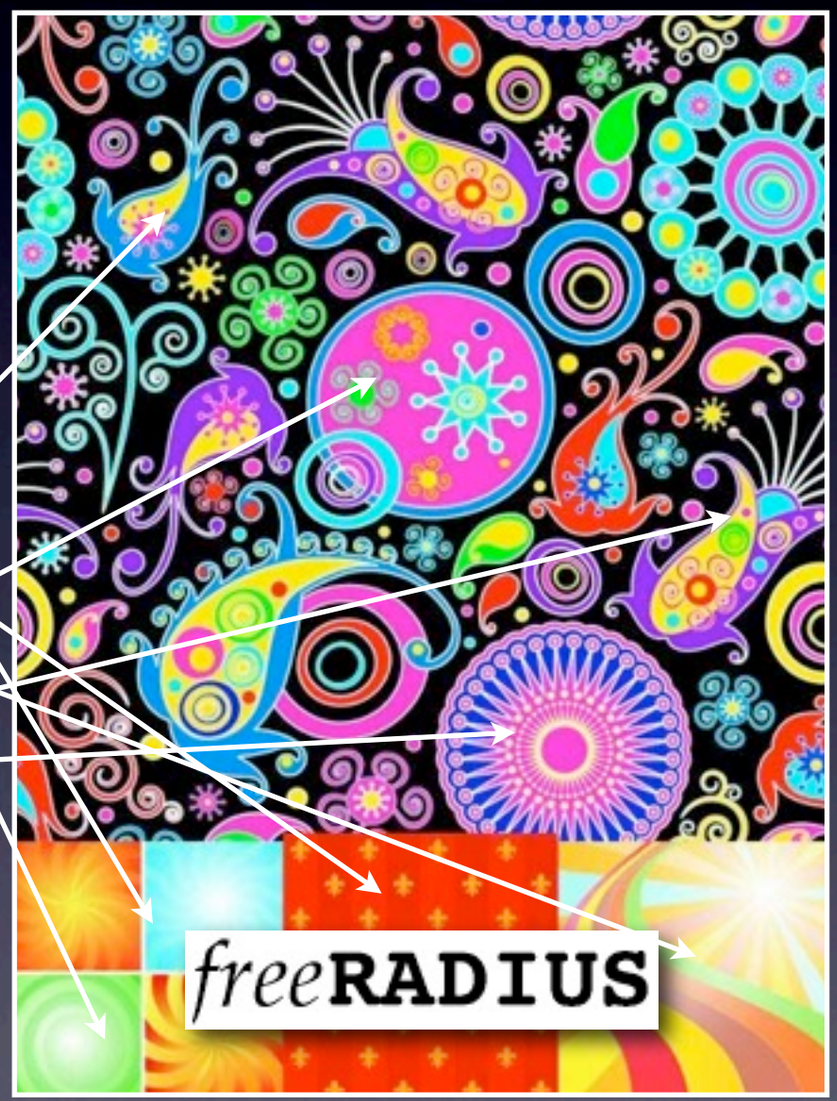
freeRADIUS

A High Performance, Open Source, Pluggable, Scalable
(but somewhat complex)
RADIUS Server

Aurélien Geron, Wifirst, January 7th 2011

freeRADIUS is...

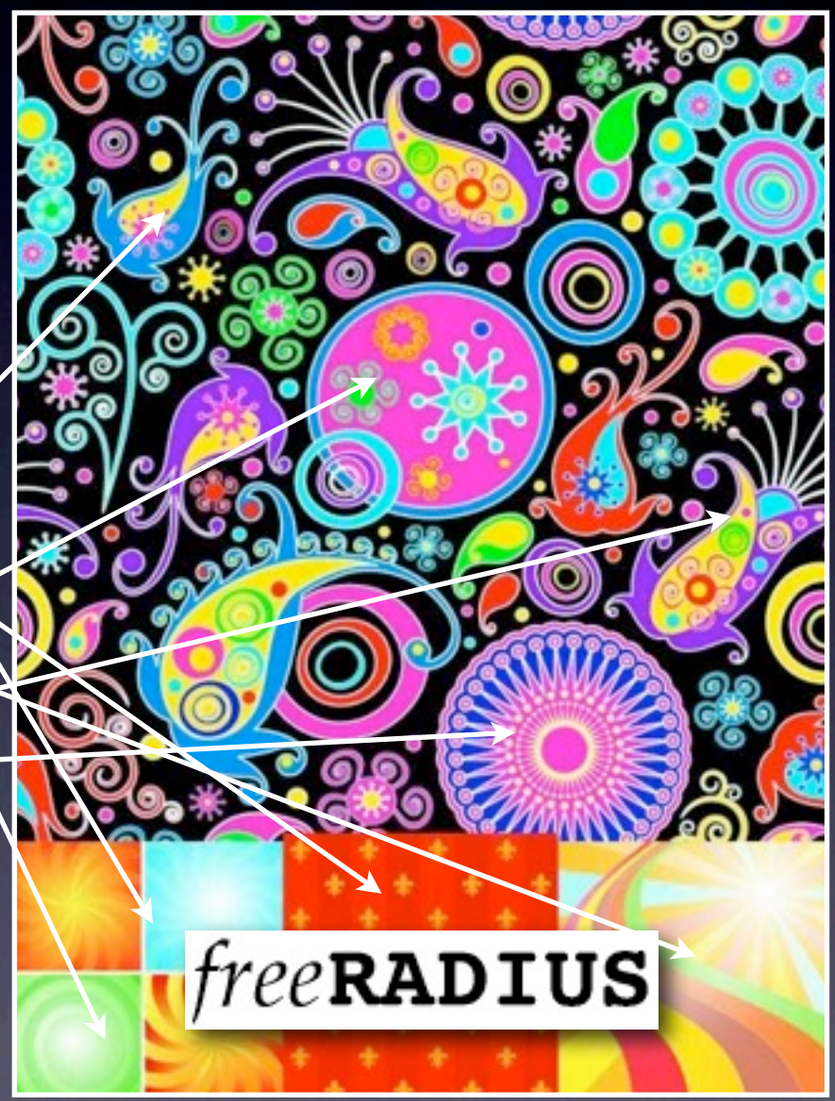
- Multiple protocoles : RADIUS, EAP...
- An Open-Source (GPLv2) server
- A powerful configu-ration system
- Many expansion modules
- The whole thing can get a bit complex



Source image: http://crshare.com/abstract-backgrounds-vector-clipart/

Roadmap

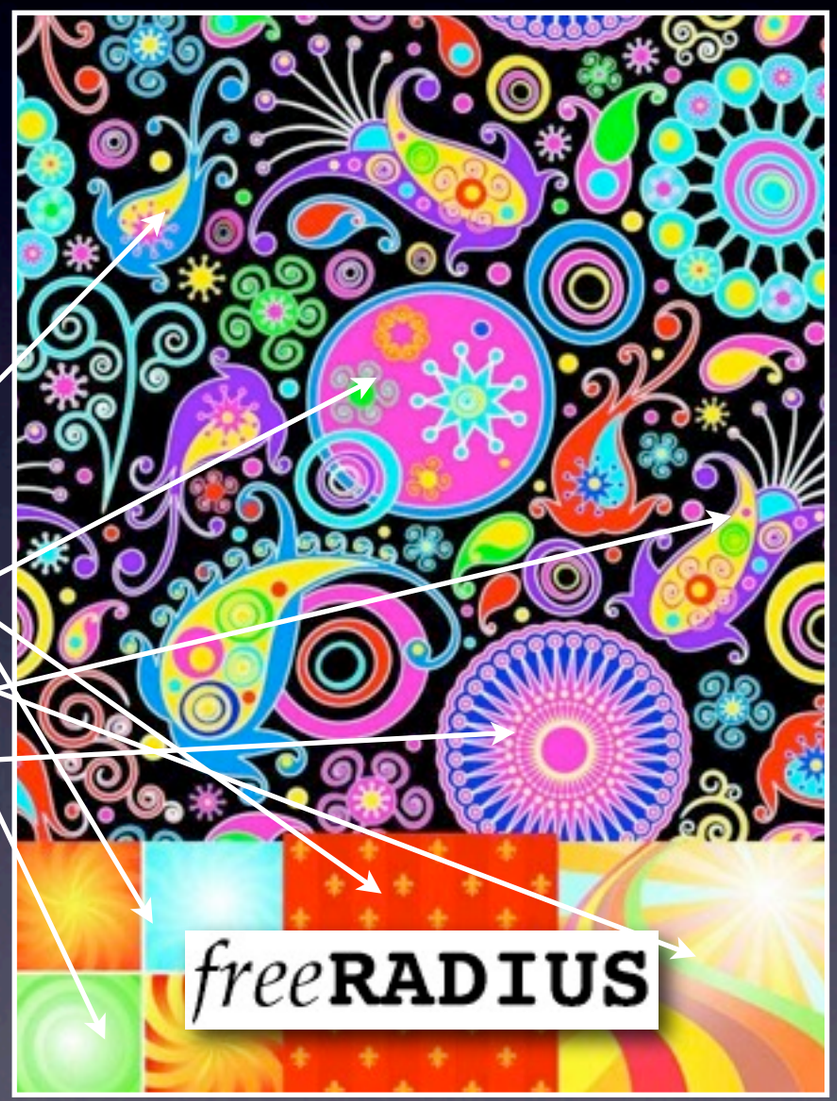
- Multiple protocoles : RADIUS, EAP...
- An Open-Source (GPLv2) server
- A powerful configu-ration system
- Many expansion modules
- Writing your own modules



Source image: http://crshare.com/abstract-backgrounds-vector-clipart/

Roadmap

- Multiple protocoles : RADIUS, EAP...
- An Open-Source (GPLv2) server
- A powerful configuration system
- Many expansion modules
- Writing your own modules

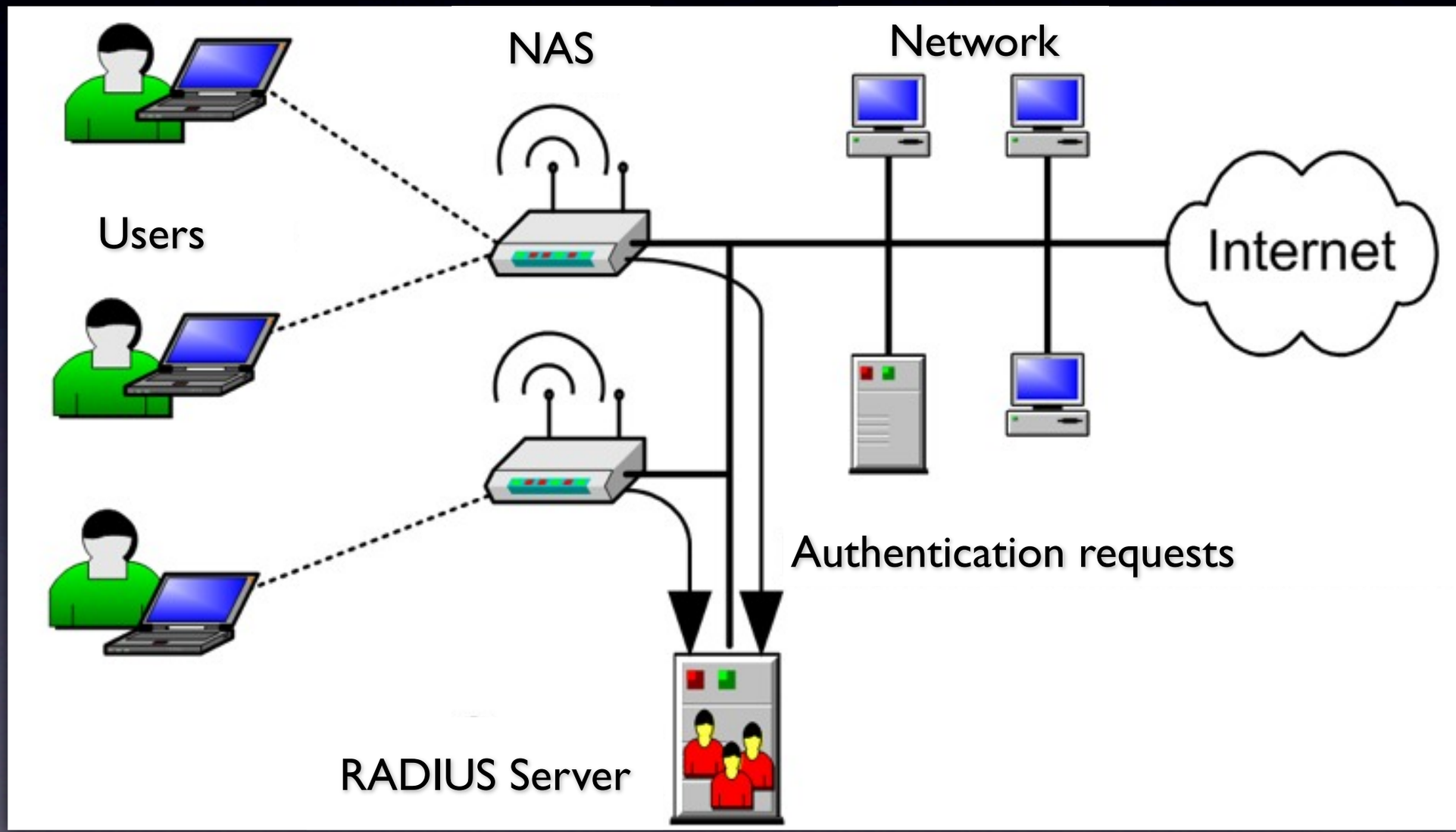


Source image: http://crshare.com/abstract-backgrounds-vector-clipart/

Before you start using freeRADIUS...

