AAA

KOMUNIKACIJSKI PROTOKOLI IN OMREŽNA VARNOST

AAA

- ***A**uthentication* : who is actually the person (computer) we are talking to
- ***A**uthorization* : does the person (computer) we are talking to have the necessary privileges to the source / use of service / …
- ***A**ccoounting* : who has at any time used a source/service/…

Content

- authentication: what is it, how can it be implemented, protocols
- authorization: how can it be implemented
- recording: system recording
- protocol for AAA

- Literature: C. Kaufman, R. Perlman, M. Speciner. Network Security – Private Communication in a Public World. Prentice Hall.

Authentication

- trust,trust
  - trust, trust, , trust...

Authentication

- two sides (Ana and Borut) are communicating and they must believe that they are actually talking to each other
  - establishing identities at the beginning
  - maintaining identity throughout the conversation
  - how can we believe that the other side is in fact the correct side
  - a side can be a person or service / program
- Ana needs to know:
  - something about Borut, with which she can recognize Borut
  - that „something“ must only be known to Ana

authentication with passwords

- Borut tells Ana his password
- possible attacks:
  - tapping (stealing inside transfer)
  - breaking into the system (stealing saved passwords)
  - guessing passwords
- defences:
  - using safe cryptographic connections
  - system / password security
  - limiting the number of trys for password guessing
- additional defence
  - Ana sends Borut a challenge which he must be able to solve

storing passwords

- passwords are being stored in all places where they are needed
  - huge vulnerability, the problem of changing
- passwords are stored in one place and used by all users
  - protection of transferring a copied to user

  we have a special node that provides service for checking password
  - special protocol

storing passwords

- We additionaly protect stored passwords with cryptographic protection
- we don’t store passwords in their original form, instead we use safeguarded unidirectional hash function *f*
  - authentication:
    1. Borut calculates *f(password)* -> g
    2. Borut sends g
    3. Ana keeps in database g and not the password. She only checks its presence g in database *(this is the correct translation)*

attacks on password

- By guessing: we limit the number of attempts
  - automaton occupies the card;
  - password is valid for a limited amount of attempts
- Limiting how long the password is valid:
  - The S/KEY One-Time Password System, RFC1760
  - A One-Time Password System, RFC2289
    - *required: find it on the internet and read about it – literature!*
    - challenge: write your own program for S/Key or invent your OTP.

attacks on password

- Stealing passwords
  - stolen blind text – change the password
  - Stolen mappings
- On the internet there are databases/services, which sistematicly calculate password mappings
  - possible defense– we salten the password
    - challenge: how to performe saltening?

address as the password

- (IP) address represents a password or a part of it
  - We trust only certain computers
- Loging is possible only from those computers
  - We trust those computers, that they finished appropriate authentication (file hosts.equiv,)
  - Only those computers are allowed to authenticate
  - required: Consider how to address the authentication at ssh?

trusted intermediaries

- *key distribution centre*
- Broker forms a key (password) for every new connection
- Short-lived keys
- *certification authority*
- Broker provides authorized passwords
- Long-lived certificates, must have option to cancel it
- Hierarchy of intermediaries

authentication people

- Using passwords
- Authentication utility
- Using biometric characteristics

- Two other options require additional hardware (which we have to trust)

passwords

- Password must not be simple: length, number of characters, which sings , ..
  - admin/admin, 1234, unique master citizen number
- Password must not be too complicated
  - NaWUwra66nu5UHAd ☹
    - challenge: Find a system that generates safe passwords.
- We change passwords systematicly
- What if we forget a password?

authentication devices

- cards
  - Only holders of informations (magnetic recording, optical recording, ...)
- Smart cards
  - They contain a computer that protects information , we need a password to access the computer...
  - Use of challenge
- Cryptographic computers
  - They form a time-depended passwords

biometric features

- Replacable password
- lack of portability

- routine, fingerprint, face identificatio, iris, voice, .

authentication process

- directly
  - Loging to a computer console
  - Remote access: telnet (TELNET Protocol, RFC 139), ssh (Does RFC exist for ssh?)
    - challenge: find other RFC documents about telnet.
- ad hoc form
- Using protocols

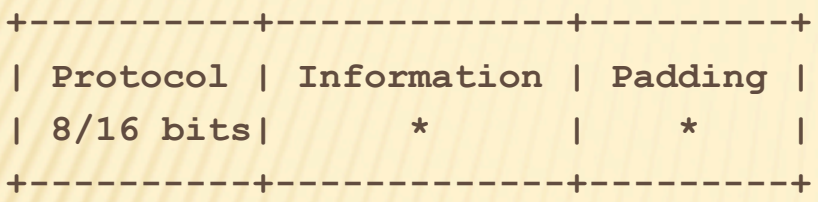
protocols for authentication

- PPP in PAP: Password authentication protocol
- CHAP: Challenge-handshake authentication protocol (MS-CHAP)
- EAP: Extensible Authentication Protocol

PPP IN PAP

- The Point-to-Point Protocol (PPP), RFC 1661
  - *challenge: find and read RFC.*
- It is replacing data-link layer
- Authentication required at the beginning of sessions

