Security of IP networks

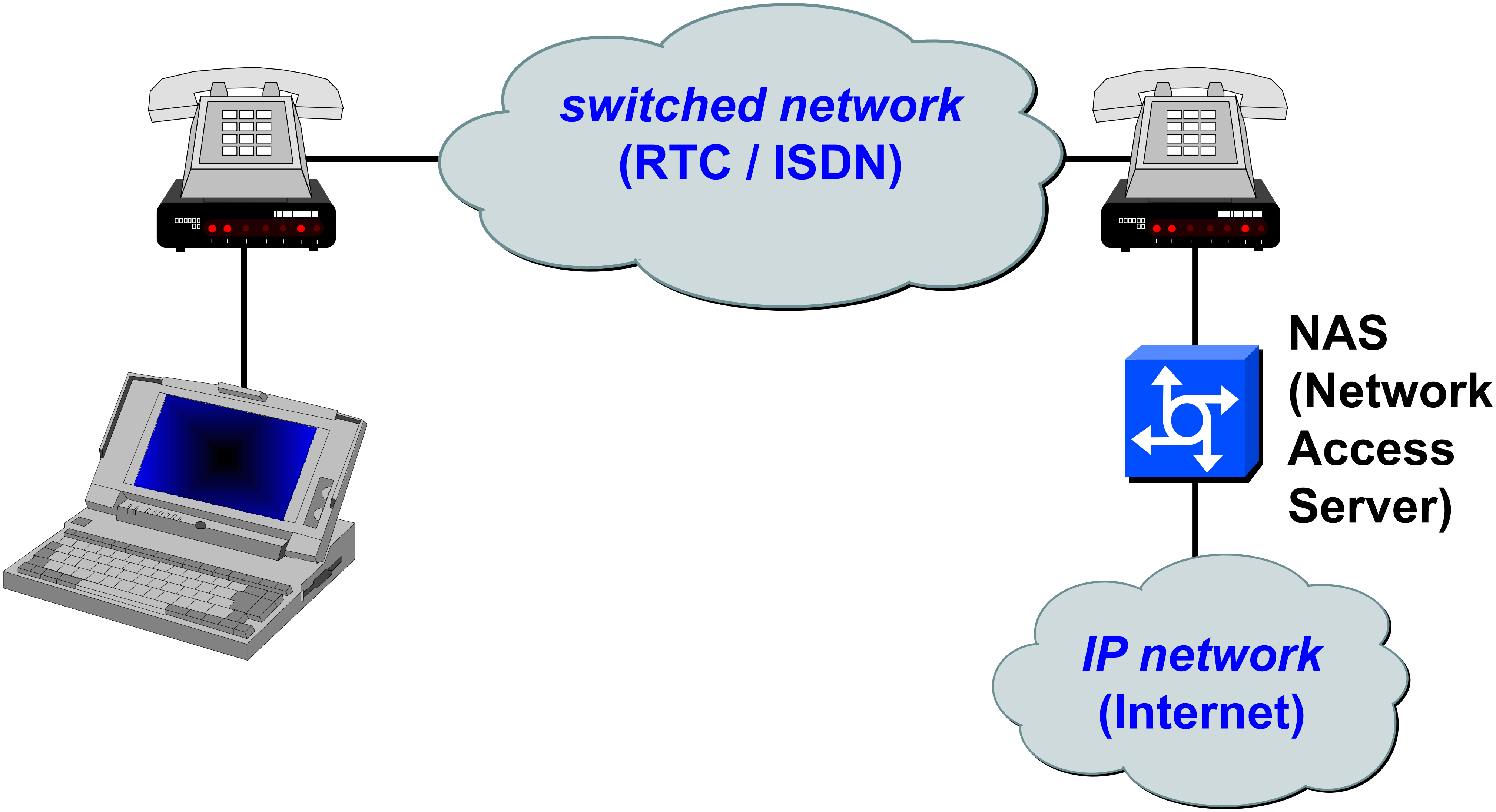
Antonio Lioy

< lioy @ polito.it >

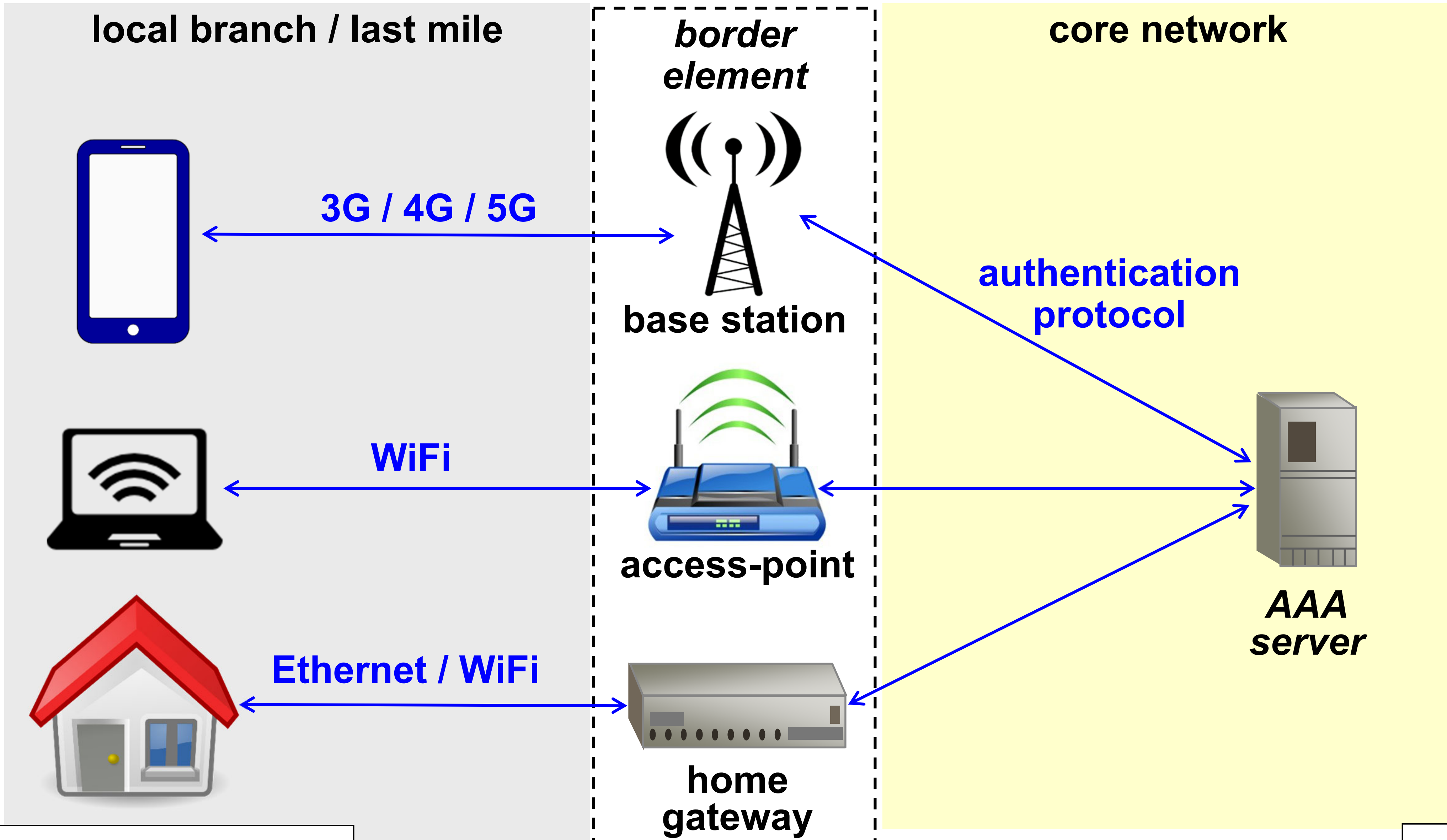
Politecnico di Torino

Dip. Automatica e Informatica

Remote access via dial-up lines



Network access (modern way)



Authentication of PPP channels

- **PPP (Point-to-Point Protocol) is able to encapsulate network packets (L3, e.g. IP) and carry them over a point-to-point link**
  - physical (e.g. PSTN, ISDN)
  - virtual L2 (e.g. xDSL with PPPoE)
  - virtual L3 (e.g. L2TP over UDP/IP)
- **activated in three sequential steps:**
  - LCP (Link Control Protocol)
    - establishing, configuring, and testing the L2 connection
    - can negotiate also authN protocol and algorithm
  - authentication (optional; PAP, CHAP, or EAP)
  - L3 encapsulation via various NCPs (Network Control Protocols)
    - e.g. IPCP (IP Control Protocol) for IP packets

Authentication of network access

- **protocols used not only for PPP**
- **PAP**
  - Password Authentication Protocol
  - user password sent in clear
- **CHAP**
  - Challenge Handshake Authentication Protocol
  - symmetric challenge-response (based on user password)
- **EAP**
  - Extensible Authentication Protocol
  - it's an authentication framework
  - … to use external techniques (e.g. challenge, OTP, TLS)

LCP Authentication-Protocol Configuration Option

- **this option contains:**
  - Type (8bit) = option type
  - Length (8bit) = length of option in bytes
  - Authentication Protocol (16bit) = protocol identifier
  - [Algorithm (8bit)] = algorithm identifier (needed when a protocol supports various ones)
- **for PAP:**
  - Type=3, Length=4, Protocol=0xC023
- **for CHAP:**
  - Type=3, Length=5, Protocol=0xC223, Algorithm=5 (for MD5)

PAP

- Password Authentication Protocol
- RFC-1334 "PPP Authentication Protocols" (Oct 1992)
  - note: defines also initial version of CHAP
- user-id and password sent in clear between Peer and Authenticator
- authentication only once when the channel is created
- very dangerous!

PAP: 2-way handshake protocol

- **(Peer > Authenticator) Authenticate-Request (code=1)**
