[Draft Working Document]

IEEE P802.16.3 Architecture and Requirements for Mobile Broadband Network Performance Measurements

**IEEE 802.16 Working Group**

**Project P802.16.3**

**Roger B. Marks, Editor**

Table of Contents

1 Scope 4

2 References 5

3 Definitions and Abbreviations 5

3.1 Definitions 5

3.2 Abbreviations 5

4 Applications 6

5 Mobile-Specific Considerations 6

6 Architecture 7

6.1 Generic Architectural Reference Model 7

6.2 Expanded Architectural Reference Model showing Public and Private Entities 7

6.3 Functional Entities 9

7 Communication Links 11

7.1 Summary of Communication Links 11

8 Protocol for registration, configuration and data transfer 11

8.1 General 11

8.2 Registration and capability exchange 12

8.3 Configuration 13

8.4 Measurement synchronization 14

8.5 Measurements upload 15

8.6 Deregistration 16

9 Data elements and message structure 17

9.1 General 17

9.2 Capabilities exchange request 17

9.3 Capability exchange response 17

9.4 Registration request 17

9.5 Registration response 18

9.6 Get Registration parameter request 18

9.7 Get Registration parameter response 18

9.8 Set Registration parameter request 19

9.9 Set Registration parameter response 19

9.10 Configuration request 19

9.11 Configuration response 19

9.12 Get Configuration Parameter request 20

9.13 Get Configuration Parameter response 20

9.14 Set Configuration Parameter request 20

9.15 Set Configuration Parameter response 20

9.16 Information Command request 21

9.17 Information command response 21

9.18 Notification request 21

9.19 Notification response 21

9.20 Public Server to Controller – Registration 22

9.21 Controller to Controller – Configuration 22

9.22 Client configuration parameters 22

9.23 Server configuration parameters 23

9.24 Test Set measurement metadata 23

9.25 Test Set measurement 24

9.26 Measurement data transfer - Client to Private Data Collector – Storage 25

9.27 Measurement data transfer - Client to Public Data Collector – Storage 25

9.28 Measurement data transfer - Public Server to Public Data Collector – Storage 25

9.29 Measurement data transfer - Private Server to Private Data Collector – Storage 25

9.30 Measurement data transfer - Private Data Collector to Public Data Collector – Storage 26

10 Radio Measurements 26

11 Application test Measurements 26

12 Considerations on privacy protection involving transmission of data from Private Data Collector to Public Data Collector 26

13 Requirements 26

**List of Figures**

Figure 1: Generic Architectural Reference Model 7

Figure 2: Application of Architectural Reference Model 8

Figure 3: Capability exchange negotiation and Registration message flows 13

Figure 4: Configuration message flow 14

Figure 5: Measurement synchronization flow 15

Figure 6: Measurement upload flow 16

Figure 7: Measurement upload flow 16

List of Tables

Table 1: Assessment of key measurement applications per stakeholder role 6

Table 2: Functional Entities 11

Table 3: Communication links among Functional Entities 11

Table 4 17

Table 5 17

Table 6 18

Table 7 18

Table 8 18

Table 9 19

Table 10 19

Table 11 19

Table 12 19

Table 13 20

Table 14 20

Table 15 20

Table 16 20

Table 17 20

Table 18 21

Table 19 21

Table 20 21

Table 21 21

Table 22: Communication links: Public Server to Controller 22

Table 23: Communication links: Controller to Controller 22

Table 24 22

Table 25 23

Table 26: Test Set measurement metadata elements 24

Table 27: Test Set measurement elements 25

Table 28: Communication links: Client to Private Data Collector 25

Table 29: Communication links: Client to Public Data Collector 25

Table 30: Communication links: Public Server to Public Data Collector 25

Table 31: Communication links: Private Server to Private Data Collector 25

Table 32: Communication links: Private Data Collector to Public Data Collector 26

[Draft] IEEE 802.16.3 Architecture and Requirements for Mobile Broadband Network Performance Measurements

# Scope

The IEEE P802.16.3 draft standard shall be developed in accordance with the P802.16.3 project authorization request (PAR) and Five Criteria Statement (IEEE 802.16-12-0489-01-Gdoc), as approved on 30 August 2012 [1]. According to the PAR, the scope of the resulting standard shall be:

*This standard specifies procedures for characterizing the performance of deployed mobile broadband networks from a user perspective. It specifies metrics and test procedures as well as communication protocols and data formats allowing a network-based server to coordinate and manage test operation and data collection.*

The standard will address the following purpose:

*By standardizing the metrics and methods, the standard provides a framework for characterizing and assessing the performance of various mobile broadband networks. By standardizing the protocols and data formats, it allows for a measurement server to collect information from a disparate set of devices on the network.*

and the following need:

*Users of broadband mobile networks, including enterprises such as corporations and governments, lack reliable, comparable data on which to base their assessment of network performance. Such data can be valuable to determine overall network quality and to pinpoint specific weaknesses, including limitations in deployment. Improved knowledge of system performance will lead the market toward more effective networks and therefore encourage the redeployment of scarce spectrum using the most efficient technologies and implementations. Also, policy makers seeking information on performance of available networks will directly benefit by the opportunity to apply the standardized metrics and methods. Researchers will also gain by the ability to compare measured performance data to simulated results and thereby assess the theoretical models. One application of such information is the assessment of technology elements proposed during standards development.*

This document specifies, in addition, the requirements to be satisfied by the IEEE P802.16.3 draft standard. In order to explain and specify those requirements, it also indicates suitable applications, and it details the architecture, functional entities, and communication links to be specified, along with a list of data to be exchanged among the entities.

# References

[1] IEEE 802.16-12-0489-01, “Approved PAR P802.16.3, with Five Criteria: *Mobile Broadband Network Performance Measurements*” ([link](http://doc.wirelessman.org/16-12-0489-01))

[2] Steven Bauer, David Clark, and William Lehr, “Understanding Broadband Speed Measurements,” MITAS Working Paper, June 2010 ([link](http://mitas.csail.mit.edu/papers/Bauer_Clark_Lehr_Broadband_Speed_Measurements.pdf))

[3] William Lehr, Steven Bauer, and David D. Clark, “Measuring Internet Performance when Broadband is the New PSTN,” *The End of the Phone System: A by-invitation Experts’ Workshop*, The Wharton School, University of Pennsylvania Philadelphia, PA, May 16-18, 2012 ([link](http://mitas.csail.mit.edu/papers/lehr_bauer_clark_pstn_transition_6_2012.pdf))

[4] “Next-Generation Measurement Architecture Standardization and Outreach Group (NMASOG) – Architecture Standards and Specifications,” Federal Communications Commission, 2012 ([link](http://apps.fcc.gov/ecfs/document/view?id=7022008017))

[5] Henning Schulzrinne, Walter Johnston, and James Miller, “Large-Scale Measurement of Broadband Performance: Use Cases, Architecture and Protocol Requirements,” September 21, 2012 ([link](http://datatracker.ietf.org/doc/draft-schulzrinne-lmap-requirements/))

# Definitions and Abbreviations

## Definitions

1. **[Term**: definition]

