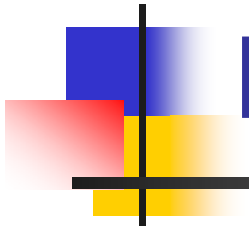
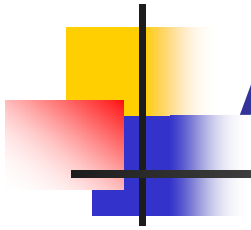


Network Services

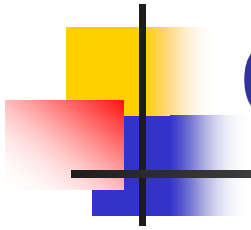
SSL/TLS, SSH



Agenda

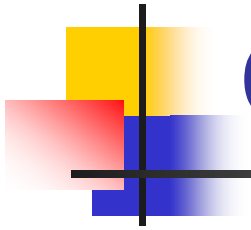
- Symmetric/Asymmetric Cryptopgraphy

Symmetric/Secret Key Cryptography

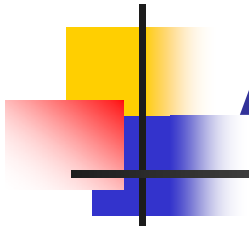


- Sender A encrypts a message m with a Key k
  - Result is e(m)
- Receiver B decrypts message e(m) with same Key k
- Key k has to be known by A+B
- Application of Key on message is a mathematical function
  - Encryption and decryption inverse functions

Asymmetric/Public Key Cryptography



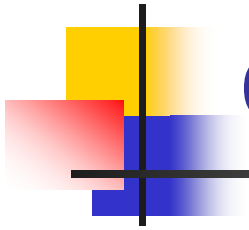
- Key consists of private part + public part
- Sender A encrypts a message m with a public key part pu
  - Result is also e(m)
- Receiver B decrypts message e(m) with private key part priv
- Public key known by anybody (also A)
- Private key ONLY known by B
- Encryption is application of public key
- Decryption is application of private key



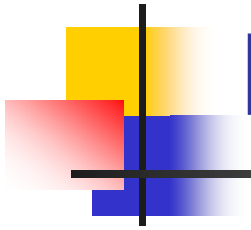
Asymmetric Signatures

- Signation done by encrypting message with private key
  - Results in Signature
  - Whole message consists of message + signature
- Verification done by decrypting message with public key
- Usually hash over message contents+header is used as signature
- Digitnal Signature Algorithm (DSA)

Combining secret and public key cryptography

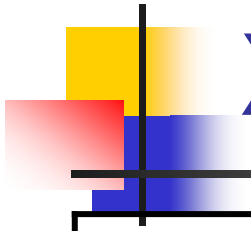


- Asymmetric algorithms
  - Rather slow
  - Used for key exchange of symmetric cryptographic algorithms
  - Key requires structure (private+public)
    - Based on large prime numbers
  - RSA, El Gamal
  - Diffie-Hellman Key exchange algorithm
- Symmetric
  - Rather fast
  - Key Usually unstructured (eg. 128bit random number)
  - DES,3DES,AES (Rindjael)



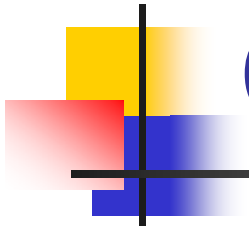
Public Key Certificates

- Critical that public key is not forged
- **Public Key Certificates**
  - Identify subjects by subjects names
    - Usually identifies a host
  - Key information about a subject (usually public key)
  - Issued by a **trusted organization** (certification authority - CA)



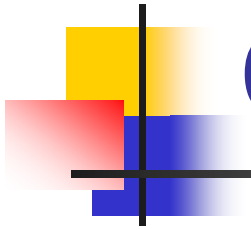
X.509 Certificate

Field entry	Description	Example
Version	Version of X.509 Standard	3
Serial Number	Assigned by CA	12345678
Algorithm Identifier	MD5 hash and RSA signing	RSA
Issuer	Cert. Authority	VeriSign
Period of Validity	Time When valid	
Subject	Describes invididum who ones the certificate	Country Austria **Common Name NWS-TUWien**
Subject's public key		RSA 0x308188…
Extensions	Vendor specific	
Signature	Issuer creates signature with its private key over certificate	0x4C2170…



