FreeRADIUS configuration

Jovana Palibrk, AMRES

NA3 T2, Sofia, 19.06.2014.

Who am I?

- jovana.palibrk@amres.ac.rs
- Academic network of Serbia
- Network security engineer
- Campus best practice task

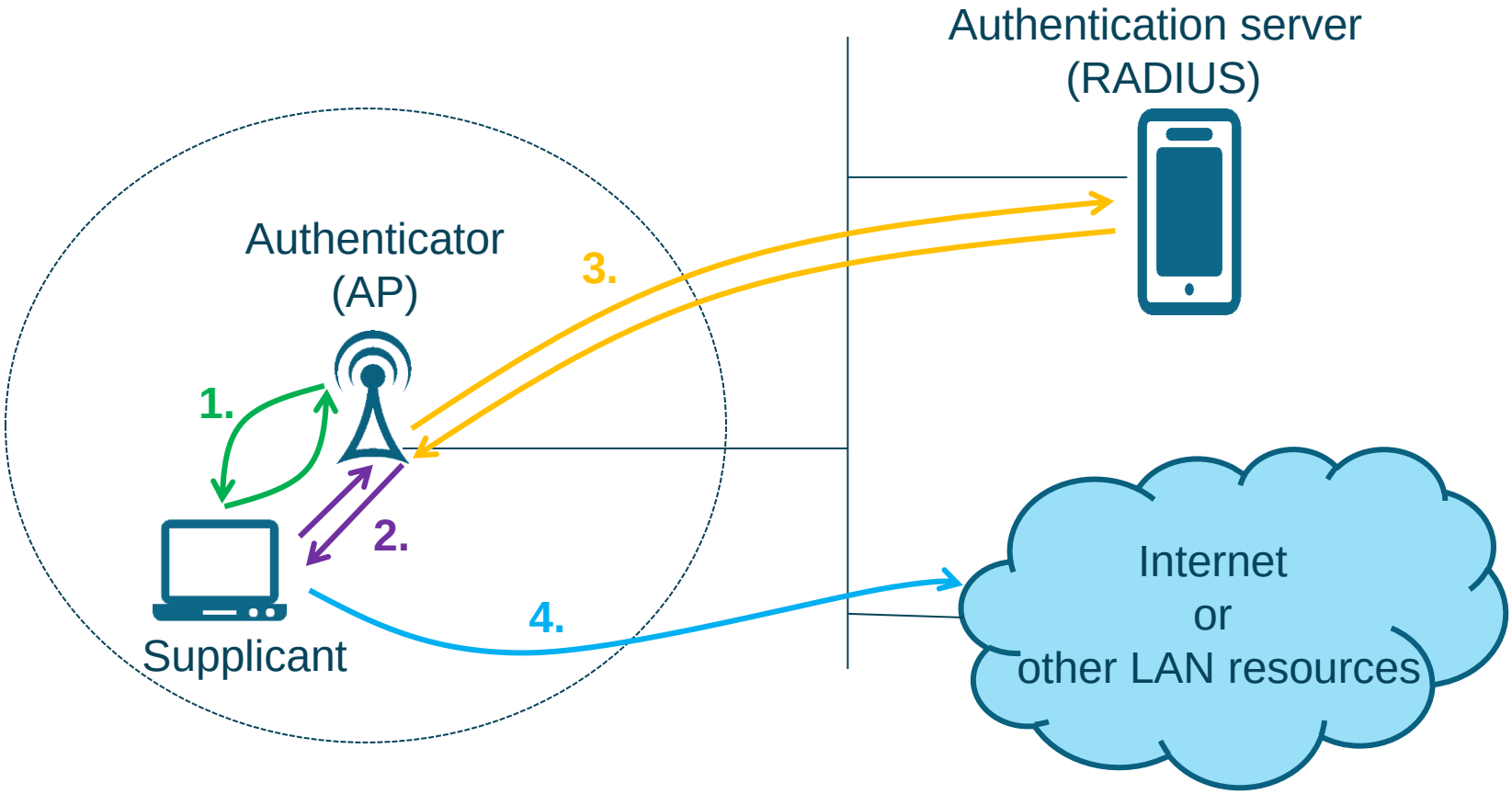
Contents

- Introduction
- FreeRADIUS platform
- FreeRADIUS server installation
- Authentication configuration
- Accounting configuration