## Abbreviations

FQDN Fully Qualified Domain Name

HO Handover

IRAT Inter Radio Access Technology

MBNPM Mobile Broadband Network Performance Measurements

PII Personally Identifiable Information

RAT Radio Access Technology

# Applications

In Table 1, we have listed key applications in tabular form, along with a list of various stakeholder roles, drawn significantly from PAR Item 5.6 (“Stakeholders for the Standard”). Table 1 also indicates an assessment of the applications of greatest interest to each stakeholder role.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Stakeholder** | | | | | | | | |
| **Measurement application** | **Governmental policy maker** | **User (individual or enterprise)** | **Cell tower operator** | **Wireless carrier / Network operator** | **Researcher** | **Standards developer** | **User device vendor** | **Application developer** | **Mobile Application Service Provider** |
| Overall data on Quality of Experience of set of networks available to consumers | x | x | x | x | x |  | x | x | x |
| Quality of Experience of a specific network |  | x | x | x | x |  | x | x | x |
| Identify limitations in deployment of a specific network |  | x | x | x |  |  |  | x |  |
| Monitor for changes in operation of a specific network |  | x |  | x |  |  |  |  |  |
| Diagnose problems in a specific network |  |  | x | x |  |  |  |  |  |
| improve knowledge of system performance |  |  |  | x | x |  |  |  | x |
| lead the market toward more effective networks | x |  |  |  |  |  |  |  | x |
| encourage the redeployment of scarce spectrum using efficient technologies and implementations | x |  | x | x |  |  |  |  |  |
| compare measured performance data to simulated results |  |  |  |  | x | x |  |  |  |
| assess theoretical models |  |  |  |  | x | x |  |  |  |
| assess technology elements proposed during standards development |  |  |  |  |  | x |  |  |  |

Table 1: Assessment of key measurement applications per stakeholder role

# Mobile-Specific Considerations

The standard shall take into consideration the specific circumstances relevant to mobility and the resultant implications on measurements. In the mobile case:

1. measurements will typically be related to a specific user device, rather than to a router on a LAN
2. a single user device can typically operate with multiple disparate network technologies
3. a single user device may connect with multiple operators
4. a user device experiences widely varying signal and network conditions
5. due to variability, far larger statistical samples may be required to draw generalized conclusions
6. significantly more metadata (including, for example, location information) is required to characterize the scenario of a specific sample
7. it may be necessary to trigger testing based on a set of environmental circumstances, such as location, rather than relying upon scenarios such as LAN quiescence as a trigger
8. active testing may be relatively more constrained due to practical issues, including data plan limits and battery consumption
9. underlying software on many mobile devices is relatively closed, and underlying network data is often relatively difficult to access

# Architecture

## Generic Architectural Reference Model

Figure 1 illustrates the generic architectural reference model. The reference model refers to five Functional Entities: Controller, Client, Server, Data Collector, and Network Parameter Host. The Functional Entities are described in more detail in subclause 6.3.

Note that the generic architectural reference model is similar to those described in other documents, such as [3], [4], and [5], but with a simplified set of communication links.



Figure 1: Generic Architectural Reference Model

## Expanded Architectural Reference Model showing Public and Private Entities

The expanded architectural reference model illustrated in Figure 2 indicates that the Measurement Client is able to communicate with two distinct forms of Measurement Server: Public and Private. Likewise, the Measurement Client is able to communicate with two distinct forms of Data Collector: Public and Private.



Figure 2: Application of Architectural Reference Model

Note that the Private Server and Private Data Collector do not register with the Controller and are unknown to it. Their identities need to be set by direct Client configuration and are not passed to the Controller. In effect, they are known only to the Client and to each other. In contrast the identities of public functional entities are known by the Controller.

The expanded architectural reference model, with additional functional entities, offers an additional set of implementation options that provide for a greater range of applications. For example, consider the Measurement Server:

* Some applications may prefer that the Measurement Server be publicly available. Such public accessibility allows the Measurement Server to provide a measurement termination point for experiments conducted by client devices belonging to general public consumers who have no access to a private Measurement Server. As a result, public Measurement Servers appear necessary to support large-scale consumer measurement campaigns.
* One limitation of a public Measurement Server is that the route to the server may not be representative of the traffic route of interest to the user. From the perspective of a large-scale consumer measurement campaign, that may not be a concern. However, from the perspective of a user, it could be a distinct weakness. In particular, some users may have a primary need for connectivity to a specific network; for example, an enterprise user may be most interested in connectivity to a corporate data server. In such cases, an appropriately-located private Measurement Server would best serve that user’s needs. A private Measurement Server could also provide additional advantages; for example, it could implement some custom measurement metrics of particular interest, and it could better protect the privacy of the user data.
* In the context of Figure 1, the Measurement Server could be public or private. However, it is possible to envision scenarios in which the functionality of the system would benefit from having both types of Measurement Server available to clients. For example, a large-scale consumer measurement campaign might have access to more data if it could convince enterprise users – those conducting measurements using a private Measurement Server – to conduct some measurements using a public Measurement Server as well.
* Note also that the public and private Measurement Servers may require different functionality. For example, the private Measurement Server may require additional authentication with respect to the Client. Also, as described in Figure 2, the public Measurement Server is provided with additional connectivity. It registers with the Controller, which allows a public Controller to select from a database of known public Measurement Servers, whereas a private Measurement Server might be known directly by the Client. Furthermore, Figure 2 indicates that the public Measurement Server submits measurement data to the Public Data Collector. For the purposes of large-scale consumer measurement campaigns, such data might be considered more reliable than data submitted by another entity. However, from the perspective of an enterprise user concerned with data privacy, such a data flow may be undesirable. However, the private Measurement Server might communicate data to the private Data Collector.

Note also a drawback to the use of the Public Server is that network operators could prioritize traffic to and from this server, which could result in measurements that inaccurately represent estimate the network performance experienced by users in practice. The use of a dual Measurement Server architecture could provide the opportunity for a check on such circumstances and could also allow controlled experiments to confirm.

These reasons help to motivate the inclusion of both Measurement Servers in the expanded architectural reference model. Likewise, we can consider the purpose of stipulating separate Public and Private Data Collector entities in the expanded architectural reference model as well. Clearly, large-scale consumer measurement campaigns require a public Data Collector, because the typical consumers lack another repository and because the campaign seeks to collect data from multiple Clients. The resulting data may be provided for public access. However, this results in a privacy dilemma. Since public users will be hesitant to volunteer for public data collection that potentially exposes their private information, it will be essential to ensure that collected public data is suitably anonymized. On the other hand, if the data is anonymous, it will not be of value to the individual users for analysis; the availability of such personalized data is the main incentive for the individual to participate in the campaign. The use of separate public and private Data Collectors provides an opportunity to resolve the dilemma. Professional or enterprise users, or any who wish to store data privately, are given the opportunity to do so, but opportunity is nevertheless provided for suitably anonymized data to be contributed to the large-scale campaign.