- You need to understand the RADIUS architecture and protocol
- as well as the EAP authentication methods, including EAP/TLS, PEAP and TTLS

RADIUS Architecture



Terminology

- User: the end-user trying to access the service
- NAS = Network Access Server = Client!
  - Historically, the NAS was located in Points of Presence (PoPs) of telco operators, to identify users trying to setup a dialup modem connection to the Internet
  - Nowadays, the NAS can also be a Wimax Base Station (or many other technologies)
  - Or some switches
  - Or WiFi access-points

AAA Server

- A RADIUS Server is a AAA server, because it handles:
  - **A**uthentication: making sure the user is who he says he is
  - **A**utorisation: tell the NAS what each user should have or should not have access to
  - **A**ccounting: log some data about each connection, such as the date and time the session started and stopped, the number of packets sent and received, etc.
- Other AAA protocoles: Diameter, TACACS

NAS Configuration

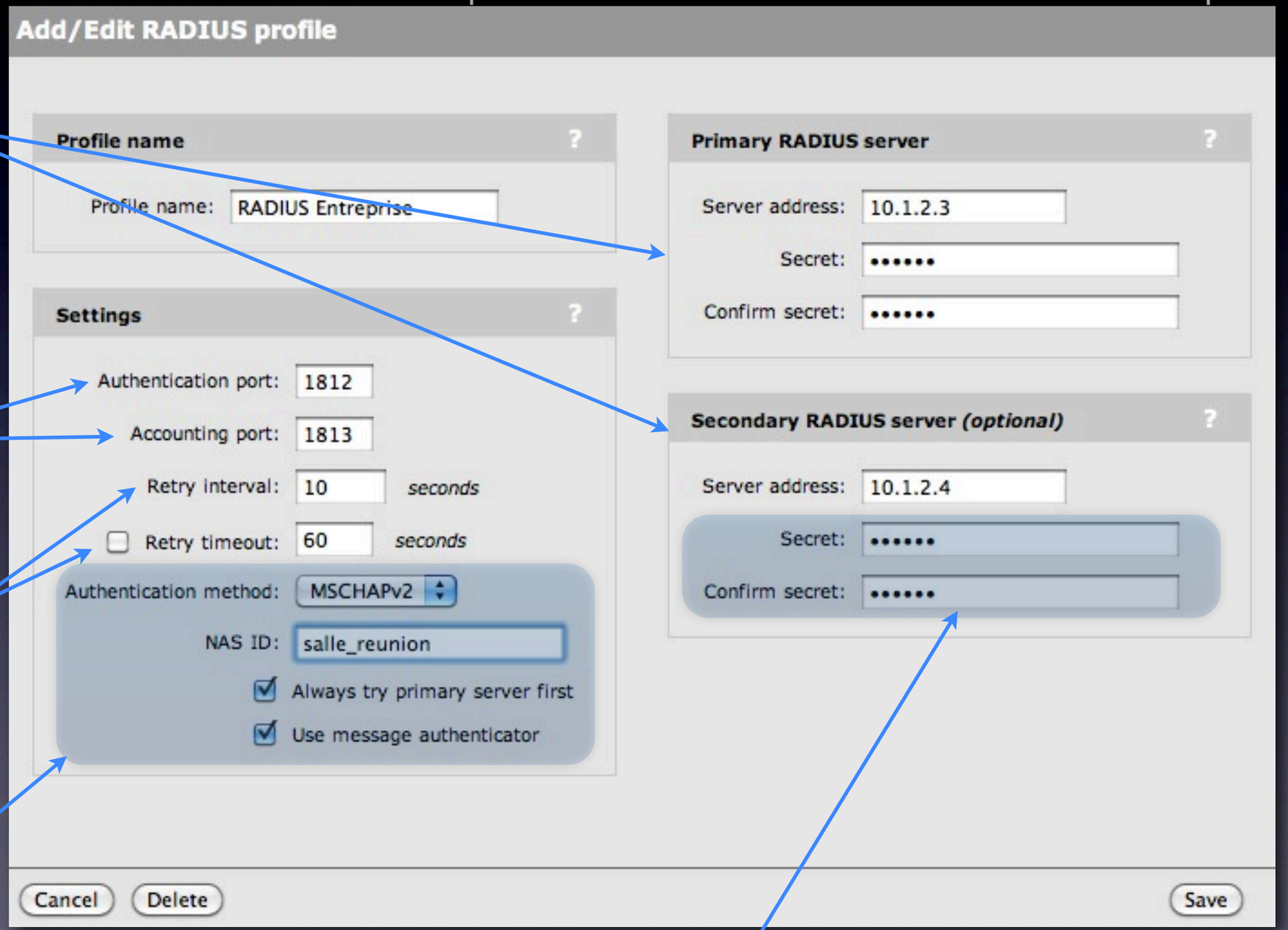
Example of the Web interface of an HP MSM310 WiFi access point

Redundent servers

Default UDP ports

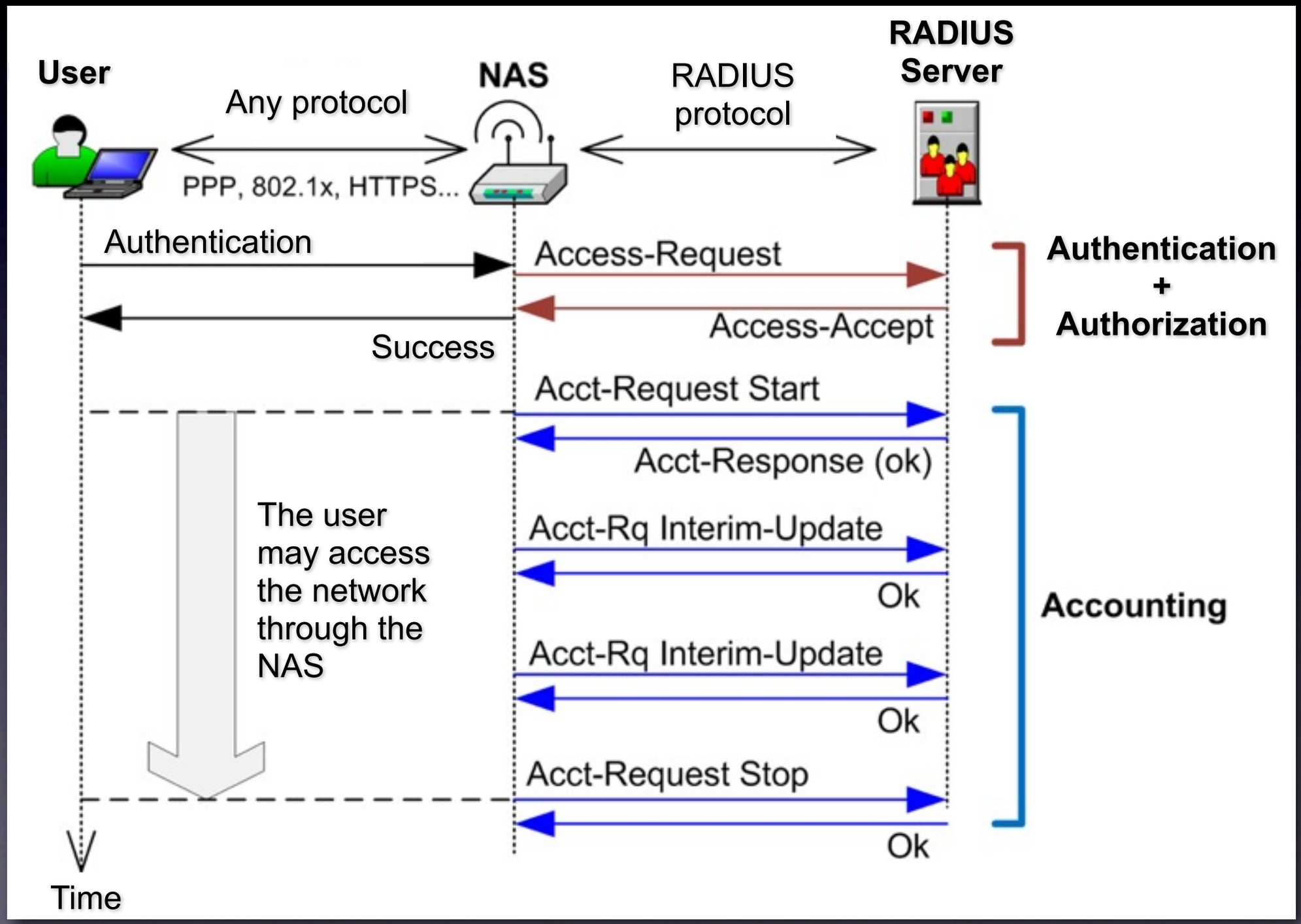
Fairly simple error handling

Other details

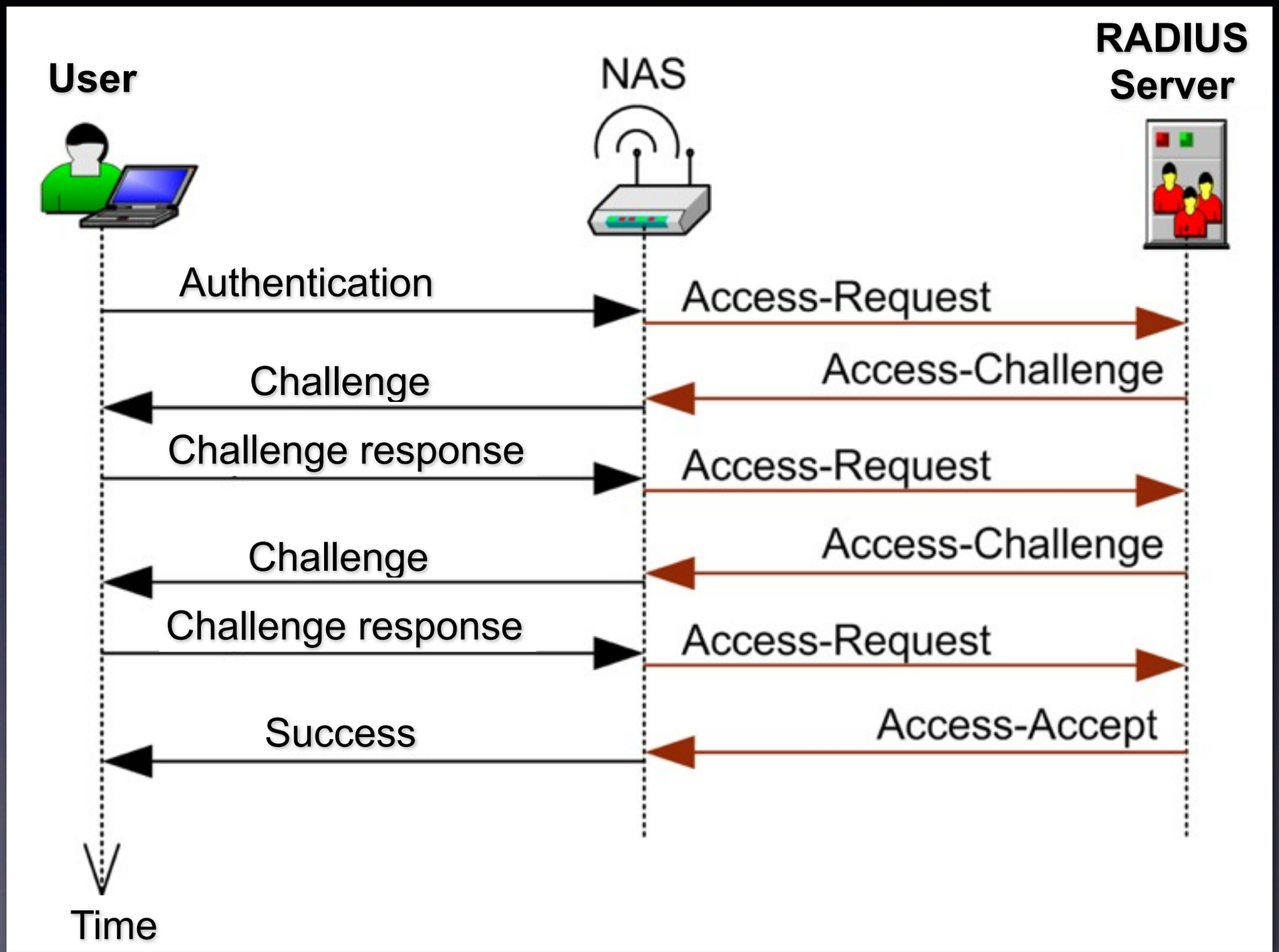


The secret allows the RADIUS server to identify the NAS

RADIUS Dialog Example



Authentication challenges



The RADIUS protocol

- It is defined in several RFCs:
  - RFC 2865 : the RADIUS protocol itself, including authentication and authorization
  - RFC 2866 : adds accounting
  - RFC 2869 : adds EAP authentication methods
  - And many more: http://freeradius.org/rfc/
- The procol relies on UDP, by default on port 1812 for authentication, and port 1813 for accounting