PPP



- protocol:
  - 0001 Padding Protocol
  - 0003 to 001f reserved (transparency inefficient)
  - 007d reserved (Control Escape)
  - 00cf reserved (PPP NLPID)
  - 00ff reserved (compression inefficient)
  - 8001 to 801f unused
  - 807d unused
  - 80cf unused
  - 80ff unused
  - c021 Link Control Protocol
  - c023 Password Authentication Protocol
  - c025 Link Quality Report
  - c223 Challenge Handshake Authentication Protocol

PAP

- Password transfer in cleantext
- Last option, if all other fail (and if we are still willing to do it)

CHAP

- PPP Challenge Handshake Authentication Protocol (CHAP), RFC 1994
  - *required: find this protocol on the internet and read it – literature!*
- Prepared for PPP use (poin to point protocol)
- Challenge-based design that Ana sends to Borut
- Transmission protocol in principle is not defined (see PPP)

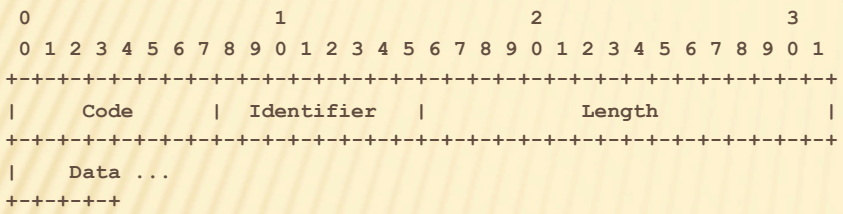
CHAP

- Three-step protocol:
  1. Ana sends a challenge
  2. Borut combines the challenge with a password and sends it back encrypted with a one-way hash function
  3. Ana verifies the if the answer is correct
- Steps in PPP protocol can be repeated for unlimited number of times
- Challenge is sent in a readable form
- password must be stored on both sides
- because the challenge is changing, it is difficult to attack with repeating

which hash function

- ppp protocol has its own control protocol LCP
- it can set various properties and also the type of a hash function
  - *challenge: where and how can we set it?*

CHAP-shape package



- Code – message code: 1 Challenge, 2 Response, 3 Success, 4 Failure
- Identifier – connection between protocol steps

MS-CHAP

- Microsoft PPP CHAP Extensions, Version 2, RFC 2759
  - *challenge: find it on the internet and read it; how is a password change conducted and what do we have to be careful of?*
- There are two versions
  - *required: how is the first version different from the second one?*
- Based on the CHAP protocol with two fundamental appendices:
  - mutual authentication
  - The ability to change paswords

EAP

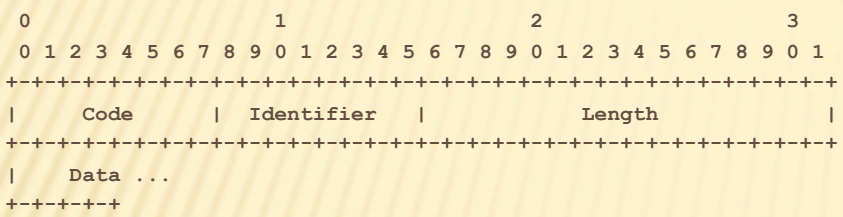
- Extensible Authentication Protocol (EAP), RFC 3748 –the basic protocol and corrections RFC5247
  - *challenge: find and read RFC*
- Framework for protocols and not a real protocol because it defines only the message format
- usually directly over the data-link layer

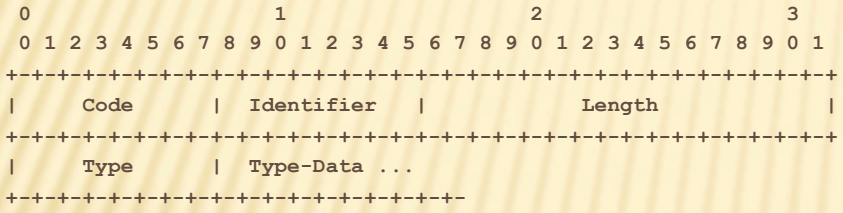
  (ppp, IEEE 802 – ethernet) and also UDP, TCP
  - *challenge: In RFC find which protocol is using UDP*
- Forwarding possibility– Authentication Server

EAP-base operation

- The client and the server (authenticator) make an agreement about the type of authentication.
- Step-protocol:
  1. Authenticator sends a request for data; ex. identification, request for authentication including the type of authentication
  2. client confirms or refuses the way of authentication
  3. steps 1. and 2. are repeated until the server identifies the client

EAP-shape package





- identical to CHAP

- request/response package
- type – what does authenticator request and how does client respond
  - 1 Identity
  - 2 Notification
  - 3 Nak (Response only)
  - 4 MD5-Challenge
  - 5 One Time Password (OTP)
  - 6 Generic Token Card (GTC)
  - 254 Expanded Types
  - 255 Experimental use

authorization

- when the user is authenticated (identified), we can check the rights that the user has
- on Unix systems usually becomes a member of a group or multiple groups, that have certain rights (*group*)
- Similiar on MS windows systems
  - *challenge: there is a RFC 2904, AAA Authorization Framework. What's it about and does it define some requirements or something else?*

