5. CTRIP Attributes

This section defines the syntax and semantics of the CTRIP attributes transported in the UPDATE message.

5.1 WithdrawnRoutes

Conditional Mandatory: False.
Required Flags: Well-known.
Potential Flags: Link-State Encapsulation (when flooding).
CTRIP Type Code: 1

The WithdrawnRoutes specifies a set of routes that are to be removed from service by the receiving CTRIP-speaker(s). The set of routes MAY be empty, indicated by a length field of zero.

5.1.1 Syntax

The WithdrawnRoutes Attribute encodes a sequence of routes in its value field. The format for individual routes is given in Section 5.1.1.1. The WithdrawnRoutes Attribute lists the individual routes sequentially with no padding as shown in Figure 1. Each route includes a length field so that the individual routes within the attribute can be delineated.

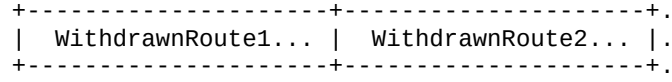

Figure 1. WithdrawnRoutes Format

5.1.1.1 Generic CTRIP Route Format

The generic format for a CTRIP route is given in Figure 2.

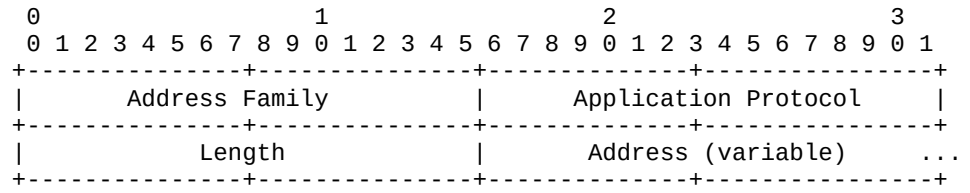

Figure 2. Generic CTRIP Route Format

Address Family:
The address family field gives the type of address for the route.
Three address families are defined in this Section:

Code	Address Family
1	Decimal Routing Numbers
2	PentaDecimal Routing Numbers
3	E.164 Numbers

This document reserves address family code 0. This document reserves address family codes 32768-65535 for vendor-specific applications (these are the codes with the first bit of the code

value equal to 1). Additional address families may be defined in the future. Assignment of address family codes is controlled by IANA. See Section 13 for IANA considerations.

Application Protocol:
The application protocol gives the protocol for which this routing table is maintained. The currently defined application protocols are:

Code	Protocol
1	SIP
2	H.323-H.225.0-Q.931
3	H.323-H.225.0-RAS
4	H.323-H.225.0-Annex-G
128	SCN general
129	TUP
130	ISUP
131	MAP
132	ISUP over IP

This document reserves application protocol code 0. This document reserves application protocol codes 32768-65535 for vendor-specific applications (these are the codes with the first bit of the code value equal to 1). Additional application protocols may be defined in the future. Assignment of application protocol codes is controlled by IANA. See Section 13 for IANA considerations.

Length:
This specifies the length of the address field, in bytes.

Address:
This is an address (prefix) of the family type given by Address Family. The octet length of the address is variable and is determined by the length field of the route.

5.1.1.2 Decimal Routing Numbers

The Decimal Routing Numbers address family is a super set of all E.164 numbers, national numbers, local numbers, and private numbers. It can also be used to represent the decimal routing numbers used in conjunction with Number Portability in some countries/regions. A set of telephone numbers is specified by a Decimal Routing Number prefix. Decimal Routing Number prefixes are represented by a string of digits, each digit encoded by its ASCII character representation. This routing object covers all phone numbers starting with this prefix. The syntax for the Decimal Routing Number prefix is:

```
decimal-routing-number  = *decimal-digit
decimal-digit           = DECIMAL-DIGIT
DECIMAL-DIGIT           = "0"|"1"|"2"|"3"|"4"|"5"|"6"|"7"|"8"|"9"
```

This DECIMAL Routing Number prefix is not bound in length. This format is similar to the format for a global telephone number as defined in SIP [REF] without visual separators and without the "+" prefix for international numbers. This format facilitates efficient comparison when using CTRIP to route SIP or H323, both of which use character based representations of phone numbers. The prefix length is determined from the length field of the route. The type of Decimal Routing Number (private, local, national, or international) can be deduced from the first few digits of the prefix.

5.1.1.3 Pentadecimal Routing Numbers

This address family is used to represent Pentadecimal Routing Numbers used in conjunction with Number Portability in some countries/regions. Pentadecimal Routing Number prefixes are represented by a string of digits, each digit encoded by its ASCII character representation. This routing object covers all routing numbers starting with this prefix. The syntax for the pentadecimal Routing Number prefix is:

```
pentadecimal-routing-number   = *pentadecimal-digit
pentadecimal-routing-digit    = PENTADECIMAL-DIGIT
PENTADECIMAL-DIGIT            = "0"|"1"|"2"|"3"|"4"|"5"|"6"|"7"|
                                "8"|"9"|"A"|"B"|"C"|"D"|"E"
```

Note the difference in alphabets between Decimal Routing Numbers and pentadecimal Routing Numbers. A pentadecimal Routing Number prefix is not bound in length.

Note that the address family, which suits the routing numbers of a specific country/region depends on the alphabets used for routing numbers in that country/region. For example, North American routing numbers SHOULD use the Decimal Routing Numbers address family, because their alphabet is limited to the digits "0" through "9". Another example, in most European countries routing numbers use the alphabet "0" through "9" and "A" through "E", and hence these countries SHOULD use the pentadecimal Routing Numbers address family.

5.1.1.4 E.164 Numbers

The E.164 Numbers address family is dedicated to fully qualified E.164 numbers. A set of telephone numbers is specified by an E.164 prefix. E.164 prefixes are represented by a string of digits, each digit encoded by its ASCII character representation. This routing object covers all phone numbers starting with this prefix. The syntax for the E.164 prefix is:

```
E164-number          = *e164-digit
E164-digit           = E164-DIGIT
E164-DIGIT           = "0"|"1"|"2"|"3"|"4"|"5"|"6"|"7"|"8"|"9"
```

This format facilitates efficient comparison when using CTRIP to route SIP or H323, both of which use character based representations of phone numbers. The prefix length is determined from the length field of the route.