The RADIUS protocol

- Each packet starts with a header containing:
  - the packet type: Access-Request, Access-Accept, Access-Reject, Access-Challenge, Accounting-Request, Accounting-Response
  - the authenticator, which is a kind of signature for the server responses (we'll come back to that)
  - packet length and packet identifier
- The body of the RADIUS packet is composed of a list of Attribute/Value Pairs (AVPs, also called NVPs = Name Value Pairs)
  - Example : User-Name="alain"

Attribute-Value Pairs

- The attributes are defined and numbered in the RFCs
  - For example, the «User-Name» attribute is number 1, and its type is ‘string’ (ie. byte-array)
- In a RADIUS packet, the name and type of an attribute are not actually present, only its number and value
- In order to configure the NAS and the server easily, it is therefore necessary to have a dictionary that lists all possible attributes, with their name, number, type, and in some cases, the possible values
- The basic types are: ‘string’ (byte array), ‘text’ (UTF-8 encoded text), ‘address’ (IPv4), ‘integer’ (32 bits unsigned integer) and ‘time’ (a timestamp, including date+time). Other RFCs completed this list with other types.

Vendor-Specific Attr.

- The «Vendor-Specific» attribute (number 26) can itself contain a list of attributes defined by a vendor company
- It contains the number of the vendor company (attributed by the IANA), followed by a list of attributes whose specifications (type, usage...) are defined by the vendor
- Quite often, the vendor company just defines one attribute «AVPair», of type 'string', and the value itself must have the following format: «AttributeName=Value»
  - Example of a value: «login-page=https://a.b.c/»
- Sometimes a vendor-specific attribute ends up so popular that it is standardized

Integrity : Server to NAS

- The authenticator is a signature of the packet that allows the NAS to make sure that the response from the server actually comes from the server itself (and that the packet was not tempered with)

How does it work?

When a NAS sends a request, it sets the authenticator field to a random value.

When the server is about to send its response to the NAS, it calculates an MD5 hash based on that response packet + the secret shared with the NAS + the random number from the request's authenticator field. It then sets the response's authenticator field to this hash value. The NAS can then calculate the same hash and make sure that it matches the response's authenticator value.

Integrity : NAS to Server

By default, nothing allows the server to make sure that a request was not forged or tempered with

- To solve this issue, a «Message-Authenticator» attribute was defined in RFC 2869
- The NAS can add this attribute to a request before sending it to the server: it is an MD5 hash calculated on the request packet + the shared secret.
- It is recommended to configure the server as to reject packets that are not signed this way.

Confidentiality

RADIUS packets are not ciphered

AVPs are sent in cleartext, with a few exceptions

The value of some attributes is ciphered

PAP Authentication

- Two authentication methods were initially specified in the RADIUS RFC: PAP and CHAP
- With PAP, the user's password is ciphered with a fairly weak algorithm (based on the shared secret), and the result is sent to the server in the `User-Password` attribute
- The server deciphers the password, makes sure that it is correct, and then answers `Access-Accept` or `Access-Reject`

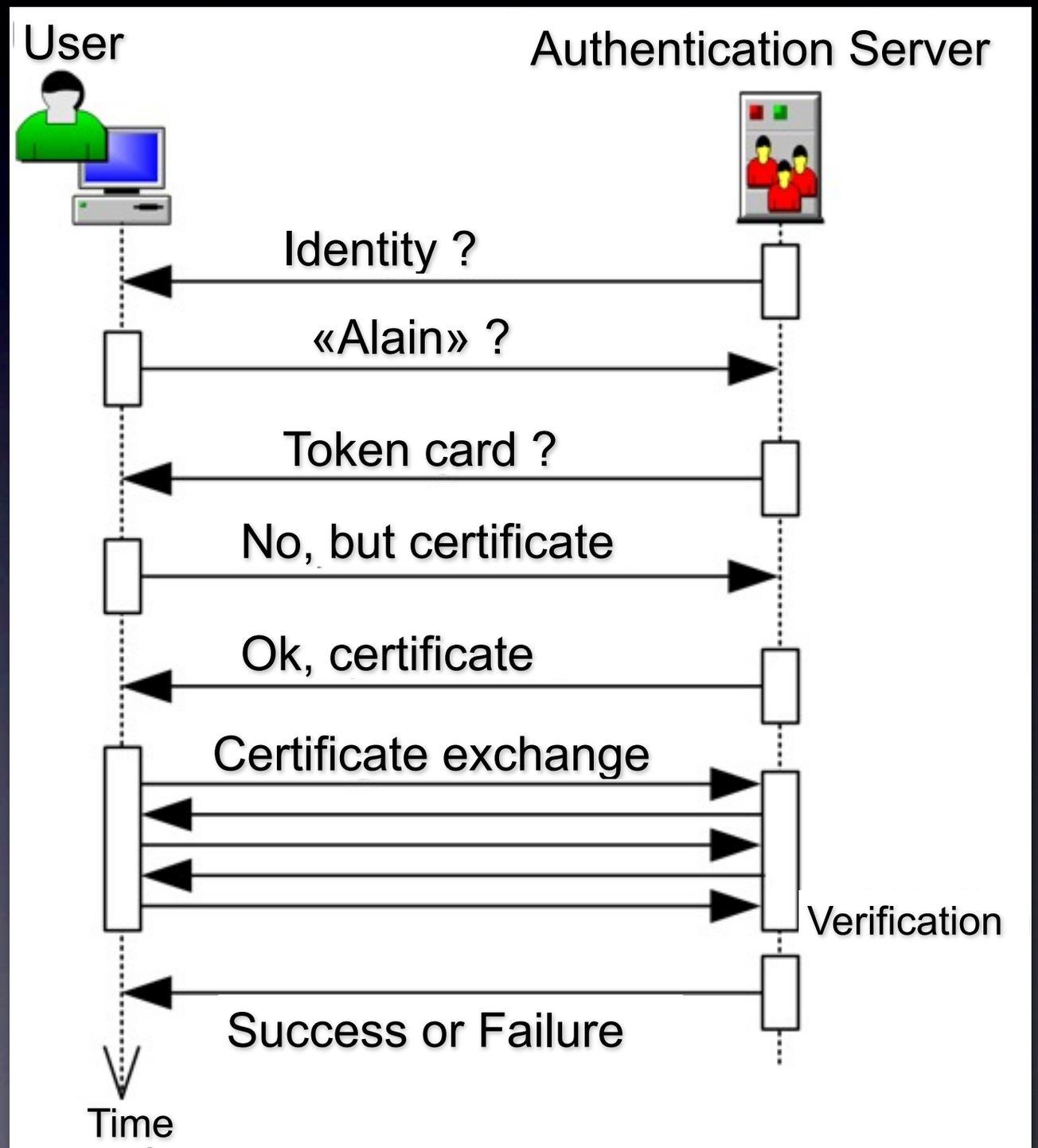
CHAP Authentication

- With CHAP authentication, the NAS sends a challenge to the user's system: it's simply a large random number
- The user's system requires the user to type his password, then it calculates an MD5 hash of that password + the challenge + a CHAP response identifier
- The hash is sent to the NAS, which then sends the CHAP-Identifier + the hash to the RADIUS server within a CHAP-Password attribute...
- ...along with the challenge value in the «CHAP-Challenge» attribute (or in the authenticator field if the challenge is 16 bytes long)
- Pros: the password is never sent on the wire (ciphered or not), since only a hash of the password is transmitted
- Cons: the RADIUS server must have access to the user's cleartext password in order to be able to check the received hash

EAP Methods

- The Extensible Authentication Protocol was defined to overcome the shortcomings of PAP and CHAP, allowing higher security and flexibility
- EAP is an independant protocol, not necessarily linked to the RADIUS protocol
- The EAP architecture only knows two actors: the user (in this case, it is called the «client»), and the authentication server (in our case, it is the RADIUS server). The NAS is out of the picture.
- During the EAP dialog, the client (ie. the user) gives his identity to the server, then an authentication method is negociated with the server, and finally the authentication itself takes place.

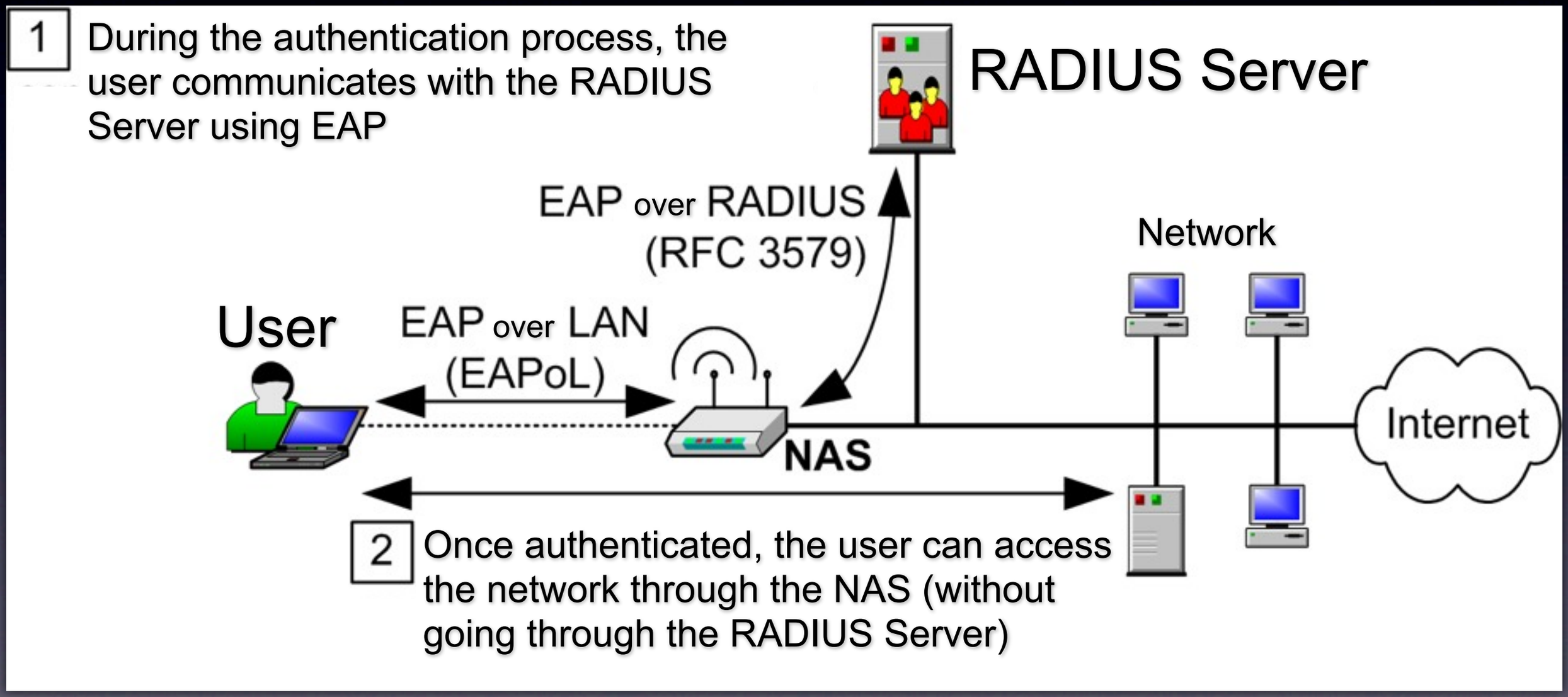
EAP Dialog


User
Authentication Server
Identity ?
«Alain» ?
Token card ?
No, but certificate
Ok, certificate
Certificate exchange
Verification
Success or Failure
Time

The 802.1x protocol

- The 802.1x protocol allows one to use both the RADIUS architecture and EAP authentication to manage connections to an Ethernet network (LAN)
- Between the user and the NAS, the EAPoL (EAP over LAN) protocol is used
- EAPoL is a variant of EAP which adds a few packet types, to allow the user to initiate the dialog (in a regular EAP dialog, the server always speaks first) and also to allow the exchange of cipher keys
- Between the NAS and the RADIUS server, the communication relies on the RADIUS protocol: the EAP packets are simply included in a special RADIUS attribute called «EAP-Message»

