NSIS Working Group
Internet Draft

Robert Hancock
Siemens/
Roke Manor Research

Document: draft-hancock-nsis-reliability-00.txt
Expires: February 2004 August 2003

Reliability Functions in the NSIS Transport Layer Protocol

Status of this Memo

Abstract

The Next Steps in Signaling working group is developing a protocol suite for signaling information about a data flow along its path in the network. The lower layer in the protocol suite, the NSIS Transport Layer Protocol (NTLP) is intended to provide a generally useful transport service for such signaling messages.

There is a long-running open question about how much (if at all) the NTLP should provide reliable message transport. There is a large amount of confusion about what this question even means, let alone how to answer it. This document identifies the possible reliability requirements for signaling protocols in general, based on past evaluations of RSVP and research in soft-state protocol performance. It makes a proposal for what kind of reliable transport functionality should be supported in the NTLP, and discusses some of the resulting impacts and constraints on the NTLP design.

Table of Contents

1 Introduction

The Next Steps in Signaling working group is developing a protocol suite for signaling information about a data flow along its path in the network. The lower layer in the protocol suite, the NSIS Transport Layer Protocol (NTLP) is intended to provide a generally useful transport service for such signaling messages. The actual signaling messages are in general originated within upper layer signaling applications, each having their own NSIS Signaling Layer Protocol (NSLP), the role of the NTLP is primarily just to move these messages around the network to the appropriate nodes. The general description of the NSIS protocol suite, including its layering structure, is provided in [1].

There is a long-running open question about how much reliability is needed in signaling messages, especially in the context of a soft-stage signaling model; the particular question relevant to the NSIS framework is how much the NTLP should provide reliable message transport, if at all. There is a large amount of confusion about what this question even means, let alone how to answer it. This document identifies the possible reliability requirements for signaling protocols in general, based on past evaluations of RSVP and research in soft-state protocol performance. It makes a proposal for what kind of reliable transport functionality should be supported in the NTLP, and discusses some of the resulting impacts and constraints on the NTLP design.

The structure of this document is as follows:
- Section 2 provides definitions and a framework for discussing reliability requirements, including a classification of message types, different types of reliability semantics, and which nodes reliability is relevant between.
- Section 3 provides highlights of the limited amount of 'research' (modeling, simulation, and practical experience) relevant to the reliability question. Most of this work relates to 'unmodified' RSVP [2], although some also has comparisons with the RSVP transport layer enhancements of [3].
- Section 4 discusses how required reliability functions could and should be split between the layers in the NSIS protocol suite.
- Section 5 discusses the implications of reliability for the NTLP design.
- Section 6 discusses the interactions between reliability and security of the protocol suite, and the NTLP in particular.
- Section 7 concludes with a proposal about how to proceed on this issue.

2 Signaling Reliability: Fundamental Concepts

The word 'reliability' has a number of connotations in the context of transport protocols, principally avoidance of message loss, re-ordering and duplication, and guarantee of data integrity. In the following, we concentrate primarily on message loss issues, since data integrity can be provided at the level of individual messages; the other properties can often be provided unilaterally if wanted by extensions at the receiver, but in general it will be signaling application dependent exactly how useful they are anyway.

2.1 What does 'Reliability' Mean?

It is a generally accepted fact that at least some signaling applications have a requirement that they should manage state in the network in a 'reliable' way, because they actually care about setting

up network state in a specific way. However, there are still several ways in which 'reliability' can be achieved from the signaling application's perspective. In particular, there are two (complementary) options:

1. Sending periodic refresh messages as in [2] can be considered a reliability mechanism. The messages could even be sent at a higher rate during an 'initialisation' period (a technique outlined in [2] and formalized in [3]).

2. Getting explicit protocol-level feedback about the 'success' of a signaling message and using that (or its absence) to repeat the message is probably the more traditional way in which 'reliability' is understood. This functionality was added to RSVP in [3].

The discussion of soft-state management in the development of the NSIS framework seems to have established that where method (1) is appropriate it should be implemented within the signaling application, and places no specific requirements on the NTLP other than to deliver individual messages. In addition, there is no dispute that some signaling applications will want to have some messages delivered with no reliability at all, and the NTLP should provide such a service. Therefore, the question on the table is:

"Should the NTLP provide a message delivery service which uses explicit feedback within the protocol to improve the reliability of operation of the overall signaling application."

2.2 Classification of Signaling Messages

We can classify the messages produced by a soft-state-based signaling application into 3 basic types, which will have different reliability requirements. These types are as follows:

1. 'Trigger' messages are signaling messages which ultimately cause an externally visible change in packet treatment for a particular data flow. Examples are installing a reservation for QoS resources for a flow, or opening a firewall pinhole, or modifying or removing such reservations or firewall configurations. Triggers have to be propagated all the way along the path that needs the state change (and maybe all the way back, e.g. RSVP PATH/RESV).