The E.164 Numbers address family and the Decimal Routing Numbers address family have the same alphabet. The E.164 Numbers address family SHOULD be used whenever possible. The Decimal Routing Numbers address family can be used in case of private numbering plans or applications that do not desire to advertise fully expanded, fully qualified telephone numbers. If Decimal Routing Numbers are used to advertise non-fully qualified prefixes, the prefixes may have to be manipulated (e.g. expanded) at the boundary between TADs. This adds significant complexity to the TAD-Border CTRIP-speaker, because, it has to map the prefixes from the format used in its own TAD to the format used in the peer TAD.

5.2 ReachableRoutes

Conditional Mandatory: False.
Required Flags: Well-known.
Potential Flags: Link-State Encapsulation (when flooding).
CTRIP Type Code: 2

The ReachableRoutes attribute specifies a set of routes that are to be added to service by the receiving CTRIP-speaker(s). The set of routes MAY be empty, as indicated by setting the length field to zero.

5.2.1 Syntax of ReachableRoutes

The ReachableRoutes Attribute has the same syntax as the WithdrawnRoutes Attribute. See Section 5.1.1.

5.2.2 Route Origination and ReachableRoutes

Routes are injected into CTRIP by a method outside the scope of this specification. Possible methods include a front-end protocol, an intra-domain routing protocol, or static configuration. Routes describing destinations on an IP network can be obtained from TRIP or ENUM through a conversion method.

5.2.3 Route Selection and ReachableRoutes

The routes in ReachableRoutes are necessary for route selection.

5.2.4 Aggregation and ReachableRoutes

To aggregate multiple routes, the set of ReachableRoutes to be aggregated MUST combine to form a less specific set.

There is no mechanism within CTRIP to communicate that a particular address prefix is not used and thus that these addresses could be skipped during aggregation. CTRIP-speakers MAY use methods outside of CTRIP to learn of invalid prefixes that may be ignored during aggregation.

If a CTRIP-speaker advertises an aggregated route, it MUST include the AtomicAggregate attribute.

5.2.5 Route Dissemination and ReachableRoutes

The ReachableRoutes attribute is recomputed at each CTRIP-speaker except where flooding is being used (e.g., within a domain). It is therefore possible for a CTRIP-speaker to change the Application Protocol field of a route before advertising that route to an external peer.

If a CTRIP-speaker changes the Application Protocol of a route it advertises, it MUST include the ConvertedRoute attribute in the UPDATE message.

5.2.6 Aggregation Specifics for Decimal Routing Numbers, E.164 Numbers, and PentaDecimal Routing Numbers

An CTRIP-speaker that has routes to all valid numbers in a specific prefix SHOULD advertise that prefix as the ReachableRoutes, even if there are more specific prefixes that do not actually exist on the PSTN. Generally, it takes 10 Decimal Routing/E.164 prefixes, or 15 pentadecimal Routing prefixes, of length n to aggregate into a prefix of length n-1. However, if a CTRIP-speaker is aware that a prefix is an invalid Decimal Routing/E.164 prefix, or pentadecimal Routing prefix, then the CTRIP-speaker MAY aggregate by skipping this prefix. For example, if the Decimal Routing prefix 19191 is known not to exist, then a CTRIP-speaker can aggregate to 1919 without 19191. A prefix representing an invalid set of PSTN destinations is sometimes referred to as a "black-hole." The method by which a CTRIP-speaker is aware of black-holes is not within the scope of CTRIP, but if a CTRIP-speaker has such knowledge, it can use the knowledge when aggregating.

5.3 NextHopAddress

Conditional Mandatory: True (if ReachableRoutes and/or WithdrawnRoutes attribute is present).
Required Flags: Well-known.
Potential Flags: None.
CTRIP Type Code: 129.

Given a route with application protocol A and destinations D, the NextHopAddress is used for obtaining the routing address for call setup with protocol A to destination D. This may or may not represent the ultimate destination of the call.

5.3.1 NextHopAddress Syntax

For generality, the next-hop address may be of various types, as identified in the Next Hop Type field. The currently defined types are:

Code	Type
1	**NH_ROUTING_NUMBER**: The routing number of the next hop is given in this attribute.
2	**NH_SCN_QUERY**: The routing number of the next hop is obtained by querying a server on the SCN.
3	**NH_IP_QUERY**: The routing number of the next hop is obtained by querying a server on the IP network.

The syntax for the NextHopAddress depends on the Next Hop Type.

5.3.1.1 Next Hop Type 1

The format of the NextHopAddress attribute for Next Hop Type 1 is given in Figure 3. The fields include the TAD number of the next hop, the Next Hop Type field, the length of the routing number pattern and the Routing Number Pattern. The Next Hop TAD indicates the domain of the next-hop. The Next Hop Type has the value 1. The Pattern Length is the length of the Routing Number Pattern in octets. The Routing Number Pattern is a substitution expression used to generate the routing number for the next hop. The format of the Routing Number Pattern is described in Section 5.3.2.

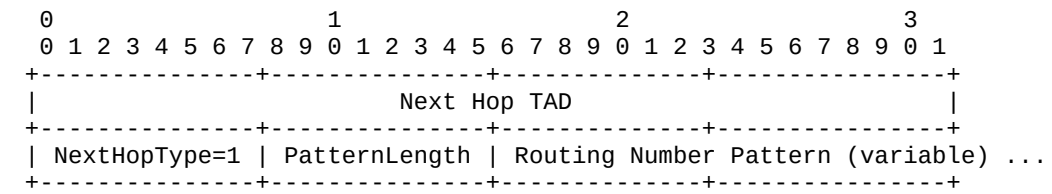

Figure 3: NextHopAddress Syntax for Next Hop Type 1

5.3.1.2 Next Hop Type 2

The format of the NextHopAddress attribute for Next Hop Type 2 is given in Figure 4. The first fields are similar to the ones of Next Hop Type 1 (Section 5.3.1.1). However, in this case, the Routing Number Pattern is a substitution expression for generating the number given as a parameter in the query. The Next Hop Type field has the value 2. In addition to the above fields, this Next Hop Type contains the protocol used for queries, the version identifier of the query protocol, the length of the Destination Point Code (DPC), the DPC, the Sub-System Number (SSN), the length of the parameter field and the parameter field.

