LTE IMS Server

Version: 2025-12-12

Table of Contents

1 Introduction

LTEIMS is an IMS standalone simple server. It has a built-in P-CSCF, I-CSCF, S-CSCF, HSS. It also allows SMS handling including SMS over SG by connecting to the Amarisoft MME.

2 Features

- Implements P-CSCF with built-in I-CSCF, S-CSCF and HSS.
- Support of SIP protocol.
- Support of MD5, AKAv1 and AKAv2 authentication.
- Support of ISIM cards using the XOR, Milenage or TUAK authentication algorithm.
- Support of IPsec (ESP/transport).
- Support of voice, video calls: MO and MT.
- Support of voice echo test.
- Support of hold.
- Support of SMS (GSM 3.40) using SIP MESSAGE and SMS over SG.
- Support of IPv4 and IPv6.
- Support of precondition and dedicated bearer using Rx interface or N5 interface.
- Support of emergency call.
- Configurable user database.
- External authentication using Cx interface.
- Command line monitor.
- Remote API using WebSocket.

3 Requirements

3.1 Hardware requirements

- LTEIMS can run on the same PC as the Amarisoft eNodeB if a simple and compact solution is needed. Otherwise, any reasonnably recent PC with at least one Gigabit Ethernet port is acceptable.
- A VoLTE compatible UE is necessary (See [VoLTE Call], page 5, note that it may depends on UE).
- A test USIM with ISIM application should be plugged into the UE. IMSI and secret key must be known. A standard USIM may also work but it depends on the UE implementation.

3.2 Known compatible UE

The Amarisoft IMS server has been tested with the following UE models:

- Samsung S5
- LG MS870

3.3 Software requirements

- A 64 bit Linux distribution. Fedora 42 is the officially supported distribution.
  The following distributions are known as compatible:
  - Fedora 22 to 42
  - Cent OS 7
  - Ubuntu 14 to 24

  Your system requires at least GLIBC 2.17.

4 Installation

The network access thru the Gigabit Ethernet port must be correctly configured.

LTEIMS can be run directly from the directory when it was unpacked. No need for explicit installation.

4.1 Fedora setup

If you want to use SMS over SG with the Amarisoft MME or precondition with QoS, you need support of SCTP protocol for which the necessary packages are not usually installed. In order to install them, do as root user:

- Fedora

  ```
  dnf install lksctp-tools kernel-modules-extra
  ```

- Ubuntu

  ```
  sudo apt-get install lksctp-tools linux-image-extra-3.13.0-24-generic
  ```

  Note that linux-image-extra package name may differ depending on your kernel version.

To verify that SCTP kernel module is running, do as root user:

```
checksctp
```

If it reports that the protocol is not supported,

- check if you have a /etc/modprobe.d/sctp-blacklist.conf file
- edit it to comment the 'blacklist sctp' line

Then reboot the PC in case the Linux kernel was upgraded too.

4.2 License key installation

LTEIMS needs a `LTEMME` license key to run. Please refer to the `ltemme` documentation.

4.3 Initial testing

- Edit the file `config/ims.cfg` to set the address of the SIP interface. Normally it is the address of the Ethernet interface that will receive SIP packets.
  You can keep the current config if you use it with the Amarisoft MME and its `config/mme-ims.cfg` config file.
- Start the program as root with:

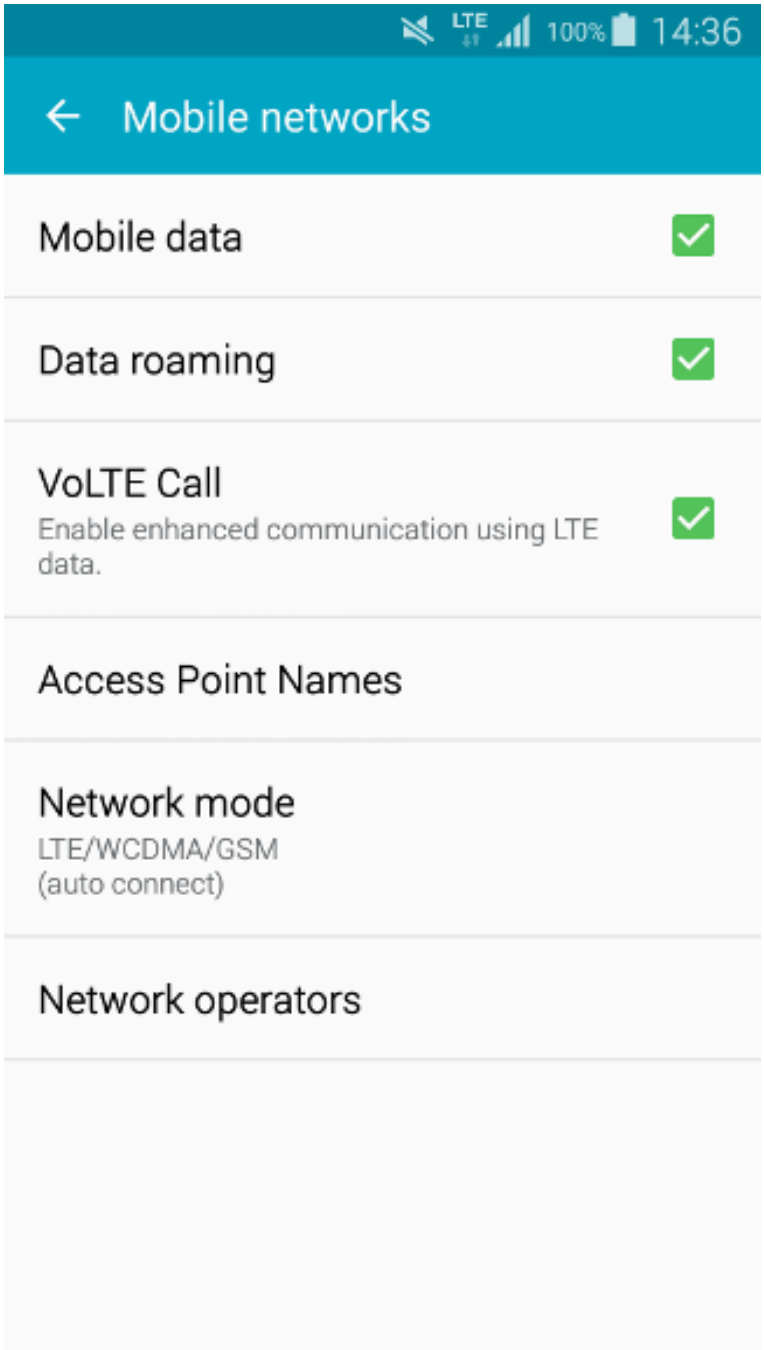
  ```
  ./lteims config/ims.cfg
  ```

  [The root access is only needed if you want IPsec support.]
- The command line interface is used to monitor the operation of LTEIMS and to change the logging options.
  Use `help` to get the list of commands and `quit` to stop the program.
- Use `users` to list the user database and registering state.

4.4 Samsung S5 configuration

Your UE must run at least Android 5.0 (Even if Android 5.0 is installed, try to update software (several times) as a sub-release is necessary).
If not, please update it.

To check your UE is configured for VoLTE, please go to `Settings/More networks/Mobile networks` of your handset and check `VoLTE Call` is checked:



We assume you are using the system with Amarisoft MME and `config/mme-ims.cfg` config file.
As there are two PDN defined, you must add them to the UE.

- Go to Settings/More networks/Mobile networks
- Turn on Data roaming
- Check VoLTE Call (If not present, it means your device is not up to date or does not support VoLTE).
- Go to Network operators, search for networks and select Amarisoft network.

- Go back to Mobile network.
- Add the first APN with the following parameters:
  - Name = Internet
  - APN = internet
  - APN type = default
- Save it and select it.
- Add second APN with following parameters:
  - Name = IMS
  - APN = ims
  - APN type = ims
- Save it and do not select it (This APN may not be displayed).
- Reboot your phone

5 Configuration reference

5.1 Configuration file syntax

The main configuration file uses a syntax very similar to the Javascript Object Notation (JSON) with few extensions.

1. Supported types:
   - Numbers (64 bit floating point). Notation: `13.4`
   - Complex numbers. Notation: `1.2+3*I`
   - Strings. Notation: `"string"`
   - Booleans. Notation: `true` or `false`.
   - Objects. Notation: `{ field1: value1, field2: value2, .... }`
   - Arrays. Notation: `[ value1, value2, .... ]`
2. The basic operations `+`, `-`, `*` and `/` are supported with numbers and complex numbers. `+` also concatenates strings. The operators `!`, `||`, `&&`, `==`, `!=`, `<`, `<=`, `>=`, `>` are supported too.
3. The numbers `0` and `1` are accepted as synonyms for the boolean values `false` and `true`.
4. {} at top level are optional.
5. " for property names are optional, unless the name starts with a number.
6. Properties can be duplicated.
   If properties are duplicated, they will be merged following [JSON merge rules], page 8, with overriding occuring in reading direction (last overrides previous).
   Ex:

```
{
    value: "foo",
    value: "bar",
    sub: {
        value: "foo"
    },
    sub: {
        value: "bar"
    }
}
```

   Will be equivalent to:

```
{
    value: "bar",
    sub: {
        value: "bar"
    }
}
```

7. Files can be included using *include* keyword (must not be quoted) followed by a string (without :) representing the file to include (path is relative to current file) and terminating by a comma.
   Arrays can't be included.
   Merge will be done as for duplicate properties.
   If *file1.cfg* is:

```
    value: "foo",
    include "file2.cfg",
    foo: "foo"
```

And *file2.cfg* is:

```
value: "bar",
foo: "bar"
```

Final config will be:

```
{
    value: "bar",
    foo: "foo"
}
```

8. A C like preprocessor is supported. The following preprocessor commands are available:

   `#define` *var expr*
   : Define a new variable with value *expr*. *expr* must be a valid JSON expression. Note that unlike the standard C preprocessor, *expr* is evaluated by the preprocessor.

