Mag. iur. Dr. techn. Michael Sonntag

Cryptography

SSL, Certificates, VPN

Institute for Information Processing and
Microprocessor Technology (FIM)
Johannes Kepler University Linz, Austria

E-Mail: sonntag@fim.uni-linz.ac.at
http://www.fim.uni-linz.ac.at/staff/sonntag.htm

Questions?

Please ask immediately!

Introduction

- General aspects
  - → Why and where to use
- Technical aspects
  - → Algorithms and their strength, required environment
- Certificates
  - → Content, PKI, revocation
- SSL
  - → Modes, protocol
- Legal aspects of cryptography
- VPNs
  - → PPTP and IPSec

Why cryptography?

- Security is a very important aspects, especially if money (or equivalents) is contained in transactions
- E-Business is usually "business at distance"
  - → You cannot see your partner
  - → You don't know your partner very well
  - → You can't know who is in the middle of your connection
  - → ...

Security is needed!

- Technical aspect of security is cryptography
  - → Encrypting data against disclosure, modification
  - → Signing data against modifications, repudiation
- Other aspects of security are also needed
  - » E.g.: Do you know what your employees actually do with data?
  - → But not discussed here!

Application areas

- Storing data encrypted
  - → Even access will not lead to disclosure
    - » Example: File encryption programs
- Transmitting data securely
  - → Encrypted transmission prevents eavesdropping
    - » Example: SSL
- Identifying your partner
  - → Preventing man-in-the-middle attacks
    - » Example: SSL
- Proof of identity
  - → Avoiding impersonation
    - » Example: Digital signatures ("Bürgerkarte")

Software components

- Several different classes of algorithms required:
  - → Hash functions: Handling the whole document takes too long
    - » Drawback: Content could be substituted!
  - → Encryption/Decryption: The same algorithm for symmetric but different for asymmetric encryption/signatures
  - → Signature: Combining a document with a private key
  - → Verification: Checking a document + signature with public key
  - → Key agreement: Creating a shared secret
    - » Even if both parties do not have a shared secret to start with!
  - → Key generation: Creating secure keys
    - » Requires e.g. secure random generators
    - » From passwords: Creating keys suitable for algorithms
- For each class several/many algorithms exist
  - → Some good, some bad (=broken, erroneous, ...)

Algorithms

- Symmetric:
  - → DES, 3DES: 54 Bit; DES is now insecure; 3DES sufficient for commercial use (frequent key changes recommended)
- Asymmetric:
  - → RSA: Classic/first asymmetric cipher (rather slow)
    - » Keysize arbitrarily (>=1024 recommended); no longer patented!
  - → AES (=Rijndael): New "standard" algorithm; different keysizes
  - → DSA: Only signatures supported, no encryption possible
- Hash:
  - → SHA-1, RIPEMD-160: 160 Bit
  - → MD5: 128 Bit (not recommended any more)
- Other:
  - → Diffie-Hellman: Key agreement without previous knowledge
    - » Generates a shared secret key

Strength of algorithms for the future

- Key length are not static:
  - → Faster computers
  - → Advances in mathematics
  - → New attacks (most dangerous of all!)
- Decicision for length must incorporate
  - → Time required for calculation
  - → Degree of security (=amount of money required for breaking)
  - → Time the calculated value should remain secure!
    - » Very often ignored!
    - » Guideline: For the next 20 years (=2024)
      - – Symmetric: ≈ 89 Bit
      - – Asymmetric: RSA, ...: ≈ 2113 Bit; DSA: ≈ 157 Bit
      - – Corresponds to a budget for an attack in 1 day of ≈ 732 million US$

Source: Lenstra, A. K.,Verheul, E. R.: Selecting Cryptographic Key Sizes. DuD 24 (2000), 166
Full article: http://security.ece.orst.edu/koc/ece575/papers/cryptosizes.pdf

Environment components

- Encryption algorithms are not all there is to be secure
- Many other elements must be taken care of:
  - → Technical "surroundings":
    - » Secure viewer: Showing exactly the content to sign and not something different
    - » Secure transmission of codes/PINs from chipcards/terminals to the CPU actually calculating signatures
    - » Physical access control/restrictions?
  - → Organizational issues:
    - » Who knows the encryption keys and where are thes stored?
    - » How to get at them in case of illness/dismissal?
    - » Who is allowed to do what? Does the equipment support these different security levels?
    - » Securing keys/certificates etc. against loss