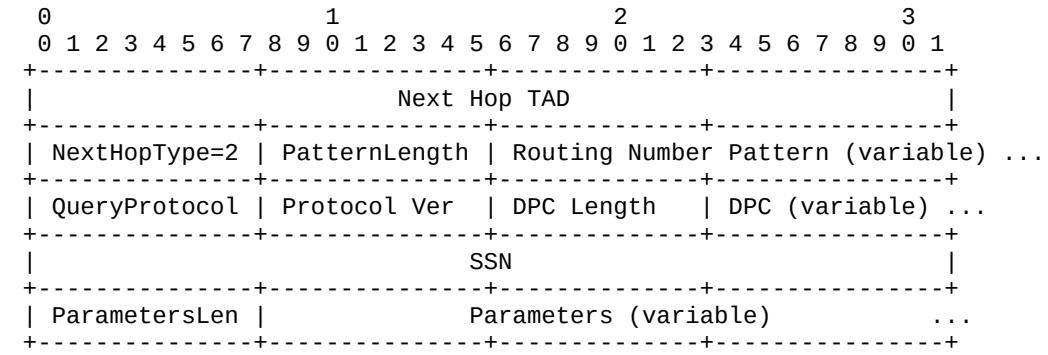

Figure 4: NextHopAddress Syntax for Next Hop Type 2

The length of the DPC is given as the number of bits. The DPC field is padded with zero bits BEFORE the value to make its length (in bits) dividable with 8. The DPC and SSN fields define the address to query.

The Query Protocol and the Query Protocol Version define the protocol used in the query. The Query Protocol and Query Protocol Version take the following values:

Query Protocol	Protocol Version	Protocol
1	**0**	**LDAP** (any version)
2	**0**	**INAP** (any version)
3	**0**	**MAP** (any version)

The Parameters field contains additional parameters for the query coded in a format depending on the values of the Query Protocol and the Query Protocol Version. The length of the parameters field is given in octets.

5.3.1.3 Next Hop Type 3

The format of the NextHopAddress attribute for Next Hop Type 3 is given in Figure 5. The first fields are similar to the ones of Next Hop Type 1 (Section 5.3.1.1). In addition to these fields, this Next Hop Type contains the protocol used for queries, the version identifier of the query protocol, the length of the server address, the server address, the length of the parameter field and the parameter field. The length fields are given as number of octets.

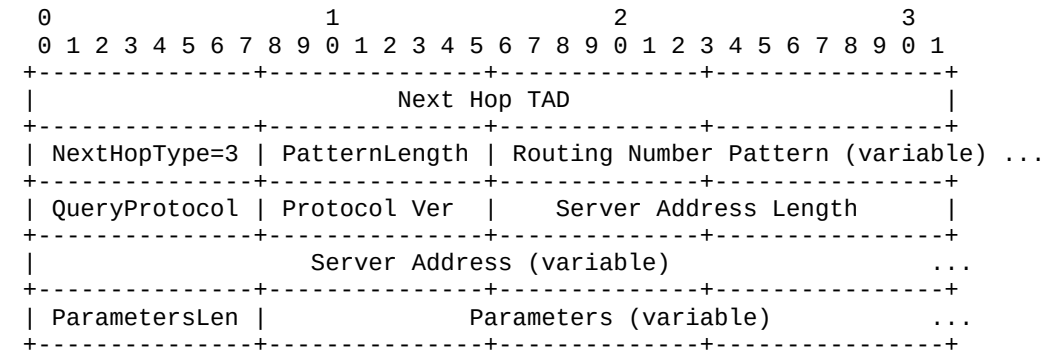

Figure 5: NextHopAddress Syntax for Next Hop Type 3

The Query Protocol and Query Protocol Version take the values defined in Section 5.3.1.2. The Parameters field contains additional parameters for the query coded in a format depending on the values of the Query Protocol and the Query Protocol Version. The length of the parameters field is given in octets.

The Server Address field contains the name or address of the next-hop server. The server address field is represented as a string of ASCII characters. It is defined as follows:

```
Server Address = host [":" port ]
host    = <   A legal Internet host domain name
              or an IPv4 address using the textual representation
              defined in Section 2.1 of RFC 1123 [REF]
              or an IPv6 address using the textual representation
              defined in Section 2.2 of RFC 2373 [REF]. The IPv6
              address MUST be enclosed in "[" and "]"
              characters.  >
port    = *DIGIT
```

If the port is empty or not given, the default port is assumed (e.g., port 5060 if the application protocol is SIP).

5.3.2 The Routing Number Pattern field

The content of the routing number pattern field is a substitution expression, which has a similar format as the NAPTR resource record of DNS [REF]. However, the flags are not used. The contents of the field MUST follow the grammar below:

```
subst_expr   = delim-char  ere  delim-char  repl  delim-char
delim-char   = "/" / "!" / ... <Any non-digit or non-flag character
               other than backslash '\'. All occurances of a delim_char
               in a subst_expr must be the same character.>
ere          = POSIX Extended Regular Expression
repl         = 1 * ( OCTET /  backref )
backref      = "\" 1POS_DIGIT
POS_DIGIT    = %x31-39                ; 0 is not an allowed backref
```

The definition of a POSIX Extended Regular Expression can be found in [REF], section 2.8.4.

The result of applying the substitution expression to the original dialed digits MUST result in a string that describes a routing number. The number is used to route the call to the next hop.

Backref expressions in the repl portion of the substitution expression are replaced by the (possibly empty) string of characters enclosed by '(' and ')' in the ERE portion of the substitution expression. N is a single digit from 1 through 9, inclusive. It specifies the N'th backref expression, the one that begins with the N'th '(' and continues to the matching ')'. For example, the ERE

```
(A(B(C)DE)(F)G)
```

has backref expressions:

```
\1  = ABCDEFG
\2  = BCDE
\3  = C
\4  = F
\5..\9  = error - no matching subexpression
```

The first character in the substitution expression shall be used as the character that delimits the components of the substitution expression. There must be exactly three non-escaped occurrences of the delimiter character in a substitution expression. Since escaped occurrences of the delimiter character will be interpreted as occurrences of that character, digits MUST NOT be used as delimiters. Backrefs would be confused with literal digits were this allowed.

5.3.3 Route Origination and NextHopAddress

When a CTRIP-speaker originates a routing object into CTRIP, it MUST include a NextHopAddress within its domain. All peer domains MUST be able to reach destination with the routing number obtained with the NextHopAddress.

5.3.4 Route Selection and NextHopAddress

CTRIP-speaker policy may prefer certain next-hops or next-hop domains over others.

5.3.5 Aggregation and NextHopAddress

When aggregating multiple routing objects with different NextHopAddress into a single routing object, a CTRIP-speaker MUST insert a new NextHopAddress, which can be used to reach all the included destinations.