Certification Authorities

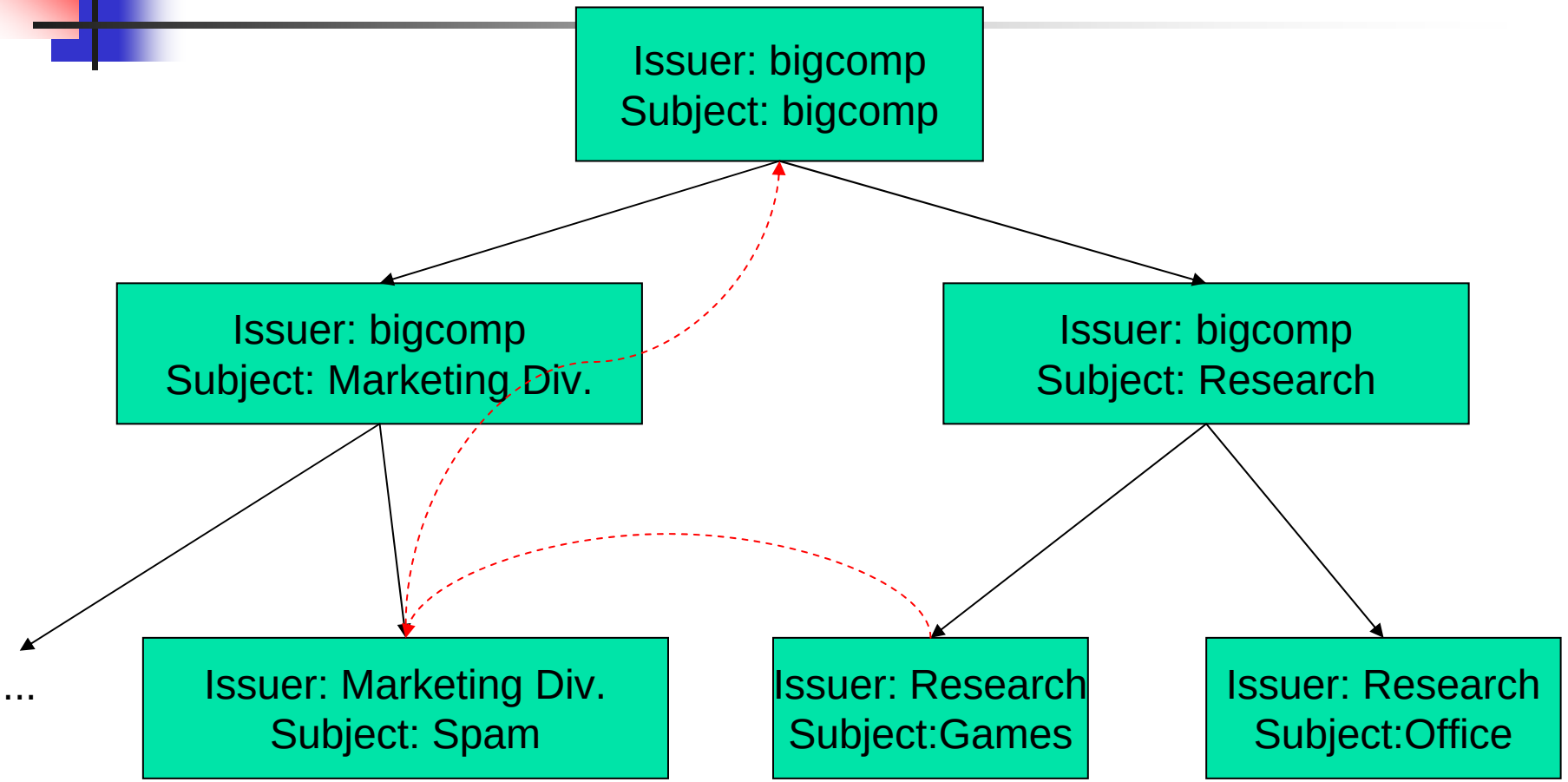
- Private authorities
  - Generate certifications strictly for their own users
    - Eg. Company for their employees' computer
  - Systems outside the company need/should not accept certificates
- Public authorities
  - Issues certificates to the general public
  - May prove identity by certificates themselves
    - Issuer and subject one and the same

