: correcte versleuteling data geeft zekerheid over identiteit
* Non-repudiatie: onweerlegbaarheid
* Codeertechnieken
  + Symmetrisch - asymmerisch
    - Secret (shared) key - public key
  + Block cipher
    - Bericht verdeeld in blokken met een bepaalde grootte
    - Blokken zijn versleuteld
  + Stream cipher
    - Symbool per symbool versleutelen
    - Geen padding nodig
    - Sneller dan block cipher

# Symmetrische encryptie

* Zelfde geheime sleutel voor encryptie & decryptie
* Voordelen: snel & kleine sleutels
* Nadeel: uitwisseling sleutel
* DES
  + Block cipher 64 bit
  + 18 functies
    - Transpositie & permutaties
    - Sleutels afgeleid van 56 bit sleutel
  + Geoptimaliseerd voor hardware
* AES
  + Opvolger DES
  + Block cipher 128 bit, sleutel [128|192|256] bit
  + Aantal ronden
    - Rondesleutels afgeleid van hoofdsleutel
    - Bytesubstitutie
    - Rijrotatie (permutatie)
    - Kolomvermenigvuldiging
    - Sleuteltoevoeging
* Block encryption modes
  + Grote hoeveelheiden data encrypteren
  + ECB
    - Hetzelfde encryptie-algoritme met dezelfde sleutel gebruiken op de verschillende blokken
    - Nadeel: informatie wordt prijsgegeven (plaintext-ciphertext pairs)
  + CBC (Cipher Block Chaining)
    - De ciphertext van het vorige blok gebruiken om de volgende plaintext blok te vervormen alvorens het te encrypten
    - Voordeel: gelijke stukken plaintext geven telkens een ander ciphertext resultaat = geen informatie prijsgeven
    - Maar: initialisatievector (IV) bepalen tussen beide partijen zodat er afspraken zijn hoe de plaintext wordt vervormd
  + CTR (Counter)
    - Teller encrypteren
    - XOR
    - Voordelen:
      * Parallelle encryptie & decryptie mogelijk
      * Preprocessing mogelijk
      * Elk blok is apart aanspreekbaar
      * Zenders die ook ontvangers zijn moeten alleen het encryptie-algoritme implementeren

# Asymmetrische encryptie

* Verschillende sleutel voor encryptie & decryptie
* 1 sleutel is bekend (public key)
* Nadelen: grote sleutels, langere berekeningen voor encryptie & decryptie (uitzondering: ECC)
* Eisen:
  + Computationeel eenvoudig om een pair te genereren
  + Private key is niet (eenvoudig) af te leiden uit de kennis van de public key
  + Computationeel eenvoudig om encryptie & decryptie te doen (met de juiste key)
* Complexere wiskundige operaties
  + (vs symmetrisch: eenvoudige operaties op bitpatronen)
* Diffie-Hellman key exchange
  + Gemeenschappelijke secret key genereren
  + Moeilijke berekeningen gebaseerd op publieke waarde & private key
  + Nadeel: geen garantie dat de bekende waarden effectief afkomstig zijn van de tegenpartij waarmee je wil communiceren 🡪 MITM mogelijk
* RSA
  + Toepassing 1: private key voor zenden, public key voor ontvangen
    - Vertrouwelijkheid: iedereen kan de public key gebruiken om het bericht te lezen 🡪 geen gegarandeerde vertrouwelijkheid
    - Authenticatie: slechts 1 persoon kan het bericht gegenereerd hebben 🡪 zekerheid over de identiteit van de zender
    - Non-repudiatie: de zender kan niet ontkennen dat hij het bericht verstuurd heeft
    - Integriteit: een MITM kan het bericht enkel lezen, niet vervangen door een ander versleuteld bericht
  + Toepassing 2: public key voor zenden, private key voor ontvangen
    - Vertrouwelijkheid: enkel de persoon met de private key kan de data lezen
    - Maar: geen authenticatie/non-repudiatie/integriteit want: iedereen kan de public key gebruikt hebben om de boodschap te genereren of via MITM te onderscheppen & vervangen
  + Voorgaande situaties: keuze integriteit vs vertrouwelijkheid
  + Maar: ook beiden mogelijk
    - 2x asymmetrische encryptie toepassen: 1x met de public key van de tegenpartij, 1x met de eigen private key
    - 1e laag: garandeert vertrouwelijkheid
    - 2e laag: garandeert authenticatie/non-repudiatie/integriteit
    - Nadeel: 2x zoveel rekenkracht nodig

# Hashing

* Hashing algoritme reduceert data tot waarde (digest)
  + Onomkeerbaar
  + Collision resistant
  + Moeilijk om H(x)=H(y) te vinden
* Voordelen: zeer snel & controle op waarde (checksum)
* Nadelen: onomkeerbaar & niet bruikbaar als encryptie
* Voorbeelden:
  + CRC32: 32 bit (geen cryptografische hash)
  + MD5: 128 bit
  + SHA-1: 160 bit
  + SHA-2: 224, 256, 384, 512 bit
  + SHA-3: 224, 256, 384, 512 bit
* Message authentication
  + Doel: integriteit & identiteit afzender bevestigen, evt. non-repudiatie
  + Encryptie (symmetrisch & asymmetrisch)
    - Soms ongewenste effecten
    - Encryptie is niet altijd gewenst
    - Integriteit indien wel encryptie?
  + Oplossing: Message Authentication Code (MAC) toevoegen
    - Stukje data dat toelaat aan de ontvanger om te verifiëren of de (cleartext) data niet werd aangepast na verzending

# Veiligheid

* AES: geen noemenswaardige beveiligingsproblemen bekend
* RSA (en andere asymmetrische algoritmes)
  + Verband public<->private key

# Toepassingen

* Digitale handtekening
* Ondertekende email
* Envelopped data
* Secure web connection
  + Client: vraagt beveiligde verbinding
  + Server: reageert met een public key
  + Client: gebruikt public key om een session key te versturen
  + Server: kan als enige de session key decrypten, want hij beschikt als enige over de private key
* Controle identiteit
  + PKI (Public Key Infrastructure)
    - Hardware, software, people, policies, procedures
    - Create, manage, distribute, store, revoke certificates
    - Manage public key encryption
  + SSL/TLS
    - Client & server: berichten uitwisselen om een bepaalde cipher suite te kiezen
    - Server: maakt keuze bekend & biedt zijn certificaat aan
    - Client: controleert inhoud certificaat; indien vertrouwd: informatie uit cert gebruiken om een gemeenschappelijke key voor symmetrische encryptie vast te leggen
    - Client: stuurt een change cipher spec: laat server weten dat alles vanaf nu geëncrypteerd zal verlopen
    - Server: stuurt een change cipher spec
  + Chain of trust:
    - Root CA -> Intermediate CA -> SSL Certificate -> Intermediate CA -> Root CA