5.3.6 Route Dissemination and NextHopAddress

When propagating routing objects to peers, a CTRIP-speaker may choose to insert a new NextHopAddress, or it may leave the NextHopAddress unchanged. It is a local policy decision of the CTRIP-speaker to decide whether to propagate or change the NextHopAddress. The CTRIP-speaker MUST ensure that the peer domains are able to reach destination with the routing number obtained with the NextHopAddress.

5.4 AdvertisementPath

Conditional Mandatory: True (if ReachableRoutes and/or WithdrawnRoutes attribute is present).
Required Flags: Well-known.
Potential Flags: None.
CTRIP Type Code: 4.

This attribute identifies the TADs through which routing information carried in an advertisement has passed. The AdvertisementPath attribute is analogous to the AS_PATH attribute in BGP. The attributes differ in that BGP's AS_PATH also reflects the path to the destination. In CTRIP, not every domain need modify the next-hop, so the AdvertisementPath may include many more hops than the actual path to the destination. The ExtendedRoutedPath attribute (Section 5.5) reflects the actual signaling path to the destination.

5.4.1 AdvertisementPath Syntax

AdvertisementPath is a variable length attribute that is composed of a sequence of TAD path segments. Each TAD path segment is represented by a type-length-value triple.

The path segment type is a 1-octet long field with the following values defined:

Value	Segment Type
1	**AP_SET**: unordered set of TADs a route in the advertisement message has traversed
2	**AP_SEQUENCE**: ordered set of TADs a route in the advertisement message has traversed

The path segment length is a 1-octet long field containing the number of TADs in the path segment value field.

The path segment value field contains one or more TAD numbers, each encoded as a 4-octets long field.

5.4.2 Route Origination and AdvertisementPath

When a CTRIP-speaker originates a route then:

- The originating CTRIP-speaker shall include its own TAD number in the AdvertisementPath attribute of all advertisements sent to CTRIP-speakers located in neighboring TADs. In this case, the TAD number of the originating CTRIP-speaker's TAD will be the only entry in the AdvertisementPath attribute.
- The originating CTRIP-speaker shall include an empty AdvertisementPath attribute in all advertisements sent to CTRIP-speakers located in its own TAD. An empty AdvertisementPath attribute is one whose length field contains the value zero.

5.4.3 Route Selection and AdvertisementPath

The AdvertisementPath may be used for route selection. Possible criteria to be used are the number of hops on the path and the presence or absence of particular TADs on the path.

The AdvertisementPath is used to prevent routing information from looping. If a CTRIP-speaker receives a route with its own TAD already in the AdvertisementPath, the route MUST be discarded.

5.4.4 Aggregation and AdvertisementPath

The rules for aggregating AdvertisementPath attributes are given in the following sections, where the term "path" used in Section 5.4.4.1 and 5.4.4.2 is understood to mean AdvertisementPath.

5.4.4.1 Aggregating Routes with Identical Paths

If all routes to be aggregated have identical path attributes, then the aggregated route has the same path attribute as the individual routes.

5.4.4.2 Aggregating Routes with Different Paths

For the purpose of aggregating path attributes we model each TAD within the path as a pair <type, value>, where "type" identifies a type of the path segment (AP_SEQUENCE or AP_SET), and "value" is the TAD number. Two TADs are said to be the same if their corresponding <type, value> are the same.

If the routes to be aggregated have different path attributes, then the aggregated path attribute shall satisfy all of the following conditions:

- All pairs of the type AP_SEQUENCE in the aggregated path MUST appear in all of the paths of routes to be aggregated.
- All pairs of the type AP_SET in the aggregated path MUST appear in at least one of the paths of the initial set (they may appear as either AP_SET or AP_SEQUENCE types).
- For any pair X of the type AP_SEQUENCE that precedes pair Y in the aggregated path, X precedes Y in each path of the initial set that contains Y, regardless of the type of Y.
- No pair with the same value shall appear more than once in the aggregated path, regardless of the pair's type.

An implementation may choose any algorithm that conforms to these rules. At a minimum, a conformant implementation MUST be able to perform the following algorithm that meets all of the above conditions:

- Determine the longest leading sequence of tuples (as defined above) common to all the paths of the routes to be aggregated. Make this sequence the leading sequence of the aggregated path.
- Set the type of the rest of the tuples from the paths of the routes to be aggregated to AP_SET, and append them to the aggregated path.
- If the aggregated path has more than one tuple with the same value (regardless of tuple's type), eliminate all but one such tuple by deleting tuples of the type AP_SET from the aggregated path.

An implementation that chooses to provide a path aggregation algorithm that retains significant amounts of path information may wish to use the procedure of Section 5.4.4.3.

5.4.4.3 Example Path Aggregation Algorithm

An example algorithm to aggregate two paths works as follows:

- Identify the TADs (as defined in Section 5.4.1) within each path attribute that are in the same relative order within both path attributes. Two TADs, X and Y, are said to be in the same order if either X precedes Y in both paths, or if Y precedes X in both paths.
- The aggregated path consists of TADs identified in (a) in exactly the same order as they appear in the paths to be aggregated. If two consecutive TADs identified in (a) do not immediately follow each other in both of the paths to be aggregated, then the intervening TADs (TADs that are between the two consecutive TADs that are the same) in both attributes are combined into an AP_SET path segment that consists of the intervening TADs from both paths; this segment is then placed in between the two consecutive TADs identified in (a) of the aggregated attribute. If two consecutive TADs identified in (a) immediately follow each other in one attribute, but do not follow in another, then the intervening TADs of the latter are combined into an AP_SET path segment; this segment is then placed in between the two consecutive TADs identified in (a) of the aggregated path.

If as a result of the above procedure, a given TAD number appears more than once within the aggregated path, all but the last instance (rightmost occurrence) of that TAD number should be removed from the aggregated path.

5.4.5 Route Dissemination and AdvertisementPath

When a CTRIP-speaker propagates a route which it has learned from another CTRIP-speaker, it shall modify the route's AdvertisementPath attribute based on the location of the CTRIP-speaker to which the route will be sent.

- When a CTRIP-speaker advertises a route to another CTRIP-speaker located in its own TAD, the advertising CTRIP-speaker MUST NOT modify the AdvertisementPath attribute associated with the route.
- When a CTRIP-speaker advertises a route to a CTRIP-speaker located in a neighboring TAD, then the advertising CTRIP-speaker MUST update the AdvertisementPath attribute as follows:

If the first path segment of the AdvertisementPath is of type AP_SEQUENCE, the local system shall prepend its own TAD number as the last element of the sequence (put it in the leftmost position).