   `#undef` *var*
   : Undefine the variable *var*.

   `#include` *expr*
   : Include the file whose filename is the evaluation of the string expression *expr*.

   `#if` *expr* Consider the following text if *expr* is true.

   `#else` Alternative of `#if` block.

   `#elif` Composition of `#else` and `#if`.

   `#endif` End of `#if` block.

   `#ifdef` *var*
   : Shortcut for `#if defined(var)`

   `#ifndef` *var*
   : Shortcut for `#if !defined(var)`

   In the JSON source, every occurrence of a defined preprocessor variable is replaced by its value.

9. Backquote strings: JSON expression can be inserted in backquote delimited strings with the `${expr}` syntax. Example: `` `abc${1+2}d` `` is evaluated as the string `"abc3d"`. Preprocessor variables can be used inside the expression. Backquote strings may span several lines.

5.1.1 JSON merge rules

Merge overriding direction depends on context, i.e source may override destination or the opposite.
JSON merge is recursive for Objects and Arrays.

Example, merging

```
{
    foo: { value: "bar" },
    same: "one",
    one: 1
}
```

with

```
{
    foo: { value: "none", second: true },
```

```
    same: "two",
    two: 1
}
```

Will become:

```
{
    foo: { value: "bar", second: true },
    same: "one",
    one: 1
    two: 1
}
```

assuming first object overrides second one.

In case of Array merging, the final array length will be the maximum length of all merged arrays.
For each element of the final array, merge will be done considering defined elements only.
Ex:

```
{
    array: [0, 1, 2, { foo: "bar" } ],
    array: [3, 4],
    array: [5, 6, 7, { bar: "foo" }, 8 ]
}
```

Will be merged to:

```
{
    array: [5, 6, 7, { foo: "bar", bar: "foo" }, 8 ],
}
```

5.2 Properties

`log_filename`
: String. Set the log filename. If no leading `/`, it is relative to the configuration file path. See [Log file format], page 43.

`log_options`
: String. Set the logging options as a comma separated list of assignments.

- *layer*.level=*verbosity*. For each layer, the log verbosity can be set to `none`, `error`, `info` or `debug`. In debug level, the content of the transmitted data is logged.
- *layer*.max_size=*n*. When dumping data content, at most `n` bytes are shown in hexa. For ASN.1, NAS or Diameter content, show the full content of the message if `n > 0`.
- *layer*.payload=[0|1]. Dump ASN.1, NAS, SGsAP or Diameter payload in hexadecimal.
- *layer*.key=[0|1]. Dump security keys (NAS and RRC layers).
- *layer*.crypto=[0|1]. Dump plain and ciphered data (NAS and PCDP layers).
- *layer*.verbose=[0|1]. If `layer` is `ipsec`, dump all packets filtering informations.
- time=[sec|short|full]. Display the time as seconds, time only or full date and time (default = time only).
- time.us=[0|1]. Dump time with microseconds precision.
- file=cut. Close current file log and open a new one.

- file.rotate=now. Move and rename to the same directory or to the directory pointed by `file.path` and open a new log file (Headers are kept).
- file.rotate=*size*. Every time log file size reaches *size* bytes, move and rename to the same directory or to the directory pointed by `file.path`, and open a new log file (Headers are kept).
  Size is an integer and can be followed by K, M or G.
- file.rotate=#*count*. Everytime number of logs in log file reaches *count*, move and rename to the same directory or to the directory pointed by `file.path`, and open a new log file (Headers are kept).
  Size is an integer and can be followed by K, M or G.
- file.path=*path*. When log rotation is enabled (`file.rotate` set), rename and move current log to this path instead of initial log path.
- append=[0|1]. (default=0). If 0, truncate the log file when opening it. Otherwise, append to it.

Available layers are: `ims`, `sip`, `media`, `rx`, `cx`, `n5`

`log_sync` Optional boolean (default = false). If true, logs will be synchronously dumped to file.
Warning, this may lead to performances decrease.

`sip_addr` Array. Each item is an object representing a SIP server socket defined as follow:

`addr` String. Set the IP address (and an optional port) on which IMS server will listen for SIP packets. The default port is 5060.

`bind_addr`
Optional string. Defines network interface on which IMS will listen.
If not specified, the addr parameter is used.

`port_min` Optional integer (Default is 10000). Defines lower bound of UDP media socket.

`port_max` Optional integer (Default is 20000). Defines upper bound of UDP media socket.

`trunk` Optional boolean (default is false). Defines the address to use for SIP trunk.

NB:

- SIP socket object can be represented by a simple string. Thus, it will represent `addr` parameter and all other parameters will use default value.
- For legacy, `sip_addr` can be a single SIP socket (Object or String) instead of an Array.

`sctp_addr`
String. Set the IP address (and an optional port) for MME connection.
This is only necessary for SMS over SG feature.

`cx_server_addr`
String. Set the IP address (and optional port) of Cx SCTP connection to the HSS.
The default port is 3868.

`cx_bind_addr`
Optional string. IP address and optional port on which the Cx SCTP connection is bound. If not set, `sctp_addr` is used.

`cx_origin_realm`
: Optional string. Defines the string sent in the Origin-Realm AVP for Cx messages. Default is set to `amarisoft.com`.

`cx_origin_host`
: Optional string. Defines the string sent in the Origin-Host AVP for Cx messages. Default is set to `ims.amarisoft.com`.

`cx_watchdog_duration`
: Optional integer (range 0 to 36000000, default = 0). Tw watchdog timer in milliseconds to send the Diameter Device Watchdog Request message. The value 0 deactives the watchdog.

`rx_server_addr`
: Optional string. Set the IP address (and optional port) of Rx SCTP connection to the MME. The default port is 3868. If not set, `cx_server_addr` is used.

`rx_bind_addr`
: Optional string. IP address and optional port on which the Rx SCTP connection is bound. If not set, `cx_bind_addr` is used.

`rx_origin_realm`
: Optional string. Defines the string sent in the Origin-Realm AVP for Rx messages. Default is set to `amarisoft.com`.

`rx_origin_host`
: Optional string. Defines the string sent in the Origin-Host AVP for Rx messages. Default is set to `ims.amarisoft.com`.

`rx_watchdog_duration`
: Optional integer (range 0 to 36000000, default = 0). Tw watchdog timer in milliseconds to send the Diameter Device Watchdog Request message. The value 0 deactives the watchdog.

`use_n5` Optional boolean (default = false). If set, the N5 interface is used instead of the Rx interface.

`n5` Optional object used to configure the N5 interface with the PCF.

- `api_root` String. apiRoot as defined in 3GPP TS 29.501: <scheme>://<host>:<port>, where <scheme> is "http" or "https".
- `transaction_timeout`
  Optional integer (range 1 to 15000, default = 4000). Defines the timeout in milliseconds for a transaction with the PCF.
- `bind_addr`
  Optional string. IP address and optional port on which the N5 TCP connection is bound. If not set, `cx_bind_addr` is used.

`domain` String. Global SIP domain used for IMPU and authentication. May be overriden at user level.
This parameter is not used to recover IMPU.

`tcp_threshold`
: Optional integer (default = 1300). Set packet threshold in bytes to use TCP instead of UDP.

`tcp_keepalive`
: Optional integer (default = 900). Time in seconds before sending keepalive on TCP connections. 0 means disabling keepalive.

`tcp_keepcount`
: Optional integer (default = 2). Number of TCP keepalive failure before releasing socket.

`session_expires`
: Optional object. Defines session timer configuration.
Parameters are the following:

  `duration` Optional integer (default = 1800). Session duration in seconds.

  `min` Optional integer (default = 90). Minimum allowed session expires.

  `refresher`
  : Optional string (default = uac). Defines session refresher when it is up to the IMS to take this decision.

  `method` Optional string (default = update). Defines SIP method to use for session refresh when IMS is the refresher.

  `force` Optional boolean (default = true). If set to true, forces the refresher in IMS request with the `refresher` value.

  For legacy purpose, `session_expires` can be an integer. If so, only `duration` is set and other parameters are kept to their default value.

`100rel` Optional boolean (default = true). Enable/disable support of provisional response

`use183` Optional boolean (default = true). Enable/disable of Session progress sending

`allow_held_invite`
: Optional boolean (default = true). If disabled, will force call initiated on hold to be unheld.

`precondition`
: Optional string (default = on). Values can be "on", "off" or "silent".
On: IMS will handled QoS according to the standard.
Off: no precondition and no dedicated bearer establishment.
Silent: dedicated bearers will be established regardless the SIP and SDP content.
Note that a Rx connection is necessary to allow dedicated bearer establishment.