## Functional Entities

Table 1 specifies the Functional Entities of the Architectural Reference Model.

|  |  |  |
| --- | --- | --- |
| **Functional Entity** | **Type** | **Description** |
| Client |  | The Client is the central element of the Architectural Reference Model. It is typically embodied as software executing on the user edge device (the Client Device), typically a mobile terminal. The measurement process is intended to collect data representative of the performance of the network from the perspective of the user edge device. In the case of passive measurements, the Client will collect performance data characterizing communications to and from the Client Device. In the case of active measurements, the Client will initiate communications, for measurement purposes, with the Server. The Client posts resultant measurement data to one or more Data Collectors. In addition, the Public Server can submit experimental results to the Public Data Collector, using the address specified by the Client. |
| Controller |  | The Controller provides information to the Client governing the measurement process. This information includes the measurement triggers (which may include day/time information as well as other specific triggering details, such as location conditions). The Controller also provides the Client with the Server and Data Collector addresses. The Client registers with the Controller to indicate its address and availability to conduct measurements. It updates its registration status as needed.  Note: Inter-controller communications for configuration sharing may be specified. |
| Server | Public | The Server serves as a communication termination, providing a data source and data recipient for active measurements initiated by the Client. The Public Server registers with the Controller to indicate its address and availability to conduct measurements. It updates its registration status as needed. The address of the Public Server is specified to the Client by the Controller.  The results of measurements collected by a wide range of Clients using the Public Server should be readily comparable. Therefore, the characteristics of the Public Server should be well known and consistent, with minimal congestion and minimal variability.  The Public Server can submit experimental results to the Public Data Collector, using the address specified by the Client.  Note: A drawback to the use of the Public Server is that network operators could prioritize traffic to and from this server, which could result in measurements that inaccurately represent estimate the network performance experienced by users in practice. |
| Server | Private | The Server serves as a communication termination, providing a data source and data recipient for active measurements initiated by the Client. The Private Server is typically hosted in a network of primary interest to the user, so that measurement of communications between the Client and the Private Server are reflective of communications conducted by the Client device outside the measurement scenario.  The address of the Private Server is specified to the Client as a result of Client configuration controlled by the user.  Note: The tests conducted with the Private server need not be identical to those conducted with the Public server. |
| Data Collector | Public | The Data Collector receives measurement results from the Client. The Client transmits to the Public Data Collector only results that are intended for public use, with appropriate controls to prevent release of personally identifiable information (PII). In the case of active measurements, such data is limited to that collected from the Public Server. When a Private Data Collector is used, the Private Data Collector may forward public results to the Public Data Collector, in which case the Client need not be responsible for that transmission. |
| Data Collector | Private | The Data Collector receives measurement results from the Client. The Client transmits to the Private Data Collector results that are intended for private use. When the Private Data Collector is used, the Private Data Collector, as directed by the Client, may forward suitably anonymized public results to the Public Data Collector, in which case the Client need not be responsible for that transmission.  The address of the Private Data Collector is specified to the Client as a result of Client configuration controlled by the user. |
| Network Parameter Host |  | The Network Parameter Host is not used.  Note: The Network Parameter Host is included in the Architectural Reference Model for information only, since such a functional entity is described in other documents ([3],[4],[5]). Those documents are primarily oriented toward fixed networks. In those cases, this entity (also known as a “Network Information Subscription Server” or “Network Parameter Server”) stores information and provides such information about the “nominal” network performance, such as the nominal service characteristics as specified in a network subscription. Such information may be available to a network operator but is generally not publicly available, so accessing this information without violating privacy concerns is problematic. Also, in the mobile case, the active network access provider depends on circumstances; for example, the link may be to a wireless LAN or a roaming cellular provider, so that subscription information may be of little relevance. Furthermore, such information is of little value in the mobile environment generally, since performance is highly dependent on many environmental parameters that vary significantly with respect to nominal performance, and information about these environmental conditions can be obtained directly through observations collected by the Client. |

Table 2: Functional Entities

# Communication Links

## Summary of Communication Links

Table 2 summarizes the communication links among Functional Entities of the Architectural Reference Model.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| To=>  From⇓ | Client | Controller | Server-Public | Server-Private | Data Collector-Public | Data Collector-Private |
| Client | - | registration (including updates) | measurement execution  (initiator) | measurement execution  (initiator) | storage  (measured data and metadata, public)  Flow control | storage  (measured data and metadata, private)  Flow control |
| Controller | configuration  (schedules/  triggers)  (including updates) | configuration | - | - | - | - |
| Server-Public | measurement execution  (responder) | registration  (including updates) | - | - | storage  (measured data and metadata, public)  Flow control | - |
| Server-Private | measurement execution  (responder) | - | - | - | - | Storage (measured data and metadata, private)  Flow control |
| Data Collector-Public | - | Notifications | - | - | - | - |
| Data Collector-Private | - | Notifications | - | - | - | - |

Table 3: Communication links among Functional Entities

# Protocol for registration, configuration and data transfer

## General

The present section introduces the basic requirements and handshakes that the selected protocol for 802.16.3 has to support.

The described operations refer to registration and capabilities negotiation, configuration, measurements upload, commands’ synchronization and deregistration:

* REGISTRATION
  + Client to Controller registration
  + Public/Private Server to Controller registration
* CONFIGURATION
  + Controller to Client configuration
  + Controller to Controller configuration
  + Controller to Server configuration
* MEASUREMENT SYNCHRONIZATION
  + Client to Controller Commands synchronization
  + Public/Private Server to Controller Commands synchronization
* MEASUREMENTS UPLOAD
  + Flow control between Client (or Server) and Data Collector.
  + Measurements upload from Client to Data Collector
  + Measurements upload from Public/Private Server to Data Collector
* DEREGISTRATION
  + Client to Controller Deregistration
  + Public/Private Server to Controller Deregistration

The underlying protocol to be used for these basic procedures has to guarantee a reliable connection. In addition, it should be possible to setup a secure connection between the peers involved in the communication, if the communication path is considered potentially unsafe. At the same time, it shall always be possible getting a monitoring point in clear text at the centralized entities, either Collector or Controller.

In the following message flows, dashed lines mean optional phases/transactions.

## Registration and capability exchange

Goal: the “Registration” mechanism allows the Client (or the Server) to register itself to a Controller, in order to perform measurements.

Prerequisite: a Client (or a Server) can retrieve the routing information to access the Controller. An example could be a DNS interaction to retrieve Controller IP address from a FQDN.

Basic handshake: the logical message flow is shown in the figure below. The protocol is following a Request/Response model.