Introduction

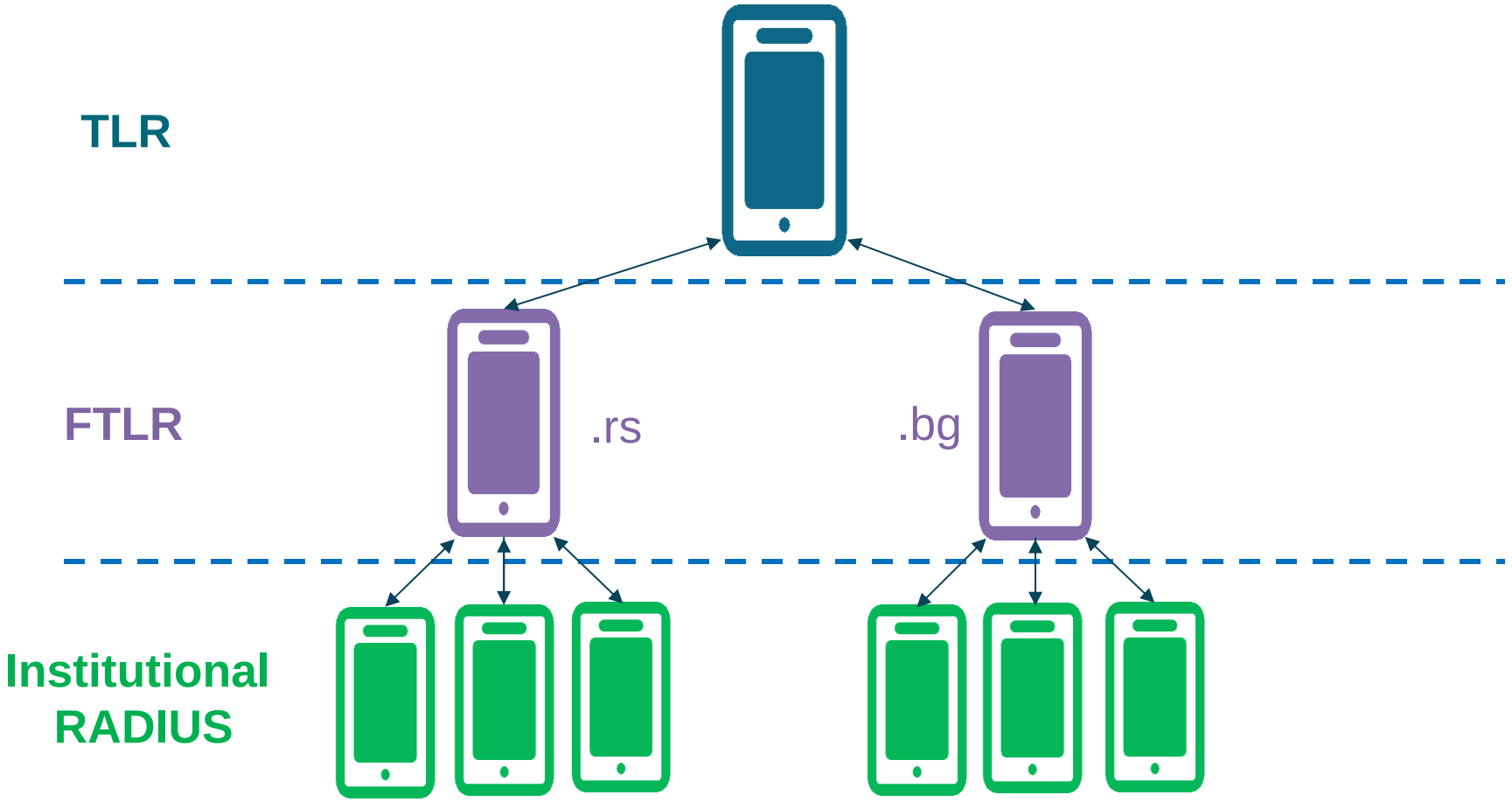
- Wireless infrastructure
- IEEE 802.1x standard
  - ***Supplicant*** – user device
  - ***Authenticator*** – access point
  - ***Authentication Server*** – *RADIUS server*