Certificates

- Public keys must be connected to certain individual/device
  - → Everyone can use/create a key, but how do you know that the person holding the private key is actually "Donald Duck" or a certain person using this pseudonym?
- Certificates connect the public key to a name
  - → As public key "match" the private keys, they are assigned too
- Certificates can contain other information
  - → E.g. server certificates can contain E-Mail of administrator
  - → Person certificates can contain restrictions or special permissions/empowerments
- Certificates are signed too so nobody can tamper with them
  - → Chicken-egg problem: Who signed the certificate?
    - » Pre-shared "master" certificate or Public Key Infrastructure (PKI)

Certificate content

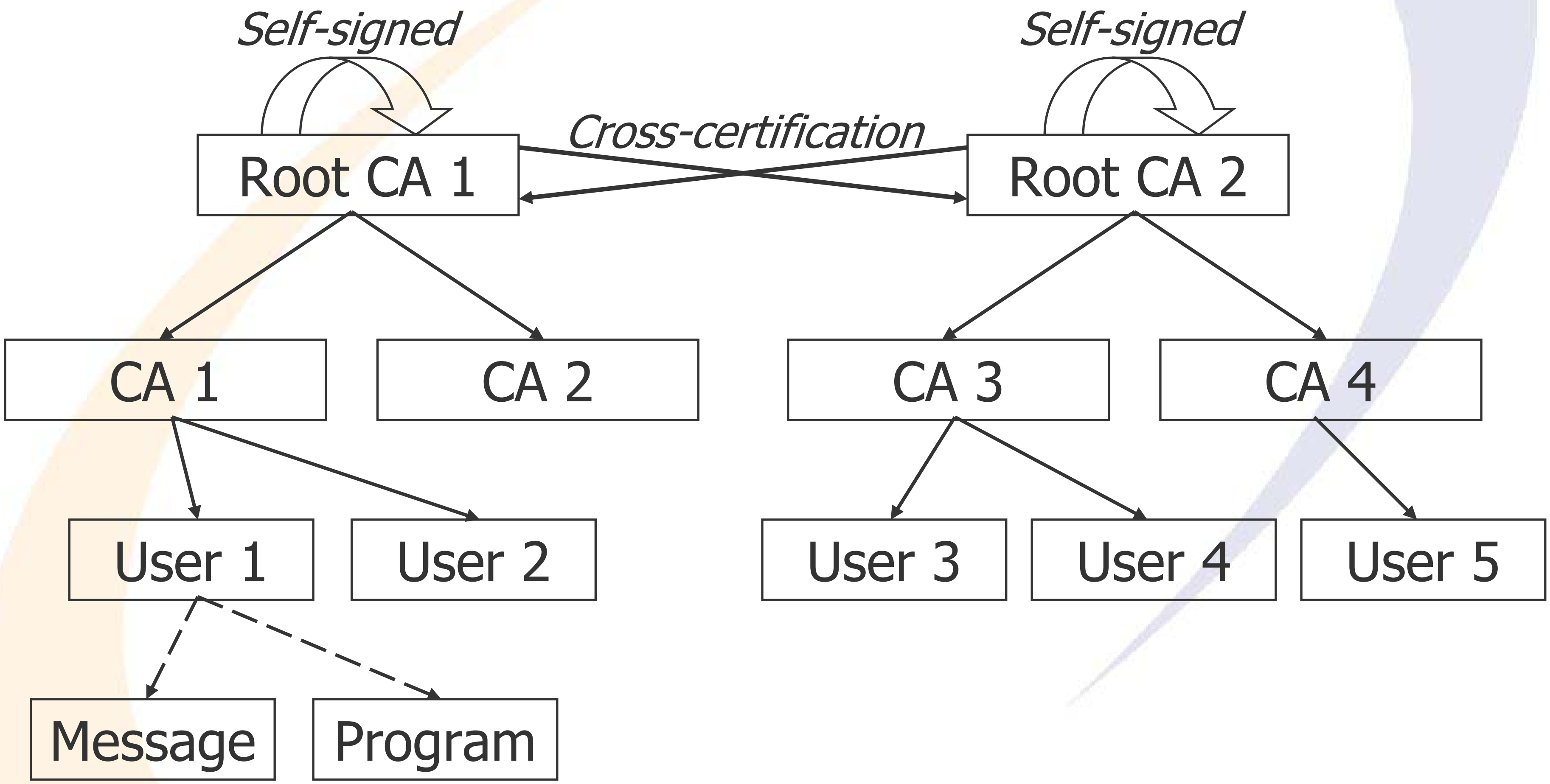
- Currently only certificates of type X.509 are of importance
  - → Several versions available
  - → Standard is not too clear
    - » Certificates from one vendor might be incompatible with those from another vendor or with some software
    - » Special problem: What data, which form, which "schema"
- No problem:
  - → Public key, including algorithm
  - → Issuer: Who "guarantees" for the association key ↔ name
  - → Version, serial number, validity, unique IDs
- Problems:
  - → Subject (=associated name): Different elements (E-Mail, additional/missing parts, ...)
  - → Extensions: Key usage, CRL distribution, contraints, etc.