  - Code (8bit) + Identifier (8bit) + Length (16bit)
  - Peer-ID Length (8bit) + Peer-ID (0-255B)
  - Passwd-Length (8bit) + Password (0-255B)
- **(Authenticator > Peer) Authenticate-Response (code=2, 3)**
  - Code (8bit) + Identifier (8bit) + Length (16bit)
  - Msg-Length (8bit) + Message (0-255B)
  - code=2 (ACK), code=3 (NAK)
- **Identifier needed to match Request and Response**
- **Authenticate-Request or -Response may be lost**
  … so, Authenticator MUST permit multiple requests

CHAP

- **RFC-1994 “PPP Challenge Handshake Authentication Protocol (CHAP)” (Aug 1996)**
- **symmetric challenge (password-based)**
  - initial challenge compulsory (at channel creation)
  - authentication request optionally repeated (with a different challenge) during transmission – decision taken by the NAS
  - challenge MUST be a nonce
- **the Authenticators that support both PAP and CHAP *must* offer CHAP first**

CHAP: 3-way handshake protocol

- **(Authenticator > Peer) Challenge (code=1)**
  - Code (8bit) + Identifier (8bit) + Length (16bit)
  - Challenge-Size (8bit) + Challenge-Value (0-255B)
- **(Peer > Authenticator) Response (code=2)**
  - Code (8bit) + Identifier (8bit) + Length (16bit)
  - Response-Size (8bit) + Response-Value (0-255B)
- **(Authenticator > Peer) Result (code= 3 Success, 4 Failure)**
  - Code (8bit) + Identifier (8bit) + Length (16bit)
- **Response-Value = md5 (Identifier || pwd || Challenge-Value)**
- **Identifier needed to match Request and Response**
- **Challenge or Response may be lost … so, Authenticator MUST resend Challenge if no Response (until retry limit)**

MS-CHAP

- **MS-CHAPv1**
  - RFC 2433 "Microsoft PPP CHAP Extensions" (oct.1998)
  - dropped by MS starting with Windows Vista
- **MS-CHAPv2**
  - RFC 2759 "Microsoft PPP CHAP Extensions, v2" (jan.2000)
  - dropped by MS starting with Win11 22H2
- **LCP negotiates CHAP algorithm 0x80 (v1) or 0x81 (v2) for option 3 (Authentication Protocol)**
- **it's a MS-specific implementation of the CHAP concepts**
  - yet supported by many other vendors (e.g. CISCO)