Access Authorization Matrix

- *access matrix* specifies the rights of the individual user groups
  - *capability list*
  - *access control list*
- stored locally in the file/files
  - similar problems as with password storage
- stored on the server
    - *challenge: How is the safety of downloaded messages and their encription?*

record

- system that will record contents of events and where and when they occurred
- Common recording form on operation systems is syslog (POSIX standard)
- Standardized also with IETF as RFC 5424, *The Syslog Protocol*.
  - *challenge: compare RFC with "man –k syslog" sites?*
  - *challenge: find other RFCs about Syslog and IETF site, where Syslog working group published documents.*

record and syslog

- Log is stored in file /var/log ...:
  - Nov 13 17:00:17 svarun0 sshd[92530]: error: PAM: authentication error for root from ip-62-129-164-36.evc.net
  - possible message levels: Emergency, Alert, Critical, Error, Warning, Notice, Info or Debug
  - *challenge: See the files in /var/log/...*

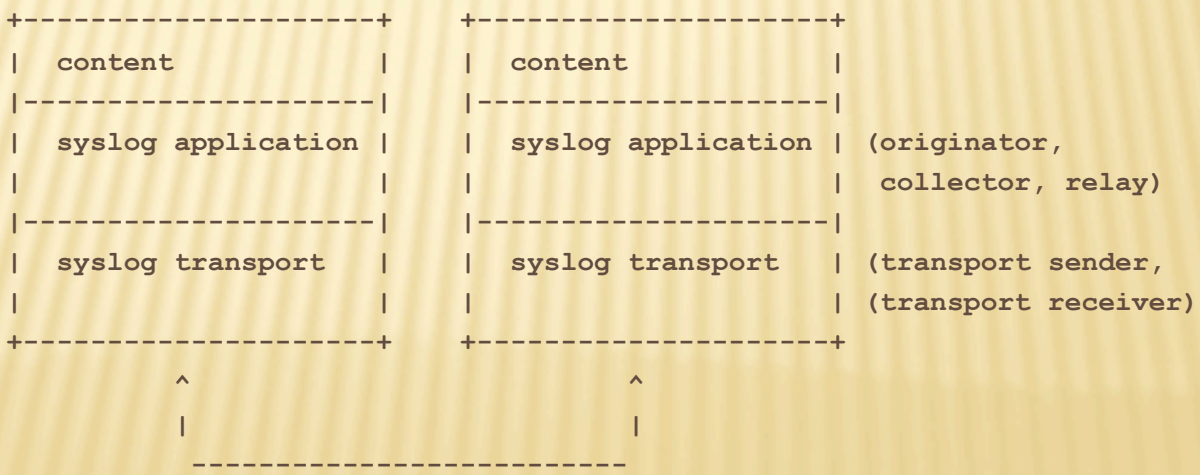
software

- on FreeBSD syslogd
- configuration in /etc/syslog.conf
  - *challenge: change the configuration so that all messages will be stored in / var / log / super-log; how to send a note to another computer?; and can we store the same note to multiple locations?*

```
security.*                      /var/log/security
auth.info;authpriv.info         /var/log/auth.log
mail.info                       /var/log/maillog
lpr.info                        /var/log/lpd-errs
ftp.info                        /var/log/xferlog
cron.*                          /var/log/cron
```

SYSLOG PROTOKOL

- Internal architecture distributes:
  - Message form and their content (RFC 5424)
  - Way of message transmision (RFC 5425)
    - *required: find RFC 5425 and look for which ingredients it speaks of– literature!*
    - challenge: find other RFCs that talk about syslog.



Syslog protokol- oblika sporočila

```
SYSLOG-MSG      = HEADER SP STRUCTURED-DATA [SP MSG]

HEADER          = PRI VERSION SP TIMESTAMP SP HOSTNAME
                  SP APP-NAME SP PROCID SP MSGID
PRI             = "<" PRIVAL ">"
PRIVAL          = 1*3DIGIT ; range 0 .. 191
VERSION         = NONZERO-DIGIT 0*2DIGIT
HOSTNAME        = NILVALUE / 1*255PRINTUSASCII

APP-NAME        = NILVALUE / 1*48PRINTUSASCII
PROCID          = NILVALUE / 1*128PRINTUSASCII
MSGID           = NILVALUE / 1*32PRINTUSASCII

TIMESTAMP       = NILVALUE / FULL-DATE "T" FULL-TIME
FULL-DATE       = DATE-FULLYEAR "-" DATE-MONTH "-" DATE-MDAY
DATE-FULLYEAR   = 4DIGIT
DATE-MONTH      = 2DIGIT  ; 01-12
DATE-MDAY       = 2DIGIT  ; 01-28, 01-29, 01-30, 01-31 based on
                  ; month/year
FULL-TIME       = PARTIAL-TIME TIME-OFFSET
PARTIAL-TIME    = TIME-HOUR ":" TIME-MINUTE ":" TIME-SECOND
                  [TIME-SECFRAC]
TIME-HOUR       = 2DIGIT  ; 00-23
TIME-MINUTE     = 2DIGIT  ; 00-59
TIME-SECOND     = 2DIGIT  ; 00-59
TIME-SECFRAC    = "." 1*6DIGIT
TIME-OFFSET     = "Z" / TIME-NUMOFFSET
TIME-NUMOFFSET  = ("+" / "-") TIME-HOUR ":" TIME-MINUTE
```

```
STRUCTURED-DATA = NILVALUE / 1*SD-ELEMENT
SD-ELEMENT      = "[" SD-ID *(SP SD-PARAM) "]"
SD-PARAM        = PARAM-NAME "=" %d34 PARAM-VALUE %d34
SD-ID           = SD-NAME
PARAM-NAME      = SD-NAME
PARAM-VALUE     = UTF-8-STRING ; characters '"', '\' and
                  ; ']' MUST be escaped.
SD-NAME         = 1*32PRINTUSASCII
                  ; except '=', SP, ']', %d34 (")

MSG             = MSG-ANY / MSG-UTF8
MSG-ANY         = *OCTET ; not starting with BOM
MSG-UTF8        = BOM UTF-8-STRING
BOM             = %xEF.BB.BF

UTF-8-STRING    = *OCTET ; UTF-8 string as specified
                  ; in RFC 3629

OCTET           = %d00-255
SP              = %d32
PRINTUSASCII    = %d33-126
NONZERO-DIGIT   = %d49-57
DIGIT           = %d48 / NONZERO-DIGIT
NILVALUE        = "-"
```