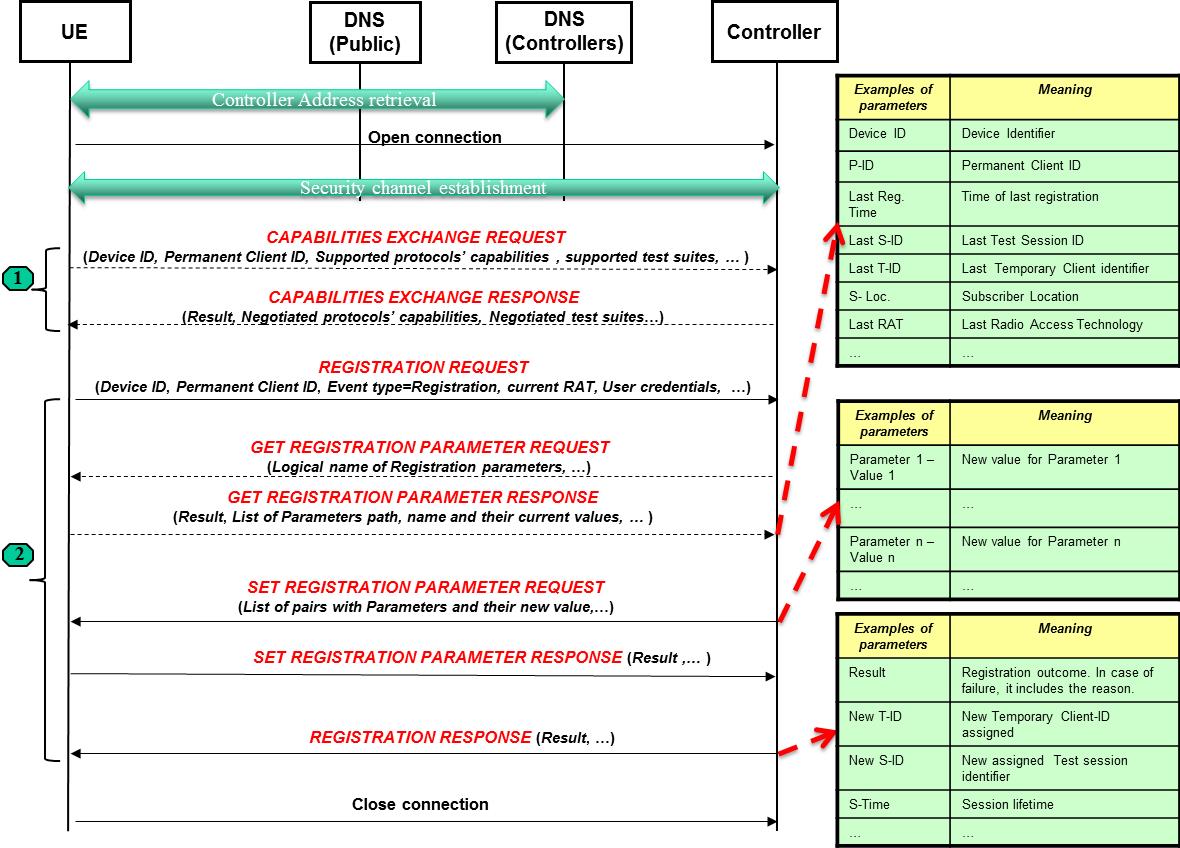


Figure 3: Capability exchange negotiation and Registration message flows

The first part of the message flow is related to the retrieval of the Controller address and the establishment of a secure reliable connection, if needed.

Then, it is possible even a negotiation phase (phase-1 in the figure), in order to setup the most suitable protocol and applications for communication. Once completed these phases, the Client is requesting the registration to the controller (phase-2 in the figure).

In turn, Controller can retrieve the path and current values of registration parameters and it shall configure the new parameters for the session, assigning, among the others, a new temporary Client-ID and a new Test Session Identifier.

The “Result” parameter is the outcome of each transaction and it is also the final outcome of the entire registration flow. Note that the dashed messages are optional in the flow.

A similar mechanism can be used by the Server for registration. In this case, the set of parameters will be a little bit different. For example, there isn’t any “Temporary Client ID” assigned to the server.

## Configuration

Goal: “Configuration” mechanism can be triggered either by the Client or by the Server in order to trigger a configuration from the Controller for a test session. It is possible also a Controller 🡪 Controller configuration, in specific scenarios where this makes sense.

Prerequisite: a Client (or a Server) is registered to the Controller.

Basic handshake: the logical message flow is shown in the figure below, in the example of UE-initiated configuration.

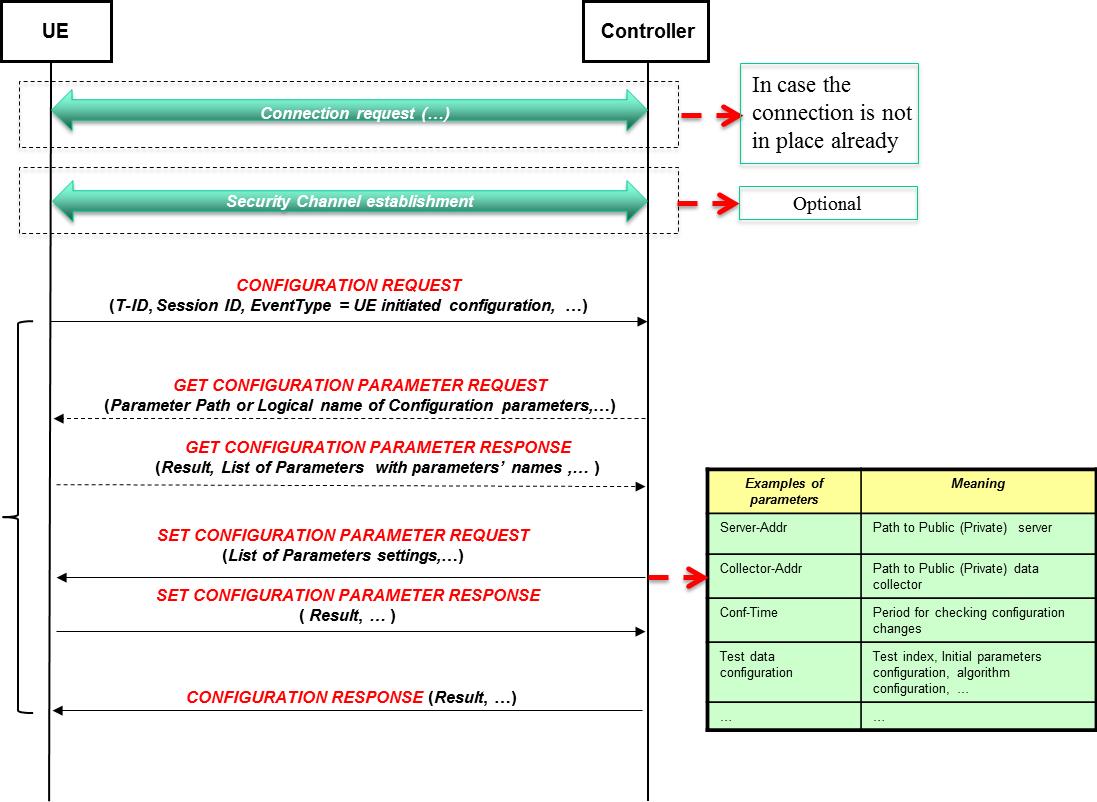


Figure 4: Configuration message flow

The “Connection request” and “Security Channel establishment” are optional phases, in case a secure connection is not in place already. The example is a configuration phase triggered by the UE: the transaction “CONFIGURATION REQUEST” / “CONFIGURATION RESPONSE” is encapsulating other transactions that allow retrieving which parameters are available for configuration (optional transaction GET CONFIGURATION PARAMETER REQUEST / RESPONSE) and setting them for the test session (transaction SET CONFIGURATION PARAMETER REQUEST / RESPONSE).

The “Result” IE is giving the outcome of each transaction and of the entire configuration procedure>. Its value is “successful” or the specific failure case, with the reason).

## Measurement synchronization

Goal: the Client (or the Server) can optionally synchronize itself with the Controller communicating either the “Start” or “Stop” of the specific measurement session.

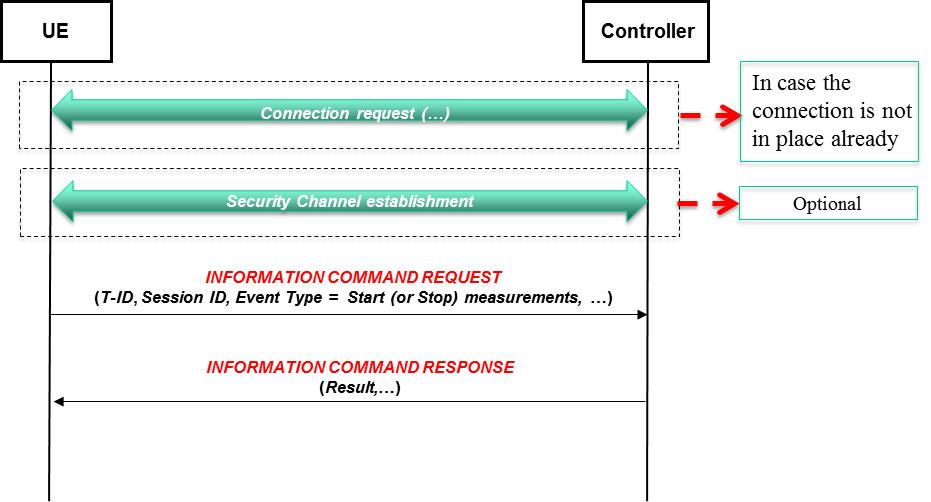


Figure 5: Measurement synchronization flow

As before, “Connection request” and “Security Channel establishment” are optional phases, in case a secure connection is not in place already. The example is an INFORMATION COMMAND REQUEST / RESPONSE to communicate either the “Start” or the “Stop” of the measurements on a specific Session ID.