MS-CHAP: extensions over CHAP

- **similar principles, different protocol**
- **common (v1 and v2):**
  - authenticator-controlled password change
  - authenticator-controlled authentication retry
  - specific failure codes
- **MS-CHAPv2 provides mutual authentication:**
  - by piggybacking a peer challenge on the Response packet
  - plus an authenticator response on the Success packet
- **each peer must know the plaintext password, or an MD4 hash of the password (thus not compatible with most password storage formats)**

The MS-CHAPv2 protocol

[PEER] [AUTHENTICATOR]

Hello >>

<< (Server Challenge, 16 byte) SC

(Client Challenge, 16 byte) CC

(Challenge-Hash) H = sha1(SC || CC || username)[0…7]

(NT-Hash) K = md4(password)

(Challenge-Response) R = des(K[0…6],H) || des(K[7…13],H) || des(K[14…20],H)

R1 *R2* *R3*

R + H + username >>

decrypt R and verify if the result matches H

(NT-Hash-Hash) NHH = md4(md4(password))

(Digest) D = sha1(NHH || R || M1)

(Authentication-Response) A = sha1(D || H || M2)

<< A

compute A' and verify if it matches A

M1 = `"Magic server to client signing constant"`

M2 = `"Pad to make it do more than one iteration"`

MS-CHAPv2: an attack

- **we have a known ciphertext-plaintext pair, R and H**
- **we need to find the three keys, K[0-6] K[7-13] K[14-20]**
- **brute-force the pwd? too much time …**
- **brute force the keys? 2^56 + 2^56 + 2^56 operations**
- **but K is just 128 bit (MD4 output) i.e. 16 B**
  - so K[14-20] has got only TWO bytes of K[14-15] padded with 0
  - we need 2^16 to find K[14-20]
- **to find K[0-6] and K[7-13] we perform only 2^56 operations and then compare the result with R1 and R2**
- **divide-and-conquer ~ 2^56 ops (<=23 hours with DES FPGA)**
- **conclusion: MS-CHAPv2 should NEVER be used anymore**
  - finally cancelled in Win11 22H2

EAP

- **RFC-3748 (extended by RFC-5247) “PPP Extensible Authentication Protocol (EAP)”**
- **a flexible L2 authentication framework**
- **authentication predefined mechanisms:**
  - MD5-challenge (similar to CHAP)
  - OTP
  - generic token card
- **other mechanisms may be added:**
  - RFC-2716 “PPP EAP TLS authentication protocol”
  - RFC-3579 “RADIUS support for EAP”