The 802.1x architecture



The main EAP methods

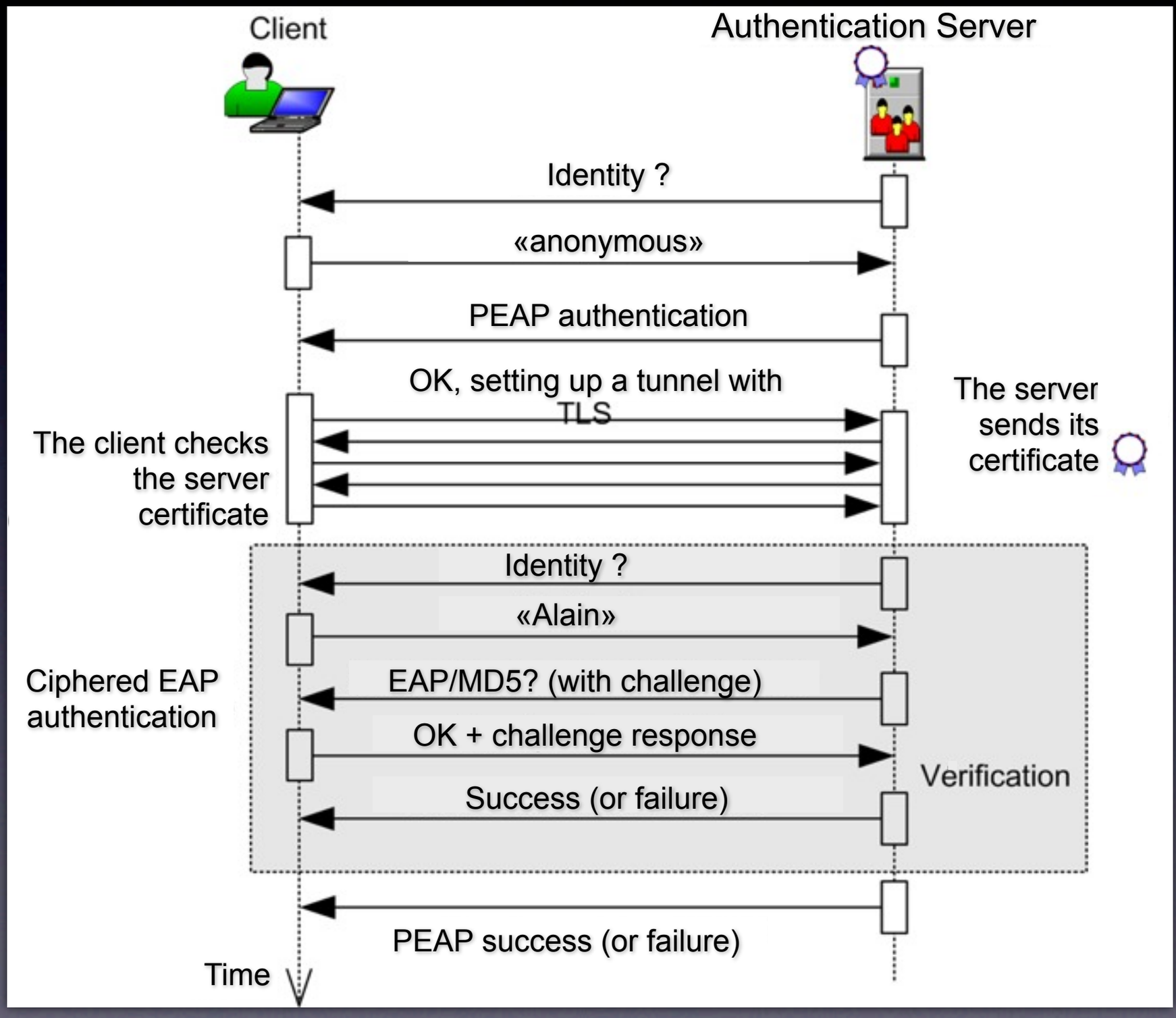
- EAP/MD5: equivalent to CHAP
- EAP/MS-CHAP-v2: similar to CHAP but it does not require the server to know the cleartext password of the user, but rather a hash of the password.
- EAP/GTC (Generic Token Card): very generic mechanism (the server sends a challenge, the user responds)
- EAP/SIM : SIM card authentication
- EAP/TLS : authentication by TLS certificates
- EAP/PEAP : establishes a TLS tunnel within which another EAP method (the inner-EAP method) can be securely used, say, EAP/MS-CHAP-v2. In this case, the whole authentication method is called PEAP/MS-CHAP-v2.

802.1x compatibility

To use a given authentication method, it is required that:

- the user's system be compatible with that method
- the RADIUS server be compatible as well
- the NAS just needs to be compatible with 802.1x: it does not care about the details of the EAP dialog, all it cares about it the outcome (accepted or rejected)

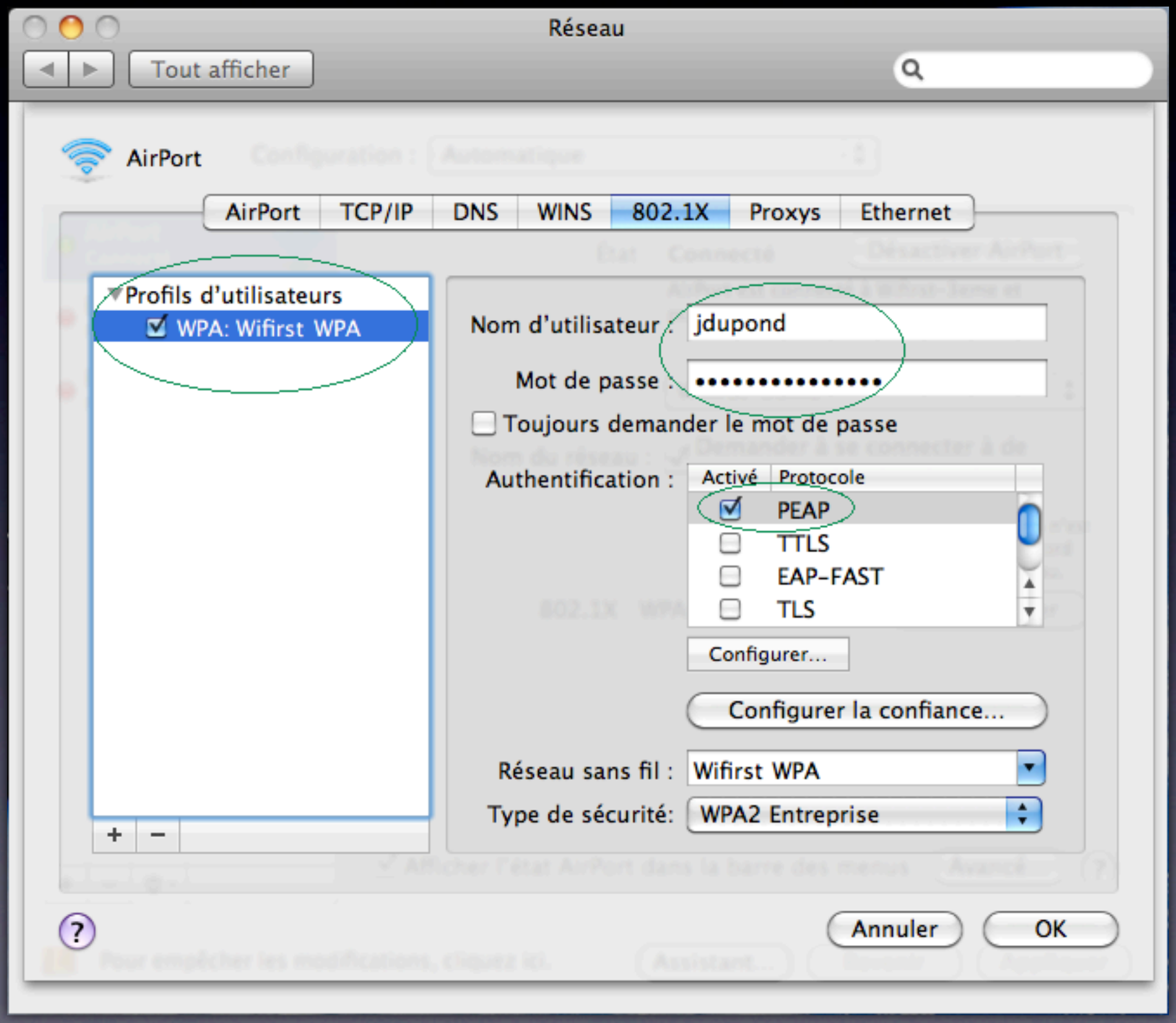
PEAP/MD5 dialog



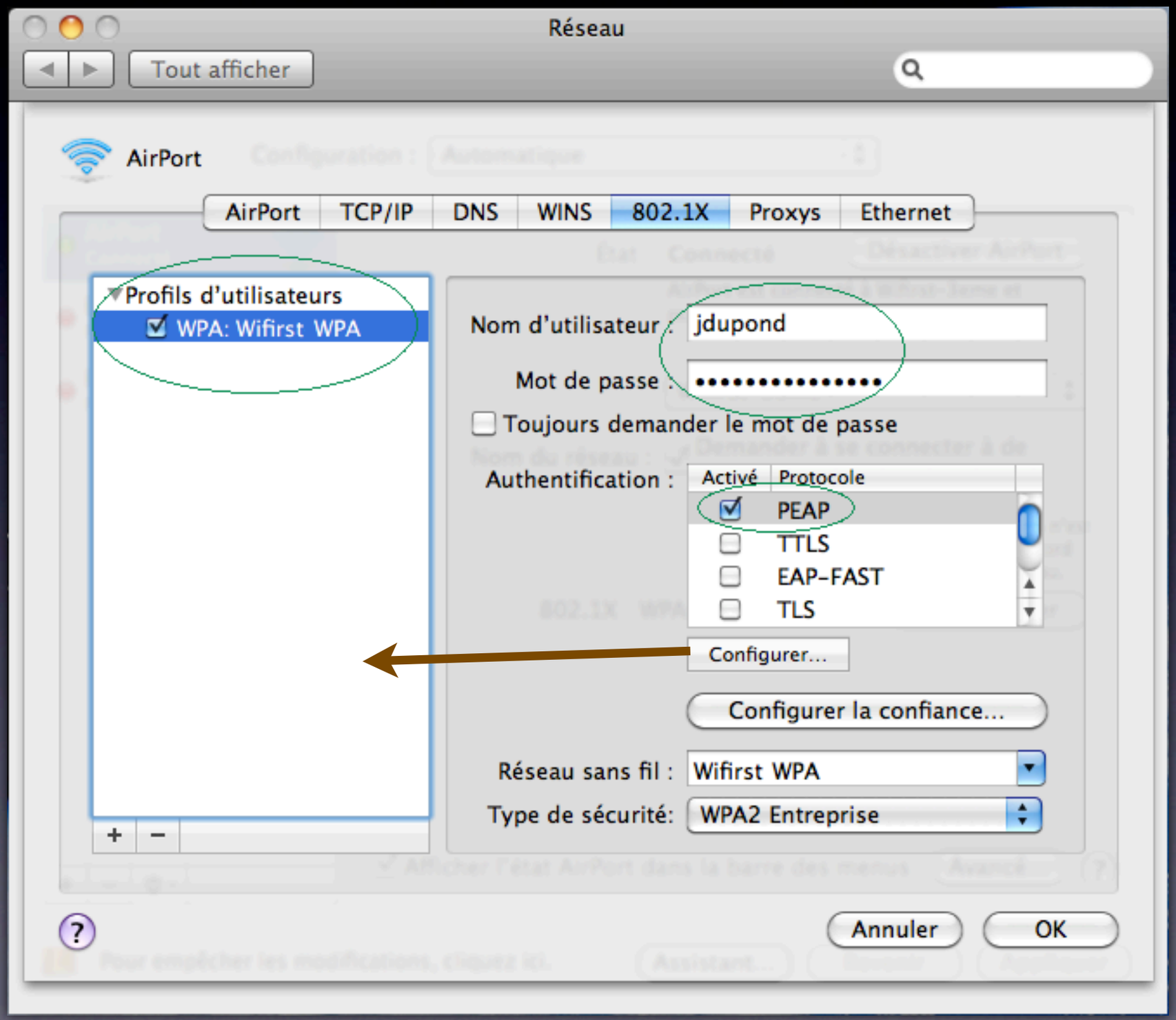
Check the certificate!

- The user must absolutely check that the server's TLS certificate is valid, otherwise there's a risk of a Man-in-the-Middle attack
- Beware: on Windows and MacOSX, when the user accepts a certificate, he implicitely accepts all certificates issued by the same certificate authority!
- To prevent this, the user must change his system configuration as follows...