## Measurements upload

Goal: this mechanism allows uploading the measurements results to the Data Collector.

In advance to the measurement upload there is the possibility to have a flow control between the Client (or Server) and the Data Collector, in order to be sure that the receiving entity is not overloaded and it can receive additional data.

Once the upload is completed, the Data Collector informs the Controller about the upload completion. The connection establishment and security channel establishment are optional phases.

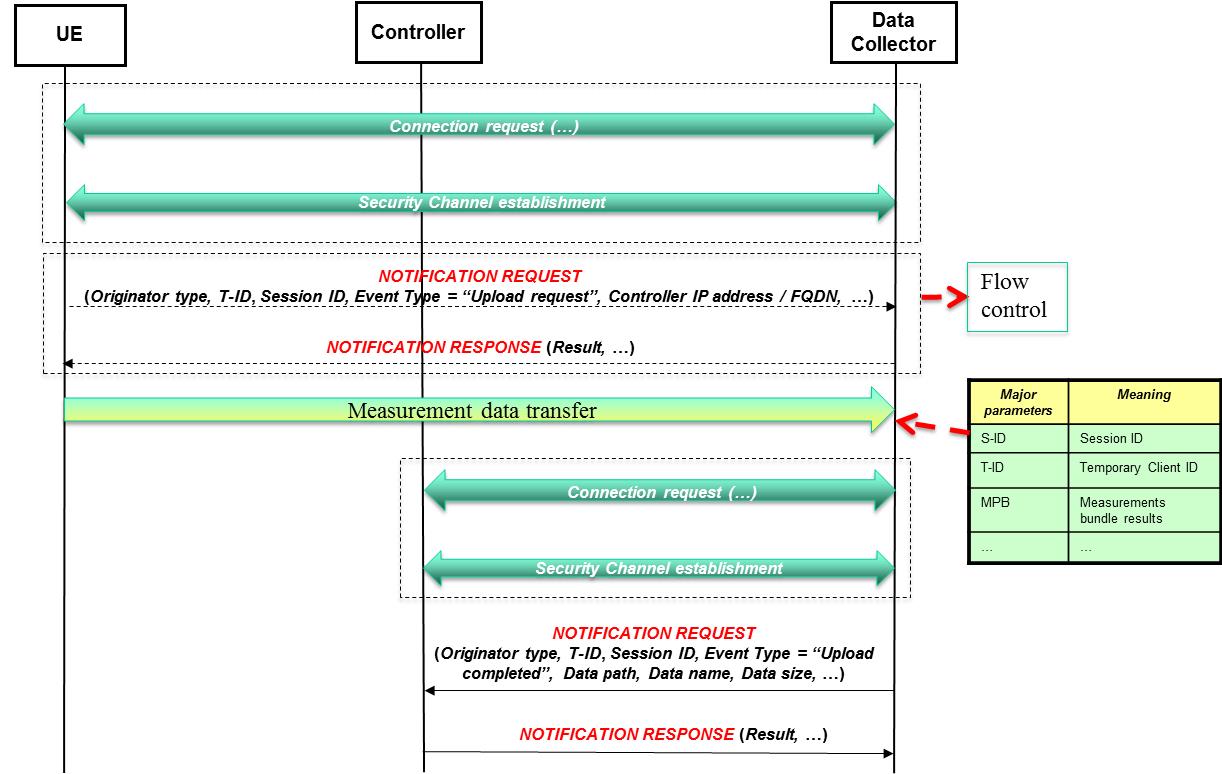


Figure 6: Measurement upload flow

## Deregistration

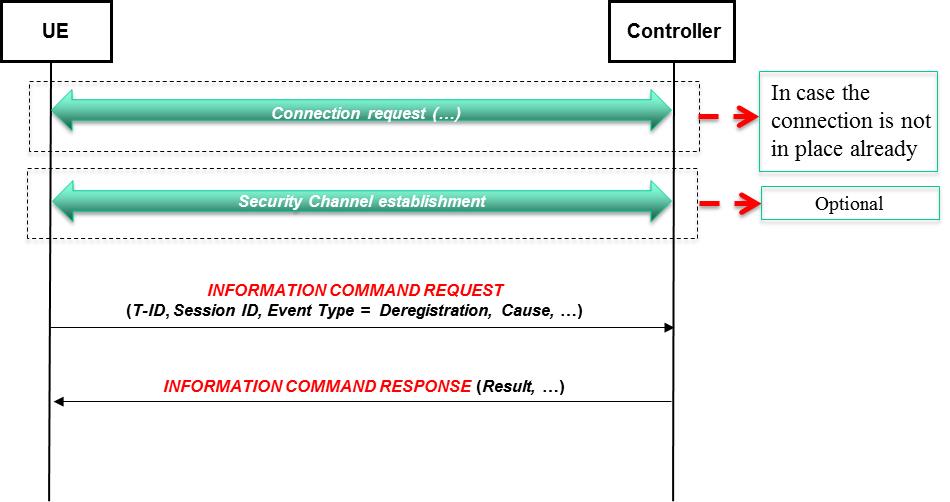


Figure 7: Measurement upload flow

Goal: “Deregistration” message flow is used to detach the Client (or Server) from the Controller.

So, all the temporary identifiers and the resources assigned by the Controller are released and they can be reassigned. In case of temporary identifiers assigned to the Client (or Server) it is recommended to avoid an immediate assignment of the same identifiers to a different entity (UE or Server) before a certain (implementation dependent) timeout. As before, the response message includes the Result of the transaction with proper failure cause in case it is an unsuccessful handshake. If the “Session lifetime” configured during registration expires, then there is the triggering of a deregistration from the Client. If this doesn’t happen, then after an additional guard timer there is the local detach of the Controller from the specific session.

# Data elements and message structure

## General

In this section it is listed the basic information elements to be exchanged between the peers and their meaning. About the “Transaction identifier” parameter that is present in all the transactions, it is a value defined by the transaction initiator and replied back by the responder.

## Capabilities exchange request

Direction: Client 🡪 Controller

Example of the scenario: [Registration and capability exchange](#_Registration_and_capability)

These are the parameters related to the capabilities negotiation between Client and Controller.