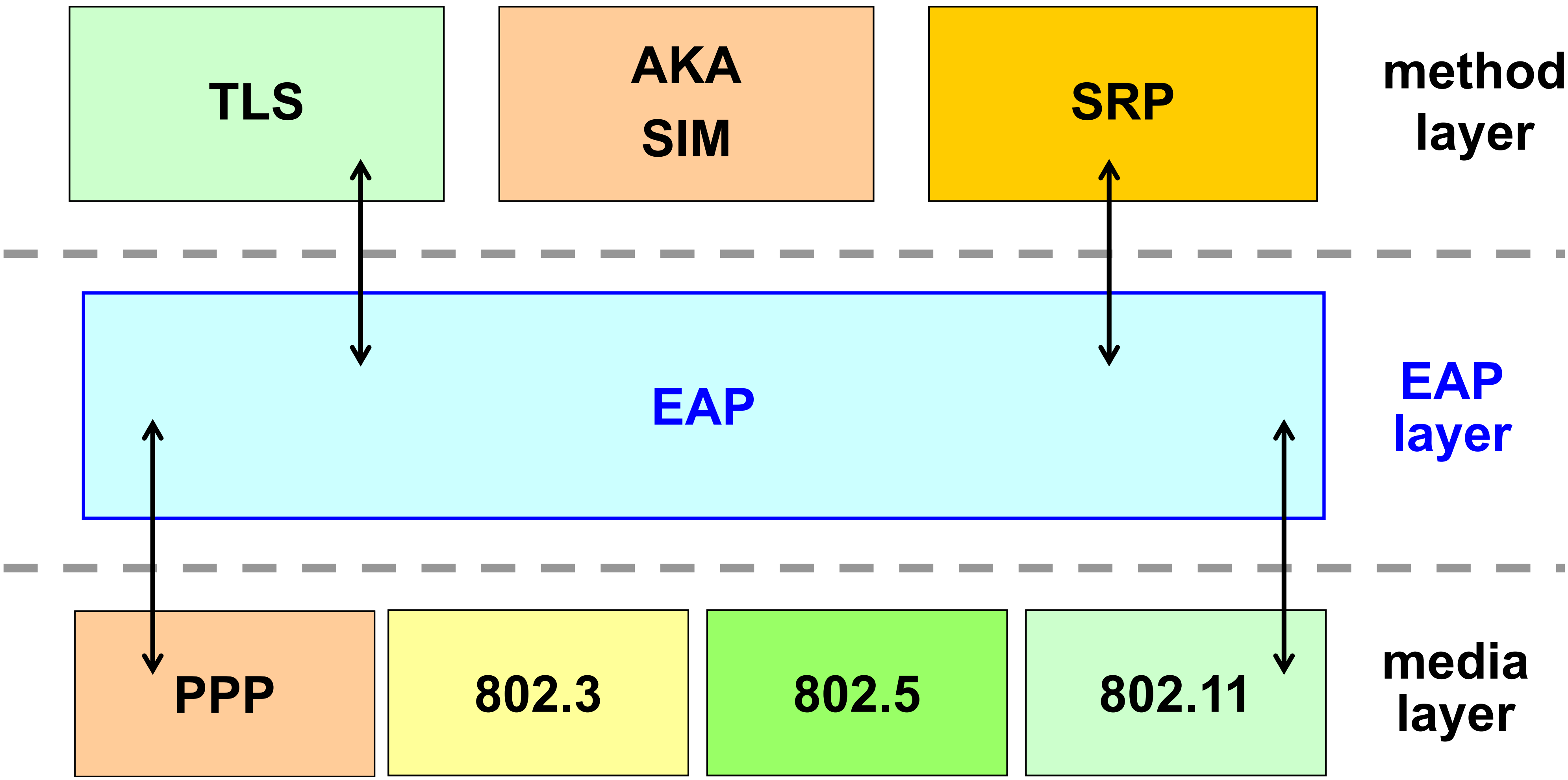
EAP - encapsulation

- **authentication data are transported via a specific EAP encapsulation protocol**
  - because L3 packets are not yet available …
- **features of EAP encapsulation:**
  - independent of IP
    - supports any link layer (e.g. PPP, 802, …)
  - explicit ACK/NAK (no windowing)
    - assumes no reordering (PPP guarantees ordering, UDP and raw IP do not!)
  - retransmission (max 3-5 retransmissions)
  - no fragmentation (must be provided by EAP methods for a payload greater than the minimum EAP MTU)

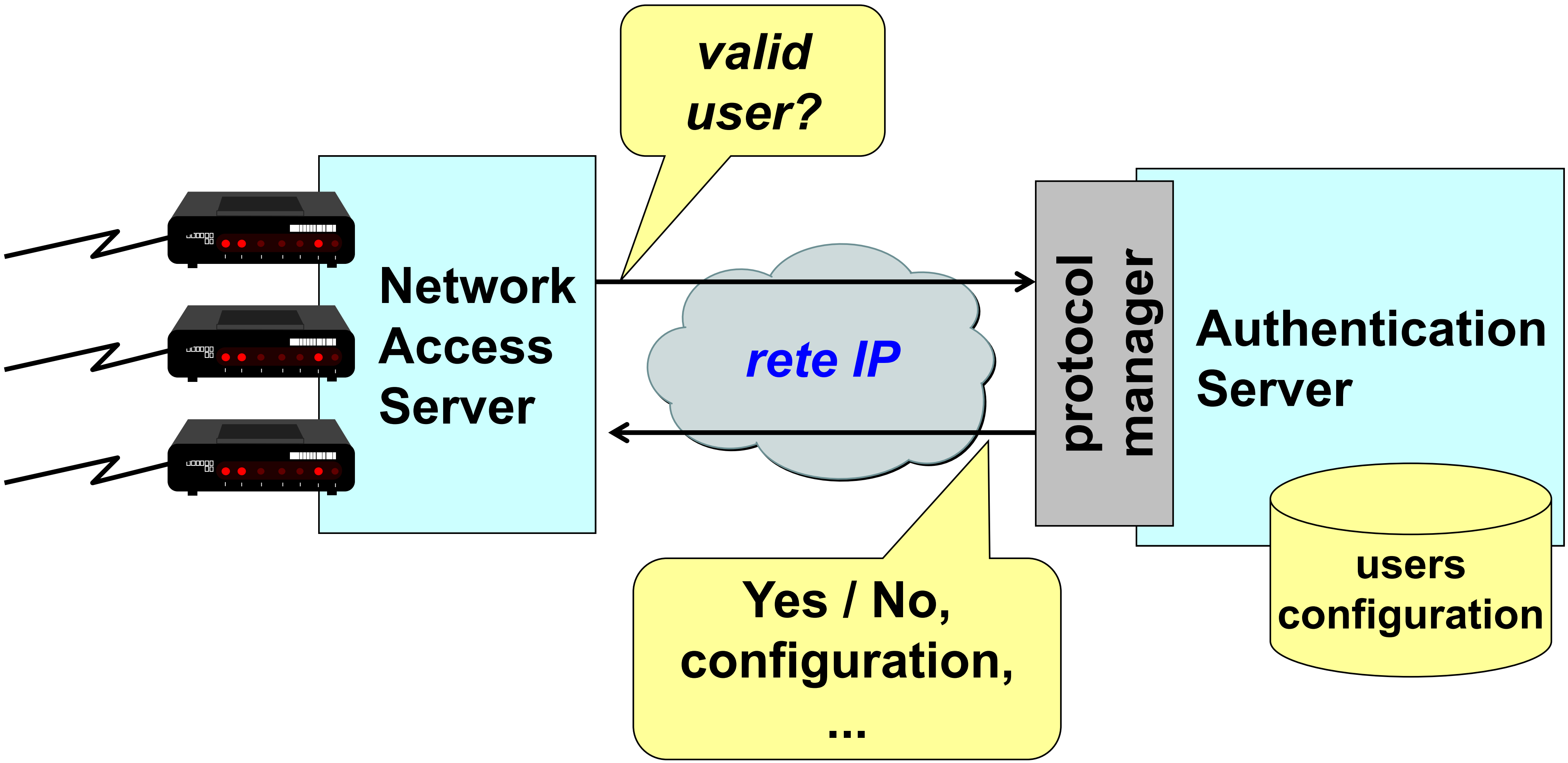
EAP

- **the link is not assumed to be physically secure**
  - EAP methods must provide security on their own
- **some EAP methods:**
  - EAP-TLS (RFC-5216) – TLS mutual authentication
  - EAP-MD5 (RFC-3748) – only EAP client authentication
  - EAP-TTLS = tunnelled TLS (to operate any authentication method protected by TLS, e.g. PAP, CHAP)
  - PEAP = TLS tunnel to protect an EAP method
  - EAP-SRP (Secure Remote Password)
  - GSS_API (includes Kerberos)
  - AKA-SIM (RFC-4186, RFC-4187)

EAP - architecture



Authentication for network access



AAA

- **the NAS manufacturers claim that security needs three functions:**
  - *Authentication* – entity's identity is authenticated based on credentials (e.g. password, OTP)
  - *Authorization* – determining whether an entity is authorized to perform a given activity or gain access to resources/services
  - *Accounting* – tracking network resource usage for audit support, capacity analysis or cost billing
- **the AS performs exactly these three functions talking with one or more NAS via one or more protocols**