See http://www.hack.org/mc/texts/x509guide.txt

Public Key Infrastructure (PKI)

- Who guarantees, that the certificate is "correct"?
    - » I.e. that the key belogs to exactly the identified person and not e.g. some imposter
  - → The signature of the certificate
  - → Who guarantees, the this signature is "correct"?
    - » ...
      - – ...
- The "top-level" certificate is self-signed
  - → Key used for signing is the one for the public key contained
  - → This certificate you "just have to trust"
    - » Obtained from a secure source, verified, ...
  - → Can also be "cross-certified": One top-level certificate is used to sign another top-level certificate and in reverse
    - » Good for few CAs (otherwisen: $O(N^2)$!)

PKI Example


Self-signed
Self-signed
Cross-certification
Root CA 1
Root CA 2
CA 1
CA 2
CA 3
CA 4
User 1
User 2
User 3
User 4
User 5
Message
Program

Certificate revocation

- Sometimes certificates must be "removed", e.g. when
  - → some attributes are incorrect (name/profession changes)
  - → private key is disclosed
  - → algorithm is now insecure
  - → no longer used (e.g. server certificates)
- although they are still valid (looked at them alone)
- Solution: Revocation lists
  - → Must (should) be consulted on each verification of a signature
  - → Must happen fast e.g. on lost keycards
    - » Legal requirement: Maximum of 3 hours
  - → Contains a timestamp
    - » Signature before the revocation must remain valid indefinitely!
- Technical standards/infrastructure still lacking for this!