PROTOKOL RADIUS

- defined in RFC 2865, *Remote Authentication Dial In User Service (RADIUS)* and RFC 2866, *RADIUS Accounting*
  - *required: find it on the internet and read about it – literature!*
  - challenge: find other RFC documents that deal with tftp and check what it say in them.
- basic functionalities:
  - authentication, authorization, recording
  - It can use other protocols for authentication
  - Look also at RFC 4962, *Guidance for Authentication, Authorization, and Accounting (AAA) Key Management*

RADIUS basic architecture

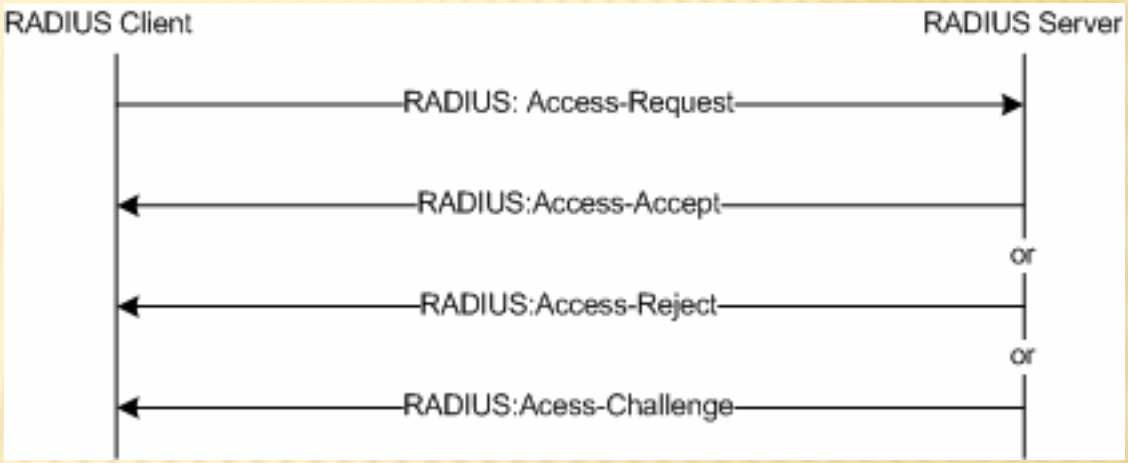
- three parties involved:
  - **user** of a service
  - **Service provider** – service provider: NAS, *Network access server*, which is also **RADIUS client**
  - **RADIUS server**
  - RADIUS server can also only be an interface for an access to the second RADISU server

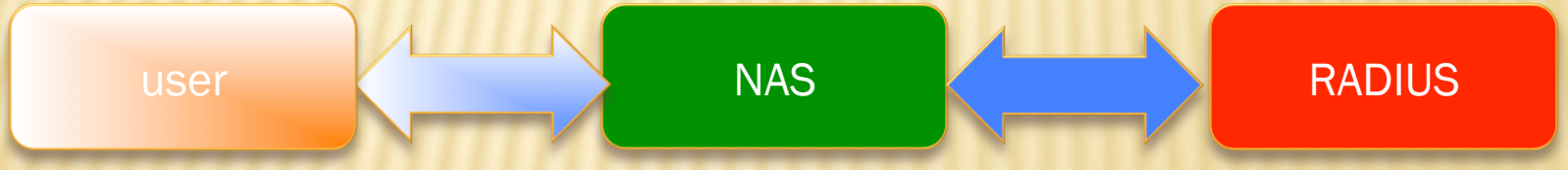
communication user-NAS

- usually directly on a data-link (!) layer
  - ppp
  - ethernet
- sometimes higher layers such as https
- safety!

Communication NAS- RADIUS(AA.)

- RADIUS protocol
  - NAS sends: *Access Request*
  - RADIUS responds: *Access Reject, Access Challenge, Access Accept*
  - If no response in a period of time, the demand is re-sent
- RADIUS can send the demand forward – *proxy*





Radius-request for access

- *Access Request message*
- Diffrent protocols – PAP, CHAP, MS-CHAP, EAP
  - *challenge: look at how MS-CHAP is supported; RFC 2548, Microsoft Vendor-specific RADIUS Attributes.*
  - *challenge: how is the support for EAP?*

Radius-denial

- *Access Reject message*
- various reasons:
  - incorrect password / username, ...
  - inadequate rights
  - further clarification may be in the message

Radius-challenge

- *Access Challenge message*
- additional password or message in different cases:
  - different password,
  - PIN code
  - established tunnel between the user and authenticator, ...
  - Something else ...