Network authentication protocols

- **RADIUS**
  - the de-facto standard
  - proxy towards other AS
- **DIAMETER**
  - evolution of RADIUS
  - emphasis on roaming among different ISP
  - takes care of security (IPsec, TLS)
- **TACACS+ (TACACS, XTACACS)**
  - originally technically better than RADIUS, achieved smaller acceptance because it was a proprietary solution (Cisco)

RADIUS

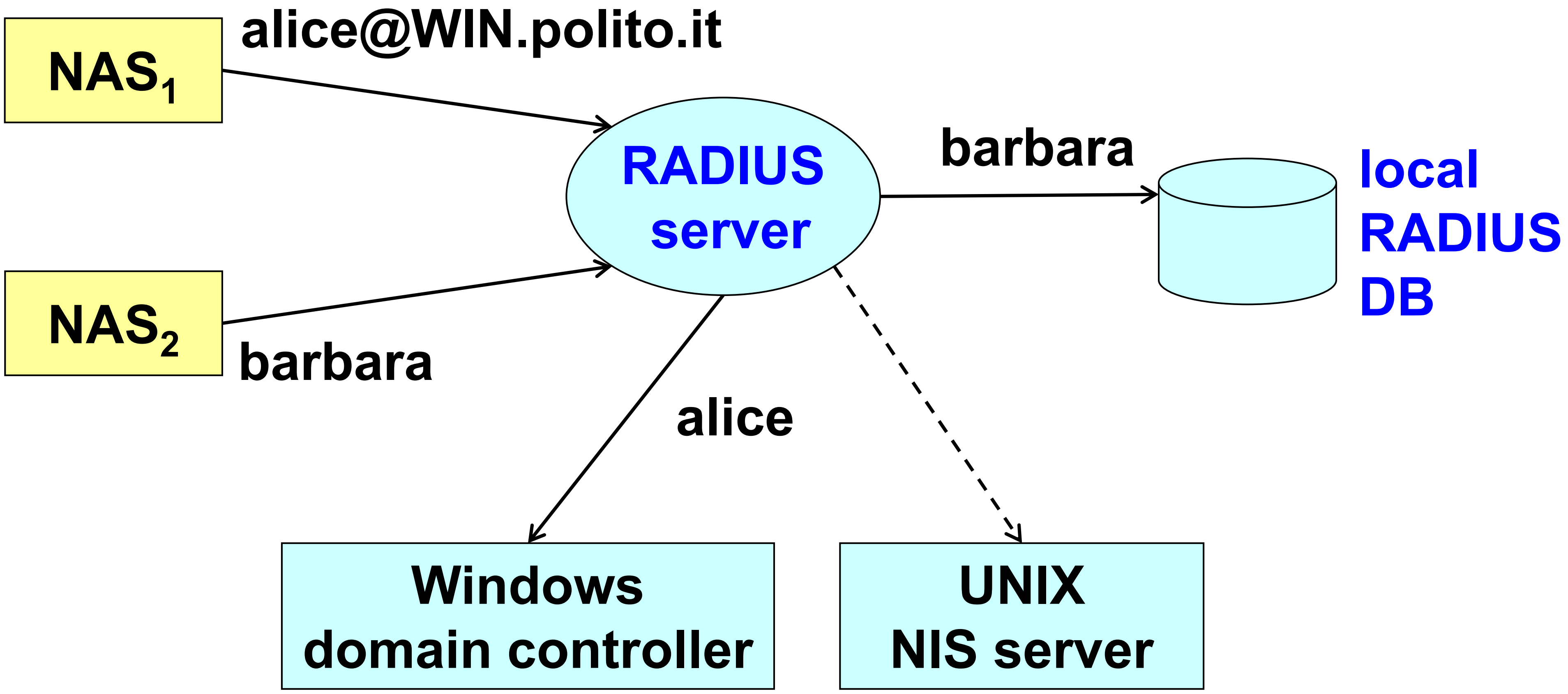
- **Remote Authentication Dial-In User Service**
- **Livingston Technologies (1991) then IETF**
- **supports authentication, authorization and accounting to control network access:**
  - physical ports (analogical, ISDN, IEEE 802)
  - virtual ports (tunnel, wireless access)
- **centralized administration and accounting**
- **client-server schema between NAS and AS**
  - port 1812/UDP (authentication) and 1813/UDP (accounting) ; unofficial ports: 1645 & 1646/UDP
  - timeout + retransmission
  - secondary server

RADIUS - RFC

- RFC-2865 (protocol)
- RFC-2866 (accounting)
- RFC-2867/2868 (tunnel accounting and attributes)
- RFC-2869 (extensions)
- RFC-3579 (RADIUS support for EAP)
- RFC-3580 (guidelines for 802.1X with RADIUS)

RADIUS proxy

- the RADIUS server may act as a proxy towards other authentication servers



Which security functionalities for Radius?

- **sniffing NAS req (if contains pwd)**
  - confidentiality, privacy
- **fake AS resp (to block valid or allow invalid user)**
- **changing AS resp (Y > N or N > Y)**
  - authN & integrity of AS resp
- **replay of AS resp (if not properly tied to NAS req)**
  - anti-replay of AS resp
- **pwd enumeration (from fake NAS)**
  - authN of NAS req
- **DoS (many NAS req from fake NAS)**
  - server scalability

RADIUS: data protection

- **packet integrity and authentication via keyed-MD5:**
  - key = shared-secret
  - client without key are ignored
- **password transmitted “encrypted” with MD5 (after padding with NUL bytes to a multiple of 128 bit):**