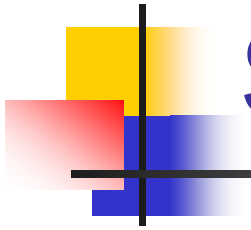


Certificates

- Validity of certificate authorities
  - Depends on browser manufactorers
    - Recognize certificates from important certificate authorities
  - Certificate Revocation Lists
    - Certificates that are no longer valid
    - No standardized way to check these lists
- Hierarchies of certificate authorities
  - Subsidiary authorities assigned by certificate authorities
    - Build a trust hierarchy
  - Not necessary to identify all identities itself
  - Not required that all parties trust all certificate authorities
    - Recursive resolution
    - Somewhere authority that is trusted must be met

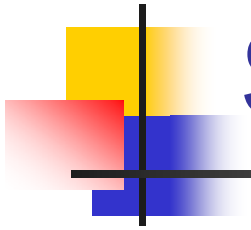
Certificate Hierarchy





SSL/TLS

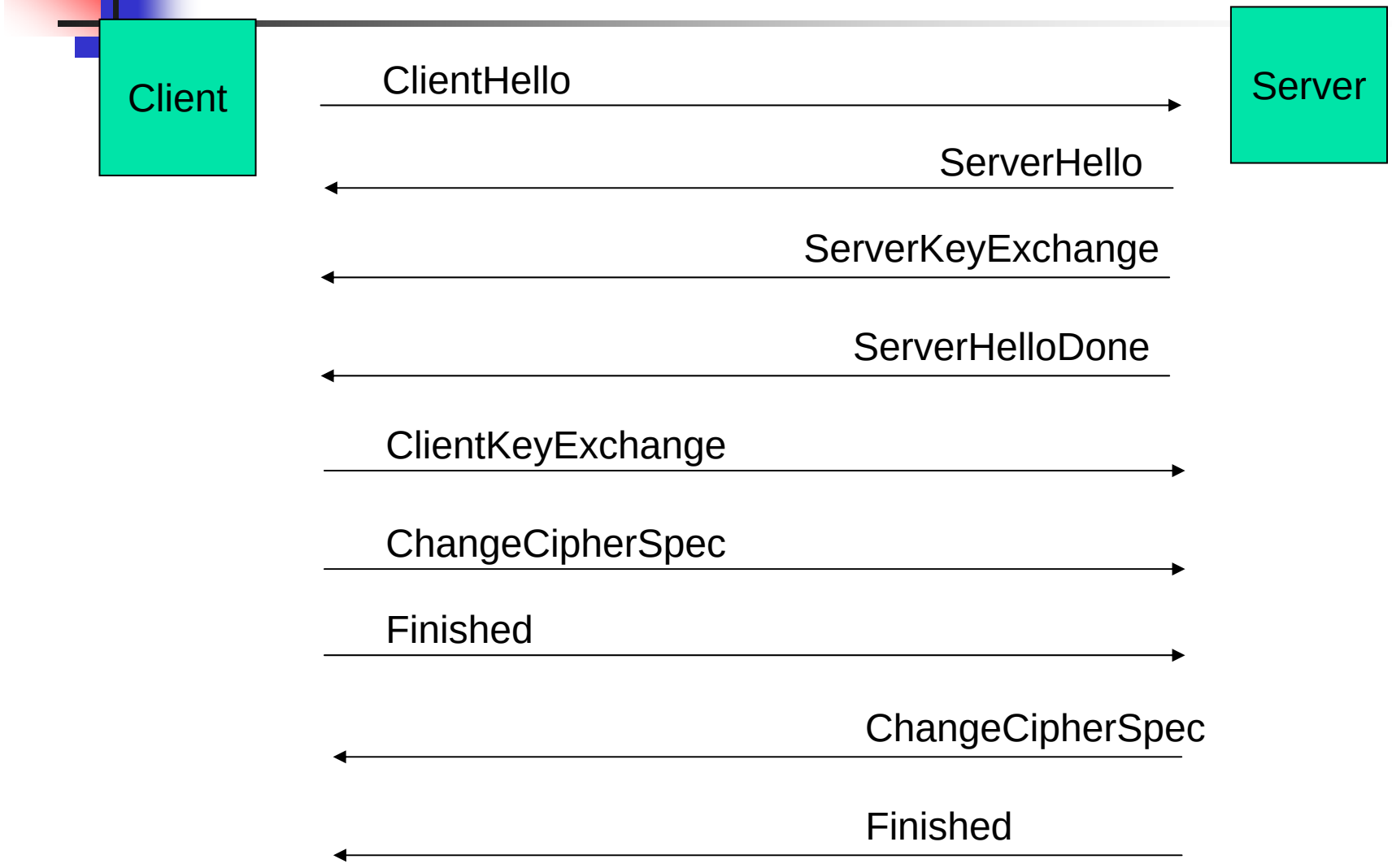
- Secure Sockets Layer (SSL)
  - Introduced by Netscape (SSL 1.0 1994)
  - Netscape Navigator ships with SSL 2.0 late 1994
- Transport Layer Security (TLS) – RFC 2246
  - TLS is successor of SSL
  - Standardized by IETF
  - Published in 1999
  - Principally new version of SSL
- Used in many applications
  - Primarily in Web applications (HTTP)
  - Also used in EMail