Introduction –
RADIUS/EAP authentication

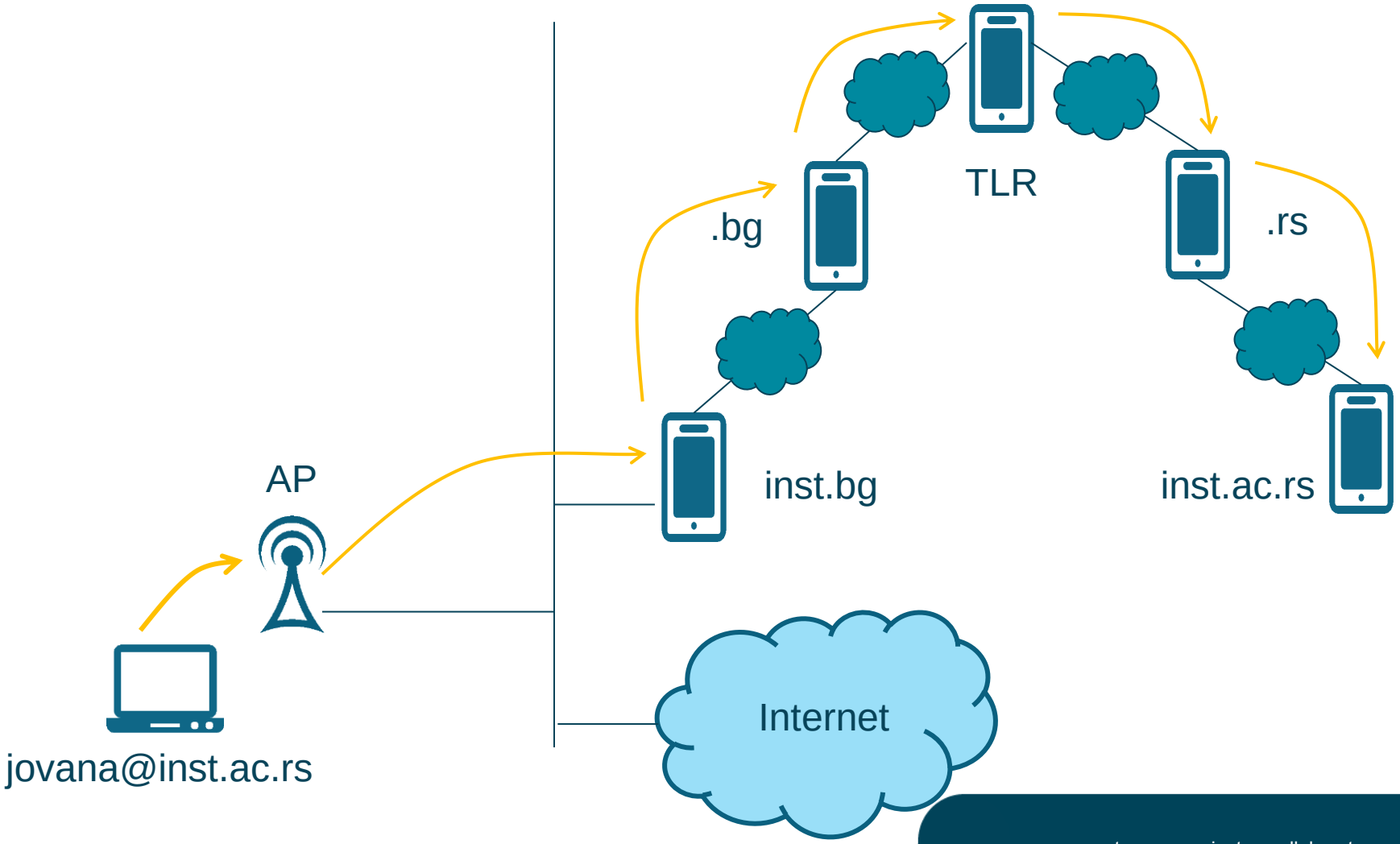


1. **Association request and response**
2. **EAP in 802.1x**
3. **EAP in RADIUS**
4. **Access to Internet or other LAN resources**