`p_called_party_id`
: Optional boolean (default is false). Enable P-Called-Party-ID header for INVITE and MESSAGE requests.

`ipsec` Optional boolean (default is true). Enable/disable support of IPsec.

`ipsec_tun_setup_script_filename`
: Optional string (default is /tmp/.ipsec_tun_setup_script). Path and name of the temporary file created to configure the IPsec TUN interface.

`ipsec_aalg_list`
: Array of strings. Each string represent IPsec authentication algorithm supported by IMS.
"null" may be used to indicate no authentication.
Supported algorithms are:

  - hmac-md5-96

- hmac-md5-128
- hmac-sha-1-96
- hmac-sha-1-160
- aes-cmac-96
- aes-gmac
- aes-gmac-us

`ipsec_ealg_list`
: Array of strings. Each string represent IPsec encryption algorithm supported by IMS.
"null" may be used to indicate no encryption.
Supported algorithms are:

  - aes-cbc
  - des-ede3-cbc
  - blowfish
  - aes-ctr
  - aes-gcm
  - aes-gcm-us

`ipsec_mtu`
: Optional integer (range 68 to 65535, default = 1400). MTU for IPsec network interface.

`ipsec_ifname`
: Optional string. If set, use this interface for IPsec. In that case, `ipsec_netns` must be set and interface must be added to the network namepsace.
In this mode, used to run lteims with non root privileges, it is up to the user to configure the tun interface.
In particular:

  - Add tun interface to specififed network namespace
  - Add `sip_addr` IP addresses to the interface
  - Set MTU (`ipsec_mtu` will be ignored)
  - Set routes
  - Set CAP_SYS_ADMIN and CAP_NET_RAW capabilities to *PROG*

  Ex:

```
setcap cap_sys_admin,cap_net_raw=ep lteims
ip tuntap add mode tun <ifname>
ip link set dev <ifname> name <ifname> netns <netns>
ip netns exec $netns ip addr add 192.168.4.1/0 dev <ifname>
ip netns exec $netns ip addr add 192.168.0.103/0 dev <ifname>
ip netns exec $netns ip link set <ifname> up mtu 1400
ip netns exec $netns ip route add default dev <ifname>
```

`ipsec_netns`
: Optional string. Must be set if `ipsec_ifname` is set. Defines network namespace name used for IPsec.

`dialog_timeout`
: Optional integer (default = 30). Time in seconds of call session. Stop call if no activity has been detected during this time.

`auth_on_register_only`
: Optional boolean (default = true). If true, don't try to authenticate other request than register (unless expires is set to 0).

`proxy_auth`
: Optional boolean (default = false). If true, use proxy authentication headers

`trunk`
: Optional object. If set, IMS will accept incoming calls from a SIP trunk and will forward unknown calls to SIP trunk (`https://tech-academy.amarisoft.com/how_to_run_volte_call_using_a_sip_trunk.wiki`).
SIP trunk must accept non authenticated requests. Parameters are the following:

  `addr`
  : String. IP address of the SIP trunk.

  `name`
  : String. SIP trunk account user name.

  `domain`
  : Optional string (default = same as `addr`). If set, use this for SIP uri domain.

  `tcp_threshold`
  : Optional integer (default = 0). If > 0, will override global `tcp_threshold` parameter.

  `anonymous`
  : Optional boolean (default = false). If true, allow to forward anonymous call to SIP trunk.

`com_addr`
: Optional string. Address of the WebSocket server remote API. See [Remote API], page 24.
If set, the WebSocket server for remote API will be enabled and bound to this address.
Default port is 9003.
Setting IP address to [::] will make remote API reachable through all network interfaces.

`com_name`
: Optional string. Sets server name. IMS by default

`com_ssl_certificate`
: Optional string. If set, forces SSL for WebSockets. Defines CA certificate filename.

`com_ssl_key`
: Optional string. Mandatory if *com_ssl_certificate* is set. Defines CA private key filename.

`com_ssl_peer_verify`
: Optional boolean (default is false). If *true*, server will check client certificate.

`com_ssl_ca`
: Optional string. Set CA certificate. In case of peer verification with self signed certificate, you should use the client certificate.

`com_log_lock`
: Optional boolean (default is false). If *true*, logs configuration can't be changed via `config_set` remote API.

`com_log_us`
: Optional boolean (default is false). If *true*, logs sent by `log_get` remote API response will have a `timestamp_us` parameters instead of `timestamp`

`com_auth` Optional object. If set, remote API access will require authentication.
Authentication mechanism is describe in [Remote API Startup], page 26, section.

- `passfile` Optional string. Defines filename where password is stored (plaintext). If not set, `password` must be set
- `password` Optional string. Defines password.
  If not set, `passfile` must be set.
- `unsecure` Optional boolean (default false). If set, allow password to be sent plaintext.
  NB: you should set it to true if you access it from a Web Browser (Ex: Amarisoft GUI) without SSL (https) as your Web Browser may prevent secure access to work.

`com_log_count`
Optional number (Default = 8192). Defines number of logs to keep in memory before dropping them.
Must be between 4096 and 2097152).

`sim_events`
Array of object. Each element defines a remote API request ([Remote API], page 24) except that `message` field is replaced by `event`.

`sim_events_loop_count`
If set, will define `loop_count` for each event of `sim_events`, See [loop_count], page 25.

`sim_events_loop_delay`
If set, will define `loop_delay` for each event of `sim_events`, See [loop_delay], page 25.

`license_server`
Configuration of the Amarisoft license server to use.
Object with following properties:

- `server_addr` String. IP address of the license server.
- `name` Optional string. Text to be displayed inside server monitor or remote API.
- `tag` Optional string. If set, server will only allow license with same tag.

Example:

```
license_server: {
    server_addr: "192.168.0.20",
    name: "My license"
}
```

`sms_expires`
Integer (default = 86400). Delay in seconds before SMS is removed from database. If a MO SMS contains the TP-Validity-Period field, it will be used instead of `sms_expires`.

`sms_hook_only`
Optional boolean (default = false). If set, when SMS is received and at least one WebSocket client has registered to `sms` event, don't process SMS internally (Only CP/RP layer will be handled).

`binding_expires`
: Integer (default = 3600, max = 864000). Default duration in seconds for registration.

`subscribe_expires`
: Integer (default = 0, max = 864000). Subscription expiration. If set to 0, use value sent by UE.

`user_agent`
: Optional string (default = Amarisoft-IMS-2025-12-12). SIP user agent.

`force_user_agent`
: Boolean (default = true). If true, IMS user-agent will always be used, else remote peer's one will be used.

`early_media`
: Boolean (default = true). Enable/disable handling of early media.

`path` Optional string (default = sip:ims.mnc001.mcc001.3gppnetwork.org). SIP path. If registration supports `path`, defines content of the `Path` header in registration response. If set to an empty string, `Path` header won't be set.

`timer_t1` Optional number (default = 2). SIP T1 timer duration in seconds.

`timer_t2` Optional number (default = 16). SIP T2 timer duration in seconds.

`timer_t4` Optional number (default = 17). SIP T4 timer duration in seconds.

`custom_headers`
: Array of object. Each object represents a custom header to add to requests and/or responses, defined as follows:

  `name` String. Header name

  `value` Optional string. Header value. If not set, replace is forced to `true` and header will only be removed if present.

  `codes` Number or array of numbers of the SIP responses on which to apply custom headers.
  `0` can be used for all codes.

  `methods` String or array of strings of the SIP method on which to apply custom headers.
  `*` can be used for all methods.
  If `codes` is set, the custom headers will be applied to the responses of the associated request.

  `replace` Optional boolean (default = false). If set, allow only one occurence of the header.

`sms_centre_address`
: Optional object used to configure the SMS centre address. Contains the following parameters:

  `type_of_number`
  : Optional enumeration "unknown", "international", "national" (default = "unknown"). SMS centre address type of number.

  `numbering_plan`
  : Optional enumeration "unknown", "national", "private" (default = "unknown"). SMS centre address numbering plan identification.

  `number` String. Contains optional '+' at first position followed by a maximum of 20 digits. SMS centre address number.

`sms_retry_delay`
: Integer. Time in s to retry SMS sending.

`echo`
: IMPU (See [impu], page 18). If set, this defines the phone number(s) for echo service.

`mt_call_sdp_file`
: String. File to use as SDP when using MT call.

`sms_message_filter`
: Optional object. Allows to define the IMS behavior for a list of SMS related messages.
Each property name represents a SMS message type. The ones currently supported are `cp_data`, `cp_ack`, `rp_data` and `rp_ack`.
Each property value is an object containing the following fields:

  `action`
  : Enumeration (`treat` (message is processed), `ignore` (message is ignored) or `reject` (message is rejected)).

  `ttl`
  : Optional integer. If set, the `reject` of `ignore` filter is applied `ttl` times. If not set, the filter is applied until it is modified.

  By default all procedures are treated.

  Example:

```
sms_message_filter: {
    cp_data: {
      action: "treat",
    },
    rp_ack: {
      action: "reject",
      ttl: 1
    }
}
```

`sms_forced_cp_cause`
: Optional integer (range 0 to 255). Allows to override the CP error cause selected by the IMS with the one configured. Set to 0 to deactivate the override.