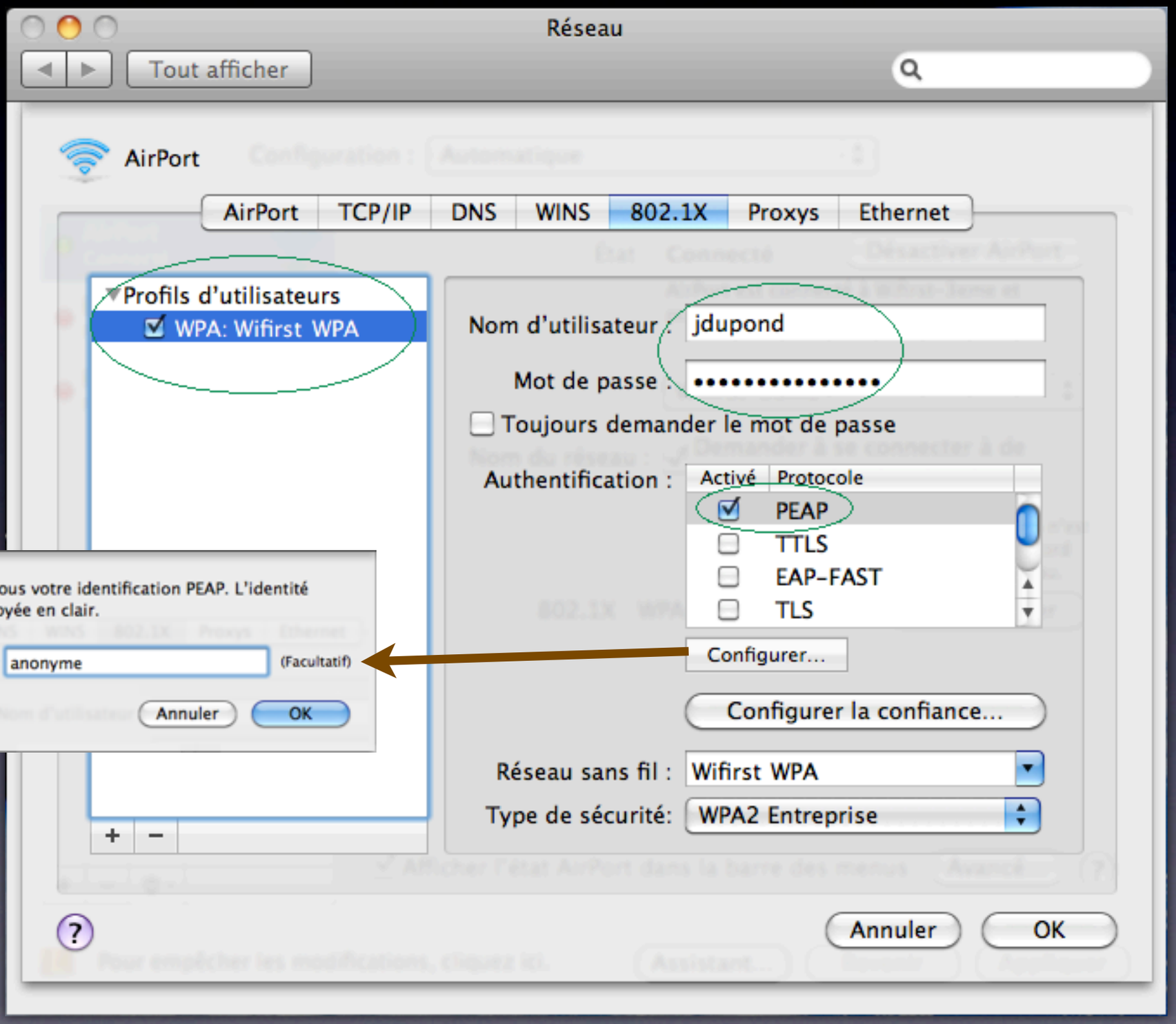
PEAP on MacOSX



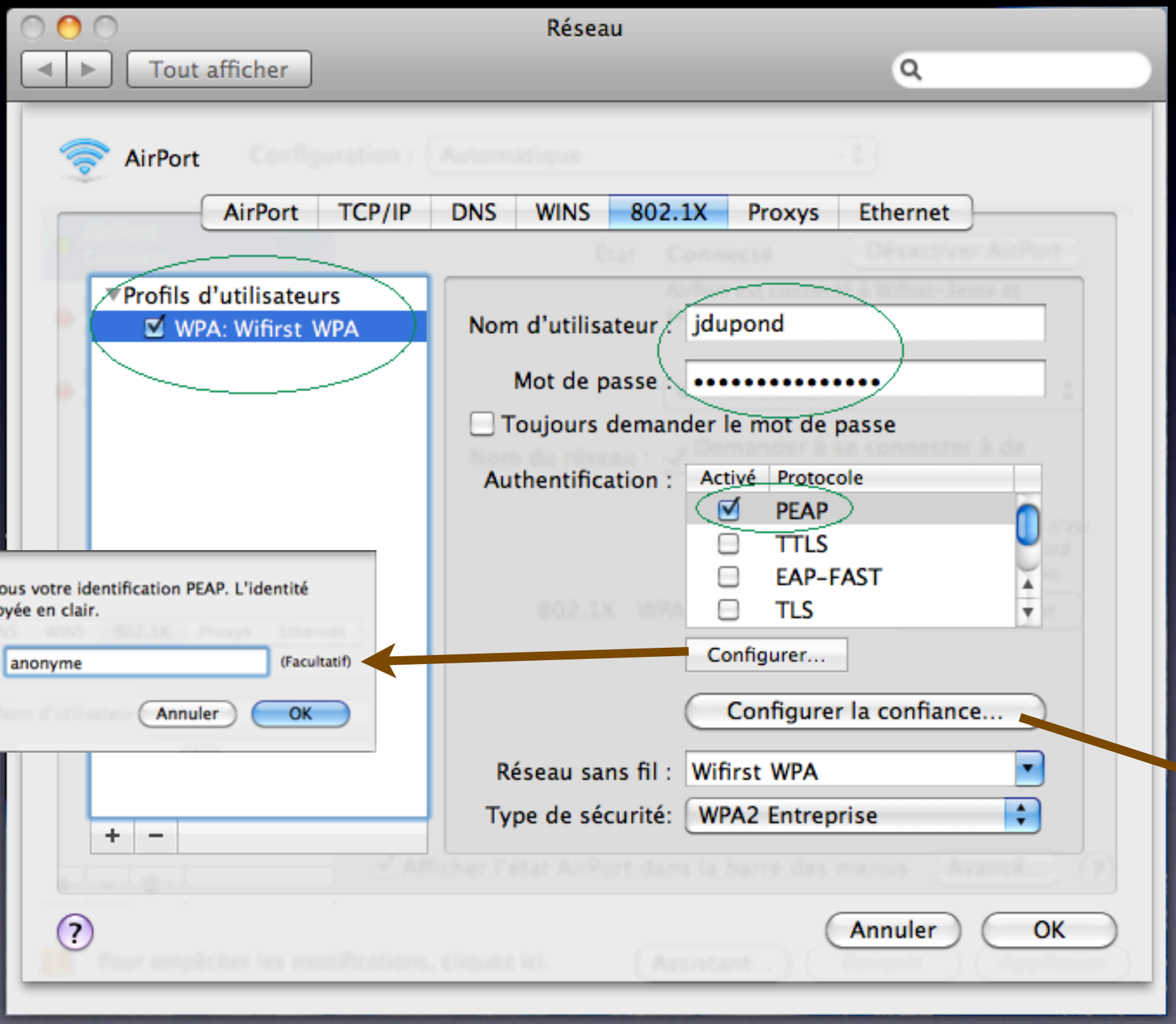
PEAP on MacOSX



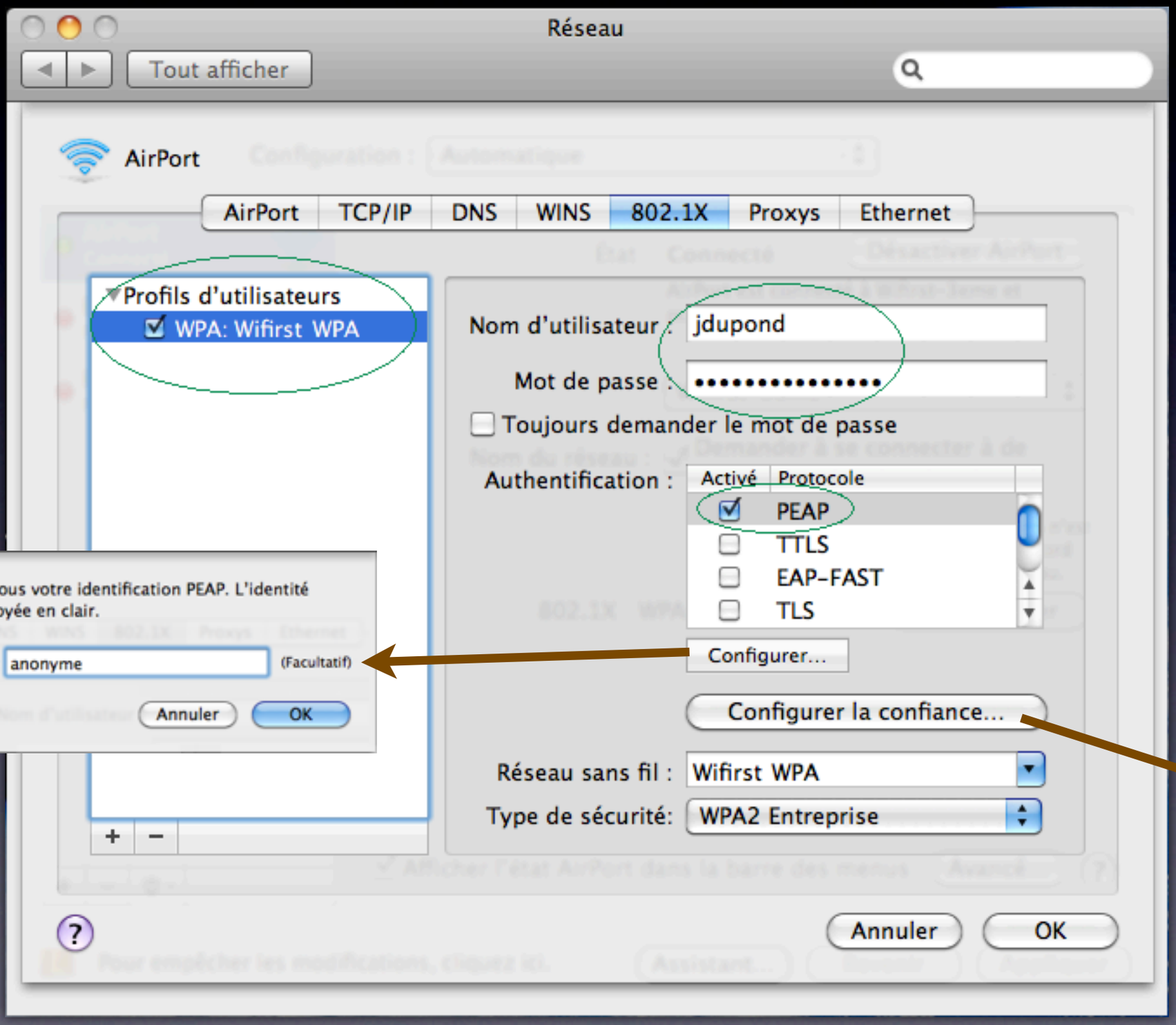
PEAP on MacOSX



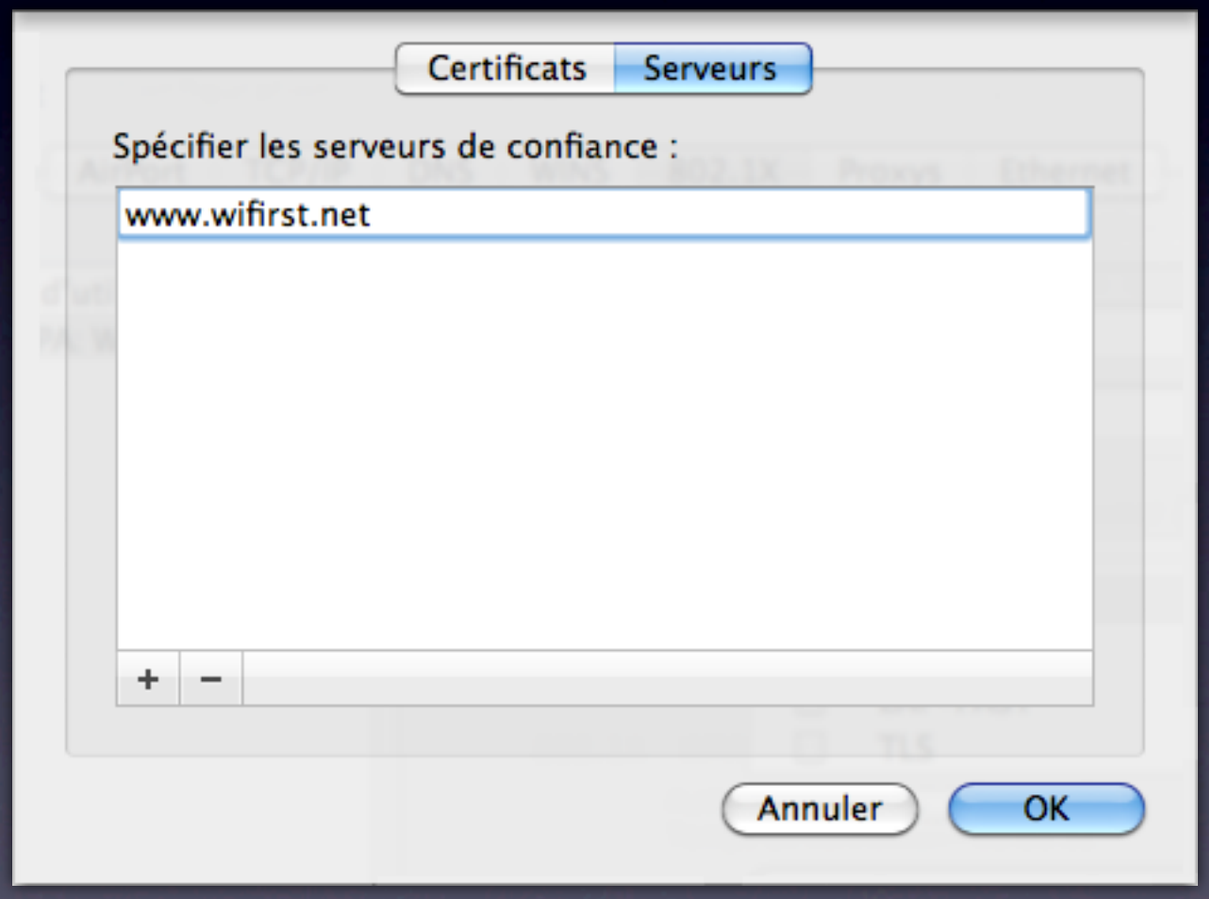
PEAP on MacOSX



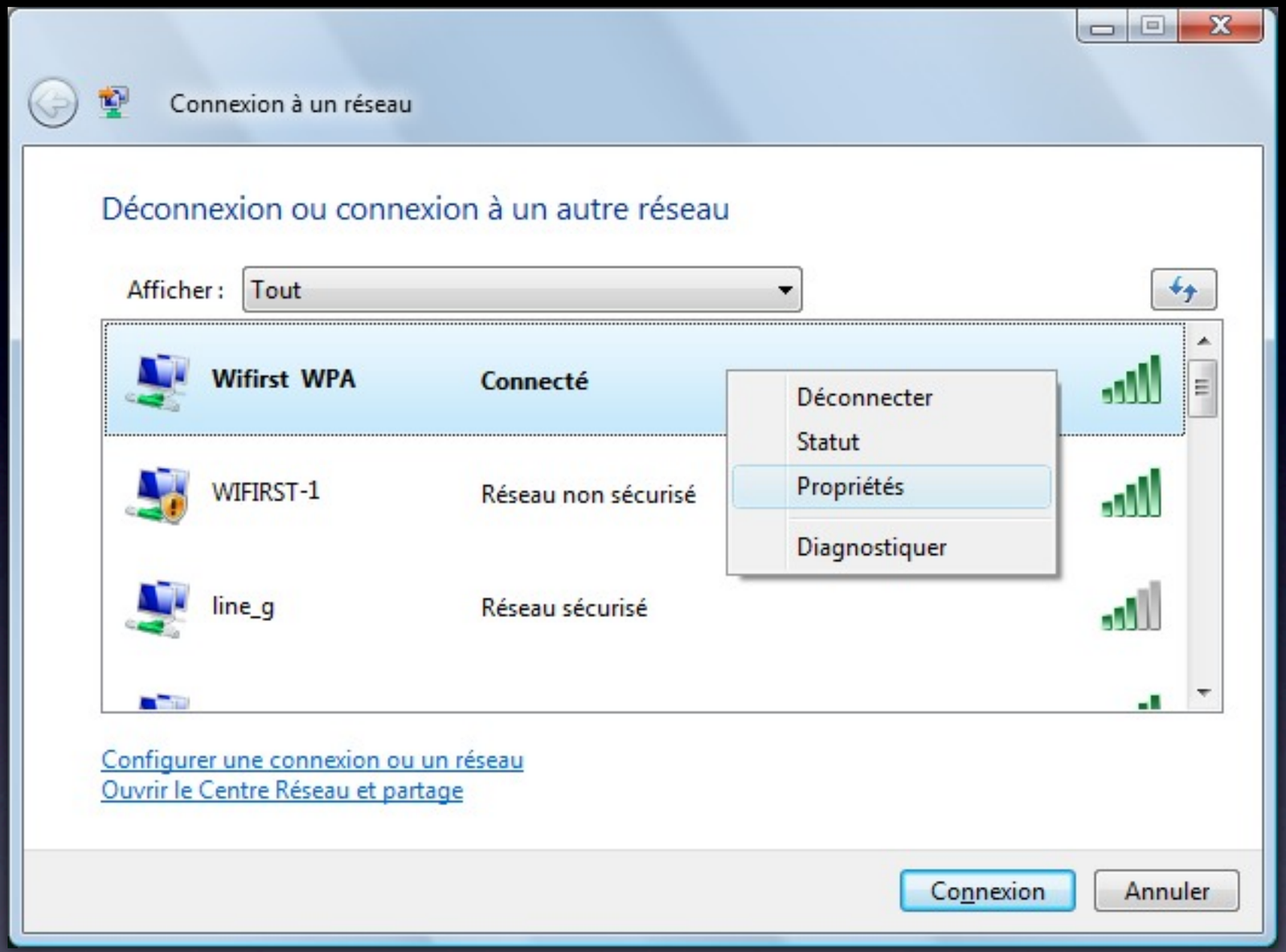
PEAP on MacOSX



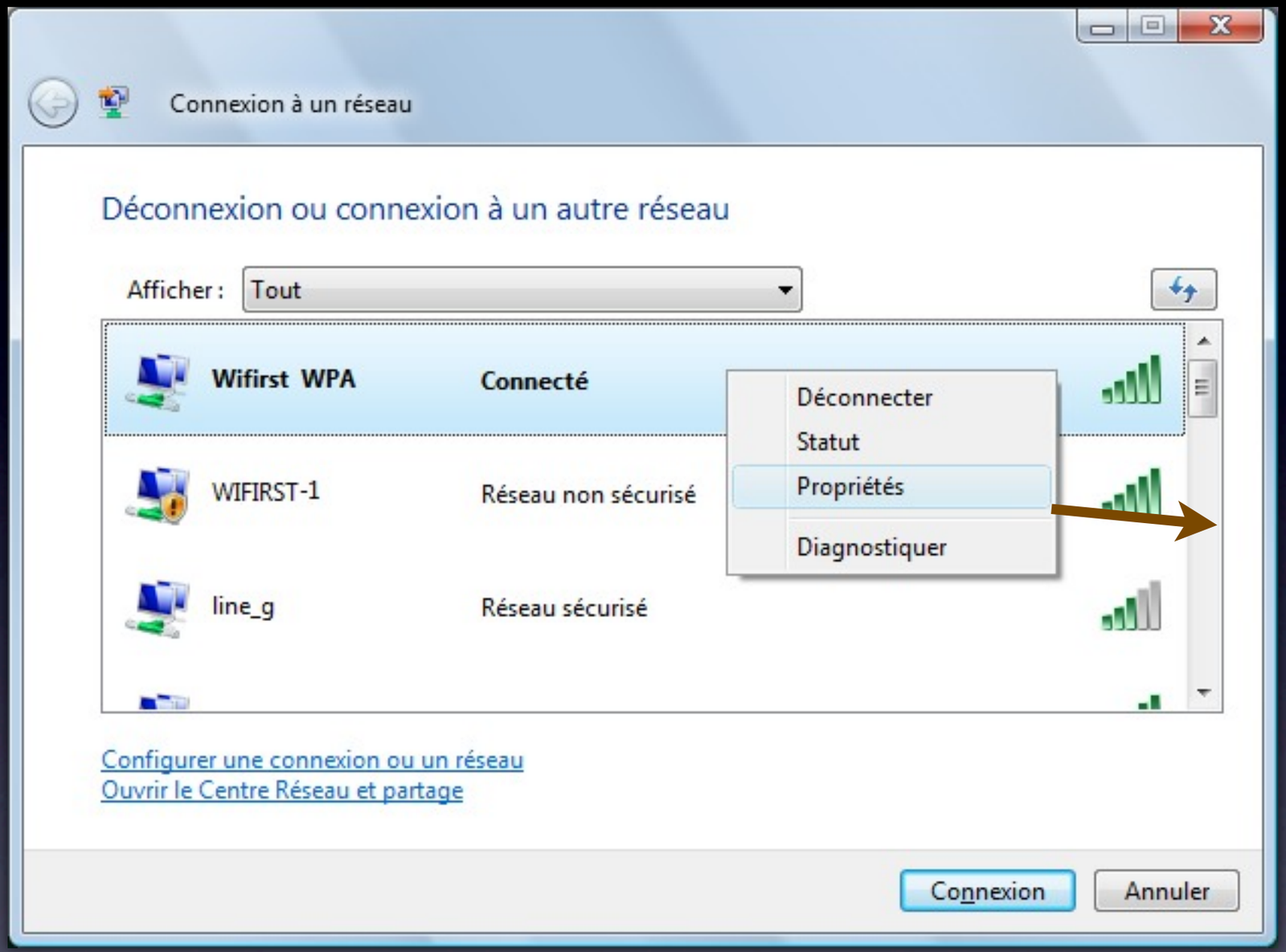
PEAP on MacOSX



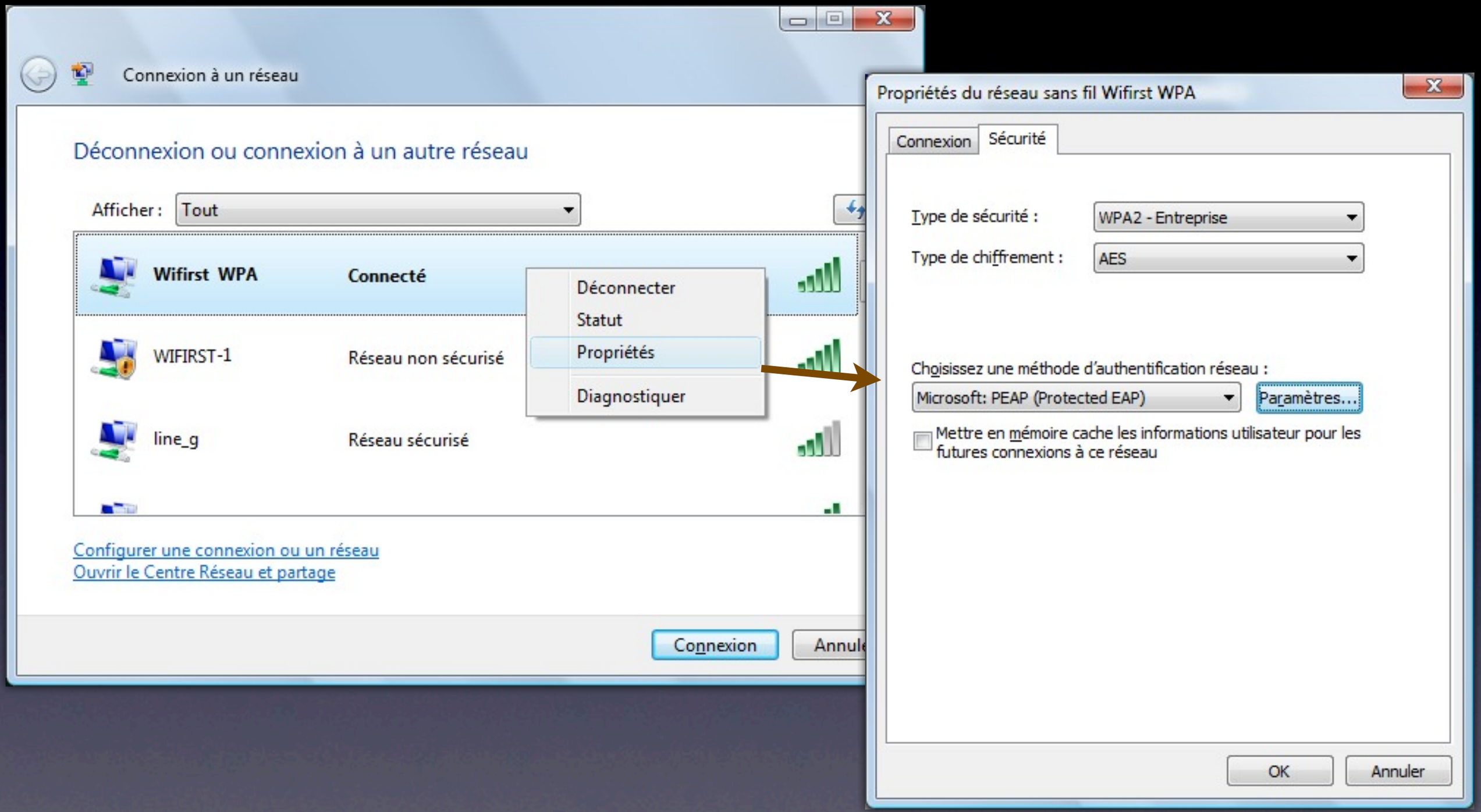
PEAP on Windows



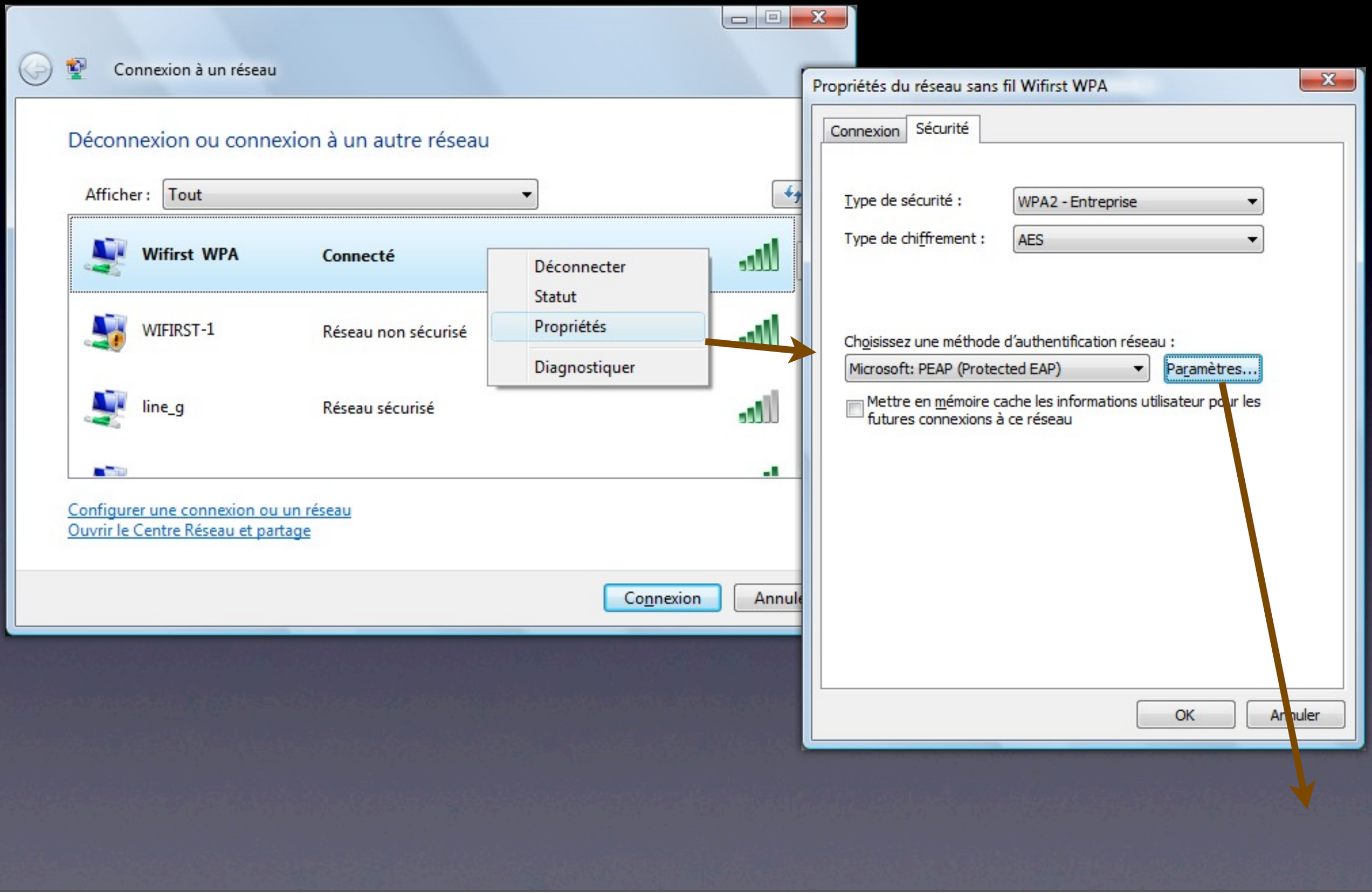
PEAP on Windows



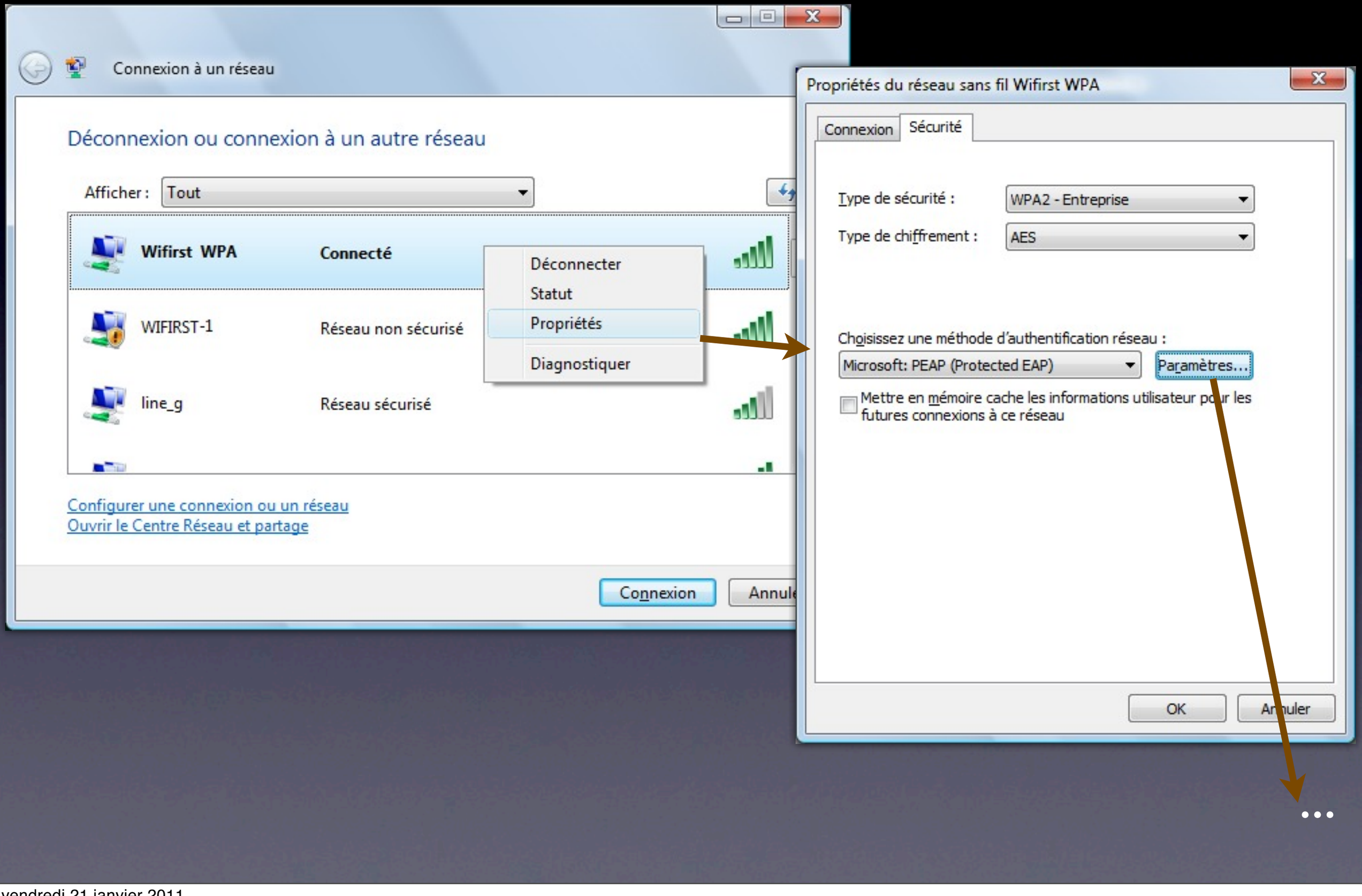
PEAP on Windows



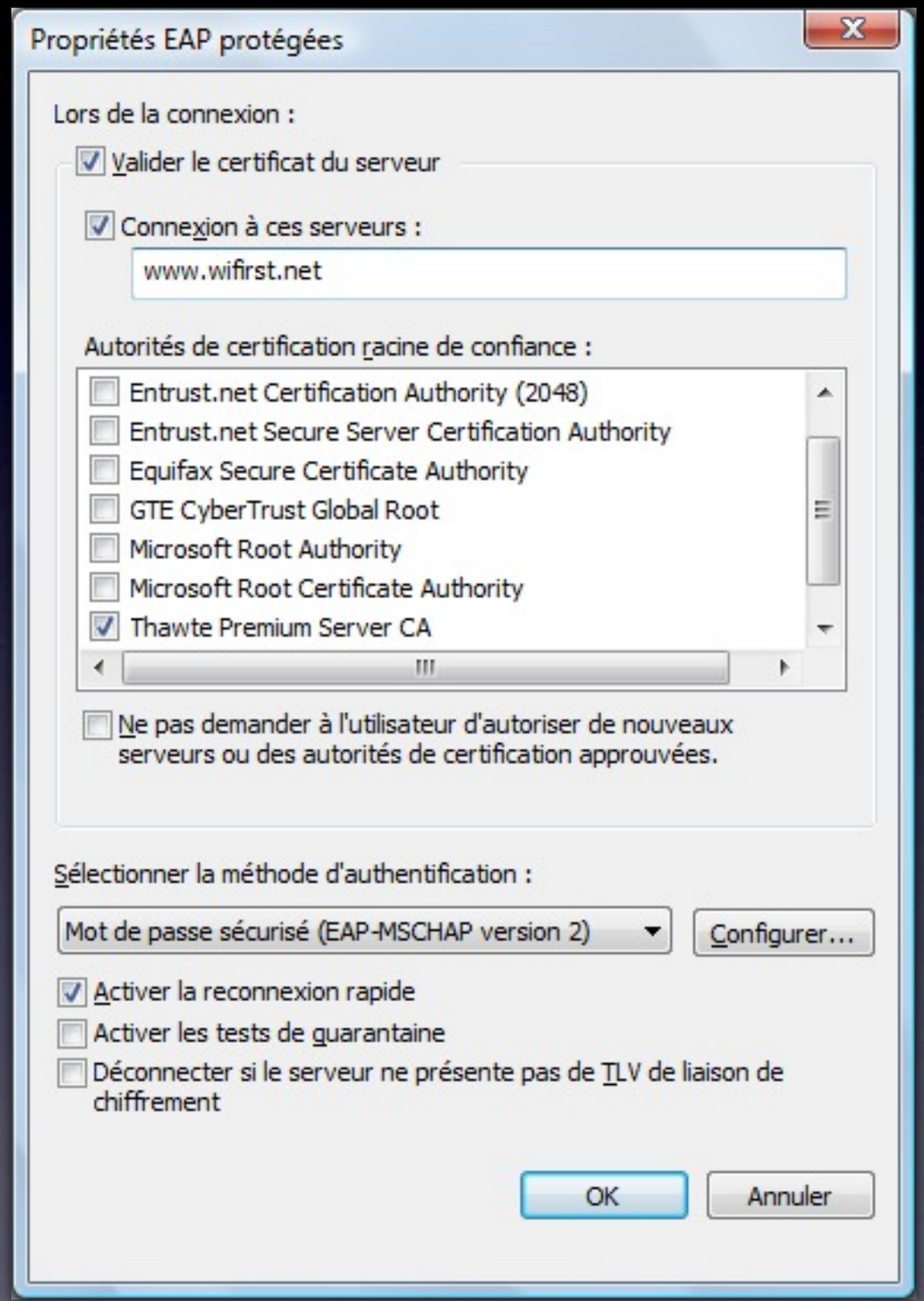
PEAP on Windows



PEAP on Windows



PEAP on Windows



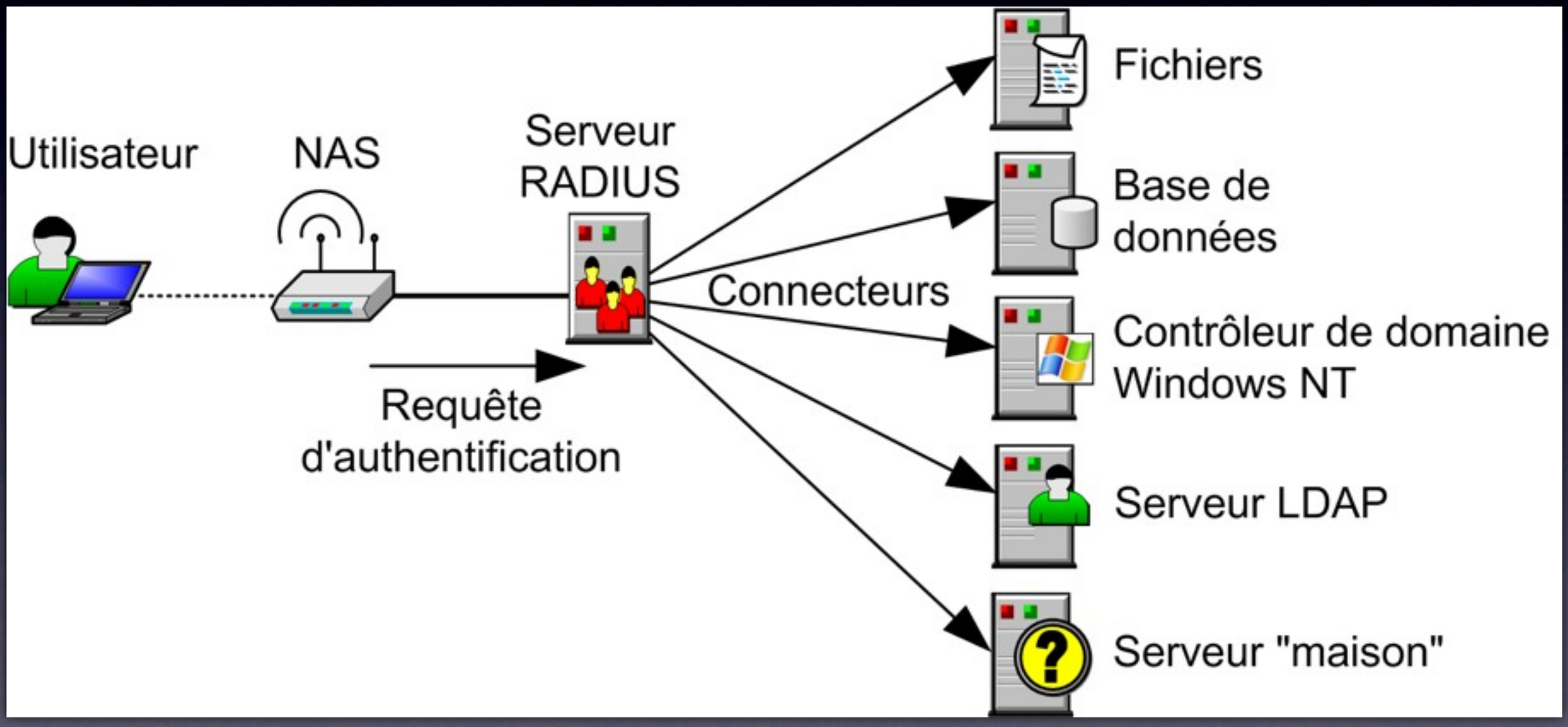
PEAP and accounting

- The PEAP tunnel only concerns authentication, not accounting