Introduction – eduroam



Introduction – eduroam



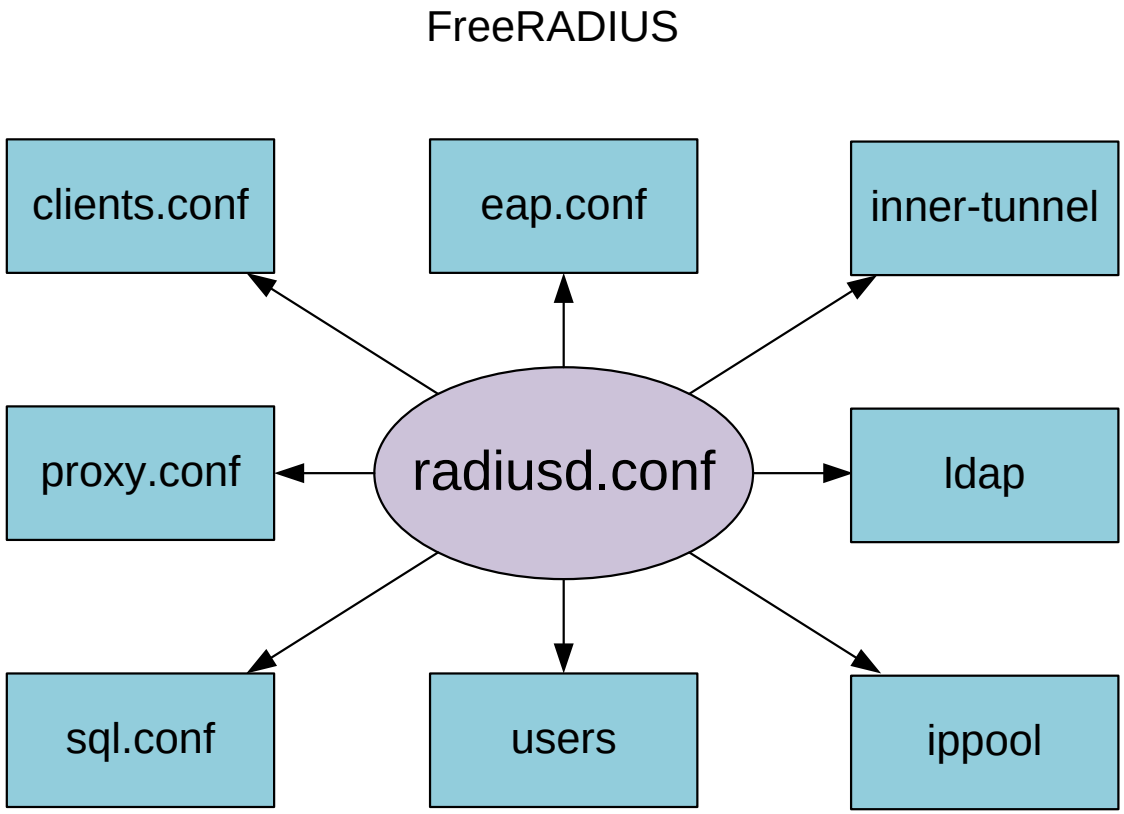
Introduction –
RADIUS/EAP authentication

- RADIUS – Remote Authentication Dial In User Service
- Networking protocol which provides centralized AAA service
  - “Who are you?” (Authentication)
  - “What services am I allowed to give you?” (Authorization)
  - “What did you do with my services while you were using them?” (Accounting)

FreeRADIUS platform

- www.freeradius.org
- *Open-source* project
- Current versions are **2.2.5** and **3.0.3**:
- Supported OSs:
  - Linux (**CentOS**, Debian, Mandriva, Red Hat, SUSE, Ubuntu)
  - FreeBSD
  - Solaris
  - OpenBSD..