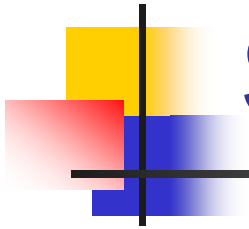


SSL

- Separate protocol for security
  - Between Application specific protocol and TCP protocol
  - Advantage: arbitrary applications may use SSL/TLS
- Different SSL protocols
  - Encryption
  - Authentication of server
  - Authentication of client
  - Continuation of previous negotiated session
- Different cipher suites
  - RSA, DH
  - DES,3DES,RC4
  - SHA,MD5

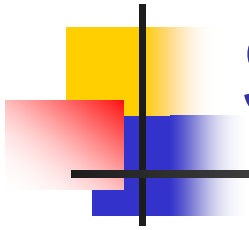
SSL – Negotiation of Encrypted Commands





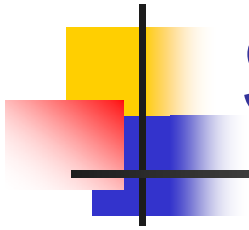
SSL Commands / 1

- ClientHello
  - Starts SSL communication between 2 parties
- Parameter
  - Version - Sends hightest version number SSL client supports (currently 3.0 for SSL, 3.1 for TLS)
  - RandomNumber - Sends a random number (includes date+time)
  - SessionID – empty in this operation mode
  - CypherSuites – cryptographic services client supports
    - Algorithms, key sizes
  - CompressionMethods
    - Must be applied before encryption
    - Not included in SSL



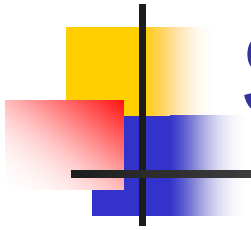
SSL Commands / 2

- ServerHello
  - Version - of SSL protocol used
  - RandomNumber - chosen by server
  - SessionID – calculated by the server
  - CypherSuite – Cryptographic parameters selected by the server from the client's previous CypherSuites parameter
  - CompressionMethod



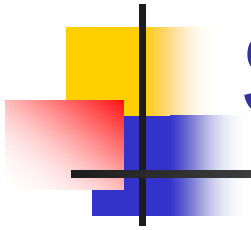
SSL Commands / 3

- ServerKeyExchange
  - Transmits public key information itself
  - Example: algorithm=RSA,
    - Sends the public key
      - (modulus and public exponent of server's public key)
  - No encryption applied here
- ServerHelloDone
  - Server has finished its negotiation



SSL Commands / 4

- ClientKeyExchange
  - Transmits Client keys information
    - Key for Symmetric encryption algorithms
    - Different keys for sending/receiving
    - Client creates keys
  - Encrypted with Server's public key
  - Completes the preliminary SSL negotation
- ChangeCipherSpec
  - Special command that "Activates" Security Services
  - "changes algorithms & keys"
- Finished
  - Message is already encrypted, has to be decrypted by other party
  - Sends key information
  - Sends all previous SSL handshake messages