Radius-confirmed

- *Access Accept message*
- RADIUS menu, that access is confirmed / authorized
  - Both the password / username as authorization
  - message can bring additional information, which NAS needs to set up services (IP address, how to establish L2TP tunel, ...); depending on the service
  - NAS may obtain additional information from other services– files, LDAP, ...

Radius- middleserver and areas

- *proxy*
- distribution of users to areas (spheres) (*realm*)
- area is defined by any set of letters, which is usually similar to the domain name
  - peter.zmeda@butale.isp
  - andrej.brodnik@fri.uni-lj.si
- Each area has its own RADIUS server

Radius- middleserver and hosting

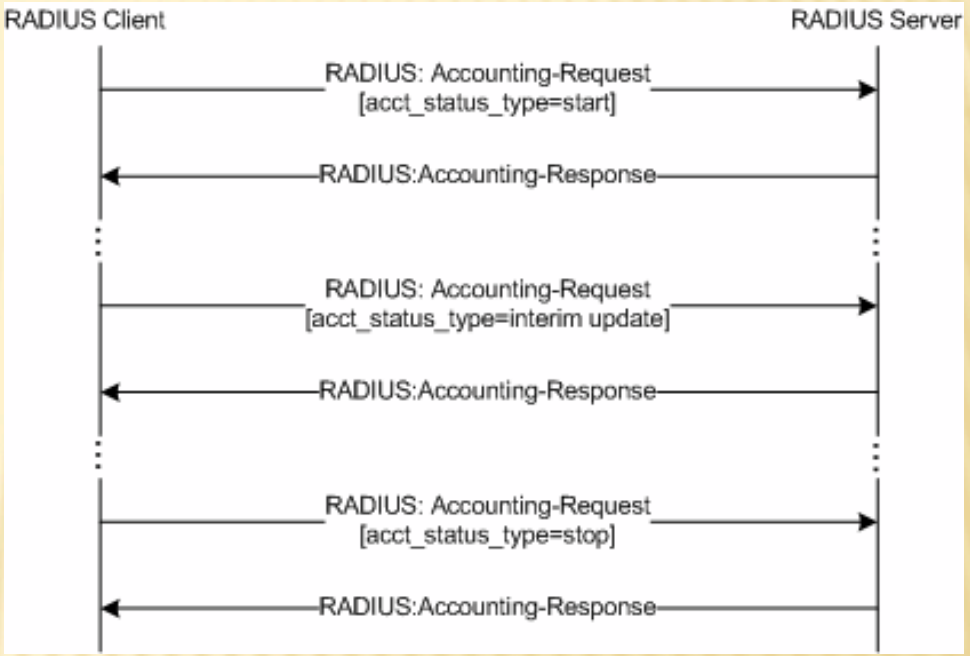
- *roaming*
- the service provider can - via the RADIUS server - allow hosting of users from other domains in his own field
- user from another area may be granted the right to use a service(Authorization)
  - Establishing collaboration among areas
  - authentication to another area

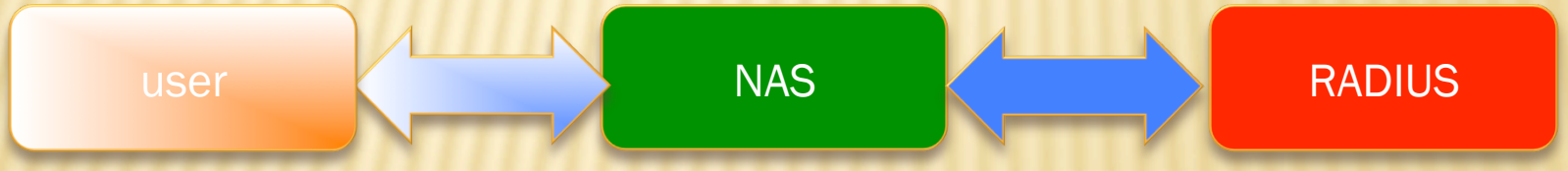
Radius- middleserver and preinter-vention

- *proxy*
- Connections beetwen servers can be safe (VPN)
- Middle server can transform the received request and send it to the right server (almost, see RFC 2865):
  - Middle server encrypts the message and sends it to the parent server
  - Parent server returns the encrypted response
    - challenge: what can the middle server change and how?

Communication NAS - RADIUS (..A)

