Specification for NACK Behavior in NLSR

Nick Gordon

October 7, 2016

Abstract

This document describes what behavior NLSR should take when receiving NACKS. The optimal behavior differs from case to case. All (?) important cases are given here.

1 Introduction

There are multiple cases to consider with respect to Interest type when describing NLSR's optimal behavior with regards to NACK processing. Generally, the importance of the data dictates the urgency of the response. We identify here two broad cases where automatic action is indicated and a third where it is not. We also identify five different NACK reasons and the response to each. Further, in each case, the mention of logging implies a higher level than normal; all important operations of the code generally emit log messages.

2 LSA, key, and validator Interests

This is the most detailed case. Generally, all three of these situations can be handled identically. They are:

- Congestion: If the network is congested, then we want to wait some time and then resend the Interest. A backoff algorithm should be used to prevent further congesting the network. A retry limit should not be used, as these three types of data are core data, and NLSR cannot function without them. As such, once we reach the ceiling of the backoff, we continue until outside intervention or the data is gotten.
- No route: If there is no route to the data, then we also can't route to that router generally. We should remove that NextHop from that FIB entry, and try another one. The normal operating procedures of the FIB entry apply. I.e. if we have a FIB entry with no NextHops, we want to remove its LSAs from the LSDB, as it is not useful for routing.
- Duplicate: We can ignore this NACK reason; this does indicate necessarily a problem.

- No reason: We can't do much about this NACK. The best recourse is to log a threatening, ominous message and hope that an operator figures out what's going on.

3 Hello Interests

This is the second interesting case. However, we note that only two cases can occur. That is, only the "No route" and "No reason" NACKs are possible:

- No route: A NACK with this reason for a Hello Interest indicates that NLSR is down on that router. That is, NFD has gotten the Interest, but does not know where to forward it. This will only happen if NLSR's prefixes are damaged or not registered altogether. We should emit a particular error message and treat that neighbor as down.
- No reason: The same strategy that is taken in the LSA case is taken here.

4 Sync Interests

We should probably ignore all NACKs gotten by Sync. This is a practical reason, as NLSR currently uses a forked version of ChronoSync. Currently, the original ChronoSync does not process NACKs, either, and so we do not want to write patches for outdated code that will be obviated by a merge back, anyway.

5 FaceStatus Dataset

When NLSR starts, it has to request the dataset of Faces from NFD. The only NACK reason that could be encountered here is "no reason", which is unfortunate. The only strategy that we can take is to retry repeatedly with a configurable retry count and timeout. After this point, NLSR should exit; it can't do *any* routing without this dataset. A possible variation on this strategy is to have a two-phase recovery, where phase 1 is instantaneous retry, meant to recover if NFD is busy for some reason, and then after that, phase 2, where retries are spaced by a wait. This phase 2 is meant to recover if NFD has not fully started up, with the wait intended to give NFD time to finish starting.

6 Nlsrc

In the case of Nlsrc-originating interests, we should only emit errors for the operator. There are two reasons for this: 1. Since Nlsrc sends only command Interests, a NACK to one of these indicates a different kind of error, and 2. As a command-line utility, it should be the operator's job to manage these errors; Nlsrc should not try to autoresolve issues. Though Nlsrc can receive NACKs, all of them should be treated identically by Nlsrc itself; log to output with the NACK reason.

7 Conclusion

We have explained the all of the places that may receive NACKs in NLSR, but we have not covered all possible NACK reasons. These are the most common NACK types. The justification for the “no reason” NACK is that there is no good “default behavior” that NLSR can default to for these cases, as we have no information.