SSL Write/Read state

- Client and Server maintain
  - Information about security services used
    - Specific Symmetric encryption algorithm
    - Specific Message integrity algorithm (Message authenthication Code)
    - Specific key material for those algorithms
      - Different for each direction!
  - Active and Pending fields for write+read state
    - Write fields for data the client/server sends
    - Read fields for date the client/server receives
    - Can only be activated when above (pending) information is complete
      - Activated by ChangeCipherSpec
    - Other Client and Server messages fills only Pending fields
- Literature
  - Stephen Thomas: "SSL and TLS Essentials: Securing the Web"

Pending/Active states – Client 1

	Write		Read	
	Act	Pnd	Act	Pnd
Encr	Null	?	Null	?
MAC	Null	?	Null	?
key	null	?	null	?

ClientHello

(Active state to null=no security , pending states are unknown)

	Write		Read	
	Act	Pnd	Act	Pnd
Encr	null	DES	Null	DES
MAC	Null	MD5	Null	MD5
key	null	?	Null	?

ServerHello

(Client knows algorithms server has selected)

ServerKeyExchange

ServerHelloDone

	Write		Read	
	Act	Pnd	Act	Pnd
Encr	null	DES	Null	DES
MAC	Null	MD5	Null	MD5
key	null	xyz	Null	xxx

ClientKeyExchange

(pending Keys are created by client)

Pending/Active states – Client 2

	Write		Read	
	Act	Pnd	Act	Pnd
Encr	DES	?	Null	DES
MAC	MD5	?	Null	MD5
key	xyz	?	Null	xxx

ChangeCipherSpec

(switch Write/Send to Active)

Finished

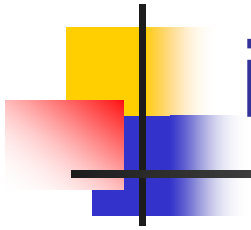
ChangeCipherSpec

(switch Read/Receive to Active)

	Write		Read	
	Act	Pnd	Act	Pnd
Encr	DES	?	DES	?
MAC	MD5	?	MD5	?
key	xyz	?	xxx	?