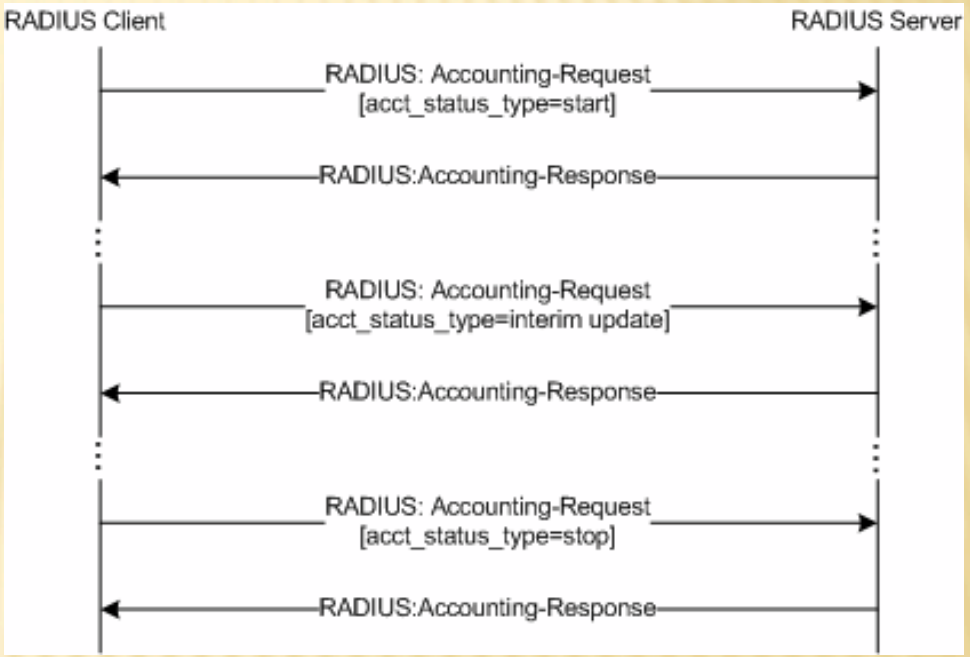
- RADIUS protocol
  - NAS sends: *Accounting Request*
  - RADIUS responds: *Accounting Response*
  - If no answer in a period of time, the request is sent again
- RADIUS can send the request forward – *proxy*





RADIUS - recording

- We can record three types of events:
  - The beginning of service use
  - further use or correction of data
  - End of use
- difference is in the content of the package, while one pair of commands is for all.



PROTOKOL RADIUS

- defined commands(example. *RPC*, *RMI*):
  - *Access Request*
  - *Access Reject, Access Challenge, Access Accept*
  - *Accounting Request*
  - *Accounting Response*
- each of the commands may have different additional features / parameters (*attributes*)

PROTOKOL RADIUS

- RFC expects UDP transport protocol
  - RADIUS is a transaction protocol– similar to http
  - Communication is step by step
  - Simplifying middle servers operations, because they don't have open connections
- UDP protocol is not safe
  - Transition to TCP/SSL
  - security on lower layers: using VPN (IPSec)

Radius protocol - signing

- signature is called *autheticator* and it is the only source of ensuring the authenticity of the package sent
- NAS and RADIUS server share a common key *secret* (*shared secret*)

Radius protocol - signing

- Signing AA. packages:
  - Client: 128-bit random number - salt
  - server (response): 128-bit number derived from the secret, package content and client salt
  - signature is used as the response authentication and does not protect requirements of the client
  - salt in the client signature is also used as salt for protection of sent password

Radius protocol - signing

- signing .. A packages:
  - Client: 128-bit number derived from secret and package content
  - server (response): 128-bit number derived from the secret, signature of client-package and package content
  - signature protects the client's request for a recording (trying to)

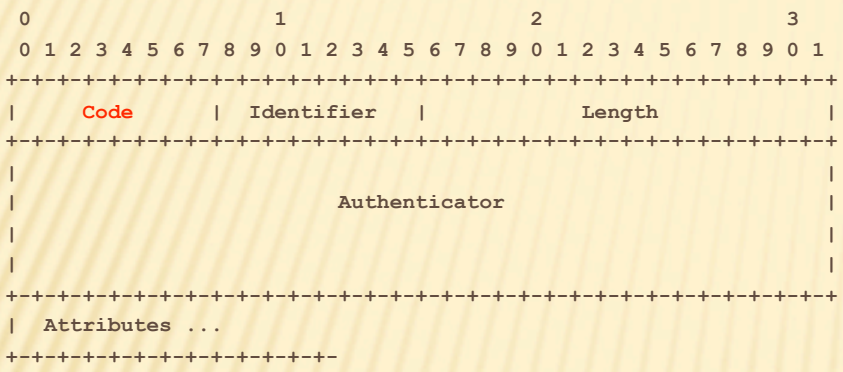
Radius protocol - Security

- Protection:
  - There is no protection against eavesdropping (hidding)
  - It's (partial) protection of the authenticity of sent packets
  - There is no protection against denial of sent contents
    - challenge: find in-depth security analysis of RADIUS protocol??

Radius protocol - Security

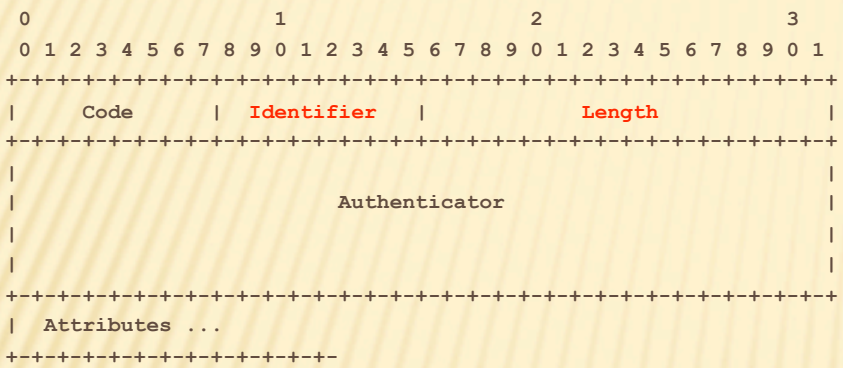
- Attacks:
  - attack by repeating
  - Middle-attacker attack
  - difference whether it is AA. part or ..A part
  - how is it with the distribution of secret and how is it distributed between the server and clients
  - challenge: lookat how handshaking with *secret works*?