If the first path segment of the AdvertisementPath is of type AP_SET, the local system shall prepend a new path segment of type AP_SEQUENCE to the AdvertisementPath, including its own TAD number in that segment.

5.5 ExtendedRoutedPath

Conditional Mandatory: True (if ReachableRoutes attribute is present).
Required Flags: Well-known.
Potential Flags: None.
CTRIP Type Code: 128.

This attribute identifies the TADs through which messages sent using this route would pass. The TADs in this path are a subset of those in the AdvertisementPath. This attribute also identifies the application protocol used in each TAD in the path.

5.5.1 ExtendedRoutedPath Syntax

ExtendedRoutedPath is a variable length attribute that is composed of a sequence of path segments. Each path segment is represented by a type-length-value triple.

The path segment type is a 1-octet long field with the following values defined:

Value	Segment Type
1	**AP_SET**: unordered set of TADs a route in the advertisement message has traversed
2	**AP_SEQUENCE**: ordered set of TADs a route in the advertisement message has traversed

The path segment length is a 1-octet long field containing the number of hop-entries in the path segment value field.

The path segment value field contains one or more hop-entries. Each hop-entry consists of a combination of a TAD number and an Application Protocol. The TAD number is encoded in a 4-octets long field. The Application Protocol is encoded in a 2-octets long field. The Application Protocol is identified with the same values as the Application Protocol field defined in Section 5.1.1.1.

5.5.2 Route Origination and ExtendedRoutedPath

When a CTRIP-speaker originates a route, it MUST include the ExtendedRoutedPath attribute.

- The originating CTRIP-speaker shall include a hop-entry containing its own TAD number and the used Application Protocol in the ExtendedRoutedPath attribute of all advertisements sent to CTRIP-speakers located in neighboring TADs. In this case, the hop-entry containing the TAD number and Application Protocol of the originating CTRIP-speaker's TAD will be the only entry in the ExtendedRoutedPath attribute.
- The originating CTRIP-speaker shall include an empty ExtendedRoutedPath attribute in all advertisements sent to CTRIP-speakers located in its own TAD. An empty ExtendedRoutedPath attribute is one whose length field contains the value zero.

5.5.3 Route Selection and ExtendedRoutedPath

The ExtendedRoutedPath MAY be used for route selection, and in most cases is preferred over the AdvertisementPath for this role. Some possible criteria to be used are the number of hops on the

path and the presence or absence of particular TADs on the path. The application protocol field MAY be used for route selection. Additional criteria include the number of application protocols on the path and the presence or absence of particular application protocols on the path.

5.5.4 Aggregation and ExtendedRoutedPath

The rules for aggregating ExtendedRoutedPath attributes are given in the following sections, where the term "path" used in Section 5.5.4.1 and 5.5.4.2 is understood to mean ExtendedRoutedPath.

5.5.4.1 Aggregating Routes with Identical Paths

If all routes to be aggregated have identical path attributes, then the aggregated route has the same path attribute as the individual routes.

5.5.4.2 Aggregating Routes with Different Paths

For the purpose of aggregating path attributes we model each hop-entry within the path as a pair <type, value>, where "type" identifies a type of the path segment (AP_SEQUENCE or AP_SET), and "value" is the combination of TAD number and Application Protocol. Two TADs are said to be the same if their corresponding <type, value> are the same.

If the routes to be aggregated have different path attributes, then the aggregated path attribute shall satisfy all of the following conditions:

- All pairs of the type AP_SEQUENCE in the aggregated path MUST appear in all of the paths of routes to be aggregated.
- All pairs of the type AP_SET in the aggregated path MUST appear in at least one of the paths of the initial set (they may appear as either AP_SET or AP_SEQUENCE types).
- For any pair X of the type AP_SEQUENCE that precedes pair Y in the aggregated path, X precedes Y in each path of the initial set that contains Y, regardless of the type of Y.
- No pair with the same value shall appear more than once in the aggregated path, regardless of the pair's type.

An implementation may choose any algorithm that conforms to these rules. At a minimum, a conformant implementation MUST be able to perform the following algorithm that meets all of the above conditions:

- Determine the longest leading sequence of tuples (as defined above) common to all the paths of the routes to be aggregated. Make this sequence the leading sequence of the aggregated path.
- Set the type of the rest of the tuples from the paths of the routes to be aggregated to AP_SET, and append them to the aggregated path.
- If the aggregated path has more than one tuple with the same value (regardless of tuple's type), eliminate all but one such tuple by deleting tuples of the type AP_SET from the aggregated path.

An implementation that chooses to provide a path aggregation algorithm that retains significant amounts of path information may wish to use the procedure of Section 5.5.4.3.

5.5.4.3 Example Path Aggregation Algorithm

An example algorithm to aggregate two paths works as follows:

- Identify the hop-entries (as defined in 5.5.1) within each path attribute that are in the same relative order within both path attributes. Two hop-entries, X and Y, are said to be in the same order if either X precedes Y in both paths, or if Y precedes X in both paths.
- The aggregated path consists of hop-entries identified in (a) in exactly the same order as they appear in the paths to be aggregated. If two consecutive hop-entries identified in (a) do not

immediately follow each other in both of the paths to be aggregated, then the intervening hop-entries (hop-entries that are between the two consecutive hop-entries that are the same) in both attributes are combined into an AP_SET path segment that consists of the intervening hop-entries from both paths; this segment is then placed in between the two consecutive hop-entries identified in (a) of the aggregated attribute. If two consecutive hop-entries identified in (a) immediately follow each other in one attribute, but do not follow in another, then the intervening hop-entries of the latter are combined into an AP_SET path segment; this segment is then placed in between the two consecutive hop-entries identified in (a) of the aggregated path.

If as a result of the above procedure, a given TAD number appears more than once within the aggregated path, all but the last instance (rightmost occurrence) of that TAD number should be removed from the aggregated path.

5.5.5 Route Dissemination and ExtendedRoutedPath

When a CTRIP-speaker propagates a route that it learned from another CTRIP-speaker, it modifies the route's ExtendedRoutedPath attribute based on the location of the CTRIP-speaker to which the route is sent.