`sms_forced_rp_cause`
: Optional integer (range 0 to 255). Allows to override the RP error cause selected by the IMS with the one configured. Set to 0 to deactivate the override.

`mms_server_bind_addr`
: Defines network interface on which MMS server will listen. It is used to configure the MMSC in the UE. MMS proxy is not supported and shall not be configured in the UE.

  Example:

```
mms_server_bind_addr: "192.168.3.1:1111"
MMSC: http://192.168.3.1:1111
```

`mms_expires`
: Optional integer (default = 86400). Delay in seconds before MMS is removed from database.

5.2.1 User database options

`ue_db`

Array of objects. Configure the user database. Each element is an entry for one user. Note that this part can be shared between Amarisoft MME and IMS. The following properties are available:

`imsi` Optional string. Set the IMSI.
Shall be present if `nai` is absent.

`nai` Optional string. Network specific identifier-based SUPI.
Shall be present if `imsi` is absent.

`multi_sim`
Optional boolean (default = false). If true, allow several UEs to have the same IMSI (useful when using several identifical test SIM cards in different UEs at the same time). They are distinguished with their IMEI. Note: it is only allowed with the XOR authentication algorithm.

`impi` String. Defines user IMPI. Must be fully filled with hostname if necessary.
If you don't know your IMPI, please look at IMS logs inside *REGISTER* request. The IMPI must match the *username* argument inside *Authorization* header.

`impu` Array of strings or array of objects.
Each string represents an IMPU. `sip`, `tel` and `urn` scheme are supported. Note that sip URI must not include hostname.
If IMPU does not start by a scheme, it is assumed to be a sip URI.
Ex:

- sip:user
- user
- tel:+33123456789
- urn:service:sos

If impu is an object, it has following members:

`impu` IMPU as defined above.

`imei` IMEI associated to this IMPU. Allows to filter calls and SMS for a specific UE.
Only relevant if multi_sim is set to true.

`anonymous`
Optional boolean (default is false). If true, allow Anonymous connection (Emergency call) and mask impu in SIP headers.

`anonymous_callback`
Optional number (default is 0). If anonymous is set, defines how long in seconds after an anonymous call has been done, the network can call back the caller. In that case, caller will be identified with a SIP IMPU derived from its IMEI.

`authentication`
Optional boolean (default is true). If false, disable authentication.

`ring_only`
: Optional boolean (default is false). If true, IMS will go up to ringing state but not further.

`precondition`
: Optional string. Values can be "on", "off" or "silent".
On mode: IMS will try to guess precondition from supported header, SDP content and/or VoLTE compatibility of client.
Off mode: no precondition and no dedicated bearer establishment.
Silent mode: dedicated bearers will be established whatever the SIP and SDP content.

`100rel`
: Optional boolean (default = true). Enable/disable 100rel support for this IMPU.

`use183`
: Optional boolean (default = true). Enable/disable of Session progress sending.

`preferred`
: Optional boolean (default: false). If set, this `impu` will be used for preferred identity and for INVITE request URI.

`asserted`
: Optional boolean (default: false). If set, this `impu` will be used for asserted identity.

`verstat`
: Optional string (default: null). If set, add verstat options to P-Asserted-URI header URI option.

`phone`
: Optional boolean (default: false). It set, add `user=phone` option to URI.

`associated`
: Optional boolean (default: true). If set to false, this `impu` will not be used for associated URI.

`early_media`
: Boolean (default = true). Enable/disable handling of early media for this IMPU.

`display_name`
: Optional string. If set, SIP headers will use this field for display name.

`domain`
: Optional string. Used to override user or global config.

`p_called_party_id`
: Optional string. If set, forces P-Called-Party-ID header for INVITE and MESSAGE requests, no matter if `p_called_party_id` global parameter is set or not.

`answer_delay`
: Optional number. If set, when doing a echo call on this IMPU, OK response to INVITE request will be delayed by `answer_delay` in seconds.

`sms_delay`
: Optional number. If set, IMS will wait `sms_delay` seconds before sending SMS to this IMPU.

`media_delay`
: Optional number. Delay sending of media packet in seconds.

`remote_control`
: Optional boolean. If set to true on a echo impu, the sending of final of INVITE request will be controled by `dialog_set` remote API with `answer` action (The delay to send answer is defined by `dialog_timeout` parameter.
A remote API `invite` event will be sent with dialog `session_id`.

`sdp_file`
: Optional string. When used in echo mode, use the SDP file to force the media sent in server SDP response.

`session_expires`
: Optional object. If set, will override global session timer parameters for that impu, See [session_expires], page 12.
If IMS is the server (MO call and echo call), caller configuration will be used, else (MT call) callee's one will be used.

`bitrate_factor`
: Optional number (Default = 1). Media bitrate are estimated from SDP. When setting up the dedicated bearer via Rx interface, the bitrate is multiplied by this number.

`audio`
: Optional boolean (default = true). If set to false, audio media will be rejected.

`video`
: Optional boolean (default = true). If set to false, video media will be rejected.

`text`
: Optional boolean (default = true). If set to false, text media will be rejected.

`rtp`
: Optional boolean (default = true). If set to false, rtp packet won't be forwarded (Can be changed via `dialog_set/media` remote API).

`rtcp`
: Optional boolean (default = true). If set to false, rtcp packet won't be forwarded (Can be changed via `dialog_set/media` remote API).

`sms`
: Optional boolean (default = true). If set to false, SMS over IMS handling will be disabled for that IMPU.

`reauthenticate`
: Optional boolean (default = false). If set to true, re-registrations will cause a re-authentication.

`match`
: Optional string. If set, defines a POSIX Extended Regular Expression. Any SIP header impu that matches it, will use this impu definition.
Ex: `"$echo"` to match all impu starting by `echo` such as `sip:echo1`...

`callflow_modifier`
: Optional object or array of objects. This allows to modify IMS behavior and messages.

Each object will define a modification in the SIP callflow. The modifier will be applied with following rules:

- On MO dialogs, callee modifiers will be used.
- On MT dialogs, caller modifiers will be used.

Check some examples ⟨undefined⟩ [Callflow modifier examples], page ⟨undefined⟩.
Each object is defined this way:

`method` Optional string. Defines on which step to apply modifier (if not set, applies on any of the following). Modifier applies on reception or sending of message depending on `dir` parameter
If set to `INVITE`, `UPDATE`, `CANCEL`, `BYE`, `NOTIFY`, `MESSAGE`, `ACK`, `PRACK`, `INFO`, `REGISTER` or `SUBSCRIBE`, modifier applies to the equivalent SIP message (received or sent). If set to `INVITE.OK`, modifier applies on final answer of INVITE request.
If set to `INVITE.TRYING`, modifier applies on 100 SIP responses.
If set to `INVITE.RINGING`, modifier applies on 180 SIP responses.
If set to `INVITE.SESSION_PROGRESS`, modifier applies on 183 SIP responses.
`MESSAGE`, modifier applies on received or sent requests.
If set to `REGISTER.SOS`, modifier applies on reception of emergency REGISTER.
If set to `REINVITE`, modifier applies on received or sent INVITE requests during an established dialog.

`log` Optional string or boolean. Add logs (`IMS` layer) for modifier. Set it false to disable. In case of string, this will be prompted in logs, else (true or not present) a default string set to `cfm#<n>` will be used.

`code` Optional number (between 100 and 699). If set, modifier will generate a SIP response with this code and modifier will apply on this message.
If code is not set, no response wil be generated.

`stop` Optional boolean (default = true). If set to true and modifier applies within a dialog, dialog will be stopped. In that case, normal callflow will be stopped.

`dir` Optional string (default = both). Can be `tx` to apply modifier on sent messages, `rx` to apply modifier on received messages or `both` for both type of messages.

`transparent`
: Optional boolean (default = false). If `true`, modifier will be applied without stopping the normal call flow.

`skip`
: Optional boolean (default = false). If `true`, message won't be sent.

`content`
: Optional string. Represent a filename used to fill body of generated message and may replace already set content with same `content_type` (Ex: use application/sdp to remove SDP).
If set to null, no content will be added but content removal will be applied.

`content_type`
: Optional string. Mandatory if `content` is set, will define response content type.

`access_network`
: Optional string (default = false). List of RAT separated by comma. If set the callflow will only be affected if `P-Access-Network-Info` matches the selected RAT. Can be `lte` (3GPP-E-UTRA), `nr` (3GPP-NR), or `n3gpp` (IEEE).

`custom_headers`
: Optional Array of object. Apply specific headers on the generated response as defined in [custom_headers], page 16, except that `code` and `methods` parameters are useless.

`impu`
: Optional string. Is set, modifier will be applied only if counterpart of the dialog matches this IMPU:
  - On MO dialogs, caller IMPU.
  - On MT dialogs, callee IMPU.

`imei`
: Optional string. Is set, modifier will be applied only if counterpart (Cf `impu`) has same IMEI provided during registration.

`ttl`
: Optional integer (> 0) (Time To Live). If set, callflow modifier will be removed after having being applied `ttl` times.