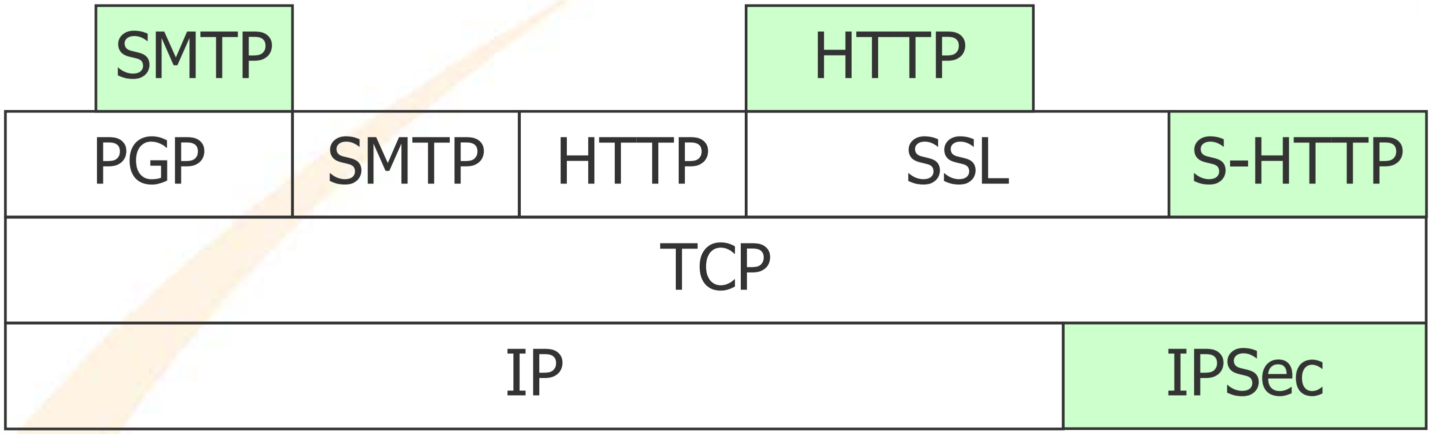
Certificates and digital signatures

- Creating/Verifying a digital signature:
  - → Encrypt values (see below) with private part of key
  - → Send document and/or encrpyted value to recipient
  - → Recipient obtains certificate of signer (however) and checks it
  - → Recipient decrypts value with public part of key and checks it
- Two kinds of signatures possible
  - → "Internal": The document is "encrypted" with the private key
    - » Verification=Decryption; reading the document takes long
      - – "Avalanche property" of good (!) algorithms: Minimal modifications lead to complete gibberish on decryption
  - → "Extneral": A Hash value is calculated and then signed
    - » Verification=Comparing the decrypted hash with the (newly) calculated one from the plaintext document; quite fast
      - – Possible problem: Finding a similar text with same hash value
      - – Quality of hash algorithm is therefore very important here!

Encryption for the WWW

- When transmitting sensitive information on the Web, the communication should be encrypted
  - → Examples: credit card numbers, company-internal forms, ...
- Currently one method is widely used: SSL
  - → Secure Socket layer: A general solution for encrypted TCP traffic; most common use with http (⇒https)
- SSL provides:
  - → Encrypted communication: Eavesdropping almost impossible
    - » Also depends on the actual algorithm used (national restrictions!)
    - » Uses symmetric cryptography for speed; numbers against replay
    - » Asymmetric cryptography used for key exchange
  - → Mutual authentication supported
    - » Uses asymmetric cryptography and certificates
  - → Configuratiojn very important (algorithms, cert. storage, ...)

Security for the WWW



- PGP: Pretty Good Privacy
- SSL: Secure Socket Layer
  - → The whole communication is secured
- S-HTTP: HTTP + security extensions
  - → Single messages are secured
- IPSec: IP Security
  - → Every communication is encrypted and/or authenticated

Authentication modes

- Either the server or both the server and the client can be authenticated
  - → For the WWW this means, authenticating only the webbrowser is not possible!
  - → Commonly, only the server is authenticated
- Authentication requires a certificate
  - → Most browsers come with a list of top-level CA certificates
  - → Unknown certificates can be imported or accepted ad-hoc
    - » Large part of CA business is based on this: No questions!
  - → For smaller companies: Create their own certificate and distribute it to partners
    - » For public: Present it on website (but is this really secure?)
  - → Webserver must have access to private key: Must be secured very well within the system!

SSL: The protocol (1)

Client **Server**

ClientHello →

ServerHello
[ServerCertificate]
[ServerKeyExchange]
[CertificateRequest]
ServerHelloDone ←

[ClientCertificate]
ClientKeyExchange
[CertificateVerify]
ChangeCipherSpec
Finished →

ChangeCipherSpec
Finished ←

↔ Encrypted [and authenticated] communication

[]: optional part

SSL: The protocol (2)

- Client-/ServerHello: Contains a random number and encryption/compression capabilities
  - → Random number: Prevents replay attacks
- S.-Certificate: Certificate including chain up to top-level CA
- ServerKeyExchange: If the server has no certificate or it cannot be used for encryption
  - → Commonly uses Diffie-Hellman Key Exchange protocol
  - → Signed by certificate to avoid man-in-the-middle attacks
- CertificateRequest: Non-anonymous server can request a client certificate
  - → Contains list of certificate types understood
  - → Contains list of DNs of accepteable CAs
    - » DN = Distinguished Name; format for name in X.509 certificates

SSL: The protocol (3)

- ServerHelloDone: Hand-off to tell client that this is all
- ClientCertificate: Certificate of the client or warning that no (matching) one is available
  - → Server cann accept without certificate or terminate protocol
- ClientKeyExchange: Client part of key exchange protocol
  - → Always required!
- CertificateVerify: Signed digest of messages
  - → To prove the knowledge of the private key for the certificate
- Finished: Encrypted & signed with (new) negotiated values
  - → Content may be sent immediately (no wait for reply required)

- ChangeCipherSpec: Switch to encryption
  - → This message is still handled according to the old algorithms!