- The outer identity is therefore the one that will be used for accounting, so if it's «anonymous», you've got a problem
- The Chargeable-User-Identity attribute was created to tackle this problem

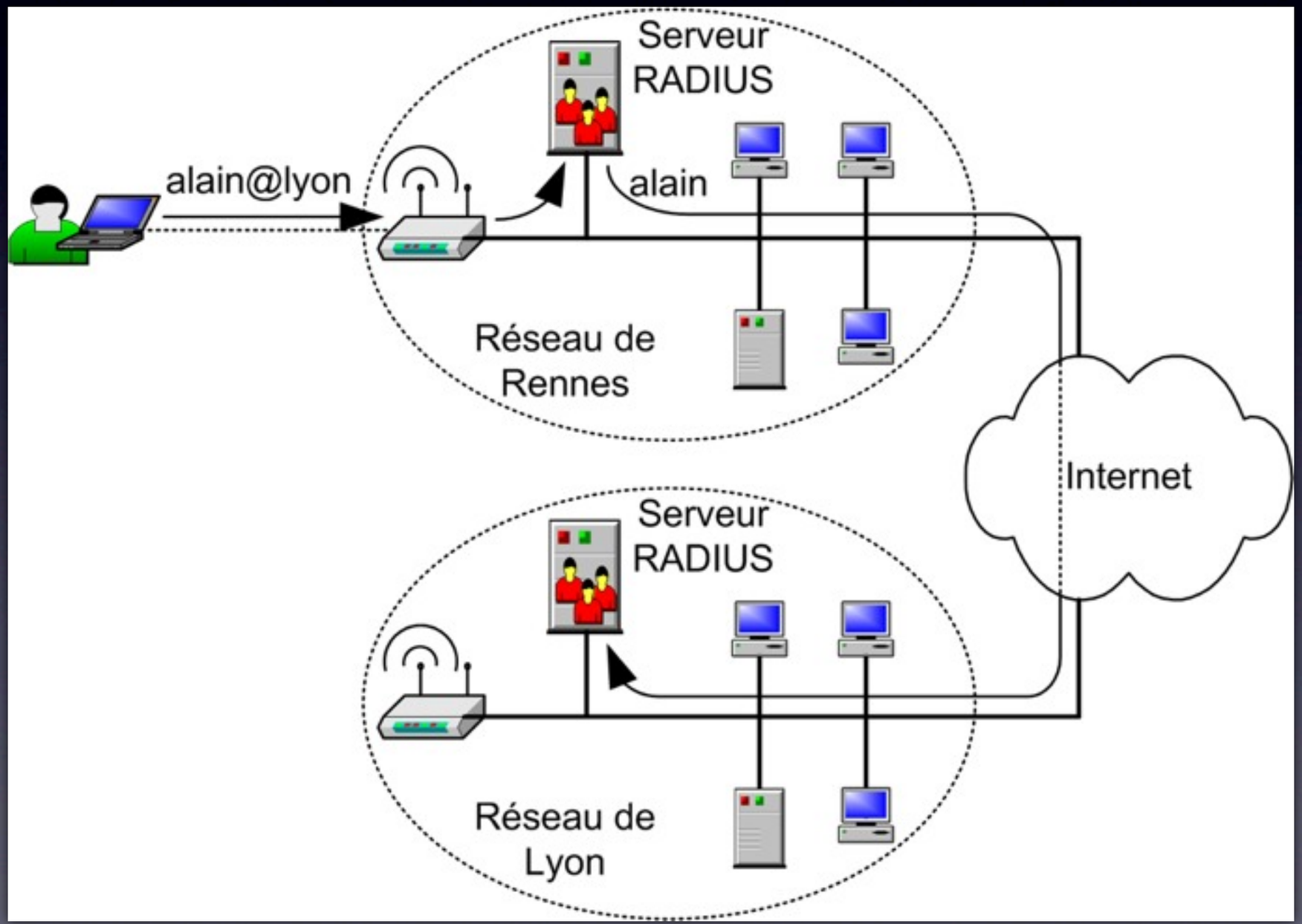
EAP/TTLS

- The EAP/TTLS authentication method is very similar to EAP/PEAP
- Its main problem is that Windows does not handle it by default: the user needs to install a software called a «TTLS supplicant»
- PEAP can only «tunnelize» EAP. The advantage of TTLS is that is also allows PAP, which makes it possible for the server to receive the cleartext password. It is sometimes useful, as we will see.

Authentication backends



What roaming is all about



Roaming terminology

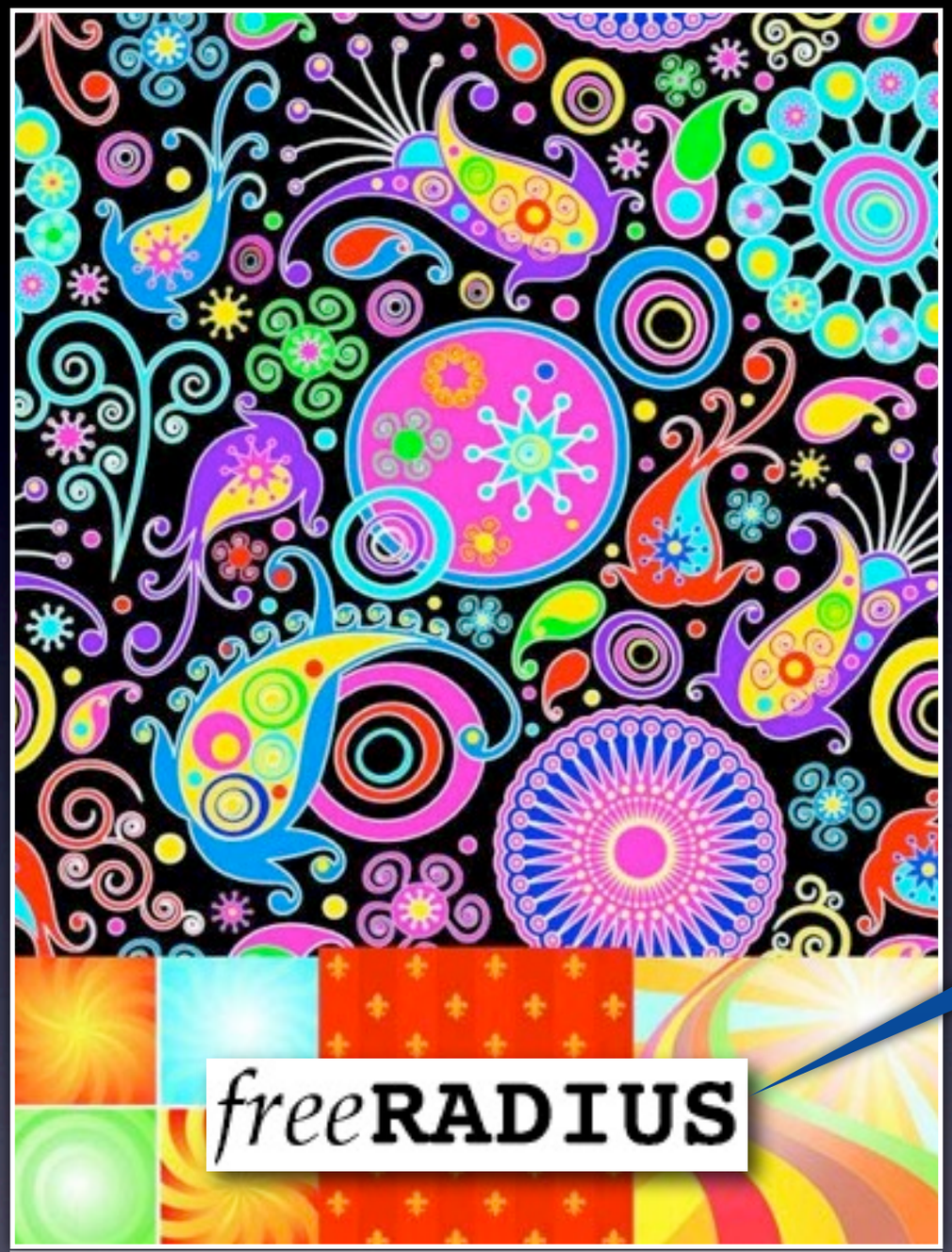
- Visited-Network : the network the user connects to