`tts`
: Optional integer (default = 0) (Time To Start). If set, callflow modifier will not be applied for the the `tts` times it has been triggered. This will differ `ttl` decreasing.

`send_delay`
: Optional number (default = 0, max = 60). Delay in seconds of SIP message sending

`send_retry`
: Optional number (default = 0, max = 60). Number of SIP message retransmission

`send_retry_delay`
: Optional number (default = `send_delay`, max = 60). Delay in seconds between two SIP message retransmission.

`ecall`
: Optional object with following properties:

`ack`
: Optional string. Modify ack received value in EmergencyCallData.Control xml content

`code`
: Deprecated. See `callflow_modifier`

`method`
: Deprecated. See `callflow_modifier`

`content`
: Deprecated. See `callflow_modifier`

`content_type`
: Deprecated. See `callflow_modifier`

`transparent`
: Deprecated. See `callflow_modifier`

`authent_type`
: Optional string (default = AKAv1). Defines minimum authentication level.
If client does no specify authentication algo, server will use this value. Else, server will allow authentication only if client provided algo is at least the one specified by this parameter.
Values are (from lowest security to highest):

`none`
: Disable authentication.

`MD5`
: MD5 digest authentication.

`AKAv1`
: AKAv1 authentication.

`AKAv2`
: AKAv2 with MD5 hash authentication.

`AKAv2-SHA-256`
: AKAv2 with SHA-256 hash authentication.

`authent_qop`
: Optional string (default = none). Defines authentication qop parameter and can be:

`none`
: Don't set qop.

`auth`
: Use `auth` qop directive.

`auth-int`
: Use `auth-int` qop directive.

`aut,auth-int`
: Let client select qop directive.

`pwd`
: Optional string. Password set for MD5 authentication. If set and *authent_type* is not set, *authent_type* is set to MD5.

`mt_call_sdp_file`
: Optional string. File to use as SDP when using MT call. Overrides global parameter.

`domain`
: Optional string. If set, overrides global config.

`auth_on_register_only`
: Optional boolean. If set, overrides global config.

`proxy_auth`
: Optional boolean (default = false). If true, use proxy authentication headers

`force_sms_over_sg`
: Optional boolean. If set, forces use of SMS over SG.

`ue_db_filename`
: Optional string. If present, store the current IMS state in a persistent file. The IMS state contains in particular the registration info and pending SMS.

6 Remote API

You can access LTEIMS via a remote API.
Protocol used is WebSocket as defined in RFC 6455 (`https://tools.ietf.org/html/rfc6455`).

Note that Origin header is mandatory for the server to accept connections.
This behavior is determined by the use of `nopoll` library.
Any value will be accepted.

To learn how to use it, you can refer to our the following tutorial (`https://tech-academy.amarisoft.com/RemoteAPI.html`).

6.1 Messages

Messages exchanged between client and LTEIMS server are in strict JSON format.

Each message is represented by an object. Multiple message can be sent to server using an array of message objects.

Time and delay values are floating number in seconds.

There are 4 types of messages:

- Request

  Message sent by client.
  Common definition:

  `message` String. Represent type of message. This parameter is mandatory and depending on its value, other parameters will apply.

  `message_id`
  Optional any type. If set, response sent by the server to this message will have same message_id. This is used to identify response as WebSocket does not provide such a concept.

  `start_time`
  Optional float. Represent the delay before executing the message.
  If not set, the message is executed when received.

  `absolute_time`
  Optional boolean (default = false). If set, `start_time` is interpreted as absolute.
  You can get current clock of system using `time` member of any response.

  `standalone`
  Optional boolean (default = false). If set, message will survive WebSocket disconnection, else, if socket is disconnected before end of processing, the message will be cancelled.

  `loop_count`
  Optional integer (default = 0, max = 1000000). If set, message will be repeated `loop_count` time(s) after `loop_delay` (From message beginning of event).
  Response will have a `loop_index` to indicate iteration number.

`loop_delay`
: Optional number (min = 0.1, max = 86400). Delay in seconds to repeat message from its `start_time`. Mandatory when `loop_count` is set > 0.

- Notification

  For some API, intermediate message may be sent by server before reception of response.
  Common definition:

  `message` String. Same as request.

  `message_id`
  : Optional any type. Same as in request.

  `time` Number representing time in seconds of the message start, relative to the beginning of the process.
  Useful to send command with absolute time.

  `notification`
  : String. Notification purpose

  `utc` Number representing UTC seconds (local clock) when the response has been generated.

- Response

  Message sent by server after any request message has been processed.
  Common definition:

  `message` String. Same as request.

  `message_id`
  : Optional any type. Same as in request.

  `time` Number representing time in seconds of the message start, relative to the beginning of the process.
  Useful to send command with absolute time.

  `utc` Number representing UTC seconds (local clock) when the response has been generated.

  `absolute_time`
  : Optional string. If `absolute_time` has been set and message is reaching LTEIMS too late, this field is present and set to `late`.

- Events

  Message sent by server on its own initiative.
  Common definition:

  `message` String. Event name.

  `time` Number representing time in seconds.
  Useful to send command with absolute time.

6.2 Startup

When WebSocket connections is setup, LTEIMS will send a first message with name set to `com_name` and type set to IMS.

If authentication is not set, message will be `ready`:

```
{
    "message": "ready",
```

```
    "type": "IMS",
    "name": <com_name>,
    "version": <software version>,
    "product": <Amarisoft product name (optional)>
}
```

If authentication is set, message will be `authenticate` :

```
{
    "message": "authenticate",
    "type": "IMS",
    "name": <com_name>,
    "challenge": <random challenge>
}
```

To authenticate, the client must answer with a `authenticate` message and a `res` parameter where:

```
res = HMAC-SHA256( "<type>:<password>:<name>", "<challenge>" )
```

`res` is a string and HMAC-SHA256 refers to the standard algorithm (`https://en.wikipedia.org/wiki/HMAC`)

If the authentication succeeds, the response will have a `ready` field set to `true`.

```
{
    "message": "authenticate",
    "message_id": <message id>,
    "ready": true
}
```

If authentication fails, the response will have an `error` field and will provide a new challenge.

```
{
    "message": "authenticate",
    "message_id": <message id>,
    "error": <error message>,
    "type": "IMS",
    "name: <name>,
    "challenge": <new random challenge>
}
```

If any other message is sent before authentication succeeds, the error `"Authentication not done"` will be sent as a response.

6.3 Errors

If a message produces an error, response will have an error string field representing the error.

6.4 Sample nodejs program

You will find in this documentation a sample program: `ws.js`.
It is located in `doc` subdirectory.
This is a nodejs program that allow to send message to LTEIMS.
It requires nodejs to be installed:

```
dnf install nodejs npm
npm install nodejs-websocket
```

Use relevant package manager instead of NPM depending on your Linux distribution.

Then simply start it with server name and message you want to send:

```
./ws.js 127.0.0.1:9003 '{"message": "config_get"}'
```

6.5 Common messages

`config_get`

Retrieve current config.

Response definition:

`type` Always "IMS"

`name` String representing server name.

`logs` Object representing log configuration.
With following elements:

`layers` Object. Each member of the object represent a log layer configuration:

`layer name`
Object. The member name represent log layer name and parameters are:

`level` See [log_options], page 9,

`max_size` See [log_options], page 9,

`key` See [log_options], page 9,

`crypto` See [log_options], page 9,

`payload` See [log_options], page 9,

`verbose` Optional boolean. See [log_options], page 9,

`count` Number. Number of bufferizer logs.

`rotate` Optional number. Max log file size before rotation.

`rotate_count`
Optional number. Max log count before rotation.

`path` Optional string. Log rotation path.

`bcch` Boolean. True if BCCH dump is enabled (eNB only).

`mib` Boolean. True if MIB dump is enabled (eNB only).

`locked` Optional boolean. If `true`, logs configuration can't be changed with `config_set` API.

`custom_headers`
Array. Current custom headers configuration See [custom_headers], page 16.

`config_set`

Change current config.
Each member is optional.
Message definition:

`logs` Optional object. Represent logs configuration. Same structure as config_get (See [config_get logs member], page 28).

All elements are optional.
Layer name can be set to `all` to set same configuration for all layers.
If set and logs are locked, response will have `logs` property set to `locked`.

`precondition`
See [precondition], page 12,

`session_expires`
See [session_expires], page 12,

`sms_retry_delay`
See [sms_retry_delay], page 17,

`sms_expires`
See [sms_expires], page 15,

`sms_hook_only`
See [sms_hook_only], page 15,

`binding_expires`
See [binding_expires], page 15,

`subscribe_expires`
See [subscribe_expires], page 16,

`auth_on_register_only`
See [auth_on_register_only], page 13,

`dialog_timeout`
See [dialog_timeout], page 13,

`p_called_party_id`
See [p_called_party_id], page 12,

`sms_message_filter`
See [sms_message_filter], page 17,

`sms_forced_cp_cause`
See [sms_forced_cp_cause], page 17,

`sms_forced_rp_cause`
See [sms_forced_rp_cause], page 17,

`mms_expires`
See [mms_expires], page 17,

`custom_headers`
Optional array. If set, will replace `custom_headers` as defined in config file (See [custom_headers], page 16) or previous call to `config_set`.