What you (don't) get!

- The server is the one specified in the certificate
  - → Not necessarily the actual webserver; this is verified by the browser, however!
    - » Difficulties for servers having different Domain Names
- Client knows the private key for its certificate (if provided)
- Revocation of certificates was checked
  - → Depends on browser; not in protocol
- Encryption, authentication, integrity, non-repudiation, no manipulation, no replay

- What you don't get:
  - → Additional certificate content (e.g. attributes) often ignored
  - → Hiding who talks to whom

Alternatives: Shared keys

- Only suitable for very small group of partners communicating
  - → See VPN later; especially VPN tunnels!
- Keys must be exchanged over a trusted channel
  - → I.e. NOT over the channel used for communicating!
- Protocols must use "Challenge-Response": The key may never be sent in cleartext!
  - » Before you don't know who is on the other side
  - → Common way: random value sent, hashed with secret key, sent back, compared to expected response
    - » No eavesdropper/man-in-the-middle can retrieve the key from it
- Not possible with SSL!
- Advantage: Usually very simple to manage
  - → Agree on a keyphrase, telephone call ⇒ works!
    - » No additional infrastructure needed (PKI, CRL, etc.)

Alternatives: Web of trust

- Similar to PKI, but distributed model
  - → Signing someone others keys to certify, that the association is correct; diverse servers for storing keys and signatures
- Based on transitivity of trust (=the signatures):
  - → A trusts B, B trusts C, C trusts D ⇒ A trusts D
- Not possible with SSL!
  - → Uses different certificate format
  - → Currently mainly used for E-Mails
- Advantage: No single point of failure
- Problem: No guaranteed decision
  - → Perhaps just no trusted connection exists; still valid & correct!
  - → CA are possible, but not necessary

"Official" certificates: Advantages / Disadvantages

- \+ Identity of person/company verified accurately
  - » More trust than a self-signed certificate
- \+ No warning messages for (modern!) browsers
- \+ Interoperability with many browsers
  - » Creating a "good" certificate is not easy!
- \+ Key length issues, etc. are taken care of
- \+ Provides reliable servers and CRL services
- − Costs money (and expires regularly, requiring a new one!)
- − May take some time to obtain (depending on CA/location)
- − Guarantees for content are small or non-existing

- Result:
  - → Public website: Indispensable (browser warning)
  - → Private/internal use: Very few reasons
    - » Except: Large companies, where managing secure and available directories and CRLs are difficult!

"Official" certificates: Obtaining one

- Fill in form for certificate
  - → Creates a "Certificate Signing Request" (CSR)
    - » Contains the certificate data, but not the private key!
- Pay the price
- CA verifies the content
  - → Usually through notarized/official documents
    - » Perhaps also personally (depending on application)
- CA creates the certificate
  - → Signed by its own private key
- CA makes the certificate available
  - → To the customer
  - → Usually also in the directory
    - » Everyone can download it

Legal aspects: Enhanced protection

- Protected communication enjois some legal benefits:
- Https:
  - → Credit card information may be sent over it
    - » Otherwise this is a breach of confidence by default!
  - → Adequate protection of privacy
    - » E.g. for medical information
  - → Reduced liability: Not using SSL might be negligence
  - → (Limited) liability of CA for wrong information
- E-Mail:
  - → Depending on the certificate this might be equal to a full manual signature
  - → Now protected by "privacy of correspondence"
    - » Other E-Mail is like postcards and therefore legally unprotected
  - → Better value as evidence

Legal aspects: Digital signatures

- Digital signatures might be equivalent to handwritten ones:
  - → Specific certificate required ("qualified certificate")
    - » Technically the same, but minimum requirements for keys, procedures, authentication of owner, ...
  - → Specific hardware required ("secure viewer", chipcards)
  - → Not for all areas possible:
    - » "Higher" forms: E.g. notarization
    - » Family law and law of succession with form requirements
      - – Not electronic legacy!
    - » Declarations of surety by private persons
      - – Very dangerous things; manual act as "warning"
- Additional: Legal presumption for the content of the message, as long as the signature is correct
  - → That the signer said this, not that the content is correct!
- Importance: Between companies, E-Government