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| Transaction Identifier | Integer | Temporary transaction identifier. |
| Device ID | Octet string | Identifier of the device |
| Permanent Client ID | Octet string | Permanent client identifier (e.g. IMSI) |
| Supported protocol capabilities | Constructed | Bundle including all necessary IEs for protocol capabilities negotiation (tbd) |
| Supported test suites | Constructed | Bundle including all necessary IEs for identifying the possible test suites and algorithms (tbd). |
| [Supported Vendor ID] | Integer | Optional - Includes an identifier to specific vendor-dependent parameters’ set. |
| Current RAT | Enumerated (GERAN, UMTS, LTE, CDMA 1x, HRPD/eHRPD, WiFi, WiMAX, …) | Identifier of the current access network the Client is attached. |
| … | … | … |

Table 4

## Capability exchange response

Direction: Controller 🡪 Client

Example of the scenario: [Registration and capability exchange](#_Registration_and_capability)

These are the parameters included in the capability exchange response:

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| Transaction Identifier | Integer | Temporary transaction identifier. |
| Result | Integer | Value signaling the transaction outcome. The list of possible values has to include the “Successful” outcome and the other possible failure scenarios. |
| Negotiated protocol capabilities | Constructed | Bundle including all necessary IEs for protocol capabilities negotiation (tbd). It is the set of parameters chosen by the Controller among those sent by the Client. |
| Negotiated test suites | Constructed | Bundle including all necessary IEs for identifying the chosen list of possible test suites and algorithms. |
| … | … | … |

Table 5

## Registration request

Direction: Client 🡪 Controller

Example of the scenario: [Registration and capability exchange](#_Registration_and_capability)

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| Transaction Identifier | Integer | Temporary transaction identifier. |
| Device ID | Octet string | Identifier of the device |
| Permanent Client ID | Octet string | Permanent client identifier (e.g. IMSI in mobile domain) |
| Event type | Enumerated | It is the IE that identifies the type of event (in this case it assumes the value “Registration”). |
| Current RAT | Enumerated | Identifier of the current access network the Client is attached. |
| User credentials | Octet string | User name and password assigned out-of-band to the Client. |
| … | … | … |

Table 6

## Registration response

Direction: Controller 🡪 Client

Example of the scenario: [Registration and capability exchange](#_Registration_and_capability)

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| Transaction Identifier | Integer | Temporary transaction identifier. |
| Result | Integer | Value signaling the transaction outcome. The list of possible values has to include the “Successful” outcome and the other possible failure scenarios. |
| New T-ID | Octet string | Temporary Identifier assigned to the Client |
| New S-ID | Octet string | Temporary Identifier assigned to the Test Session |
| R-Time | Octet string(4) | Re-registration interval (how often the Client should re-register to the controller). |
| S-Time | Octet string(4) | Session lifetime in seconds |
| … | … | … |

Table 7

## Get Registration parameter request

Direction: Controller 🡪 Client

Example of the scenario: [Registration and capability exchange](#_Registration_and_capability)

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| Transaction Identifier | Integer | Temporary transaction identifier. |
| LogParam name | Constructed | List of logical names for the registration parameters. In the response there will be the current path/names as known by the Client. |
| … | … | … |

Table 8

## Get Registration parameter response

Direction: Client 🡪 Controller

Example of the scenario: [Registration and capability exchange](#_Registration_and_capability)

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| Transaction Identifier | Integer | Temporary transaction identifier. |
| Result | Integer | Value signaling the transaction outcome. |
| Device ID | Octet string | Temporary Identifier assigned to the Client |
| P-ID | Octet string | Permanent client identifier |
| Last Reg. Time | Constructed | Time/day of Last registration; “Null” in case it is the first registration. It allows to limit the rate of subsequent registrations. |
| Last S-ID | Octet string | Last Test Session identifier; “Null” in case it is the first registration. |
| Last T-ID | Octet string | Last Temporary Identifier assigned to the Client; “Null” in case it is the first registration. |
| S-Loc | Octet string | Location of the Client (e.g. Location Area - Cell identity) |
| Last RAT | Enumerated | Last Radio Access Technology; “Null” in case it is the first registration. |
| Current Registration Parameters list | Constructed | List of path, name and current values of the registration parameters (tbd). |
| … | … | … |

Table 9

## Set Registration parameter request

Direction: Controller 🡪 Client

Example of the scenario: [Registration and capability exchange](#_Registration_and_capability)

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| Transaction Identifier | Integer | Temporary transaction identifier. |
| Time/day | Constructed | Time and day of registration |
| New Registration Param. Settings | Constructed | List of registration parameters to set with name, [path] and values. |
| … | … | … |

Table 10

## Set Registration parameter response

Direction: Client 🡪 Controller

Example of the scenario: [Registration and capability exchange](#_Registration_and_capability)

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| Transaction Identifier | Integer | Temporary transaction identifier. |
| Result | Integer | Value signaling the transaction outcome. |
| … | … | … |

Table 11

## Configuration request

Direction: Client 🡪 Controller

Example of the scenario: [Configuration](#_Configuration)

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| Transaction Identifier | Integer | Temporary transaction identifier. |
| T-ID | Octet string | Temporary Identifier of the device |
| S-ID | Octet string | Test Session identifier |
| Event type | Enumerated | It is the IE that identifies the type of event (in this case it assumes the value “UE-initiated Configuration”). |
| … | … | … |

Table 12

## Configuration response

Direction: Controller 🡪 Client

Example of the scenario: [Configuration](#_Configuration)

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| Transaction Identifier | Integer | Temporary transaction identifier. |
| Result | Integer | Value signaling the transaction outcome. |
| … | … | … |

Table 13

## Get Configuration Parameter request

Direction: Controller 🡪 Client

Example of the scenario: [Configuration](#_Configuration)

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| Transaction Identifier | Integer | Temporary transaction identifier. |
| Conf. Param. name | Constructed | List of logical names for the configuration parameters. In the response there will be the current names as identified by the Client. |
| … | … | … |

Table 14

## Get Configuration Parameter response

Direction: Client 🡪 Controller

Example of the scenario: [Configuration](#_Configuration)

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| Transaction Identifier | Integer | Temporary transaction identifier. |
| Result | Integer | Value signaling the transaction outcome. |
| Current Conf. Param. list | Constructed | List of name, [path] and current values of the Configuration parameters. |
| … | … | … |

Table 15

## Set Configuration Parameter request

Direction: Controller 🡪 Client

Example of the scenario: [Configuration](#_Configuration)

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| Transaction Identifier | Integer | Temporary transaction identifier. |
| Server-Addr | Octet String | IP Address or FQDN of a “default” [Public] Server to (optionally) connect to |
| Collector-Addr | Octet String | IP address or FQDN of a [Public] Data Collector to connect to |
| Conf. Time | Integer | Period of time for next configuration checking: how often Client should check Controller for configuration changes (Unit: tbd) |
| New Conf. Param. Settings | Constructed | List of test parameters to set with name, [path] and new values. |
| … | … | … |

Table 16

## Set Configuration Parameter response

Direction: Client 🡪 Controller

Example of the scenario: [Configuration](#_Configuration)

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| Transaction Identifier | Integer | Temporary transaction identifier. |
| Result | Integer | Value signaling the transaction outcome. |
| … | … | … |

Table 17

## Information Command request

Direction: Client 🡪 Controller

Example of the scenario: [Measurement synchronization](#_Measurement_synchronization) or [Deregistration](#_Deregistration)

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| Transaction Identifier | Integer | Temporary transaction identifier. |
| T-ID | Octet string | Temporary Identifier of the device |
| S-ID | Octet string | Test Session Identifier |
| Event type | Enumerated | It is the IE that identifies the type of event (it can assume for example values as “Start”, “Stop” or “Deregistration”). |
| [Cause] | Integer | This optional field details the reason for deregistration. |
| … | … | … |

Table 18

## Information command response

Direction: Controller 🡪 Client

Example of the scenario: [Measurement synchronization](#_Measurement_synchronization) or [Deregistration](#_Deregistration)

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| Transaction Identifier | Integer | Temporary transaction identifier. |
| Result | Integer | Value signaling the transaction outcome. |
| … | … | … |

Table 19

## Notification request

Direction: Client 🡪 Data Collector or Data Collector 🡪 Controller

Example of the scenario: [Measurements upload](#_Measurements_upload)

Used to send information from the originator to the destination peer.