`100rel` See [100rel], page 12,

`tcp_threshold`
See [tcp_threshold], page 11,

`tcp_keepalive`
See [tcp_keepalive], page 11,

`tcp_keepcount`
See [tcp_keepcount], page 12,

`log_get` Get logs.
This API has a per connection behavior. This means that the response will depend on previous calls to this API within the same WebSocket connection.
In practice, logs that have been provided in a response won't be part of subsequent request unless connection is reestablished. To keep on receiving logs, client should send a new `log_get` request as soon as the previous response has been received.
If a request is sent before previous request has been replied, previous request will be replied right now without considering specific min/max/timeout conditions.

Message definition:

`min` Optional number (default = 1). Minimum amount of logs to retrieve. Response won't be sent until this limit is reached (Unless timeout occurs).

`max` Optional number (default = 4096). Maximum logs sent in a response.

`timeout` Optional number (default = 1). If at least 1 log is available and no more logs have been generated for this time, response will be sent.

`allow_empty`
Optional boolean (default = false). If set, response will be sent after timeout, event if no logs are available.

`rnti` Optional number. If set, send only logs matching rnti.

`ue_id` Optional number. If set, send only logs with matching ue_id.

`layers` Optional Object. Each member name represents a log layer and values must be string representing maximum level. See [log_options], page 9.
If *layers* is not set, all layers level will be set to *debug*, else it will be set to *none*.
Note also the logs is also limited by general log level. See [log_options], page 9.

`short` Optional boolean (default = false). If set, only first line of logs will be dumped.

`headers` Optional boolean. If set, send log file headers.

`start_timestamp`
Optional number. Is set, filter logs older than this value in milliseconds.

`end_timestamp`
Optional number. Is set, filter logs more recent than this value in milliseconds.

`max_size` Optional number (default = 1048576, i.e. 1MB). Maximum size in bytes of the generated JSON message. If the response exceeds this size, the sending of logs will be forced independently from other parameters.

Response definition:

`logs` Array. List of logs. Each item is a an object with following members:

`data` Array. Each item is a string representing a line of log.

`timestamp`
Number. Milliseconds since January 1st 1970. Not present if `com_log_us` is set in configuration.

`timestamp_us`
: Number. Microseconds since January 1st 1970. Only present if `com_log_us` is set in configuration.

`layer` String. Log layer.

`level` String. Log level: *error*, *warn*, *info* or *debug*.

`dir` Optional string. Log direction: *UL*, *DL*, *FROM* or *TO*.

`ue_id` Optional number. UE_ID.

`cell` Optional number (only for PHY layer logs). Cell ID.

`rnti` Optional number (only for PHY layer logs). RNTI.

`frame` Optional number (only for PHY layer logs). Frame number (Subframe is decimal part).

`channel` Optional string (only for PHY layer logs). Channel name.

`src` String. Server name.

`idx` Integer. Log index.

`headers` Optional array. Array of strings.

`discontinuity`
: Optional number. If set, this means some logs have been discarded due to log buffer overflow.

`microseconds`
: Optional boolean. Present and set to true if `com_log_us` is set in configuration file.

`log_set` Add log.
Message definition:

`log` Optional string. Log message to add. If set, *layer* and *level* are mandatory.

`layer` String. Layer name. Only mandatory if *log* is set.

`level` String. Log level: *error*, *warn*, *info* or *debug*. Only mandatory if *log* is set.

`dir` Optional string. Log direction: *UL*, *DL*, *FROM* or *TO*.

`ue_id` Optional number. UE_ID.

`flush` Optional boolean (default = false). If set, flushes fog file.

`rotate` Optional boolean (default = false). If set, forces log file rotation.

`cut` Optional boolean (default = false). If set, forces log file reset.

`log_reset`
: Resets logs buffer.

`license` Retrieves license file information.
Response definition:

`products` String. List of products, separated by commas.

`user` String. License username.

`validity` String. License end of validity date.

`id` Optional string. License ID.

`id_type` Optional string. License ID type. Can be `host_id` or `dongle_id`

`uid` Optional string. License unique ID.

`filename` Optional string. License filename.

`server` Optional string. License server URL.

`server_id`
Optional string. License server ID.

`quit` Terminates lteims.

`help` Provides list of available messages in *messages* array of strings and events to register in *events* array of strings.

`stats` Report statistics for LTEIMS.
Every time this message is received by server, statistics are reset.
Warning, calling this message from multiple connections simultaneously will modify the statistics sampling time.

Response definition:

`cpu` Object. Each member name defines a type and its value cpu load in % of one core.

`instance_id`
Number. Constant over process lifetime. Changes on process restart.

`counters` Object. List of counters, with following sub members:

`messages` Object. Each member name is the message name and its value is its occurence.
To get list of message, type *cevent help msg* in LTEIMS monitor.

`errors` Object. Each member name is the error name and its value is its occurence.
To get list of message, type *cevent help error* in LTEIMS monitor.

`register` Register client for messages generated by LTEIMS. Message definition:

`register` Optional string or array of string. List of messages to register to.
Can be `users_update`, `sms`

`unregister`
Optional string or array of string. List of messages to unregister.
Can be `users_update`, `sms`

`ipsec` Report ipsec SAs.

Response definition:

`SAs` Array. List of object representing a security association witth following definition:

`type` String. IP version, can be `IPv4` or `IPv6`.

`dir` String. Direction, can be `in` or `out`.

`spi` Number. SPI.

`ue_id` Number. Associated `ue_id`.

`mode` String. ESP type, can be `tunnel` or `transport`

`src` String. Source IP address.

`dst` String. Destination IP address.

`tun_src` Optional string. Tunnel source IP address.

`tun_dst` Optional string. Tunnel destination IP address.

`src_prefix`
Number. Source network prefix.

`dst_prefix`
Number. Destination network prefix.

`authent_key`
String. Authentication key in hexadecimal form (Empty string authentication is disabled).

`cipher_key`
String. Ciphering key in hexadecimal form (Empty string ciphering is disabled).

6.6 LTE messages

`users_get`
Get users state.
Message definition:

`registered_only.`
Optional boolean (default = false). If set, only registered user will be dumped.

Response definition:

`users` Array of object. Each item represents a user with following parameters:

`impi` String. IMPI of user (IP Multimedia Private identity).

`force_sms_over_sg`
Optional boolean. Current SMS over SG forcing state.

`bindings` Array of object. One for each contact binding:

`uri` String. Contact URI.

`impu` Array of strings. List of associated IMPUs.

`q` Number. Contact priority.

`video` Optional boolean. Video support.

`sms` Optional boolean. SMS pending.

`imei` Optional string. IMEI.

`expires` Integer. Number of seconds before binding expiration.

`dialogs` Array of object. One for each current dialog:

`remote` String. IMPI of remote user.

`sms` Integer. Number of pending SMS.

`users_add`
Add users.
Message definition:

`users.` Array of object. Same as info in configuration file: See [ue_db], page 18.

`user_set` Configure user.
Message definition:

`impi` String. IMPI of user to configure.

`force_sms_over_sg`
Optional boolean. Set/unset forcing of SMS over SG.

`impu_set` Configure impu.
Message definition:

`impu` String. IMPU to configure.

`*` Same parameters as `impu` configuration object. See [impu configuration], page 18.

`impu_add` Add impu.
Message definition:

`impu` String or object. IMPU to add See [impu configuration], page 18.

`impi` String. User for which to add IMPU. Can be set to `echo` to add it to the echo list.

`impu_del` Remove impu.
Message definition:

`impu` String or object. IMPU to add See [impu configuration], page 18.

`impi` String. User for which to remove IMPU. Can be set to `echo` to remove it from the echo list.

`sms` Send SMS.
Message definition:

`impi` Optional string. IMPI of user (IP Multimedia Private identity).

`impu` Optional string. If IMPI is not set, try to get user from IMPU (IP Multimedia Public identity).

`text` String. SMS text to send.

`sender` Optional string. Sets SMS sender.

`validity` Optional integer (Default = 86400). Validity period in seconds.

`binary` Optional string. If set (and `text` is not set), must be a base64 string representing binary data of the TP-User-Data.

`binary_hex`
Optional string. If set (and `text` is not set), must be an hexadecimal string representing binary data of the TP-User-Data.

`tp_udl` Optional integer. Used when `binary` or `binary_hex` is set. If present, it sets the TP-User-Data-Length field. If not present, the TP-User-Data-Length field is set to the number of octets of the `binary` or `binary_hex` field.

`tp_udhi_present`
Optional boolean (default is false). When `binary` or `binary_hex` is set, indicates if TP-User-Data start with a user-data header or not.

`pid` Optional integer (default is 0). Defines protocol identifier.

`dcs` Optional integer (default is 4). Defines data coding scheme. If the `text` parameter was provided, it's up to the user to ensure that the `dcs` value is coherent with the encoding automatically selected (7 bit default GSM alphabet or UCS2).

`sos` Optional boolean (default = false). If set, will only try to reach emergency registered UEs.

`sms_flush`
Flush pending SMS.
Message definition:

`impi` String. IMPI of user (IP Multimedia Private identity).

`mms` Send MMS.
Message definition:

`filename` String. File name to send. Extensions jpg, jpeg, png, gif and txt are supported.