- When a CTRIP-speaker advertises a route to another CTRIP-speaker located in its own TAD, the advertising CTRIP-speaker MUST NOT modify the ExtendedRoutedPath attribute associated with the route.
- If the CTRIP-speaker has not changed the NextHopAddress attribute, then the CTRIP-speaker MUST NOT change the ExtendedRoutedPath attribute.
- Otherwise, the CTRIP-speaker changed the NextHopAddress and is advertising the route to a CTRIP-speaker in another TAD. The advertising CTRIP-speaker MUST update the ExtendedRoutedPath attribute as follows:

    - If the first path segment of the ExtendedRoutedPath is of type AP_SEQUENCE, the local system shall prepend a hop-entry containing its own TAD number and used Application Protocol as the last element of the sequence (put it in the leftmost position).
    - If the first path segment of the ExtendedRoutedPath is of type AP_SET, the local system shall prepend a new path segment of type AP_SEQUENCE to the ExtendedRoutedPath, including a hop-entry containing its own TAD number and used Application Protocol in that segment.

5.6 AtomicAggregate

Conditional Mandatory: False.
Required Flags: Well-known.
Potential Flags: None.
CTRIP Type Code: 6.

The AtomicAggregate attribute indicates that a route may traverse domains not listed in the ExtendedRoutedPath. If a CTRIP-speaker, when presented with a set of overlapping routes from a peer CTRIP-speaker, selects the less specific route without selecting the more specific route, then the CTRIP-speaker includes the AtomicAggregate attribute with the routing object.

5.6.1 AtomicAggregate Syntax

This attribute has length zero (0); the value field is empty.

5.6.2 Route Origination and AtomicAggregate

Routes are never originated with the AtomicAggregate attribute.

5.6.3 Route Selection and AtomicAggregate

The AtomicAggregate attribute may be used in route selection – it indicates that the ExtendedRoutedPath may be incomplete.

5.6.4 Aggregation and AtomicAggregate

If any of the routes to aggregate has the AtomicAggregate attribute, then so MUST the resultant aggregate.

5.6.5 Route Dissemination and AtomicAggregate

If a CTRIP-speaker, when presented with a set of overlapping routes from a peer CTRIP-speaker, selects the less specific route (see Section 0) without selecting the more specific route, then the CTRIP-speaker MUST include the AtomicAggregate attribute with the routing object (if it is not already present).

An CTRIP-speaker receiving a routing object with an AtomicAggregate attribute MUST NOT make the set of destinations more specific when advertising it to other CTRIP-speakers, and MUST NOT remove the attribute when propagating this object to a peer CTRIP-speaker.

5.7 LocalPreference

Conditional Mandatory: False.
Required Flags: Well-known.
Potential Flags: None.
CTRIP Type Code: 7.

The LocalPreference attribute is only used intra-domain, it indicates the local CTRIP-speaker's preference for the routing object to other CTRIP-speakers within the same domain. This attribute MUST NOT be included when communicating to a CTRIP-speaker in another domain, and MUST be included over intra-domain links.

5.7.1 LocalPreference Syntax

The LocalPreference attribute is a 4-octet unsigned numeric value. A higher value indicates a higher preference.

5.7.2 Route Origination and LocalPreference

Routes MUST NOT be originated with the LocalPreference attribute to inter-domain peers. Routes to intra-domain peers MUST be originated with the LocalPreference attribute.

5.7.3 Route Selection and LocalPreference

The LocalPreference attribute allows one CTRIP-speaker in a domain to calculate a preference for a route, and to communicate this preference to other CTRIP-speakers within the domain.

5.7.4 Aggregation and LocalPreference

The LocalPreference attribute is not affected by aggregation.

5.7.5 Route Dissemination and LocalPreference

A CTRIP-speaker MUST include the LocalPreference attribute when communicating with peer CTRIP-speakers within its own domain. A CTRIP-speaker MUST NOT include the LocalPreference attribute when communicating with CTRIP-speakers in other domains. LocalPreference attributes received from inter-domain peers MUST be ignored.

5.8 MultiExitDisc

Conditional Mandatory: False.
Required Flags: Well-known.
Potential Flags: None.
CTRIP Type Code: 8.

When two TADs are connected by more than one set of peers, the MultiExitDisc attribute may be used to specify preferences for routes received over one of those links versus routes received over other links. The MultiExitDisc parameter is used only for route selection.

5.8.1 MultiExitDisc Syntax

The MultiExitDisc attribute carries a 4-octet unsigned numeric value. A higher value represents a more preferred routing object.

5.8.2 Route Origination and MultiExitDisc

Routes originated to intra-domain peers MUST NOT be originated with the MultiExitDisc attribute. When originating a route to an inter-domain peer, the MultiExitDisc attribute may be included.

5.8.3 Route Selection and MultiExitDisc

The MultiExitDisc attribute is used to express a preference when there are multiple links between two domains. If all other factors are equal, then a route with a higher MultiExitDisc attribute is preferred over a route with a lower MultiExitDisc attribute.

5.8.4 Aggregation and MultiExitDisc

Routes with differing MultiExitDisc parameters MUST NOT be aggregated. Routes with the same value in the MultiExitDisc attribute MAY be aggregated and the same MultiExitDisc attribute attached to the aggregated object.

5.8.5 Route Dissemination and MultiExitDisc

If received from a peer CTRIP-speaker in another domain, a CTRIP-speaker MAY propagate the MultiExitDisc to other CTRIP-speakers within its domain. The MultiExitDisc attribute MUST NOT be propagated to CTRIP-speakers in other domains.

An CTRIP-speaker may add the MultiExitDisc attribute when propagating routing objects to a CTRIP-speaker in another domain. The inclusion of the MultiExitDisc attribute is a matter of policy, as is the value of the attribute.

5.9 Communities

Conditional Mandatory: False.
Required Flags: Not Well-Known, Independent Transitive.
Potential Flags: None.
CTRIP Type Code: 9.

A community is a group of destinations that share some common property.

The Communities attribute is used to group destinations so that the routing decision can be based on the identity of the group. Using the Communities attribute should significantly simplify the distribution of routing information by providing an administratively defined aggregation unit.

Each TAD administrator may define the communities to which a particular route belongs. By default, all routes belong to the general Internet Telephony community.

As an example, the Communities attribute could be used to define an alliance between a group of Internet Telephony service providers for a specific subset of routing information. In this case, members of that alliance would accept only routes for destinations in this group that are advertised by other members of the alliance. Other destinations would be more freely accepted. To achieve this, a member would tag each route with a designated Community attribute value before disseminating it. This relieves the members of such an alliance, from the responsibility of keeping track of the identities of all other members of that alliance.