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| Transaction Identifier | Integer | Temporary transaction identifier. |
| Originator type | Enumerated | (Client, Server, Data Collector, …) |
| T-ID | Octet string | Temporary Identifier of the device |
| S-ID | Octet string | Test Session Identifier |
| Event type | Enumerated | It is the IE that identifies the type of event (e.g. “Upload request” or “Upload completed”). |
| [Controller-Addr] | Octet String | Optional - IP address or FQDN of the Controller |
| … | … | … |

Table 20

## Notification response

Direction: Controller 🡪 Client or Data Collector 🡪 Client

Example of the scenario: [Measurements upload](#_Measurements_upload)

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| Transaction Identifier | Integer | Temporary transaction identifier. |
| Result | Integer | Value signaling the transaction outcome. |
| [Event details] | Constructed | Optional - Details about the event (e.g. Data path, Data Name, Data size) |
| … | … | … |

Table 21

## Public Server to Controller – Registration

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| tbd |  |  |
|  |  |  |

Table 22: Communication links: Public Server to Controller

## Controller to Controller – Configuration

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| tbd |  |  |
|  |  |  |

Table 23: Communication links: Controller to Controller

## Client configuration parameters

Here’s a list of parameters to be considered for Client configurations. They should be included into “New Conf. Param. Settings” within “Set Parameter Request” operation.

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| Data Usage Limit | Integer/kilobytes | The maximum number of bytes the Client will transfer in a test session |
| Distance Threshold | Integer/meters | Land distance from previous test a Client must exceed to perform its next test set |
| Time Threshold | Integer/minutes | Time since previous test a Client must exceed to perform its next test set |
| Location Threshold | Integer/meters | Accuracy threshold above which Client will not initiate test set |
| > Application Test Set | Constructed | List of indexes on the application tests to perform |
| >> Test Parameters | Constructed | For each test, list of test-specific parameters / values |
| >> Test algorithm | Constructed | For each test, if applicable, the index to the chosen algorithm to adopt for the test. |
| >> Trigger condition | Constructed | For each test, if applicable, the specification of the trigger condition for the test. |
| >> Measurements configuration | Constructed | The configuration of which measurements to report for each test. |
| >>> Reporting configuration | Constructed | If applicable, the configuration of the measurements reporting. For example, the configuration of “bins” intervals for a specific measurement. |
| > Radio Measurements | Constructed | List of radio measurements (see section 10 for the list of possible measurements). |
| >> Radio Measurements configuration | Constructed | The configuration of which radio measurements to report. |
| >>> Radio Meas. Reporting configuration | Constructed | If applicable, the configuration of the measurements reporting. For example, the configuration of “bins” intervals for a specific measurement. |
| Battery Threshold | Integer/Percentage | The battery level below which the Client won’t perform any test nor submit measurements to Data Collector |
| Sampling Flag | Enumerated( No Sampling, Sampling) | Flag to signal sampling user plane |
| Sampling type | Enumerated( Uniform, Begin Session, Adaptive, …) | It is the configuration of the sampling method |
| Sampling interval | Integer | If applicable, it signals the pace for sampling. |
| [additional conditional triggering parameters] | … | … |

Table 24

## Server configuration parameters

The server configuration parameters are a subset of the Client configuration parameters:

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| Data Usage Limit | Integer/kilobytes | The maximum number of bytes the Server will transfer in a test session |
| Time Threshold | Integer/minutes | Time since previous test a Server must exceed to perform its next test set |
| > Application Test Set | Constructed | List of indexes on the application tests to perform |
| >> Test Parameters | Constructed | For each test, list of test-specific parameters / values |
| >> Test algorithm | Constructed | For each test, if applicable, the index to the chosen algorithm to adopt for the test. |
| >> Trigger condition | Constructed | For each test, if applicable, the specification of the trigger condition for the test. |
| >> Measurements configuration | Constructed | The configuration of which measurements to report for each test. |
| >>> Reporting configuration | Constructed | If applicable, the configuration of the measurements reporting. For example, the configuration of “bins” intervals for a specific measurement. |
| Sampling configuration | Constructed | User plane Sampling configuration |
| Sampling Flag | Enumerated( No Sampling, Sampling) | Flag to signal sampling user plane |
| Sampling type | Enumerated( Uniform, Begin Session, Adaptive, …) | It is the configuration of the sampling method |
| Sampling interval | Integer | If applicable, it signals the pace for sampling. |
| [additional conditional triggering parameters] | … | … |

Table 25

## Test Set measurement metadata

|  |
| --- |
| **Description** |
| Test Set measurement metadata elements |
| Note: Elements to be considered as public or private, subject to privacy policy review. |
| Public Server Identifier  Cellular carrier  Network type  Network technology  (in case of IRAT HO, it can be present both source and target, otherwise only one RAT).  Network identifier  Base station identifier  Cell identifier  (in case of mobility, source and target cell can be present, otherwise only one Cell ID is there. In case of WCDMA diversity, the “best cell” can be reported).  Cell location code  (e.g. Location Area, Tracking Area, Routing Area, Service Area, if available)  Home Network Mobile County Code  Home Network Mobile Network Code  Roaming state  [Visited Network Mobile County Code]  [Visited Network Mobile Network Code]  Wi-Fi radio state  Wi-Fi connection state  Enterprise identifier  Latitude  Longitude  Altitude  Speed of travel  Direction of travel  Location Accuracy  Location data provider  SIM Mobile County Code  SIM Mobile Network Code  SIM provider  Power adapter state  Battery state  Battery charge level  Battery temperature  Battery voltage  Device unique identifier  Device manufacturer  Device brand  Device model  Device operating system type/version  BIOS Identifier  CPU information  CPU activity/load  Screen resolution  Free disk/RAM space  Free memory card storage space  Number of running apps  Sampling flag  (set if user plane sampling is applied)  Failure summary flag  (set if any error occurred during tests). |
| Test conditions  Temporary Client (or Server) identity  Test Session Identity |

Table 26: Test Set measurement metadata elements

## Test Set measurement

The Test Set measurement data are the results of the configured tests. The following is the structure of the bundle including these results. Each measurement will have its proper format according to this framework.

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| Measurement Start time | Constructed | Time/day of measurement start |
| Measurement Stop time | Constructed | Time/day of measurement stop |
| >Measurement bundle | Constructed | Envelope to include measurement results |
| >> Length of IE | Integer | Length of the bundle |
| >>> Meas. Identifier | Integer | Identifier of a measurement, as set during configuration phase. |
| >>> Meas. Length | Integer | Length of the specific measurement |
| >>> Meas. Container | Constructed | Depending on the measurement identifier, there can be several information included: either only measurement values or both parameters and measurements related to the scenario under test. |
| >>> Meas. Identifier | Integer | Identifier of a measurement, as set during configuration phase. |
| >>> Meas. Length | Integer | Length of the specific measurement |
| >>> Meas. Container | Constructed | As described before. |
| … | … | … |

Table 27: Test Set measurement elements

## Measurement data transfer - Client to Private Data Collector – Storage

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| T-ID | Octet string | Temporary Identifier of the device |
| S-ID | Octet string | Temporary Identifier assigned to the Test Session |
| Controller identifier | Octet string | IP address and FQDN of the Controller involved in the session |
| Test Set – Public | Constructed | Test Set measurement report - Public |
| Metadata Set – Public | Constructed | Test Set measurement condition report – Public  *Note: Test Set measurement metadata elements, considered as public based on privacy policy review* |
| Test Set – Private | Constructed | Test Set measurement report - Private |
| Metadata Set – Private | Constructed | Test Set measurement condition report – Private  *Note: Test Set measurement metadata elements, considered as private based on privacy policy review* |