`from` String. Sender phone number.

`to` String. Receiver phone number.

`sos` Optional boolean (default = false). If set, will only try to reach emergency registered UEs.

`mms_server`
Display the address of the MMS server or return an error if the MMS server is not started. Response definition:

`address` String. Address of the MMS server.

`mt_call` Initiate a mobile terminating call.
Message definition:

`impi` String. IMPI (IP Multimedia Private identity) of user to call.

`impu` String. IMPU (IP Multimedia Public identity) of user to call.

`contact` String. Contact SIP uri of user to call.

`sdp_file` Optional string. Define file to use as sdp. Will override *mt_call_sdp_file* parameter.

`sip_file` Optional string. Deprecated. Same as `sdp_file`.

`caller` Optional string. Use it to force caller IMPU. If IMPU is in user database, the P-Asserted-Identity header will be added.

`sos` Optional boolean (default = false). If set, will only try to reach emergency registered UEs.

`duration` Optional number. If set, call duration in seconds (The server will close the dialog).

Response definition:

`session_id`
String. If call has started, provides its session ID.

`dialog_get`
Get list of current pending dialogs.
Dialog will persist 30s after being stopped. Message definition:

`session_id`
Optional string. If set, filter on session ID.

Response definition:

`dialogs` Array of object representing dialogs as follow:

`session_id`
String. Dialog session ID.

`state` String. Dialog state, can be `init`, `ringing`, `start`, `hold` or `stop`.

`type` String. Dialog type, can be `server`, `echo` or `mt call`

`to` Callee IMPU.

`from` Caller IMPU.

`mt_dialog`
Optional string. In case of server dialog, session id of associated MT dialog.

`mo_dialog`
Optional string. In case of client dialog, session id of associated MO dialog.

`date` Integer. Dialog creation time in seconds since 1st January 1970.

`duration` Number. Number of seconds since dialog has started.

`event_list`
Array of object representing events that has occurred during dialog lifetime.
Each element have the following definition:

`type` String. Event type, can be `state`, when a state change occurs, `send` and `recv` when receiving or sending message.

`timestamp`
Number. Event time in seconds since dialog creation.

`state` String. Dialog state when event has occurred as defined above.

`medias` Array of object representing media state.
Each media is an object having following definition:

`type` String. Media type, can be `audio`, `video` or `text`.

`qos` String. QoS state, can be:

- disabled: QoS not enabled, IETF mode used.
- required: QoS required but not yet initiated.
- pending: QoS dedicated bearer establishment in progress.
- erab_set: QoS done.

`dir` String. Media current direction, can be `sendrecv`, `sendonly`, `recvonly` or `inactive`.

`rtp_addr` String. RTP packets destination address.

`rtp_recv_count`
Integer. Number of RTP packets received.

`rtp_send_count`
Integer. Number of RTP packets sent.

`rtp_block`
Optional boolean. If true, outgoing RTP packet are blocked.

`rtcp_addr`
String. RTCP packets destination address.

`rtcp_recv_count`
Integer. Number of RTCP packets received.

`rtcp_send_count`
Integer. Number of RTCP packets sent.

`rtcp_block`
Optional boolean. If true, outgoing RTCP packet are blocked.

`dialog_set`
Perform action on dialog Message definition:

`session_id`
String. Session ID of dialog to stop. If set to `*`, applies to the latest dialog found (useful when only one dialog available).

`action` String. Action to perform.

Action	**Description**
stop	Forces termination of the dialog.

answer	Triggers INVITE final answer of an echo called configured with `remote_control` option.
reinvite	Forces sending of INVITE if the dialog is started.
hold	If call is not held locally, hold it, unhold it. Applies on MT and ECHO calls only.
downgrade	Downgrades call by removing one media.
media	Modify media behavior.
ecall-info	In case eCall has been set during INVITE, forces sending of SIP INFO message to trigger MSD report.
session-expires-stop	Stop any pending session refresh mechanism.

`code` Optional integer. If action is `answer`, forces SIP code.
It action is `stop`, set protocol cause of SIP reason header.

`reason` Optional string. If action is `stop`, set reason text of SIP reason header.

`reason_protocol`
: Optional string. If action is `stop`, set protocol of SIP reason header.

`sdp_file`
: Optional string. Applicable on MT call dialogs only. Forces a new SDP to be sent.

`media`
: Optional string. If action is `downgrade`, defines media to remove (Can be `audio`, `video` or `text`).
If action is `hold`, hold/unhold only applies to this type of media.
If action is `media`, applies only on this media, else apply to all media types.

`rtp`
: Optional boolean. If action is `media`, set it to false to block rtp packet streaming or true to re-enable it.

`rtcp`
: Optional boolean. If action is `media`, set it to false to block rtcp packet streaming or true to re-enable it.

`unregister`
: Force a network deregistration of a binding. Message definition:

`uri`
: String. Binding URI (Address of Record)

`event`
: Optional string (default = deactivated). Notify event.

6.7 Remote events

Some messages (events) may be sent by LTEIMS without client sollicitation.
To receive them, you need to register to those events via [remote event registration], page 32.
The received JSON will have a `message` property with the events name.

Ex:
Register to `<event name>` event:

```
{
  message: "register",
  register: "<event name>"
}
```

Message received:

```
{
  message: "<event name>",
  ...
}
```

Here is the list of events generated by LTEIMS:

`sms`
: Generated by SMS reception:

`sender`
: String. SMS originator.

`destination`
: String. SMS destination.

`text`
: String. SMS text.

`binary`
: String. If `text` is not set, base64 encoded string of SMS data.

`dcs`
: Integer. Data coding scheme.

`date`
: Integer. SMS sending date in seconds since January 1st 1970.

`users_update`
: Event generated when a change occurs on a user (Registration, call, sms...).

  `users_update`
  : Array of object. Each item represents a user (See [users_get], page 33).

`dialog`
: Generated when a dialog's state has changed.

  `dialog` Object. Same as [dialog_get], page 36,

6.8 Examples

1. Config
   1. Client sends

```
{
    "message": "config_get",
    "message_id": "foo"
}
```

   2. Server replies

```
{
    "message_id": "foo",
    "message": "config_get",
    "name": "UE",
    "logs": {
        "phy": {
            "level": "error",
            "max_size": 0
        },
        ...
        "rrc": {
            "level": "debug",
            "max_size": 1
        }
    }
}
```

2. Error
   1. Client sends

```
{
    "message": "bar",
    "message_id": "foo"
}
```

   2. Server replies

```
{
    "message_id": "foo",
    "message": "bar",
    "error": "Unknown message: bar"
}
```

7 Callflow modifier examples

7.1 Call barring

To mimic call barring, you may use a callflow modifier.
For that, add a impu in `echo` list:

```
{
    impu: "sip:barring",
    match: '^\\+33',
    callflow_modifier: [{
        method: "INVITE",
        code: 403,
        stop: true,
        log: "Call barring",
        custom_headers: [{
            name: "Reason",
            value: "SIP; cause=21; text=\"Call barring active\"",
            replace: true,
        }],
    }],
},
```

Here is how it works:

- `impu: "sip:barring"` is to identify impu.
- `match: '^\\+33'` will associate any INVITE request sent to an impu starting by `'+33` to our impu. When calling such number, as dialog will be identified as an MO dialog, the callflow modifiers of our impu will be used.
- The callflow modifier just states, that on INVITE request:
  - send 403 response
  - stop dialog immediately
  - replace or add a `Reason` header set by LTEIMS by a custom one.

7.2 Call forward

To mimic call barring, you may use a callflow modifier.
For that, modify the impu impacted by call forwarding:

```
{
    impu: "tel:123456",
    callflow_modifier: [{
        method: "INVITE",
        code: 302,
        stop: true,
        log: "Call forward",
        custom_headers: [{
            name: "Contact",
            value: "<tel:666>",
            replace: true,
        }],
    }],
```

```
},
```

The callflow modifier just states, that on INVITE request:

- send 302 response
- stop dialog immediately
- replace or add a `Contact` header set by LTEIMS by a custom one.

Note that you can use `impu_set` remote API to add/remove such modifier to enable/disable call forwarding at runtime.
To add, just set `callflow_modifier` with the forward modifier array in your `impu_set` message.
To remove, just set `callflow_modifier` without the forward modifier in your `impu_set` message.

8 Command line monitor reference

The following commands are available:

`help` Display the help. Use `help` *`command`* to have a more detailed help about a command.

`log [`*`log_options`*`]`
: Display the current log state. If *log_options* are given, change the log options. The syntax is the same as the `log_options` configuration property.

`mme` Lists MME connections

`sms` *`impi`* `or` *`impu`* *`text`*
: Send a SMS to the user identified by *impi* or *impu* if *impi* has not been found.

`sms_flush` *`impi`*
: Flush pending SMS.

`mms` *`filename`* *`from`* *`to`*
: Send a MMS to the user identified by *to*. Extensions jpg, jpeg, png, gif and txt are supported. For any other extension value, the content type is interpreted as octet stream.

`mms_server`
: Display the address of the MMS server or return an error if the MMS server is not started.

