Implementation Experience with Extended network tuples for the AA Protocol

Philip Budne
phil@Shiva.COM

Shiva Corporation
1 Cambridge Center
Cambridge, Ma 02142

ABSTRACT

This paper describes Experience from implementing the extended AA routine tuples described in ‘‘Extended network tuples for the AA Protocol’’ released in February 1992.

1. Implementations

The following experimental implementations have been done (with limitations and changes to be described); Tom Evans of Webster has implemented an `atalkad` that can parse extended routes as described in my memo, and has modified the Webster MPG to accept such routes.

I have modified K-STAR to send and accept extended routes in `aaROUTEQ` and `aaROUTE` packets, as well from atalkad.

2. `flags` field values

In ‘‘Extended network tuples for the AA Protocol’’ I describe a very limited legal set of values for the arouteTuple flags field. In practice this set is too small.

Values described in my previous memo;

`0x20` ‘‘Host’’
`0x22` ‘‘Async’’
`0x40` ‘‘Net’’ with zero low-order ones in broadcast
`0x41` ‘‘Net’’ with eight low-order ones in broadcast
`0x42` ‘‘Net’’ with sixteen low-order ones in broadcast
`0x43` ‘‘Net’’ with twenty-four low-order ones in broadcast
`0x80` ‘‘Kbox’’
`0x90` ‘‘Core Kbox’’

The following additional value must be accepted for compatibility with existing atalkad implementations;

`0x81` ‘‘Kbox Etalk’’

Lastly, existing versions of K-STAR will send routes for connected ports with the 0x80 (old ‘‘from AA’’ bit) set. While my previous memo prohibits ever setting the AA bit on packets sent on the wire the following values should be reserved;

`0x88` K-STAR own LTalk
`0x89` K-STAR own ETalk
`0x98` K-STAR core gw own LTalk

This behavior in K-STAR exists to ensure that any router that does a full route replace based on the new route (ie; the LocalTalk of a Core Gateway) will not allow the route to age (and thus lose the ability to communicate with the core gateway)!

Many strategies exist to defend against needing this;

[1] NEVER replace a configured route.

[2] Never remove the ''configured'' property of a route.

[3] Keep ''core gateway'' and ''trusted gateway'' lists seperate from the routing table, so that the routing table need not be consulted for the `ipRus`test.

[4] more here?!

The problem remains that any gateway which sends aroutes for connected ports in `aaROUTEQ/aaROUTE` may cause problems with existing implementations (particularly if the advertisement is at a lower cost than the initial route from atalkatab). It would therefore be prudent to allow advertisement of such routes with the ''from AA'' field.

3. Route packing for atalkad

While atalkad has traditionally delivered initial routes in the exact order in which they appear in `atalkatab` this presents a problem. Delivery of multiple initial routing packets by atalkad is based on the construction of an array of arouteTuples that is sent in groups of 64. Doing this in the presense of extended arouteTuples could cause an extended tuple to be split across packets.

Three strategies could be used to overcome this problem;

[1] Add a new ''pad'' tuple type to add to the array before storing an extended tuple that would cause a boundary crossing.

[2] Add pad entries as above, but detect and elide them before transmission (send some packets with only 63 routes).

[3] Reorder the routes, so that all extended tuples appear first. This strategy was adopted.

4. Problems with double tuples

Tom has observed several problems with my design to allow the one tuple of an extended tuple pair to be transmitted without the high order bit of the hops field set.

4.1. Exact Broadcast

Since the motivation for the ''exact broadcast address'' (the `node` field of the addiotional tuple) is to allow directed broadcast on networks where the correct directed broadcast address cannot be specified using a multiple of 8 low-order one bits, it is pointless to allow old routers to see such routes. Tom's atalkad always sets the high-order bit of the hops field if an exact broadcast address is present.

4.2. Hopcount Inequities

Some existing implementations advertise initial static `K` routes via RTMP at lower cost than others. While this inequity is merely irritating at present, it means that any router which understands extended arouteTuples, but advertises them at anything short of bargain basement costs will lose out to extended aroute blind ''wholesale'' routers.

Several strategies exist to solve this problem;

[1] Mandate low initial hopcount values, so that all routers agree.

[2] Set the high order bit on the hops fields of both arouteTuples, so that the extended route is only accepted by modified routers, and the extended route is always visible. Tom has adopted this strategy.

4.3. Discussion

Assuming adoption of strategy 2, the above changes make all double tuple routes completely invisible to unupdated routers. This lifts the requirement that the new extended tuple bear ANY resemblance (in particular the flags field) to the current arouteTuple. The extended tuple could even contain algorithm selectors as described in my ‘‘ROUTEX’’ proposal!

5. Extended Tuple Order

Tom offered the following observation on tuple order (note the points raised here are moot if the high order bit of BOTH hops fields are set!);

If you have an extended network with a range of ONE (i.e. network 10-10), then what an IPTalk-1 router is going to see (upon receiving an AA packet) net 10 at 1 hop followed by net 10 at 0x80 hops.

Given the ambiguity in Apple’s pseudocode for RTMP, it is likely that a router may interpret the second tuple as a POISON (notify neighbour) and the route may be killed a few microseconds after it was born.

Reversing the order of the tuples makes this sequence work (0x80 hop-count ignored or entered as POISON, 1-hop-count sets it alive and OK).

It would be possible to "get around" this with the "current" order by forcing all "10-10" type extended networks to "10" (an old-style tuple), but this would give inconsistent routing tables as visible from SNMP - the "expectation" is that non-zero-range-end means "extended network".

Also, maybe you’d better specify in the document that extended networks with a range of one SHOULD NOT be "collapsed" into non-extended tuples in case some bright spark thinks of doing it?

6. Specification of hops field

If the hops field of both tuples is to be flagged with 0x80, then the low five bits should be treated as hop-count, and the remaining two bits must be zero.