FreeRADIUS platform



FreeRADIUS installation

- Before FreeRADIUS installation:
  - Make sure your system has ***gcc***, ***glibc***, ***binutils***, and ***gmake*** installed before trying to compile
- Other dependencies (based on modules that you need):
  - Openssl, openssl-devel – needed for FR EAP module to work
  - LDAP (if you have LDAP database)
  - MySQL

FreeRADIUS installation

- Installation (with output redirection):

```
./configure  -flags > text.file
make
make install (root privileges)
```

- You can use `-flags` to customize the settings (use `--help` to see all available flags)

FreeRADIUS installation

```
[root@radius freeradius-server-2.1.11]# ./configure --with-openssl > config.txt

configure: WARNING: snmpget not found - Simultaneous-Use and checkrad.pl may not work
configure: WARNING: snmpwalk not found - Simultaneous-Use and checkrad.pl may not work
configure: WARNING: pcap library not found, silently disabling the RADIUS sniffer.
configure: WARNING: silently not building rlm_counter.
configure: WARNING: FAILURE: rlm_counter requires:  libgdbm.
configure: WARNING: FAILURE: rlm_dbm requires:  (ndbm.h or gdbm/ndbm.h or gdbm-ndbm.h)
   (libndbm or libgdbm or libgdbm_compat).
configure: WARNING: silently not building rlm_dbm.
configure: WARNING: the TNCS library isn't found!
configure: WARNING: silently not building rlm_eap_tnc.
configure: WARNING: FAILURE: rlm_eap_tnc requires:  -lTNCS.
configure: WARNING: silently not building rlm_eap_ikev2.
configure: WARNING: FAILURE: rlm_eap_ikev2 requires:  libeap-ikev2 EAPIKEv2/connector.h.
configure: WARNING: silently not building rlm_ippool.
configure: WARNING: FAILURE: rlm_ippool requires:  libgdbm.
configure: WARNING: silently not building rlm_pam.
configure: WARNING: FAILURE: rlm_pam requires:  libpam.
configure: WARNING: silently not building rlm_python.
configure: WARNING: FAILURE: rlm_python requires:  Python.h.
configure: WARNING: silently not building rlm_sql_iodbc.
configure: WARNING: FAILURE: rlm_sql_iodbc requires: libiodb.
```

FreeRADIUS installation

```
configure: WARNING: silently not building
  rlm_ippool.

configure: WARNING: FAILURE: rlm_ippool requires:
  libgdbm.
```

FreeRADIUS installation

- **raddb** - FreeRADIUS directory:

```
cd /usr/local/etc/raddb
```

- All configuration files and modules are located in **raddb**, to list them use:

```
ls -la
```

FreeRADIUS installation

- Starting the server

```
radiusd
```

- Stopping the server

```
killall radiusd
```

- Check if the radius deamon will start (with default configuration)
- Starting the server in debugging mode:

```
radiusd -X
```

FreeRADIUS installation

```
Listening on authenticatio address * port 1812
Listening on accounting address * port 1813
Listening on command file /usr/local/var/run/radiusd/radiusd.sock
Listening on authentication address 127.0.0.1 port 18120 as server
  inner-tunnel
Listening on proxy address * port 1814
Ready to process requests.
```

CTRL + C

Authentication configuration

- Which EAP type to deploy
- EAP type configuration
- Virtual server configuration
- NAS client parameter configuration
- Connecting FreeRADIUS with user database
- Processing of Auth requests

Which EAP type to deploy

- Supported EAP authentication types (by FreeRADIUS):
  - **EAP-TLS**
  - **EAP-TTLS**
  - **PEAP**
  - EAP-GTC
  - LEAP
  - EAP-MD5

Which EAP type to deploy

- If your ID management infrastructure supports X.509 client certificates – then you can use **EAP-TLS**

- If your ID management infrastructure uses username/password:
  - Passwords in clear-text or as NT-hash? – **EAP-TTLS**, **PEAP**
  - If the passwords are in any other format - then you can use only **EAP-TTLS**