`mt_call` *`callee`* `[-d` *`duration`*`]` `[`*`sip_file`*`]` `[`*`caller`*`]`
: Initiate a mobile terminating call.
callee can be IMPI, IMPU or contact URI.
sip_file Define file to use as sdp. Will override *mt_call_sdp_file* parameter.
caller can be used to force caller IMPU. If IMPU is in user database, the P-Asserted-Identity header will be added.
duration duration of the call in seconds before server closes it.

`dialog` Lists all dialogs.

`dialog_set` *`dialog-index`* *`action`*
: Perform action on dialog.
Can be `stop`, `hold` or `reinvite` (Cf [dialog_set], page 37).

`quit` Stop the program and exit.

9 Log file format

9.1 IMS, SIP

When a message is dumped, the format is:

```
time layer dir id message
```

`time`	Time using the selected format.
`layer`	Indicate the layer.
`dir`	FROM or TO or - (No direction associated).
`id`	For `IMS`, represents a unique ID associated with a UE binding. For `SIP`, represents a unique ID associated to a SIP dialog.
`message`	Log message.

9.2 CX, RX

When a message is dumped, the format is:

```
time layer dir addr message
```

`time`	Time using the selected format.
`layer`	Indicate the layer.
`dir`	FROM or TO or - (No direction associated).
`addr`	Source IP address for incoming messages. Destination IP address for outgoing messages.
`message`	Log message.

9.3 MEDIA

When a message is dumped, the format is:

```
time layer id dir protocol/media message
```

`time`	Time using the selected format.
`layer`	Indicate the layer.
`dir`	FROM or TO or - (No direction associated).
`id`	SIP associated dialog id.
`protocol`	Can be either `RTP` or `RTCP`.
`media`	Media type: `audio`, `video` or `text`.
`message`	Log message.

10 Change history

10.1 Version 2025-12-12

- OpenSSL library is upgraded to 3.5.4
- removed deprecated DES-CBC IPsec encryption algorithm
- added `session-expires-stop` action for `dialog_set` remote API
- callflow modifiers are logged by default
- added `send_retry` to callflow modifier
- added `send_retry_delay` to callflow modifier
- added `ecall.ack` to callflow modifier
- added `skip` to callflow modifier
- added `media` parameter for remote API `dialog_set/media`
- added `match` to impu configuration
- added `ttl` parameter to `sms_message_filter` configuration object
- `Content-Length` header can now be modified by callflow modifiers
- added `auth-int` qop directive
- added callflow modifier examples for call barring and call forward
- added `allow_held_invite` parameter
- added `proxy_auth` parameter

10.2 Version 2025-09-19

- added `sms` parameter to `impu` configuration
- added `AES-GCM-US` IPsec cipher
- added `AES-GMAC-US` IPsec authentication
- added `verstat` to `impu` configuration
- added `phone` to `impu` configuration
- added `reauthenticate` to `impu` configuration
- added `anonymous` to `SIP trunk` configuration
- added `use183` global parameter and for `impu`
- added `event` for `unregister` remote API
- added `ecall-info` message forwarding for PSAP

10.3 Version 2025-06-13

- Add Early media support
  - added `early_media` parameter to global and `impu` configuration
- added `rtp` and `rtcp` parameters to `impu` configuration
- added `mms_server` remote API and monitor command

10.4 Version 2025-03-14

- added `audio` and `text` parameter to `impu` configuration
- added `log` option to callflow modifier
- added `ipsec_tun_setup_script_filename` parameter
- added allow header removal in `custom_headers`
- added `mms_server` remote API and monitor command

10.5 Version 2024-12-13

- added `anonymous_callback` IMPU parameter
- added `send_delay` to callflow modifier
- added `ecall-info` to `dialog_set` remote API

10.6 Version 2024-09-13

- added `bitrate_factor` parameter to `impu` configuration
- added `video` parameter to `impu` configuration
- added media parameter for `action_set/hold` remote API to hold a specific media only
- added SMS status report adn SMS command support
- added `license` remote API
- added `tts` parameter to `callflow_modifier` object
- `com_logs_lock` parameter is renamed to `com_log_lock`. `com_logs_lock` is still supported for backward compatibility
- added `com_log_us` parameter
- added `reason` to `dialog_stop` monitor command
- added `sos` parameter to `sms` remote API
- added NG eCall support

10.7 Version 2024-06-14

- OpenSSL library is upgraded to 1.1.1w
- added `ipsec_ifname` and `ipsec_netns` parameters to allow IPsec with non root privileges
- added `AKAv2-SHA-256` authentication scheme
- added `AES-GCM` IPsec cipher
- added `AES-GMAC` IPsec authentication

10.8 Version 2024-03-15

- added `ttl` parameter to `callflow_modifier` object
- added `reason_protocol` parameter to `dialog_set.stop` remote API
- added `sms_delay` parameter to `impu` configuration
- added SIP trunk documentation
- added support of '+' in `sms_centre_address`
- added `ipsec_mtu` parameter
- updated SIP code list

10.9 Version 2023-12-15

- session timer support added
- added `loop_count` and `loop_delay` to remote API messages
- added `sim_events`, `sim_events_loop_count` and `sim_events_loop_delay`
- added `com_ssl_ca` parameter for SSL verification
- added `dialog` remote API event
- added `session_expires` parameter to `impu` object
- added `duration`, `min`, `refresher`, `method` and `force` to `session_expires` object

10.10 Version 2023-09-08

- added `media` action to `dialog_set` remote API to allow blocking or not rtp and/or rtcp streams
- `sms_centre_address` parameter is added
- `ipsec` remote API added
- `tcp_threshold` can now be changed during runtime with `config_set` remote API
- `sms_flush` remote API added

10.11 Version 2023-06-10

- N5 interface support is added
- added `impu_add` and `impu_del` remote API
- added `echo` monitor command
- added `auth` qop parameter
- `com_logs_lock` parameter added to disable logs configuration change via remote API

10.12 Version 2023-03-17

- `com_addr` parameter now uses [::] address instead of 0.0.0.0 in the delivered configuration file to allow IPv6 connection
- added `media_delay` to callflow modifier
- `auth_on_register_only` default value is changed to true instead of false
- added `REGISTER.SOS` callflow modifier
- added `INVITE.RINGING` callflow modifier
- added `INVITE.SESSION_PROGRESS` callflow modifier
- added `REINVITE` callflow modifier
- added code -2 to callflow modifier
- added `code` and `reason` parameters for `dialog_set/action` stop remote API
- added `downgrade` action and `media` parameter to `dialog_set` remote API

10.13 Version 2022-12-16

- added `custom_headers` for `callflow_modifier`
- added `path` parameter for registration
- added `dialog_set` remote API and monitor command. Replaces `dialog_stop`, `reinvite` and `dialog_answer` APIs
- added `hold` action on `dialog_set` remote API and monitor command
- added `imei` and `impu` filters on `callflow_modifier`
- added `sdp_file` to `dialog_set/reinvite` remote API
- added `utc` parameter to remote API response messages

10.14 Version 2022-09-16

- "ipsec debug" monitor is now deprecated. Set ipsec.verbose to 1 in log configuration
- `callflow_modifier` is added to impu definition and allows multiple modifiers
- allowed `callflow_modifier` based on `P-Access-Network-Info` header
- `tcp_keepalive` default value changed to 15 minutes

- `tcp_keepcount` parameter is added
- `custom_headers` parameter is added to `config_get` and `config_set` remote API
- `date` parameter is added to `sms` remote API event

10.15 Version 2022-06-17

- OpenSSL library is upgraded to 1.1.1n
- `code` parameter in `impu` objet now accepts the value -1 to ignore an incoming SIP message
- `tcp_keepalive` is added to control keepalive on TCP sockets
- allowed dual emergency/non emergency registration
- `sos` option is added to `mt_call` and `sms` remote APIs
- `start_timestamp` and `end_timestamp` are added to `log_get` API

10.16 Version 2022-03-18

- IMEI is now taken into account to improve calls between UEs using the same IMSI
- `binding_expires` parameter maximum value is increased from 3600 to 864000

10.17 Version 2021-12-17

- a new `sdp_file` parameter is added to force the SDP used in echo mode
- a new `answer_delay` parameter is added to IMPU configuration to delay final answer to INVITE request
- a new `INVITE.OK` option has been added to `method` parameter of IMPU configuration to apply `code` on final answer to INVITE request.
- a new `remote_control` option has been added to IMPU configuration. It can be handled via a new `invite` remote API event and a new `dialog_answer` remote API.
- a new `transparent` option has been added to IMPU configuration to avoid affecting normal callflow.
- a new `impu_set` remote API is added to dynamically update impu configurations.
- `REGISTER` handling has been added to impu `method` parameter.
- `INVITE.TRYING` handling has been added to impu `method` parameter.
- `license` monitor command is added

10.18 Version 2021-09-17

- the minimum GLIBC version is now 2.17
- logs can be displayed with microseconds precision
- `precondition` global parameter syntax is updated. Legacy boolean values are still available but we recommend to use `on` and `off` instead
- `force_user_agent parameter` is added to avoid overriding the UE user-agent string when forwarding SIP packets between UEs
- `prefered` parameter is renamed to `preferred`. Legacy name is still supported
- the logging format is further described

11 License

Abbreviations

APN	Access Point Name
IMPU	IP Multimedia Public Identity
IMPI	IP Multimedia Private Identity