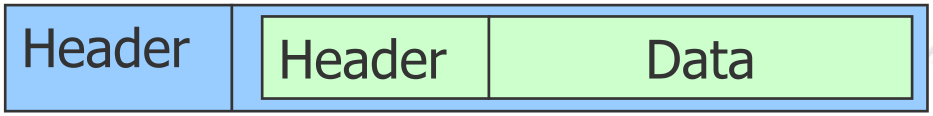
VPNs

- VPN = Virtual Private Network
  - → A private network across a public medium
    - » Replacement of leased lines by encrypted/authenticated communication using the "ordinary" and common internet
  - → Especially important for mobile workers
    - » Always "virtually" located in the home network with all possibilities there (telephone, server access, etc.)
  - → Other application: Branch offices
    - » The Internet serves as the company backbone
- Obviates the need for a firewall
  - → Everything is encrpyted and authenticated
    - » Filtering would be impossible anyway
  - → But does NOT secure against "internal" attacks
    - » Internet is protected against, Intranet must be secure itself!
- Transparent for users (apart from establishing perhaps)!

- Obviates the need for a firewall
  - → Everything is encrpyted and authenticated
    - » Filtering would be impossible anyway
  - → But does NOT secure against "internal" attacks
    - » Internet is protected against, Intranet must be secure itself!
  - → Does NOT apply in "split" configurations
    - » Some traffic is sent through the tunnel (e.g. file server access)
    - » Some traffic is sent to the Internet (e.g. webbrowser)
- Disadvantages:
  - → Traffic can no longer be compressed
    - » Must happen before or at the tunnel endpoint
  - → No QoS (as often available with leased lines)
  - → Sometimes difficult to set up
  - → Powerful hardware needed for encrypting larger bandwidth

Tunneling

- Transport of packets from one protocol over another one
  - → Done by "packing" the original packets into new packtes of the outer protocol
  - → Transparent to upper layers
  - → Can also be used identically, i.e. packing IP into IP
    - » This is used e.g. by IPSec (+ additional information)
  - → Reasons: not suported (e.g. IPX), unroutable (NetBUI), illegal addressing (192.168.?.?), ...
- Source: Encapsulation
  - → Adding a new header (and perhaps a new trailer)



- Destination: Extraction
  - → Passes the content on in some locally defined manner

PPTP

- Point to Point Tunneling Protocol (version of PPP)
  - → Supports IP, IPX, NetBUI
- Client-Server-Model
- Rather easy to set up (and client is integrated into windows)
- Can be transported across NAT (with additional software, ...)
- Client authentication by username/password
  - → Several old and very insecure algorithms/protocols exist!
  - → Server is not authenticated in most implementations!
- Encryption of content optional
- No key management protocols
  - → Key remains the same for the whole communication!
- No integrity check for packets

IPSec

- IP Security Protocol (intended for IPv6, but used also for v4)
  - → No full layer 3 support: No multicasts, static routing only
    - » Static routing: No dynamically "redirecting" the tunnel; the encapsulated packets can be routed in any way
- Allows (and implementations support) a multitude of authentication, encryption, hash and compression protocols
- Mutual authentication of packets and endpoints
- Key exchange protocol
  - → New key for each tunnel and regularly changing keys
- Encryption of complete content
- Supports IP only

