**3GPP TSG-CT WG4 Meeting #96C4-200842**

**E-Meeting, 24th – 28th February 2020**

|  |
| --- |
| *CR-Form-v12.0* |
| **CHANGE REQUEST** |
|  |
|  | **29.571** | **CR** | **0191** | **rev** | **-** | **Current version:** | **16.2.0** |  |
|  |
| *For* [***HE******LP***](http://www.3gpp.org/3G_Specs/CRs.htm#_blank)*on using this form: comprehensive instructions can be found at* [*http://www.3gpp.org/Change-Requests*](http://www.3gpp.org/Change-Requests)*.* |
|  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Proposed change affects:*** | UICC apps |  | ME |  | Radio Access Network |  | Core Network | **X** |

|  |
| --- |
|  |
| ***Title:***  | Pattern of Ipv4AddrMask |
|  |  |
| ***Source to WG:*** | Huawei |
| ***Source to TSG:*** | CT4 |
|  |  |
| ***Work item code:*** | SBIProtoc16 |  | ***Date:*** | 2020-02-07 |
|  |  |  |  |  |
| ***Category:*** | **F** |  | ***Release:*** | Rel-16 |
|  | *Use one of the following categories:****F*** *(correction)****A*** *(mirror corresponding to a change in an earlier release)****B*** *(addition of feature),* ***C*** *(functional modification of feature)****D*** *(editorial modification)*Detailed explanations of the above categories canbe found in 3GPP [TR 21.900](http://www.3gpp.org/ftp/Specs/html-info/21900.htm). | *Use one of the following releases:Rel-8 (Release 8)Rel-9 (Release 9)Rel-10 (Release 10)Rel-11 (Release 11)Rel-12 (Release 12)**Rel-13 (Release 13)Rel-14 (Release 14)Rel-15 (Release 15)Rel-16 (Release 16)* |
|  |  |
| ***Reason for change:*** | The pattern of data model Ipv4AddrMask is not exact enought. The Mask of IPV4 address should be between 0 and 32, but in current pattern it can be any characters. |
|  |  |
| ***Summary of change:*** | Correct pattern of data model Ipv4AddrMask. |
|  |  |
| ***Consequences if not approved:*** | The pattern of data model Ipv4AddrMask is not exact enought. |
|  |  |
| ***Clauses affected:*** | 5.2.2, A.2 |
|  |  |
|  | **Y** | **N** |  |  |
| ***Other specs*** |  | **X** |  Other core specifications  | TS/TR ... CR ...  |
| ***affected:*** |  | **X** |  Test specifications | TS/TR ... CR ...  |
| ***(show related CRs)*** |  | **X** |  O&M Specifications | TS/TR ... CR ...  |
|  |  |
| ***Other comments:*** | This CR will introduce backward compatible new features in the OpenAPI specification file of TS29571\_CommonData.yaml, TS29503\_Nudm\_SDM.yaml, TS29505\_Subscription\_Data.yaml. |
|  |  |
| ***This CR's revision history:*** |  |

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*The start of changes\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

### 5.2.2 Simple Data Types

This clause specifies common simple data types.