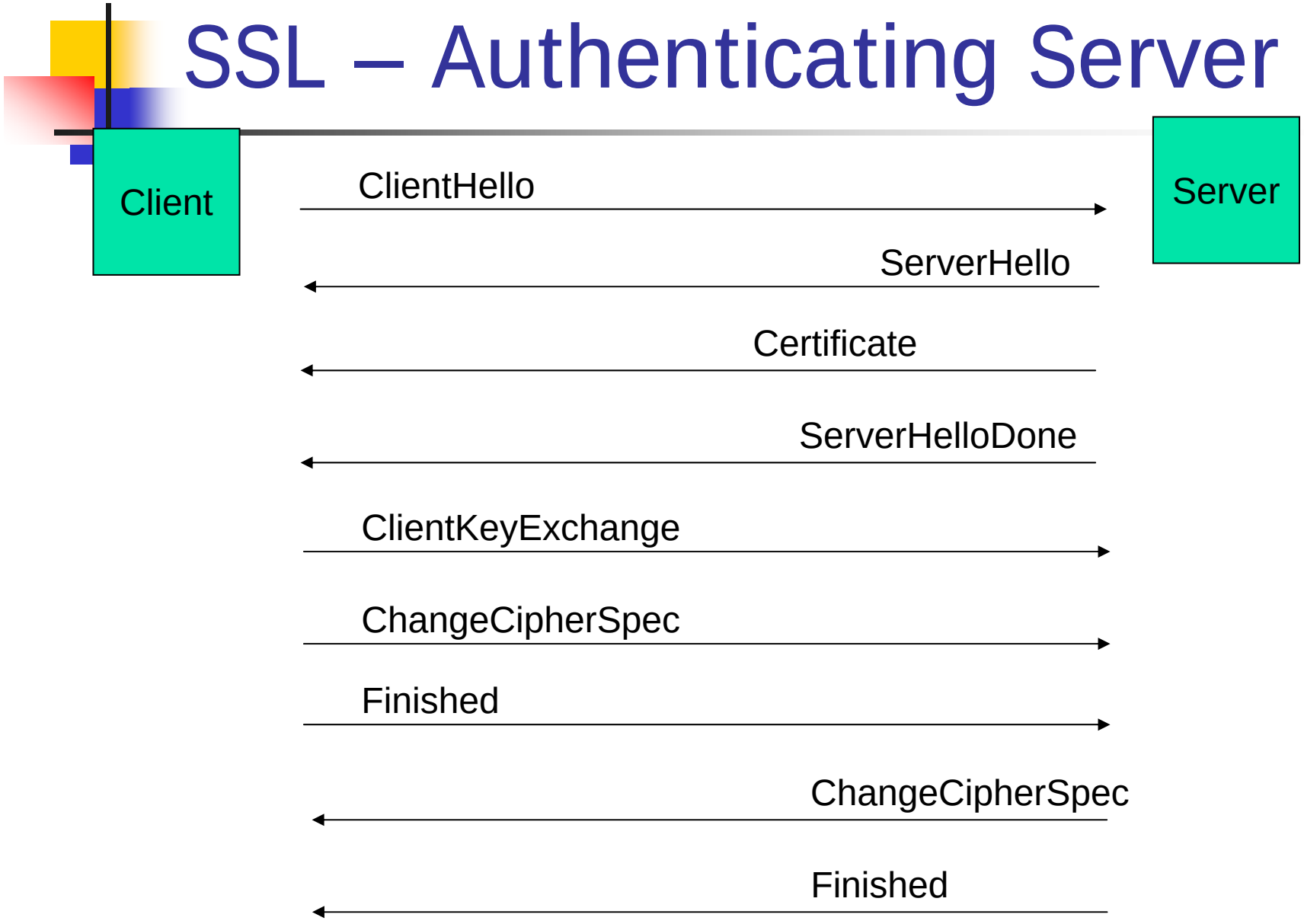
Finished

SSL – Authenticating Server's identity

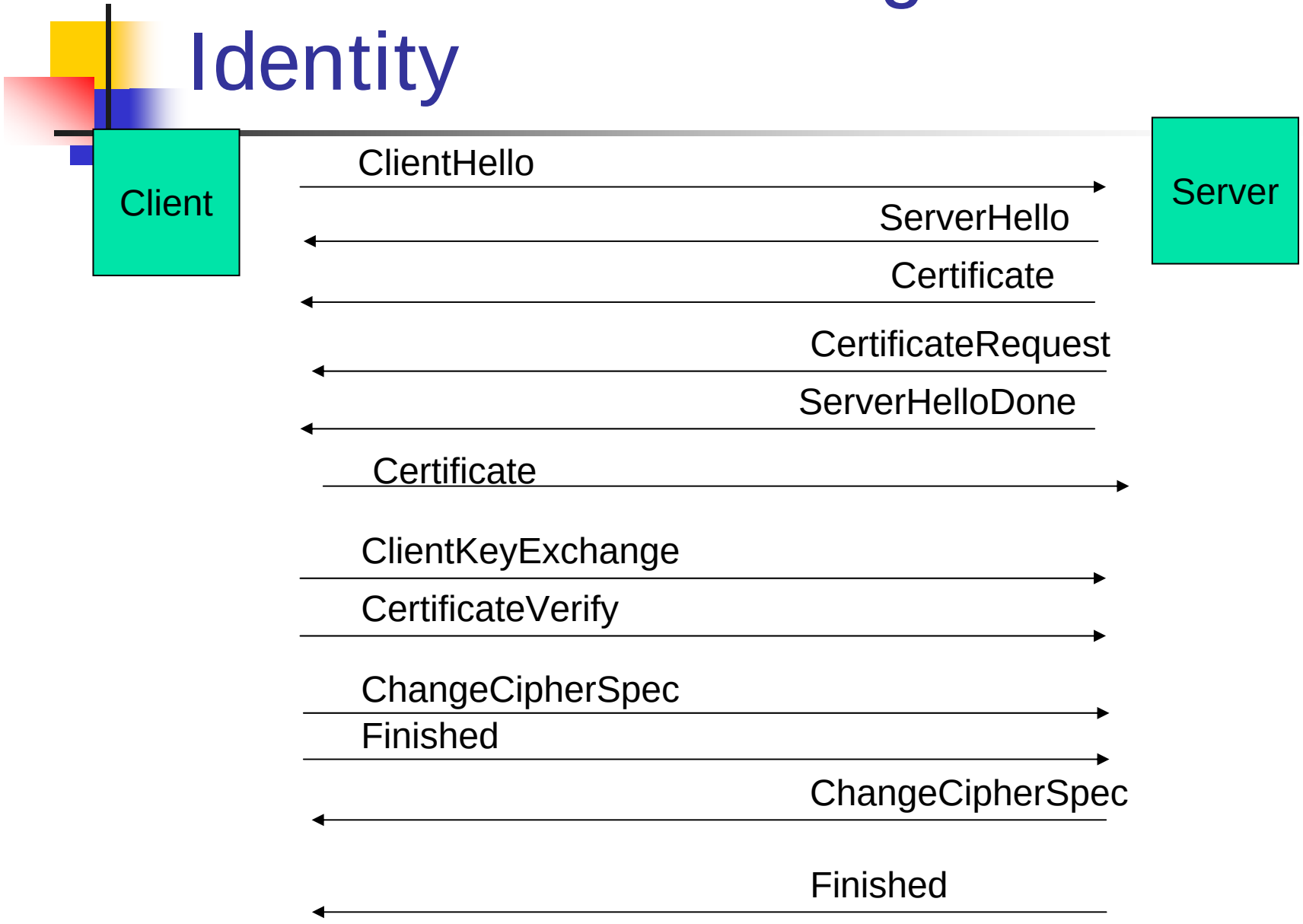


- Server sends certificate message
  - Certificate with Public key
- Client verifies validity of certificate
  - Certificate Signatures, Validity Times, Revocation Status
  - Checks domain name of web site with domain name stored in certificate (Subject)
    - Eg. Server located at "www.mydomain.org" and certificate valid only for www.otherdomain.org
  - Client's ClientKeyExchange uses public key in certificate
    - Sometimes another public key may be used
      - Example US Export restrictions (cryptographic key lengths)

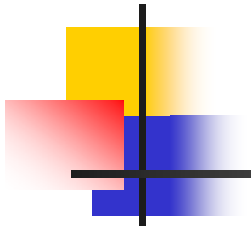
SSL – Authenticating Server



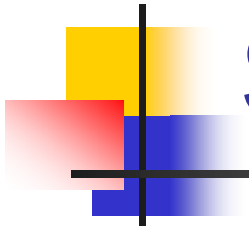
SSL – Authenticating Client's Identity



SSL – Authenticating Client's Identity

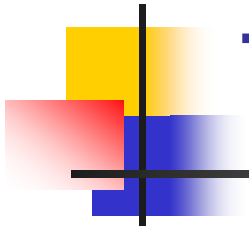


- Server wants to authenticate the Client's identity
  - Server indicates wish to authenticate Client's identity by sending a CertificateRequest message
  - Client sends its own Certificate within Certificate message
    - Client's public key within the certificate is used for signatures only – no encryption