Which EAP type to deploy

	clear-text	NT-hash	MD5 hash	Salted MD5 hash	SHA1 hash	Salted SH1 hash	Unix Crypt
PAP	o	o	o	o	o	o	o
CHAP	o	x	x	x	x	x	x
Digest	o	x	x	x	x	x	x
MS-Chap	o	o	x	x	x	x	x
PEAP	o	o	x	x	x	x	x
EAP-MSCHAPv2	o	o	x	x	x	x	x
Cisco LEAP	o	o	x	x	x	x	x
EAP-GTC	o	o	o	o	o	o	o
EAP-MD5	o	x	x	x	x	x	x
EAP-SIM	o	x	x	x	x	x	x

EAP type configuration

raddb/eap.conf

```
$ cd /usr/local/etc/raddb/
$ joe eap.conf
```

EAP type configuration

raddb/eap.conf

```
eap {
      default_eap_type = ttls
      timer_expire     = 60
      ignore_unknown_eap_types = no
      cisco_accounting_username_bug = no

   tls {
       certdir = ${confdir}/certs
       cadir = ${confdir}/certs
       private_key_password = whatever
       private_key_file =${certdir}/private.key
       certificate_file = ${certdir}/server.pem
       CA_file = ${cadir}/ca.pem
       dh_file = ${certdir}/dh
       random_file = /dev/urandom
       fragment_size = 1024
       include_length = yes
       check_crl = no
       cipher_list = "DEFAULT"
       }
```

```
ttls {
   default_eap_type = md5
   copy_request_to_tunnel = no
   use_tunneled_reply = no
   virtual_server = "inner-tunnel"
     }

peap {
    default_eap_type = mschapv2
    copy_request_to_tunnel = no
    use_tunneled_reply = no
    virtual_server = "inner-tunnel"
      }

 mschapv2 {
       }

 }
```

EAP type configuration

raddb/eap.conf

```
eap {
      default_eap_type = ttls
      . . .
  tls {
       . . .
       private_key_file =${certdir}/private.key
       certificate_file = ${certdir}/server.pem
       CA_file = ${cadir}/ca.pem
       . . .
       }

   ttls {
        default_eap_type = md5

copy_request_to_tunnel = no
use_tunneled_reply = no
virtual_server = "inner-tunnel"
         } . . .
```

CTRL + K + F

CTRL + K + X

Virtual server creation

- Two virtual servers
  - First one processes requests before the EAP tunnel is established (“outer-tunnel”)
  - Second one processes requests inside the EAP tunnel (“inner-tunnel”)
- Location:
  - `raddb/sites-available/default`
  - `raddb/sites-available/inner-tunnel`
- Virtual servers are activated by creating symbolic link to a sites-enabled directory:
  - `raddb/sites-enabled/`

Virtual server creation

raddb/sites-available/outer-tunnel


```
$ cd sites-available
$ ls -la
-rw-r----- 1 root root 19174 Jun 14 15:30 default
-rw-r----- 1 root root 12328 Jun 14 15:30 inner-tunnel
$ cp default outer-tunnel
$ joe outer-tunnel
```

Virtual server creation

raddb/sites-available/outer-tunnel

```
server outer-tunnel {
authorize {
        preprocess
        chap
        mschap
        digest
        suffix
        eap
        files
        expiration
        logintime
        pap

}
authenticate {
        Auth-Type PAP {
                pap
        }
        Auth-Type CHAP {
                chap
        }
        Auth-Type MS-CHAP {
                mschap
        }
        digest
        unix
        eap
}
preacct {
        preprocess
        acct_unique
        suffix
        files
}
accounting {
        detail
        unix
        radutmp
        exec
        attr_filter.accounting_response
}
session {
        radutmp
}
post-auth {
        reply_log
        exec
        Post-Auth-Type REJECT {
                attr_filter.access_reject
        }
}
pre-proxy {
}
post-proxy {
        eap
}
}
```

Virtual server creation

raddb/sites-available/outer-tunnel

```
server outer-tunnel {
authorize {
. . .
pre-proxy {
}
post-proxy {
        eap
}
}
```

CTRL + K + V

CTRL + K + U

Virtual server creation

raddb/sites-available/inner-tunnel

```
$ cd sites-available
$ joe inner-tunnel
```