Radius protocol - form package



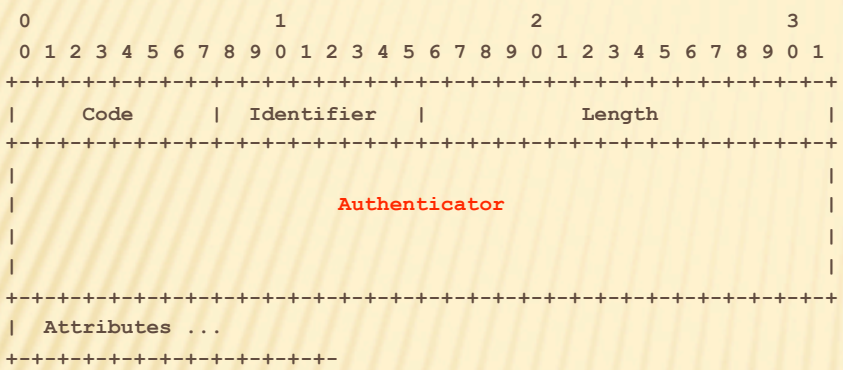
- Code – code command:
  - (1) Access-Request
  - (2) Access-Accept
  - (3) Access-Reject
  - (4) Accounting-Request
  - (5) Accounting-Response
  - (11) Access-Challenge
  - (12) Status-Server (trial)
  - (13) Status-Client (trial)
  - (255) Reserved

Radius protocol - form package



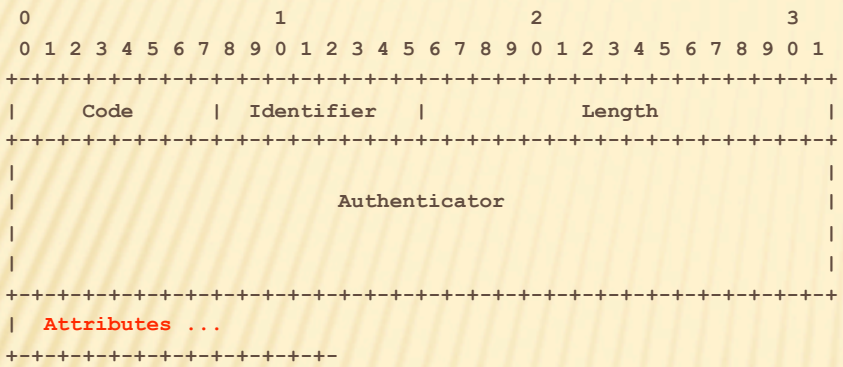
- Identifier – RADIUS protocol is a step-by-step protocol and client must know the answer to any request received. Length – length of the entire packet including the header in bytes
  - minimum length is 20 and the largest 4096
  - if the package is larger, it is reduced to length, if it is shorter, it is discarded

Radius protocol - form package



- Autheticator – ,,signature'' of package of lenght 16 bytes:
  - AA. request: 128 bit random number
  - AA. response: *MD5(Code • ID • Length • RequestAuth • Attributes • Secret)*
  - ..A request: *MD5(Code • ID • Length • 00^{16} • Attributes • Secret)*
  - ..A response: *MD5(Code • ID • Length • RequestAuth • Attributes • Secret)*
  - operation • is contact (concatenation)

Radius protocol - form package

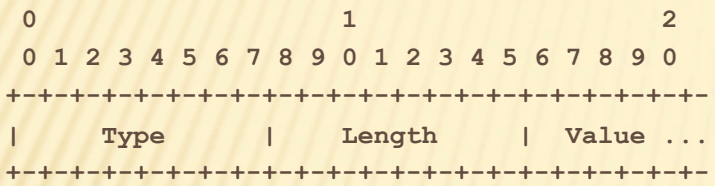


- Attributes – Additional parameters of the command that was sent

Protocol-Radius attributes

- number of possible attributes is 256
- request: the users must have the option of adding their own attributes
- Values of attributes are to be arbitrary: number, date, time, string, …

Radius attributes



- *TLV* record
- *Type* – which attribute it is
- *Length* – number of bytes to record the value of the attribute
- *Value* – value of attribute
  - text: UTF-8 encoded, length greater than 0 and a maximum length of 256 bytes
  - series: an arbitrary string, length greater than 0 and a maximum length of 256 bytes
  - Address: 32-bit recording
  - Integer: 32 bit recording
  - Time: 32 bit value from 00:00:00 1.1.1970 UTC (standard attributes do no use)

Protocol Radius- attributes

- Attributes walk-through:
  - (1) User-Name
  - (2) User-Password
  - (3) CHAP-Password

Protocol Radius - attributes: pass-word

- Password is encrypted using salt in authenticator (*RA*) and shared secret (*S*):
  - Password is divided into 128-bit parts p [1. n]
  - b[1]= MD5(S • RA); c[1]= p[1] XOR b[1]
  - …
  - b[i]= MD5(S • c[i-1]); c[i]= p[i] XOR b[i]

Protocol-Radius attributes

- Attributes walk-through:

- (4) NAS-IP-Address
- (5) NAS-Port
- (6) Service-Type
- (7) Framed-Protocol
- (8) Framed-IP-Address
- (9) Framed-IP-Netmask
- (10) Framed-Routing
- (11) Filter-Id
- (12) Framed-MTU
- (13) Framed-Compression
- (14) Login-IP-Host
- (15) Login-Service
- (16) Login-TCP-Port
- (17) (unassigned)
- (18) Reply-Message
- (19) Callback-Number
- (20) Callback-Id
- (21) (unassigned)
- (22) Framed-Route
- (23) Framed-IPX-Network
- (24) State

Protocol-Radius attributes

- Attributes walk-through:

- (25) Class
- (26) Vendor-Specific
- (27) Session-Timeout
- (28) Idle-Timeout
- (29) Termination-Action
- (30) Called-Station-Id
- (31) Calling-Station-Id
- (32) NAS-Identifier
- (33) Proxy-State
- (34) Login-LAT-Service
- (35) Login-LAT-Node
- (36) Login-LAT-Group
- (37) Framed-AppleTalk-Link
- (38) Framed-AppleTalk-Network
- (39) Framed-AppleTalk-Zone
- (40-59) recording
- (60) CHAP-Challenge
- (61) NAS-Port-Type
- (62) Port-Limit
- (63) Login-LAT-Port

Protocol-Radius attributes

- Attributes walk-through:– recording:

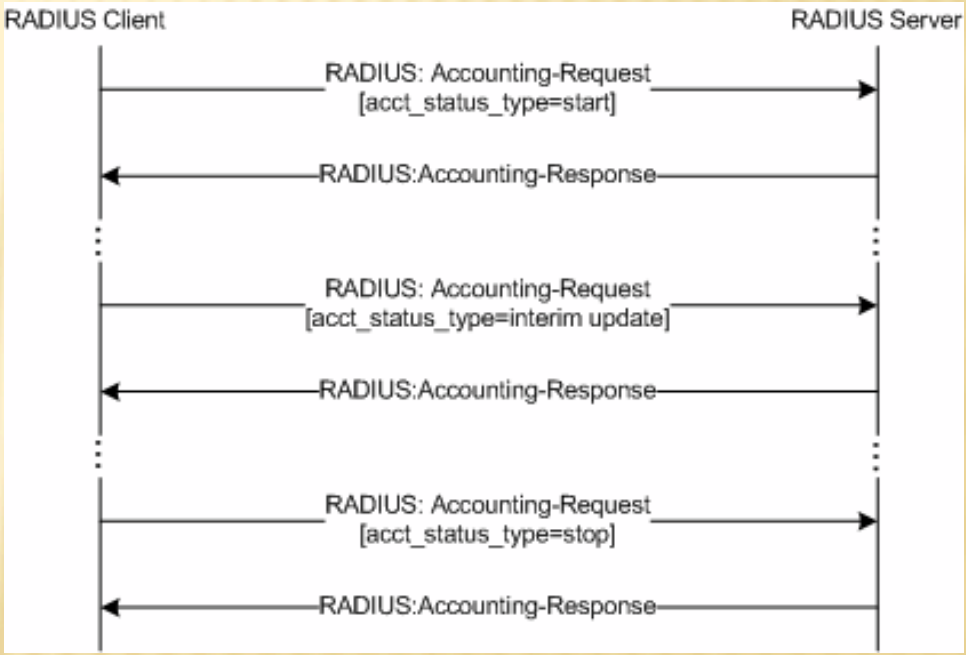
- (40) Acct-Status-Type
- (41) Acct-Delay-Time
- (42) Acct-Input-Octets
- (43) Acct-Output-Octets
- (44) Acct-Session-Id
- (45) Acct-Authentic
- (46) Acct-Session-Time
- (47) Acct-Input-Packets
- (48) Acct-Output-Packets
- (49) Acct-Terminate-Cause
- (50) Acct-Multi-Session-Id
- (51) Acct-Link-Count

- *challenge: How‘s it like with attributes 52-59 and 64-255?*
- *challenge: How‘s it like with attributes 17 and 21?*

Protocol Radius - recording

- *Acct-Status-Type* and *Acct-Session-Id* serve to support the record within one session on the service offered by NAS

status:
- (1) Start
- (2) Stop
- (3) Interim-Update
- (7) Accounting-On
- (8) Accounting-Off
- (9-14) Reserved for Tunnel Accounting
- (15) Reserved for Failed



software

- On FreeBSD (Linux): freeradius
- configuration in the/usr/local/etc/radiusd.conf
  - *challenge: find the manual and just set a file and run the server.*
  - *challenge: where is the shared secret stored and how it is shared between the server and clients?*
  - *challenge: where are notes being kept?*
  - *challenge: how can RADIUS use other services for authentication*

DIAMETER

- Defined in RFC 3588, *Diameter Base Protocol* in RFC 5719, 5729
  - *required: find it on the internet and read about it – literature!*
  - challenge: find the remaining RFC documents dealing with tftp and check what it says in them.
- Primarily security response to the RADIUS
- is not entirely consistent with the RADIUS

DIAMETER

- differences between RADIUS and DIAMETER:
  - More secure transmission protocols (TCP, ...)
  - integrated network security (SSL, IPsec)
  - More attributes are possible (32-bit)
- Software: freeDiameter