  - Client proves that it posseses the certificate by submitting a CertificateVerify message
    - Encrypted with private key
    - Over key information + all previous SSL handshake messages exchanged by both systems



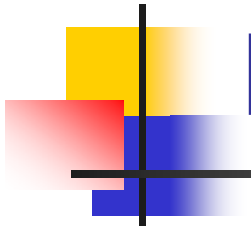
SSL - Limitations

- Protocol limitations
  - Requires connection-oriented transport protocol such as TCP
  - Does not support non-repudiation
- Tool limitations
  - Relies on other components such as cryptographic algorithms
- Environmental limitation
  - Security provided only on the transmission network
  - The path to the network and from the network is not secured



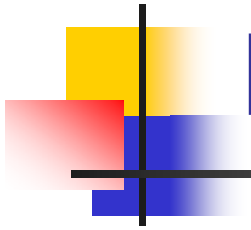
TLS – Differences to SSL

- Protocol version 3.1
- More procedures for potential and actual security alerts
  - 23 instead of 12
  - Eg. Certificate-Revoked
- Message authentication standardized
  - Uses H-MAC (hashed Message Authentication Code)
    - Combines (Sequence number,TLS protocol message type, TLS version, Message length, Message contents)
  - Instead of SSL combination of key information and application data
- More cipher suites