Virtual server creation

raddb/sites-available/inner-tunnel

```
server inner-tunnel {
authorize {
        suffix
        update control {
               Proxy-To-Realm := LOCAL
        }
        eap
        files
        expiration
        logintime
        pap
}
authenticate {
        Auth-Type PAP {
                pap
        }
        Auth-Type CHAP {
                chap
        }
        Auth-Type MS-CHAP {
                mschap
        }
        unix
        eap
}
session {
        radutmp
}
post-auth {
        Post-Auth-Type REJECT {
                attr_filter.access_reject
        }
}
pre-proxy {
}
post-proxy {
        eap
}
```

Virtual server creation

raddb/sites-enabled


```
$ cd ..
$ cd /sites-enabled
$ ln -s /usr/local/etc/raddb/sites-available/outer-tunnel
$ ls -la
default -> ../sites-available/default
inner-tunnel -> ../sites-available/inner-tunnel
outer-tunnel -> /usr/local/etc/raddb/sites-
  available/outer-tunnel
```

Virtual server creation

raddb/clients.conf

```
$ cd ..
$ joe clients.conf
```

Client parameter configuration

raddb/clients.conf

```
client AP-library {
   ipaddr              = 192.168.1.25
   secret              = mYs3cr3t
   shortname           = AP1
   nastype             = other
   virtual_server      = outer-tunnel
}

client radius2 {
   ipaddr               = 192.168.6.34
   secret               = uRs3cr3t
   shortname            = radius2
   nastype              = other
   virtual_server       = outer-tunnel
}
```

Client parameter configuration

raddb/clients.conf

```
client localhost {
        ipaddr = 127.0.0.1
        secret = testing123
        virtual_server = outer-tunnel
        require_message_authenticator = no
}
```

CTRL + K + X

Connecting to user database

- User database:
  - LDAP – Lightweight Directory Access Protocol
  - FreeRADIUS users file
- Additional configuration lines should be added to inner-tunnel
- Configuration of additional modules depends of database type

Connecting to user database - LDAP

- LDAP configuration file `/raddb/modules/ldap`

```
ldap {
    server = "localhost"
    identity = "uid=reader,ou=SystemAccounts,dc=bg,dc=ac,dc=rs"
    password = b1g$3cr3t
    basedn = "ou=People,dc=bg,dc=ac,dc=rs"
     ...
```

- Mapping between RADIUS and LDAP attributes is configured in `/raddb/ldap.attrmap`

```
checkItem       SMB-Account-CTRL-TEXT       acctFlags
checkItem       Expiration                  radiusExpiration
checkItem       Cleartext-Password          userPassword
checkItem       User-Name                   uid
#checkItem      Pool-Name                   ismemberof
```

Connecting to user database
- LDAP – `inner-tunnel`

```
authorize {
        suffix
        update control {
                Proxy-To-Realm := LOCAL
        }
        eap
        files
        ldap
        expiration
        logintime
        pap
}
authenticate {
        Auth-Type PAP {
                pap
        }
```

Connecting to user database - FR users file

- Manipulation with authentication requests
- Adding configuration parametar ***files*** to inner-tunnel:

```
server inner-tunnel {
authorize {
        auth_log
        eap
        files
        mschap
        pap
}
```

Connecting to user database - FR users file

```
$ cd /usr/local/etc/raddb
$ joe users
```

```
sofia Cleartext-Password:= “cbp“
```

CTRL + K + V

CTRL + K + X

Processing of Auth requests

- Do we want to process the requests only localy or some authentication requests requires proxying to another server?
- Relevant configuration file is `raddb/proxy.conf`

Processing of Auth requests
`proxy.conf` – Local

```
proxy server {
        default_fallback = no
}
home_server localhost {
        type = auth+acct
        ipaddr = 127.0.0.1
        port = 1812
        secret = testing123
        response_window = 20
        zombie_period = 40
        revive_interval = 120
        status_check = status-server
        check_interval = 30
        num_answers_to_alive = 3
}
realm workshop.bg {
        authhost    = LOCAL
        accthost    = LOCAL
        User-Name   = "%{Stripped-User-Name}"
}
realm LOCAL {
}
realm NULL {
}
```