Another example use of the Communities attribute is with aggregation. It is often useful to advertise both the aggregate route and the component more-specific routes that were used to form the aggregate. These information components are only useful to the neighboring CTRIP peer, and perhaps the TAD of the neighboring CTRIP peer, so it is desirable to filter out the component routes. This can be achieved by specifying a Community attribute value that the neighboring peers will match and filter on. That way it can be assured that the more specific routes will not propagate beyond their desired scope.

5.9.1 Syntax of Communities

The Communities attribute is of variable length. It consists of a set of 8-octet values, each of which specifies a community. The first 4 octets of the Community value are the Community TAD Number and the next 4 octets are the Community ID.

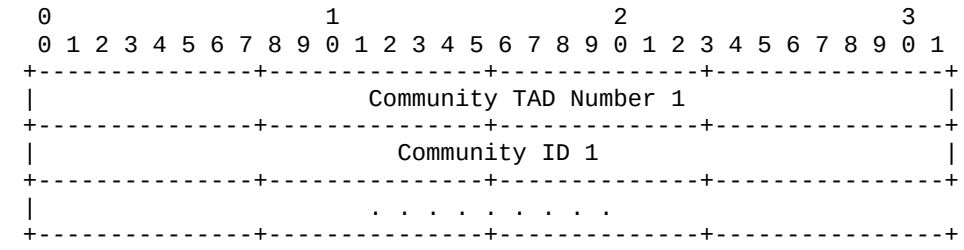

Figure 6: Communities Syntax

For administrative assignment, the following assumptions may be made:

- The Community attribute values starting with a Community TAD Number of 0x00000000 are hereby reserved.

The following communities have global significance and their operation MUST be implemented in any Community attribute-aware CTRIP CTRIP-speaker.

- NO_EXPORT (Community TAD Number = 0x00000000 and Community ID= 0xFFFFFF01). Any received route with a community attribute containing this value MUST NOT be advertised outside of the receiving CTRIP TAD.

Other community values MUST be encoded using an TAD number in the four most significant octets. The semantics of the final four octets (the Community ID octets) may be defined by the TAD (e.g., TAD 690 may define research, educational, and commercial community IDs that may be used for policy routing as defined by the operators of that TAD).

5.9.2 Route Origination and Communities

The Communities attribute is not well-known. If a route has a Communities attribute associated with it, the CTRIP-speaker MUST include that attribute in the advertisement it originates.

5.9.3 Route Selection and Communities

The Communities attribute may be used for route selection. A route that is a member of a certain community may be preferred over another route that is not a member of that community. Likewise, routes without a certain community value may be excluded from consideration.

5.9.4 Aggregation and Communities

If a set of routes is to be aggregated and the resultant aggregate does not carry an AtomicAggregate attribute, then the resulting aggregate should have a Communities attribute that contains the union of the Community attributes of the aggregated routes.

5.9.5 Route Dissemination and Communities

A CTRIP-speaker may manipulate the Communities attribute before disseminating a route to a peer. Community attribute manipulation may include adding communities, removing communities, adding a Communities attribute (if none exists), deleting the Communities attribute, etc.

5.10 DomainTopology

Conditional Mandatory: False.
Required Flags: Well-known, Link-State encapsulated.
Potential Flags: None.
CTRIP Type Code: 10.

Within a TAD, each CTRIP-speaker must know the status of other CTRIP-speakers so that CTRIP-speaker failure can be detected. To do this, each CTRIP-speaker advertises its internal topology to other CTRIP-speakers within the domain. When a CTRIP-speaker detects that another CTRIP-speaker is no longer active, the information sourced by that CTRIP-speaker can be deleted (the Adj-TRIB-In for that peer may be cleared). The DomainTopology attribute is used to communicate this information to other CTRIP-speakers within the domain.

A CTRIP-speaker MUST send a topology update each time it detects a change in its internal peer set. The topology update may be sent in an UPDATE message by itself or it may be piggybacked on an UPDATE message which includes ReachableRoutes and/or WithdrawnRoutes information.

When a CTRIP-speaker receives a topology update from an internal CTRIP-speaker, it MUST recalculate which CTRIP-speakers are active within the TAD via a connectivity algorithm on the topology.

5.10.1 DomainTopology Syntax

The DomainTopology attribute indicates the CTRIP-speakers with which the CTRIP-speaker is currently peering. The attribute consists of a list of the CTRIP Identifiers with which the CTRIP-

speaker is currently peering, the format is given in Figure 7. This attribute MUST use the link-state encapsulation as defined in Section 4.3.2.4.

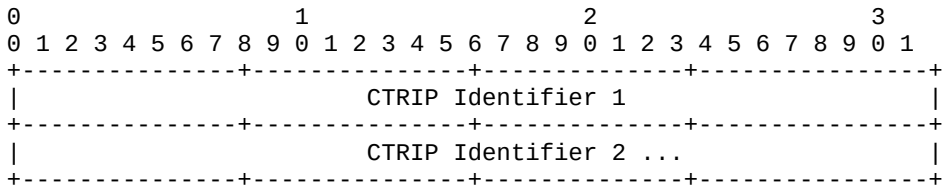

Figure 7: DomainTopology Syntax

5.10.2 Route Origination and DomainTopology

The DomainTopology attribute is independent of any routes in the UPDATE. Whenever the set of internal peers of a CTRIP-speaker changes, it MUST create an UPDATE with the DomainTopology Attribute included listing the current set of internal peers. The CTRIP-speaker MUST include this attribute in the first UPDATE it sends to a peer after the peering session is established.

5.10.3 Route Selection and DomainTopology

This attribute is independent of any routing information in the UPDATE. When a CTRIP-speaker receives an UPDATE with an DomainTopology attribute, it MUST compute the set of CTRIP-speakers currently active in the domain by performing a connectivity test on the DomainTopology as given by the set of originated DomainTopology attributes. The CTRIP-speaker MUST locally purge the Adj-TRIB-In for any CTRIP-speaker that is no longer active in the domain. The CTRIP-speaker MUST NOT propagate this purging information to other CTRIP-speakers as they will make a similar decision.

5.10.4 Aggregation and DomainTopology

This information is not aggregated.

5.10.5 Route Dissemination and DomainTopology

A CTRIP-speaker MUST ignore the attribute if received from a peer in another domain. An CTRIP-speaker MUST NOT send this attribute to an inter-domain peer.

5.11 ConvertedRoute

Conditional Mandatory: False.
Required Flags: Well-known.
Potential Flags: None.
CTRIP Type Code: 12.