HTTPS

- HTTPS (HTTP over TLS) – RFC 2818
  - HTTP Client starts with sending TLS ClientHello
  - Standard Port 443
- Upgrading to TLS within HTTP/1.1 – RFC 2817
  - Allows secured and unsecured HTTP to share the same port
  - Client may send an HTTP/1.1 request with an "Upgrade: TLS/1.0" header field
    - Server may either respond with normal response or switch to secured TLS communication
  - If the Upgrade is mandatory the client must send an OPTIONS request with an Upgrade TLS/1.0 header field
  - Server may respond to normal request with with "426 Upgrade Required" response
    - The request requires secure communication



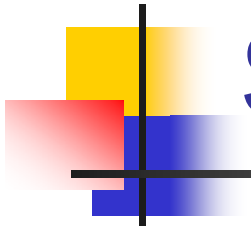
HTTPS / Example

Client:

OPTIONS * HTTP/1.1
Host: dsg.infosys.tuwien.ac.at
Upgrade: TLS/1.0
Connection: Upgrade

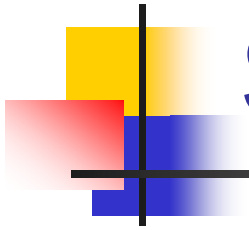
Server:

HTTP/1.1 101 Switching Protocols
Upgrade: TLS/1.0, HTTP/1.1
Connection: Upgrade



Secure Shell (SSH)

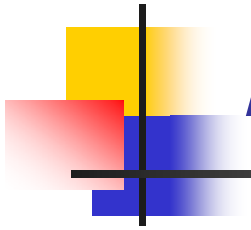
- RFCs 4250-4256, and others
- "Protocol for secure remote login and other secure network services over an insecure network"
- SSH Standard means for secure shell access on Unix machines
- Supports Automatic host key authentication
  - Clients that come from one particular HOST can automatically be authenticated



SSH Transport Layer Protocol

- Supports
  - Strong encryption
  - Server authentication
  - Integrity protection
  - May support compression
- Supports different algorithms
  - Key Exchange (eg. Diffie-Helmann)
    - Used to exchange keys between client / server
  - Server Host Key Algorithms (ssh-rsa,ssh-dss)
  - Encryption Algorithms (symmetric) (aes128, 3des,...)
    - Data encryption
  - Mac Algorithms (hmac-md5, mac-sha1, ...)
    - For generating message authentication code
  - Compression Algorithms (zlib)
- Algorithms negotiated during Key Exchange Messages

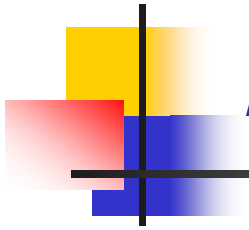
SSH Transport Layer Protocol / 1



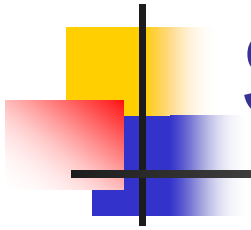
- Usually over TCP/IP, Standard Port 22
- Client initiates connection to server

1. Server responds with identification string
   - Example: Server sends SSH-2.0-OpenSSH_3.9p1
2. Client sends also identification string
3. Server sends Key Exchange Init
   - Includes supported algorithms
4. Client sends also Key Exchange Init
   - Includes supported algorithms

SSH Transport Layer Protocol / 2

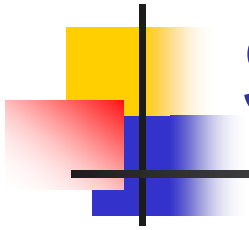


5. Client sends Key Exchange Message
   - Eg. Diffie-Hellman GEX Request
6. Server replies Key Exchange Reply
7. Client sends Diffie-Hellman GEX Init
8. Server sends Diffie-Hellman GEX Reply
9. Client sends "New Keys"
   - From this point on all communication is encrypted



SSH Channels

- Channels are means for communicating with SSH
  - Each channels has a specific number
  - Multiple channels possible at the same time
  - SSH_MSG_CHANNEL_REQUEST
- Channels
  - X11 Forwarding ("x11" parameter)
  - Starting a remote Command ("exec" command)
  - Starting a remote shell ("shell")



Summary

- Most important cryptography
  - SSL/TLS
  - SSH