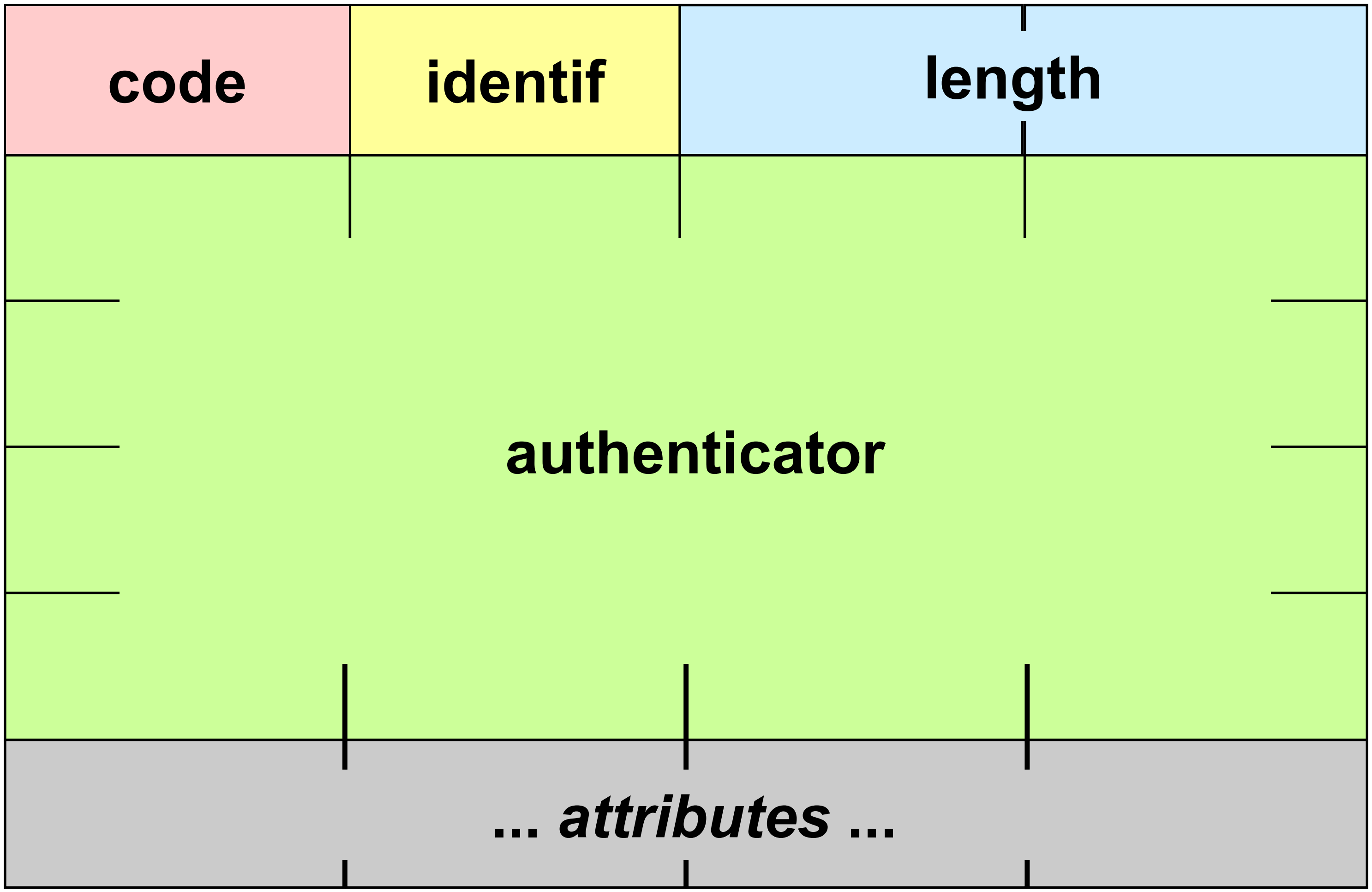
$$\textbf{password} \oplus \textbf{md5(key+authenticator)}$$

RADIUS

- **user authentication via PAP, CHAP, token-card and EAP**
  - CISCO provides a free server for CryptoCard
  - others support SecurID
- **attributes in TLV form, easily extensible without modification to installed base (by ignoring any unknown Type):**

attribute type – length – value

RADIUS - format



RADIUS – packet types

- **ACCESS-REQUEST**
  - contains access credentials (e.g. username + pwd)
- **ACCESS-REJECT**
  - access is denied (e.g. due to bad username/pwd)
- **ACCESS-CHALLENGE**
  - requests additional info from the user (e.g. a PIN, token code, secondary password)
- **ACCESS-ACCEPT (** ***parameters*** **):**
  - access is granted + network parameters are given
    - for SLIP/PPP: Framed-Protocol, Framed-IP-Address, Framed-IP-Netmask, MS-primary-DNS-server, MS-Secondary-DNS-server, …
    - for terminal: host, port

RADIUS - authenticator

- **double purpose:**
  - server reply authentication and no replay
  - masking the password
- **in Access-Request:**
  - it is named Request Authenticator
  - 16 byte randomly generated by the NAS
- **in Access-Accept / Reject / Challenge**
  - it is named Response Authenticator
  - it is computed via a keyed-digest:

md5(code || ID || length || RequestAuth || attributes || secret)