Processing of Auth requests
`proxy.conf` – Local

```
proxy server {
        default_fallback = no
}
. . .
realm workshop.bg {
        authhost    = LOCAL
        accthost    = LOCAL
        User-Name   = "%{Stripped-User-Name}"
}
realm LOCAL {
}
realm NULL {
}
```

CTRL + K + V

CTRL + K + X

Processing of Auth requests
`proxy.conf` – Local + Proxy

```
home_server radius2 {
        type = auth+acct
        ipaddr = 192.168.14.15
        port = 1812
        secret = r@diu$
        response_window = 20
        zombie_period = 40
        revive_interval = 120
        status_check = status-server
        check_interval = 30
        num_answers_to_alive = 3
}
home_server_pool radius2 {
        home_server = radius2
}
realm DEFAULT {
        pool = radius2
        nostrip
}
```

Testing

- eapol_test - http://deployingradius.com/scripts/eapol_test/
  - EAP testing tool
  - Part of wpa supplicant
- Command
  - `eapol_test -c ttls-pap.conf -s testing123`

Testing

```
$ cd /usr/local/etc/raddb
$ joe ttls-pap.conf
```

Testing - ttls-pap.conf

```
#
# eapol_test -c ttls-pap.conf -s testing123
#
network={
        ssid="example"
        key_mgmt=WPA-EAP
        eap=TTLS
        identity="sofia@workshop.bg"
        anonymous_identity="anonymous@workshop.bg"
        password="cbp"
        phase2="auth=PAP"

        #
        #  Uncomment the following to perform server
  certificate validation.
#       ca_cert="/etc/raddb/certs/ca.der"
}
```

CTRL + K + X

Testing - testing123

```
client localhost {
        ipaddr = 127.0.0.1
        secret = testing123
        virtual_server = outer-tunnel
        require_message_authenticator = no
}
```

Testing

```
$ cd /usr/local/etc/raddb
$ joe ttls-pap.conf
$ eapol_test -c ttls-pap.conf -s testing123
```

Accounting configuration

- Depends of whether the devices that you use as NAS supports RADIUS Acct (Cisco, Lancom)
- MySQL configuration:
  - Create a table (table examples can be found in `raddb/sql/mysql/`)
  - Create a user with write priviledges
- FreeRADIUS configuration:
  - Create accounting queries in *`something`*`.conf` in `raddb/sql/mysql/`
  - Edit `raddb/sql.conf`

Accounting configuration
raddb/sql.conf

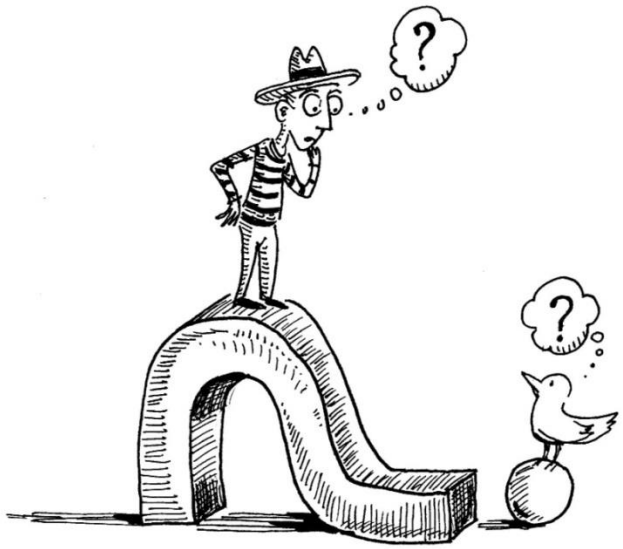
```
sql ws-test {
        . . .
        server = “192.168.14.23"
        login = “jupiter"
        password = “s@turn"
        radius_db = "radius"
        acct_table1 = “table1“
        acct_table2 = “table1"
        . . .
        $INCLUDE sql/${database}/something.conf
}
```

Accounting configuration

raddb/sites-available/outer-tunnel

```
...
preacct {
        preprocess
        acct_unique
        suffix
        files
}
accounting {
        ws-test
        detail
        unix
        radutmp
        exec
        attr_filter.accounting_response
}
session {
        radutmp
}
```

Questions?



Thank you!