Table 5.2.2-1: Simple Data Types

|  |  |  |
| --- | --- | --- |
| Type Name | Type Definition | Description |
| Binary | string | String with format "binary" as defined in OpenAPI Specification [3] |
| BinaryRm | string | This data type is defined in the same way as the "Binary" data type, but with the OpenAPI "nullable: true" property. |
| Bytes | string | String with format "byte" as defined in OpenAPI Specification [3], i.e, base64-encoded characters, |
| BytesRm | string | This data type is defined in the same way as the "Bytes" data type, but with the OpenAPI "nullable: true" property. |
| Date | string | String with format "date" as defined in OpenAPI Specification [3] |
| DateRm | string | This data type is defined in the same way as the "Date" data type, but with the OpenAPI "nullable: true" property. |
| DateTime | string | String with format "date-time" as defined in OpenAPI Specification [3] |
| DateTimeRm | string | This data type is defined in the same way as the "DateTime" data type, but with the OpenAPI "nullable: true" property. |
| DiameterIdentity | string | String containing a Diameter Identity, according to clause 4.3 of IETF RFC 6733 [18].Pattern: '^([A-Za-z0-9]+([-A-Za-z0-9]+)\.)+[a-z]{2,}$' |
| DiameterIdentityRm | string | This data type is defined in the same way as the "DiameterIdentity" data type, but with the OpenAPI "nullable: true" property. |
| Double | number | Number with format "double" as defined in OpenAPI Specification [3] |
| DoubleRm | number | This data type is defined in the same way as the "Double" data type, but with the OpenAPI "nullable: true" property. |
| DurationSec | integer | Unsigned integer identifying a period of time in units of seconds.  |
| DurationSecRm | integer | This data type is defined in the same way as the "DurationSec" data type, but with the OpenAPI "nullable: true" property. |
| Float | number | Number with format "float" as defined in OpenAPI Specification [3] |
| FloatRm | number | This data type is defined in the same way as the "Float" data type, but with the OpenAPI "nullable: true" property. |
| Uint16 | integer | Unsigned 16-bit integers, i.e. only value between 0 and 65535 are permissible. |
| Uint16Rm | integer | This data type is defined in the same way as the "Uint16" data type, but with the OpenAPI "nullable: true" property. |
| Int32 | integer | Integer with format "int32" as defined in OpenAPI Specification [3] |
| Int32Rm | integer | This data type is defined in the same way as the "Int32" data type, but with the OpenAPI "nullable: true" property. |
| Int64 | integer | Integer with format "int64" as defined in OpenAPI Specification [3] |
| Int64Rm | integer | This data type is defined in the same way as the "Int64" data type, but with the OpenAPI "nullable: true" property. |
| Ipv4Addr | string | String identifying a IPv4 address formatted in the "dotted decimal" notation as defined in in IETF RFC 1166 [4].Pattern: '^(([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])$' |
| Ipv4AddrRm | string | This data type is defined in the same way as the "Ipv4Addr" data type, but with the OpenAPI "nullable: true" property. |
| Ipv4AddrMask | string | String identifying a IPv4 address mask formatted in the "dotted decimal" notation as defined in in IETF RFC 1166 [4].Pattern: '^(([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])(\/([0-9]|[1-2][0-9]|3[0-2]))$' |
| Ipv4AddrMaskRm | string | This data type is defined in the same way as the "Ipv4AddrMask" data type, but with the OpenAPI "nullable: true" property. |
| Ipv6Addr | string | String identifying an IPv6 address formatted according to clause 4 of IETF RFC 5952 [5]. The mixed IPv4 IPv6 notation according to clause 5 of IETF RFC 5952 [5] shall not be used.Pattern: '^((:|(0?|([1-9a-f][0-9a-f]{0,3}))):)((0?|([1-9a-f][0-9a-f]{0,3})):){0,6}(:|(0?|([1-9a-f][0-9a-f]{0,3})))$'andPattern: '^((([^:]+:){7}([^:]+))|((([^:]+:)\*[^:]+)?::(([^:]+:)\*[^:]+)?))$' |
| Ipv6AddrRm | string | This data type is defined in the same way as the "Ipv6Addr" data type, but with the OpenAPI "nullable: true" property. |
| Ipv6Prefix | string | String identifying an IPv6 address prefix formatted according to clause 4 of IETF RFC 5952 [5]. Pattern: '^((:|(0?|([1-9a-f][0-9a-f]{0,3}))):)((0?|([1-9a-f][0-9a-f]{0,3})):){0,6}(:|(0?|([1-9a-f][0-9a-f]{0,3})))(\/(([0-9])|([0-9]{2})|(1[0-1][0-9])|(12[0-8])))$'andPattern: '^((([^:]+:){7}([^:]+))|((([^:]+:)\*[^:]+)?::(([^:]+:)\*[^:]+)?))(\/.+)$' |