RADIUS - some attributes

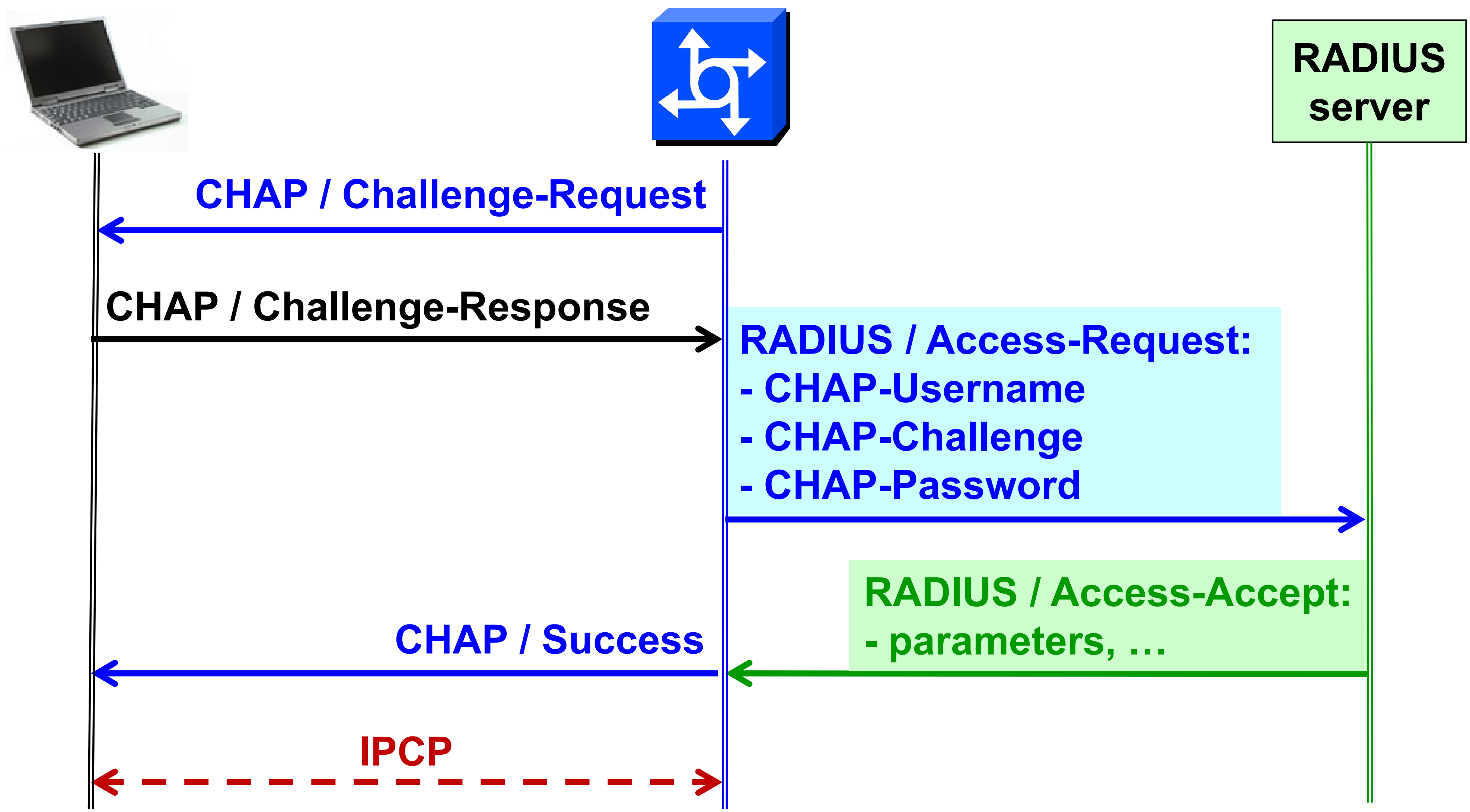
type	length	value

- **type = 1 (User-Name)**
  - value = text, network access identifier (NAI), DN
- **type = 2 (User-Password)**
  - value = password ⊕ md5 (key || RequestAuthent.)
- **type = 3 (Chap-Password)**
  - value = user CHAP response (128 bit)
- **type = 60 (CHAP-Challenge)**
  - value = challenge from the NAS to the user

NAI (Network Access Identifier)

- RFC-2486
- NAI = username [@ realm]
- all devices must support NAI up to 72 byte long
- the exact syntax for username and realm is in the RFC (note that only ASCII characters < 128 are allowed, but all of them are allowed)
- note that the username is the one used in the PPP authentication phase (does not necessarily match the application username)

Example - CHAP + RADIUS



IEEE 802.1x

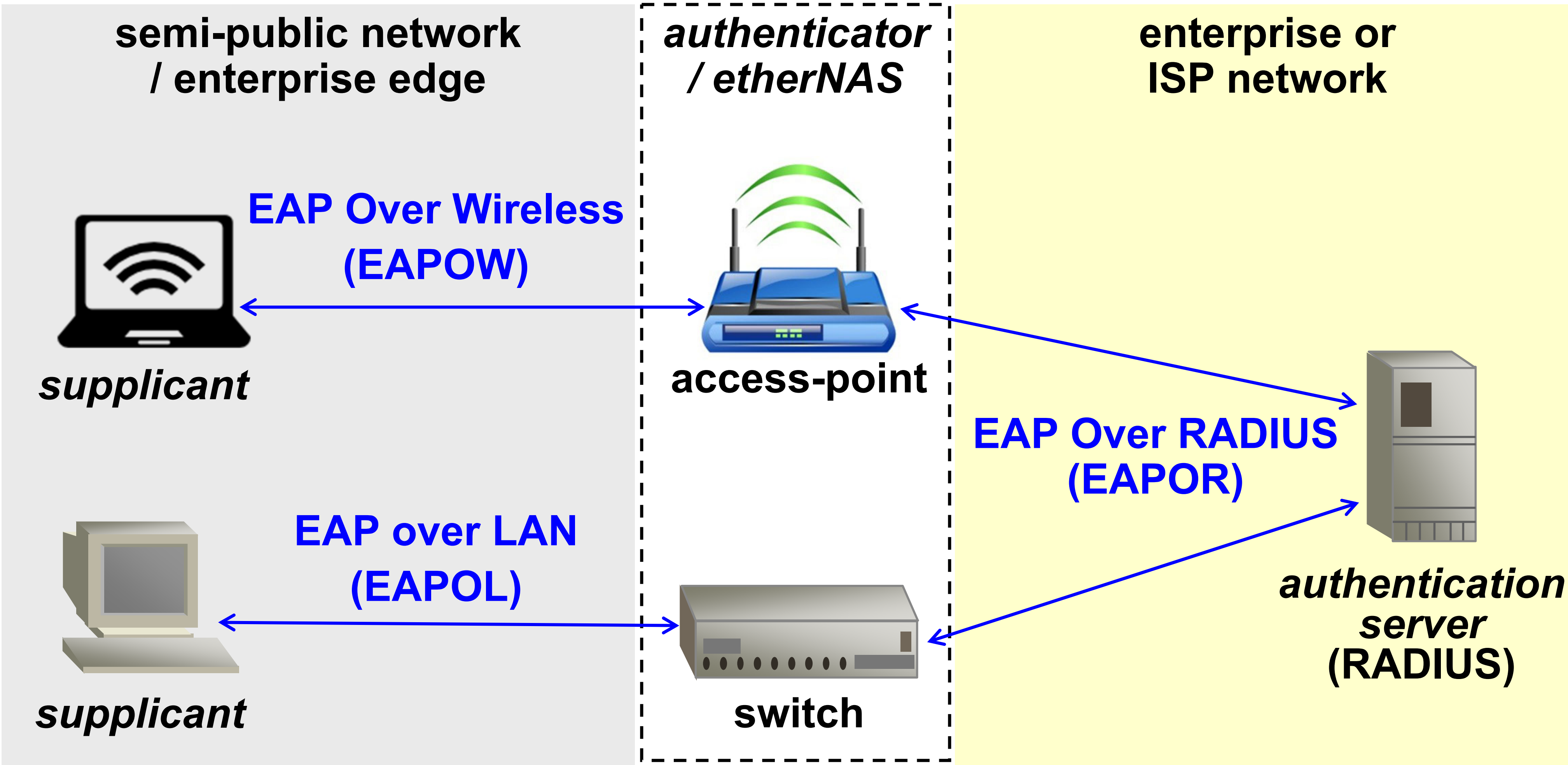
- **Port-Based Network Access Control:**
  - L2 authentication architecture
  - useful in a wired network to block access
  - absolutely needed in wireless networks
- **first implementations (long ago):**
  - Windows-XP and Cisco wireless access-points