- Home-Network : the network the user comes from (where his account is configured)
- Realm : the part of the user's login that allows the server to tell what his Home-Network is:
  - `alain@lyon` in the previous example
  - the format can vary, for example: `fti/d9fl3g`
- Home-Server : the RADIUS server where the user is configured
- Proxy-Server : the RADIUS server of the Visited-Network

Roaming terminology

- Say we are a network operator, and we have settled a roaming agreement with operator X:
  - Roaming-in: is when a subscriber of operator X connects to our network (X owes us some money)
  - Roaming-out: is when one of our subscribers connects to X's network (we owe X some money)
  - Reconciliation: is when we compare our connect logs to those of operator X, and we agree on how much we owe them, and how much they owe us
  - Data Clearing House (DCH): is a company that takes care of reconciliation for large operators

Tunnels and proxying

- When a tunneled authentication method (PEAP or TTLS) is used in a roaming context, then the Proxy-Server only sees the external identity of the user (for example, «anonymous»)
- The user must therefore configure his external identity in such a way that the Proxy-Server can forward the requests to the right Home-Server (for example anonymous@lyon)
- If the internal identity also contains a realm, the Home-Server itself runs the risk of proxying the internal request to another RADIUS server: the packet's data would not be protected anymore:
    - this is why proxying is usually deactivated for internal EAP requests



Questions?