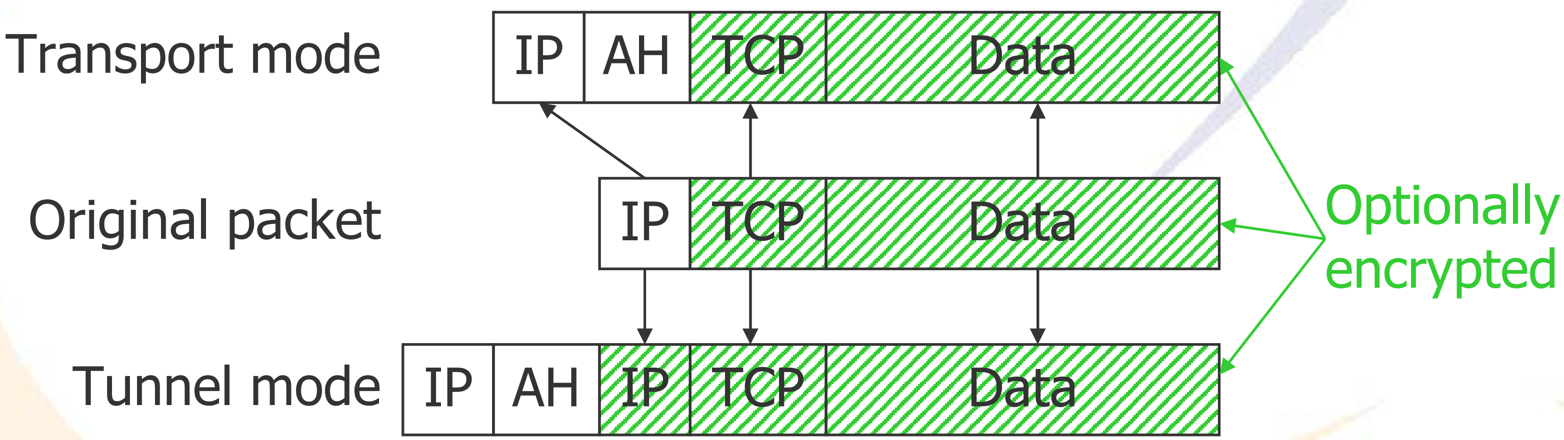
IPSec vs. PPTP

- PPTP is easier to set up
  - → Username and password, no certificates, CAs, CRLs needed
    - » IPSec also supports Pre-Shared-Keys; wizards for setup sometimes availbale (depending on vendor)
- IPSec is much more secure
  - → Keys exchanged during usage
  - → Algorithms supported are more secure
- PPTP can go over NAT
  - → This might be good or bad, however!
- IPSec implementations have fewer weaknesses
  - → Microsoft PPTP implementation has (still) many weaknesses
- IPSec supports IP only

When possible, use IPSec!

IPSec: Modes

- Transport mode: Only data in encrypted
  - → Header remains publicly visible!
  - → Additional small header added
  - → Used for secure connections (Host-Host communication)
    - » Rather rarely used
- Tunnel mode: Complete packet is encrypted
  - → Completely new IP header added (in addition to ESP header)
  - → Used for VPNs (LAN-LAN tunnel)



IPSec

- AH protocol: Authentication Header
  - → Cryptographic checksum over packet
    - » No modification on transport, identified peer was sender
  - → Includes the complete header ⇒ NAT impossible
- ESP protocol: Encapsulation Security Payload
  - → Encryption of whole packet
  - → DES, MD5, SHA must be supported, anything als can be
- IPComp: Compression protocol
  - → To be used optionally before encryption
- IKE: Internet Key Exchange
    - » Optional protocol (⇒ manual configuration otherwise)
  - → Agreeing on a shared secret for authentication/encryption
  - → Uses e.g. Diffie-Hellman or master keys

IPSec limitations

- IPSec does not work with dynamic IPs
  - → One fixed and one dynamic is still possible, as long as the dynamic side is the initiator
  - → If both sides have dynamic IPs, DynDNS (and software support) is necessary
    - » IPSec works on the level of IP, therefore it only understands IP addresses; Name→IP address resolution must be external!
- No NAT: Use IPSec "afterwards" (e.g. router appliance)
  - → Or directly on the same router (first NAT, then IPSec)
    - » But then its probably better to use an IPSec LAN-LAN tunnel!
- Very complex: Small errors might lead to working solution, but reduce security significantly
- Interoperability sometimes lacking, but improving
- Debugging is difficult: Everything is encrypted, ...!

Other security aspects

- Using VPNs, SSL, digital signatures is nice (and necessary!), but does not solve all problems:
  - → Denial of Service
  - → Endpoint security (storing those creditcard numbers)
  - → Users: Security is cumbersome and therefore circumvented
  - → Cryptography is only as secure as the key storage
    - » Who uses really good passwords/passphrases?
    - » How is the "backup" of the password organized (bank safe)?
  - → Physical security? Social engineering? Internal attacks?
- But security is also not self-serving:
  - → Value of goods to be secured vs. cost of protection

Holistic view required for encompassing security!