Table 28: Communication links: Client to Private Data Collector

## Measurement data transfer - Client to Public Data Collector – Storage

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| T-ID | Octet string | Temporary Identifier of the device |
| S-ID | Octet string | Temporary Identifier assigned to the Test Session |
| Controller identifier | Octet string | IP address and FQDN of the Controller involved in the session |
| Test Set – Public | Constructed | Test Set measurement report |
| Metadata Set – Public | Constructed | Test Set measurement condition report – Public  *Note: Test Set measurement metadata elements, considered as public based on privacy policy review.* |

*Note: The Public Data Collector could receive data from* *the* *Private Data Collector (per 8.14) rather than from the Client, obviating the need to duplicate over-the-air communication from the Client. Data propagating via the Private Data Collector might differ due to obfuscation techniques, etc.*

Table 29: Communication links: Client to Public Data Collector

## Measurement data transfer - Public Server to Public Data Collector – Storage

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| Server Identifier | Octet string | Unique identifier per server. |
| S-ID Session ~~Experiment~~ Identifier | Octet string | Provides a record of each experiment, for later validation by correlation with Client Test Set measurement report. |
| Controller identifier | Octet string | IP address and FQDN of the Controller involved in the session |
| tbd | … | … |

Table 30: Communication links: Public Server to Public Data Collector

## Measurement data transfer - Private Server to Private Data Collector – Storage

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| Server Identifier | Octet string | Unique identifier per server. |
| S-ID Session ~~Experiment~~ Identifier | Octet string | Provides a record of each experiment, for later validation by correlation with Client Test Set measurement report. |
| Controller identifier | Octet string | IP address and FQDN of the Controller involved in the session |
| tbd | … | … |

Table 31: Communication links: Private Server to Private Data Collector

## Measurement data transfer - Private Data Collector to Public Data Collector – Storage

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Type/Units** | **Description** |
| Temporary Server ID | Octet string | Temporary identifier of the private server assigned by the Controller |
| S-ID | Octet string | Temporary Identifier assigned to the Test Session |
| Controller identifier | Octet string | IP address and FQDN of the Controller involved in the session |
| Test Set – Public | Constructed | Test Set measurement report - Public |
| Metadata Set – Public | Constructed | Test Set measurement condition report – Public  *Note: Test Set measurement metadata elements, considered as public based on privacy policy review.* |

Table 32: Communication links: Private Data Collector to Public Data Collector

# Radio Measurements

The radio measurements that can be supported are:

* Min / Max / Mean Signal Strength (RSSI)
* Interference measurement (e.g. Downlink SIR)
* Chip energy over noise (Ec/No)
* tbd…

# Application test Measurements

The application measurements that should be supported could be considered in conjunction with the outcome of the IETF IPPM “IP Performance metrics” working group.

Here’s a link to current status for this IETF WG: <http://www.ietf.org/mail-archive/web/ippm/current/msg03274.html>

In addition to active testing with specific test packets designed for “ad-hoc” measurements (for example delays), it could be considered also measurements obtained analyzing normal traffic to typical hosts/services, in order to check the real end-user experience.

# Considerations on privacy protection involving transmission of data from Private Data Collector to Public Data Collector

[tbd]

[location obfuscation, etc.?]

[usage guidelines?]

The dialogue between the UE (or the Server) and the Controller has to be protected, for example using HTTPS mechanism, when there is the exchange of privacy sensitive information. For the same phases, it is necessary that the two peers involved in a dialogue can authenticate one each other.

The same for the dialogue between Client (or Server) and (Private/Public) Data Collector or between different Controllers, again in case of privacy-sensitive information exchange.

In addition, the privacy-sensitive information stored in the Data Collectors have to be protected, if for example they are related to single users and not anonymous users or group of users.

In any case, the access to the measurement results and configuration data in the Controller and in the Data Collector has to be protected with adequate IT mechanisms.

The Controller is also providing temporary identities to the Client and to the test session, in order to protect the real identities of the UE performing the tests.

# Requirements

1. The standard shall specify procedures for characterizing and assessing the performance of deployed mobile broadband networks from a user perspective.
2. The standard shall specify metrics broadly applicable to all IP-based mobile broadband networks.
3. The standard should reference metrics specified by IETF (particularly from the IP Performance Metrics (IPPM) Working Group) whenever feasible.
4. The standard shall specify to consider also radio quality measurements, if possible.
5. The standard shall specify to consider measurements related to different radio conditions, if possible (e.g. the latency between last sent/received packet on a source cell and first sent/received packet on a target cell, after mobility).
6. The standard shall specify test procedures.
7. The standard shall specify procedures for a measurement server to collect information from a disparate set of user devices on the network.
8. The standard shall specify communication and data exchange protocols and data formats allowing a network-based server to coordinate and manage test operation and data collection.
9. The standard shall be implementable in software.
10. The standard should be compatible with implementation by any IP-based server in conjunction with any IP-based user device.
11. The standard should consider how to minimize (consistent with an overall optimized solution) the cost burden on the user device due to the extent that data transfer may be subject to a fee from the carrier, may interfere with other active user device processes, and may drain the user device power.
12. The standard shall specify procedures for measuring including uplink throughput rate, downlink throughput rate, latency, and jitter.
13. The standard shall specify procedures for quantifying packet loss and timeouts.
14. The standard shall specify procedures for collecting and transmitting various types of metadata, to include carrier network, network type, cell ID, user device make/model, network policy information, and radio resource control parameters, if available. The metadata will include the test conditions, Client (or Server) temporary identity and the test session identification.
15. The standard shall specify how test results are validated.
16. The standard shall specify how the measurements are built (e.g. one measurement per session, periodical measurements, measurements on trigger).
17. The standard shall specify procedures for collecting and transmitting user device location and location accuracy associated with measurement events.
18. The standard shall specify procedures for reducing user device location accuracy for privacy protection.
19. The standard shall specify procedures to ensure that Personally Identifiable Information (PII) is treated sensitively and protected from unauthorized disclosure.
20. The standard shall specify procedures to manage and respond to user consent authorization with regard to PII.
21. The standard shall specify anonymization procedures.
22. The standard shall provide for control of the tradeoff between cost and performance, so that cost-driven users can reduce the number of measurements and the thoroughness of measurements to obtain lower-cost operation, albeit with less complete information. The standard should recommend means of estimating and reporting the statistical validity of a set of measurement data.
23. The standard shall specify procedures based on active probing.
24. The standard should specify procedures based on passive measurements.
25. The standard shall support the needs of the public and research communities for collection of openly accessible anonymized data.
26. The standard is defining a general method to avoid that test traffic takes any advantage or network resource preemption against normal traffic.
27. The standard shall specify a method to uniquely identify the specific UE, Controller and Server host involved in a test session. At the same time, UE private identity has to be protected.
28. The standard shall specify how the UE gets informed about the temporary identifiers assigned for the specific test session.
29. These identifiers don’t prevent the unique identification of the test results.
30. The standard should specify a method to transfer the test session identifiers from the UE to the service host, before the test completion.
31. The standard shall specify how an authorized external entity/system can query the test results of a specific session,
32. The standard shall consider which requirements have to be satisfied by the “external entity/system” for authorization/authentication, data transfer reliability and data confidentiality.
33. The standard shall specify the protocol used to transfer such test results to the external system and the data format.
34. [addition requirements (tbd)]