http://standards.ieee.org/getieee802/download/802.1X-2001.pdf

IEEE 802.1x

- **authentication and key-management framework:**
  - may derive session keys for use in packet authentication, integrity and confidentiality
  - standard algorithms for key derivation (e.g. TLS, SRP, …)
  - optional security services (authentication or authentication+encryption)

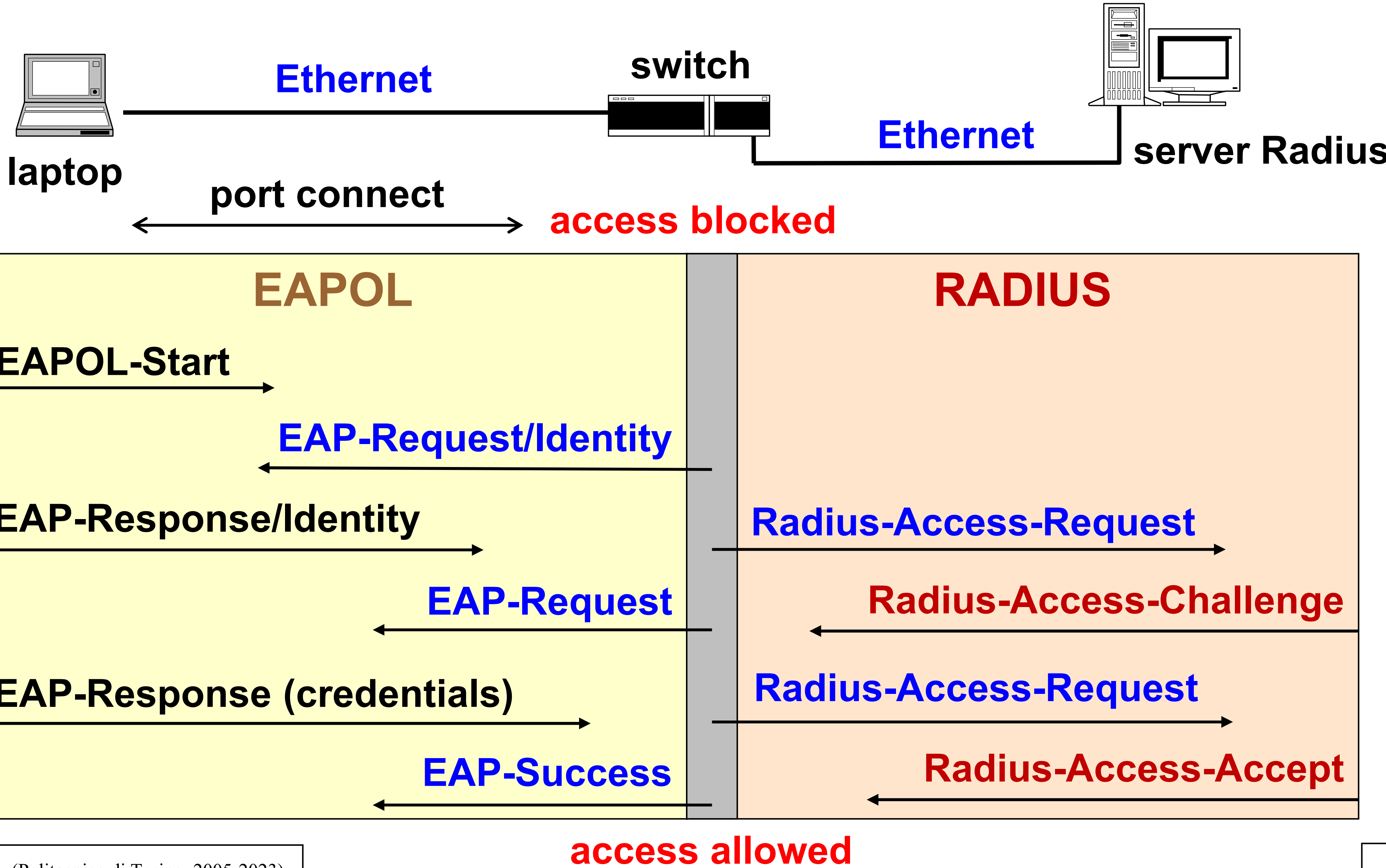
802.1x - architecture



802.1x - advantages

- **exploits the application level for the actual implementation of the security mechanisms**
  - direct dialogue between supplicant and AS
    - NIC and NAS operate as “pass-through device”
  - no change needed on NIC and NAS to implement new mechanisms
  - perfect integration in AAA

802.1x - messages



eduroam

- **WiFi access at research institutes (Italy, Europe, …)**
  - (21/11/2021) 106 countries
  - uses 802.1x + RADIUS federation