The ConvertedRoute attribute indicates that an intermediate CTRIP-speaker has altered the route by changing the route's Application Protocol. For example, if a CTRIP-speaker receives a route with Application Protocol X and changes the Application Protocol to Y before advertising the route to an external peer, the CTRIP-speaker MUST include the ConvertedRoute attribute. The attribute is an indication that the advertised application protocol will not be used end-to-end, i.e., the information advertised about this route is not complete.

5.11.1 ConvertedRoute Syntax

This attribute has length zero (0); the value field is empty.

5.11.2 Route Origination and ConvertedRoute

Routes are never originated with the ConvertedRoute attribute.

5.11.3 Route Selection and ConvertedRoute

The ConvertedRoute attribute may be used in route selection – it indicates that advertised routing information is not complete.

5.11.4 Aggregation and ConvertedRoute

If any of the routes to aggregate has the ConvertedRoute attribute, then so MUST the resultant aggregate.

5.11.5 Route Dissemination and ConvertedRoute

If a CTRIP-speaker changes the Application Protocol of a route before advertising the route to an external peer, the CTRIP-speaker MUST include the ConvertedRoute attribute.

5.12 IPDestination

Conditional Mandatory: False.
Required Flags: Well-known.
Potential Flags: None.
CTRIP Type Code: 130.

The IPDestination attribute indicates the source of the information, if the route has been imported from another routing protocol. The attribute is used by the converting entity to avoid exporting routes back to the same protocol from which the information originates.

5.12.1 IPDestination Syntax

The IPDestination attribute carries a 1-octet unsigned numeric value. The currently defined values are the following:

Value	Originating protocol
0	**CTRIP:** The information has been inserted into CTRIP without conversion. The terminal is thus not on an IP network.
1	**ENUM:** The information has been converted from ENUM to CTRIP. The terminal is an IP terminal.
2	**TRIP**: The information has been converted from TRIP to CTRIP. The terminal is an IP terminal.
3	**OTHER:** The source of the information is unknown. The terminal is an IP terminal.

For routes without the IPDestination attribute, the value of the attribute defaults to 0 (CTRIP).

5.12.2 Route Origination and IPDestination

Routes leading to terminals on an SCN network are never originated with the IPDestination attribute. Routes leading to terminals on an IP networks MUST be originated with an IPDestination attribute with a value corresponding to the source of the information.

5.12.3 Route Selection and IPDestination

The IPDestination attribute may be used in route selection. A non-zero value indicates that the terminal is on an IP network.

5.12.4 Aggregation and IPDestination

Routes with differing IPDestination value MUST NOT be aggregated. Routes with the same value in the IPDestination attribute MAY be aggregated and the same IPDestination attribute attached to the aggregated object.

5.12.5 Route Dissemination and IPDestination

A CTRIP-speaker MUST NOT change the IPDestination attribute.

5.13 NumberPortabilityState

Conditional Mandatory: False.
Required Flags: Well-known.
Potential Flags: None.
CTRIP Type Code: 131.

The NumberPortabilitState attribute indicates the state of a route that is changing due to number portability. The attribute improves synchronization of the removal of the previous advertisement with the distribution of the new advertisement.

5.13.1 NumberPortabilityState Syntax

The IPDestination attribute carries a 1-octet unsigned numeric value. The following values are defined:

Value	State
1	**NP_READY_TO_MOVE:** The previous domain has signaled that the route can be replaced by the new route. This value is attached to the replaced route.
2	**NP_MOVE_COMPLETED:** The new domain advertises the new route. This value is attached to the replacing route.

Routes without the NumberPortabilityState are in the normal state (**NP_NORMAL**).

5.13.2 Route Origination and NumberPortabilityState

Routes which are not being moved or which not recently have been moved MUST NOT be originated with the NumberPortabilityState attribute.

During the movement of a route, the following procedure is followed:

1. The previous advertiser signals that the route can be replaced by originating the route with an attached NumberPortabilityState attribute signaling the state NP_READY_TO_MOVE. The other attributes SHOULD be the same as in the previous advertisement of the route.
2. When the new advertiser receives a route in the NP_READY_TO_MOVE state, a new route for the prefix can be installed. The first advertisement for the new route has a NumberPortabilityState attribute with the NP_MOVE_COMPLETED state.
3. When the previous advertiser receives an advertisement in the NP_MOVE_COMPLETED state from any peer for the same route that itself advertises in the NP_READY_TO_MOVE state, the own advertisement MUST be removed.
4. All consecutive advertisements for the moved route MUST NOT contain any NumberPortabilityState attribute.
5. A route in the NP_READY_TO_MOVE state SHOULD only be active in one (1) day, and MUST NOT be active longer than three (3) days. If the previous advertiser does not receive

any advertisement for the route in question in the NP_MOVE_COMPLETED state during this time, the own advertisement MUST be withdrawn.

6. A route in the NP_MOVE_COMPLETED state SHOULD only be active in one (1) day, and MUST NOT be active longer than three (3) days. After this time, the route is advertised without a NumberPortabilityState attribute.

5.13.3 Route Selection and NumberPortabilityState

The NumberPortabilityState attribute MUST be used in route selection, according to the following rules:

1. A route in the NP_READY_TO_MOVE state MUST be assigned a preference of one fourth (1/4) or less of the preference of a similar route without the attribute.
2. A route in the NP_MOVE_COMPLETED state MUST be assigned a preference equal to or higher than the preference of a similar route without the attribute.

5.13.4 Aggregation and NumberPortabilityState

Routes with differing NumberPortabilityState value MUST NOT be aggregated. Routes with the same value in the NumberPortabilityState attribute MAY be aggregated and the same NumberPortabilityState attribute attached to the aggregated object. However, aggregation of attributes with non-zero value in the NumberPortabilityState attribute is NOT recommended due to the short duration of the state.

5.13.5 Route Dissemination and NumberPortabilityState

A CTRIP-speaker MUST NOT change the NumberPortabilityState attribute.

5.14 Considerations for Defining New CTRIP Attributes

Any proposal for defining new CTRIP attributes should specify the following:

- the use of this attribute,
- the attribute's flags,
- the attribute's syntax,
- how the attribute works with route origination,
- how the attribute works with route aggregation, and
- how the attribute works with route dissemination and the attribute's scope (e.g., intra-domain only like LocalPreference)

IANA will manage the assignment of CTRIP attribute type codes to new attributes.