| Ipv6PrefixRm | string | This data type is defined in the same way as the "Ipv6Prefix" data type, but with the OpenAPI "nullable: true" property. |
| MacAddr48 | string | String identifying a MAC address formatted in the hexadecimal notation according to clause 1.1 and clause 2.1 of IETF RFC 7042 [17].Pattern: '^([0-9a-fA-F]{2})((-[0-9a-fA-F]{2}){5})$' |
| MacAddr48Rm | string | This data type is defined in the same way as the "MacAddr48" data type, but with the OpenAPI "nullable: true" property. |
| SupportedFeatures | string | A string used to indicate the features supported by an API that is used as defined in clause 6.6 in 3GPP TS 29.500 [25].The string shall contain a bitmask indicating supported features in hexadecimal representation:Each character in the string shall take a value of "0" to "9" or "A" to "F" and shall represent the support of 4 features as described in table 5.2.2-3. The most significant character representing the highest-numbered features shall appear first in the string, and the character representing features 1 to 4 shall appear last in the string. The list of features and their numbering (starting with 1) are defined separately for each API. If the string contains a lower number of characters than there are defined features for an API, all features that would be represented by characters that are not present in the string are not supported. |
| Uinteger | integer | Unsigned Integer, i.e. only value 0 and integers above 0 are permissible.  |
| UintegerRm | integer | This data type is defined in the same way as the "Uinteger" data type, but with the OpenAPI "nullable: true" property. |
| Uint32 | integer | Unsigned 32-bit integers, i.e. only value 0 and 32-bit integers above 0 are permissible.  |
| Uint32Rm | integer | This data type is defined in the same way as the "UInt32" data type, but with the OpenAPI "nullable: true" property. |
| Uint64 | integer | Unsigned 64-bit integers, i.e. only value 0 and 64-bit integers above 0 are permissible.  |
| Uint64Rm | integer | This data type is defined in the same way as the "Uint64" data type, but with the OpenAPI "nullable: true" property. |
| Uri | string | String providing an URI formatted according to IETF RFC 3986 [6].  |
| UriRm | string | This data type is defined in the same way as the "Uri" data type, but with the OpenAPI "nullable: true" property. |
| VarUeId | string | String represents the SUPI or GPSI.Pattern: "^(imsi-[0-9]{5,15}|nai-.+|msisdn-[0-9]{5,15}|extid-[^@]+@[^@]+|.+)$". |
| VarUeIdRm | string | This data type is defined in the same way as the "VarUeId" data type, but with the OpenAPI "nullable: true" property. |
| TimeZone | string | String with format "<time-numoffset>" optionally appended by "<daylightSavingTime>", where:- <time-numoffset> shall represent the time zone adjusted for daylight saving time and be encoded as time-numoffset as defined in clause 5.6 of IETF RFC 3339 [10];- <daylightSavingTime> shall represent the adjustment that has been made and shall be encoded as "+1" or "+2" for a +1 or +2 hours adjustment.Example: "-08:00+1" (for 8 hours behind UTC, +1 hour adjustment for Daylight Saving Time). |
| TimeZoneRm | string | This data type is defined in the same way as the "TimeZone" data type, but with the OpenAPI "nullable: true" property. |
| StnSr | string | String representing the STN-SR as defined in clause 18.6 of 3GPP TS 23.003 [7]. |
| StnSrRm | string | This data type is defined in the same way as the "StnSr" data type, but with the OpenAPI "nullable: true" property. |
| CMsisdn | string | String representing the C-MSISDN as defined in clause 18.7 of 3GPP TS 23.003 [7]).Pattern: "^[0-9]{5,15}$". |
| CMsisdnRm | string | This data type is defined in the same way as the "CMsisdn" data type, but with the OpenAPI "nullable: true" property. |
| DayOfWeek | integer | Integer between and including 1 and 7 denoting a weekday. "1" shall indicate "Monday", and the subsequent weekdays shall be indicated with the next higher numbers. "7" shall indicate "Sunday". |
| TimeOfDay | string | String with format "partial-time" or "full-time" as defined in clause 5.6 of IETF RFC 3339 [10]. Examples: "20:15:00", "20:15:00-08:00" (for 8 hours behind UTC). |

Table 5.2.2-2: Reused OpenAPI data types

|  |  |
| --- | --- |
| Type Name | Description |
| boolean | As defined in OpenAPI Specification [3] |
| integer | As defined in OpenAPI Specification [3] |
| number | As defined in OpenAPI Specification [3] |
| string | As defined in OpenAPI Specification [3] |
| NOTE Data types defined in OpenAPI Specification [3] do not follow the UpperCamel convention for data types in 3GPP TS 29.501 [2] |

Table 5.2.2-3: Meaning of a Hexadecimal Character in SupportedFeatures Type

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Character | Feature n+3supported | Feature n+2supported | Feature n+1supported | Feature nsupported |
| "0" | no | no | no | no |
| "1" | no | no | no | yes |
| "2" | no | no | yes | no |
| "3" | no | no | yes | yes |
| "4" | no | yes | no | no |
| "5" | no | yes | no | yes |
| "6" | no | yes | yes | no |
| "7" | no | yes | yes | yes |
| "8" | yes | no | no | no |
| "9" | yes | no | no | yes |
| "A" | yes | no | yes | no |
| "B" | yes | no | yes | yes |
| "C" | yes | yes | no | no |
| "D" | yes | yes | no | yes |
| "E" | yes | yes | yes | no |
| "F" | yes | yes | yes | yes |
| NOTE 1 "n" shall be i \* 4 + 1, where "i" is zero or a natural number, i.e permissible values of "n" are 1, 5, 9, …NOTE 2 If a feature is not defined, it shall be indicated with value "no". |

For example, if only the first feature defined in the feature list is set to 1, the corresponding SupportedFeatures attribute would have a value of "1", or "001" (any amount of 0's to the left of the 1 would result into an equivalent feature list). If we have 32 features defined, and only the last feature in a feature list is set to 1, the corresponding SupportedFeatures attribute would have a value of "80000000".

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Next change\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## A.2 Data related to Common Data Types

openapi: 3.0.0

***(… text not shown for clarity …)***

#

# COMMON SIMPLE DATA TYPES

#

 Binary:

 format: binary

 type: string

 BinaryRm:

 format: binary

 type: string

 nullable: true

 Bytes:

 format: byte

 type: string

 BytesRm:

 format: byte

 type: string

 nullable: true

 Date:

 format: date

 type: string

 DateRm:

 format: date

 type: string

 nullable: true

 DateTime:

 format: date-time

 type: string

 DateTimeRm:

 format: date-time

 type: string

 nullable: true

 DiameterIdentity:

 type: string

 pattern: '^([A-Za-z0-9]+([-A-Za-z0-9]+)\.)+[a-z]{2,}$'

 DiameterIdentityRm:

 type: string

 pattern: '^([A-Za-z0-9]+([-A-Za-z0-9]+)\.)+[a-z]{2,}$'

 nullable: true

 Double:

 format: double

 type: number

 DoubleRm:

 format: double

 type: number

 nullable: true

 DurationSec:

 type: integer

 DurationSecRm:

 type: integer

 nullable: true

 Float:

 format: float

 type: number

 FloatRm:

 format: float

 type: number

 nullable: true

 Int32:

 format: int32

 type: integer

 Int32Rm:

 format: int32

 type: integer

 nullable: true

 Int64:

 type: integer

 format: int64

 Int64Rm:

 format: int64

 type: integer

 nullable: true

 Ipv4Addr:

 type: string

 pattern: '^(([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])$'

 example: '198.51.100.1'

 Ipv4AddrRm:

 type: string

 pattern: '^(([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])$'

 example: '198.51.100.1'

 nullable: true

 Ipv4AddrMask:

 type: string

 pattern: '^(([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])(\/([0-9]|[1-2][0-9]|3[0-2]))$'

 example: '198.51.0.0/16'

 Ipv4AddrMaskRm:

 type: string

 pattern: '^(([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])(\/.+)$'

 example: '198.51.0.0/16'

 nullable: true

 Ipv6Addr:

 type: string

 allOf:

 - pattern: '^((:|(0?|([1-9a-f][0-9a-f]{0,3}))):)((0?|([1-9a-f][0-9a-f]{0,3})):){0,6}(:|(0?|([1-9a-f][0-9a-f]{0,3})))$'

 - pattern: '^((([^:]+:){7}([^:]+))|((([^:]+:)\*[^:]+)?::(([^:]+:)\*[^:]+)?))$'

 example: '2001:db8:85a3::8a2e:370:7334'

 Ipv6AddrRm:

 type: string

 allOf:

 - pattern: '^((:|(0?|([1-9a-f][0-9a-f]{0,3}))):)((0?|([1-9a-f][0-9a-f]{0,3})):){0,6}(:|(0?|([1-9a-f][0-9a-f]{0,3})))$'

 - pattern: '^((([^:]+:){7}([^:]+))|((([^:]+:)\*[^:]+)?::(([^:]+:)\*[^:]+)?))$'

 example: '2001:db8:85a3::8a2e:370:7334'

 nullable: true

 Ipv6Prefix:

 type: string

 allOf:

 - pattern: '^((:|(0?|([1-9a-f][0-9a-f]{0,3}))):)((0?|([1-9a-f][0-9a-f]{0,3})):){0,6}(:|(0?|([1-9a-f][0-9a-f]{0,3})))(\/(([0-9])|([0-9]{2})|(1[0-1][0-9])|(12[0-8])))$'

 - pattern: '^((([^:]+:){7}([^:]+))|((([^:]+:)\*[^:]+)?::(([^:]+:)\*[^:]+)?))(\/.+)$'

 example: '2001:db8:abcd:12::0/64'

 Ipv6PrefixRm:

 type: string

 allOf:

 - pattern: '^((:|(0?|([1-9a-f][0-9a-f]{0,3}))):)((0?|([1-9a-f][0-9a-f]{0,3})):){0,6}(:|(0?|([1-9a-f][0-9a-f]{0,3})))(\/(([0-9])|([0-9]{2})|(1[0-1][0-9])|(12[0-8])))$'

 - pattern: '^((([^:]+:){7}([^:]+))|((([^:]+:)\*[^:]+)?::(([^:]+:)\*[^:]+)?))(\/.+)$'

 nullable: true

 MacAddr48:

 type: string

 pattern: '^([0-9a-fA-F]{2})((-[0-9a-fA-F]{2}){5})$'

 MacAddr48Rm:

 type: string

 pattern: '^([0-9a-fA-F]{2})((-[0-9a-fA-F]{2}){5})$'

 nullable: true

 SupportedFeatures:

 type: string

 pattern: '^[A-Fa-f0-9]\*$'

 Uinteger:

 type: integer

 minimum: 0

 UintegerRm:

 type: integer

 minimum: 0

 nullable: true

 Uint16:

 type: integer

 minimum: 0

 maximum: 65535

 Uint16Rm:

 type: integer

 minimum: 0

 maximum: 65535

 nullable: true

 Uint32:

 format: int32

 type: integer

 minimum: 0

 Uint32Rm:

 format: int32

 type: integer

 minimum: 0

 nullable: true

 Uint64:

 format: int64

 type: integer

 minimum: 0

 Uint64Rm:

 format: int64

 type: integer

 minimum: 0

 nullable: true

 Uri:

 type: string

 UriRm:

 type: string

 nullable: true

 VarUeId:

 type: string

 pattern: '^(imsi-[0-9]{5,15}|nai-.+|msisdn-[0-9]{5,15}|extid-[^@]+@[^@]+|.+)$'

 VarUeIdRm:

 type: string

 pattern: '^(imsi-[0-9]{5,15}|nai-.+|msisdn-[0-9]{5,15}|extid-[^@]+@[^@]+|.+)$'

 nullable: true

 TimeZone:

 type: string

 TimeZoneRm:

 type: string

 nullable: true

 StnSr:

 type: string

 StnSrRm:

 type: string

 nullable: true

 CMsisdn:

 type: string

 pattern: '^[0-9]{5,15}$'

 CMsisdnRm:

 type: string

 pattern: '^[0-9]{5,15}$'

 nullable: true

 DayOfWeek:

 type: integer

 minimum: 1

 maximum: 7

 description: integer between and including 1 and 7 denoting a weekday. 1 shall indicate Monday, and the subsequent weekdays shall be indicated with the next higher numbers. 7 shall indicate Sunday.

 TimeOfDay:

 type: string

 description: String with format partial-time or full-time as defined in clause 5.6 of IETF RFC 3339. Examples, 20:15:00, 20:15:00-08:00 (for 8 hours behind UTC).

***(… text not shown for clarity …)***

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*The end of changes\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*