Trigger messages can loosely be distinguished as causing 'hard' or 'soft' changes; for example, a QoS trigger merely changes the performance of the network in handling a flow, whereas a firewall trigger will probably affect whether the flow is possible at all. However, even 'soft' triggers may have 'hard' consequences (e.g. in generating accounting records in the QoS case) and, therefore, we

won't worry about this distinction. The goal for our signaling transport solution is that messages which have the significance of a trigger are rapidly delivered to all nodes which need to see them.

As well as initialization (where loss delays session establishment by a refresh period) and termination (where loss delays resource release by a cleanup period), there may be circumstances where two independent triggers need to be sent mid-session. This might be to modify a reservation path in a mobility scenario or carry out some merging operations. Such triggers have to be sequenced reliably, and in particular the first delivered promptly. Without positive feedback, race conditions occur; these are not just pathological cases but are observed 'in the wild' (e.g. the RSVP merging discussion in [4]).

2. 'Refresh' messages are signaling messages which confirm existing state within a node (e.g. extending a cleanup timer) but which don't otherwise affect flow treatment. Refresh messages can be generated and absorbed at each signaling node (the RSVP approach), or only at flow endpoints (e.g. as in several alternative QoS signaling proposals, such as YESSIR[5] and Boomerang[6]). In either case, loss of a certain number K (often K=3) of successive messages causes any reservation state to be removed at that node and (in the RSVP case) along the remainder of the path.

Normally, the problem of lost refresh messages is ignored, since the probability of losing several messages in sequence can be made very small. However, there is at least an indirect relationship with the reliability question, since K must be large enough to reduce the risk of losing a session to an acceptable level. This means that either the cleanup period after session termination is very long if a teardown 'trigger' message is not used (or lost), or the refresh period must be reduced, thereby increasing the message processing load.

3. Signaling applications may produce other types of message, which aren't triggers or refreshes, and/or have no well-defined reliability requirements (e.g. messages which provide notification of errors that may be transient). We won't analyse the impact of such messages, other than to note that they may exist.

The basic question of this document is:

"What role should the NTLP play in ensuring prompt execution of signaling triggers, and how should it handle signaling refreshes to minimize network load and session failure?"

2.3 Where is Reliability Required?

Logically, reliability is an attribute of the manner of communication between a pair of nodes, implicitly incorporating any intermediate nodes between them which are taking part in that communication. The question is, which nodes should we consider reliability between:

1. The endpoints of the data flow - this is not possible in general, since these nodes might not even be NSIS-aware.
2. The 'outermost' signaling-application-aware nodes on the data path - on the assumption that if triggers and refreshes are delivered appropriately over this scope, all other signaling nodes will also automatically be in step as well.
3. Any pair of adjacent signaling-application-aware nodes - so signaling operations (triggering and refreshing) can be done with appropriate performance locally, even if there is no end-to-end change.
4. Any pair of adjacent NSIS-aware nodes (even nodes not aware of the particular signaling application in question). Note that since (according to [1]) the NTLP does not store signaling application state, these nodes cannot be message sources or sinks, and therefore provision of the functionality at this level could only be considered a backup to providing it at levels (2) or (3).

If NSIS is only interested in solutions where signaling state is updated in response to end-to-end application requirements, then (2) would probably be sufficient. However, at least some scenarios require local adaptation to changed network conditions without incurring end to end delays if possible (this 'local repair' functionality can be found in the base RSVP specification [2]). Logically, such local signaling exchanges might take place between any pair of nodes which store (per-flow) signaling application state. If anything, 'reliability' for such exchanges is even more important than for end-to-end exchanges, since the former occur mid-session where latency is critical, whereas the latter occur mainly at session start and end where latency is much less of an issue. In addition, while reliability only between adjacent NTLP peers might be desirable for NTLP-internal operations, it is not directly required as the mechanism for ensuring appropriate delivery of signaling application messages (and may even be sub-optimal as a mechanism for that).

Therefore, the assumption from this section is:

"The appropriate delivery of signaling application triggers and refreshes needs to be ensured between pairs of adjacent signaling application aware nodes (which store per-flow state); the problem cannot be forced out to data flow senders and receivers or their signaling proxies."

2.4 What Reliability Semantics are Appropriate?

There are several possible 'classes' of reliability that can be considered for the delivery of a signaling message by the NTLP. Informally, and roughly in order, they are:

Class 0: No reliability - the NTLP just accepts the message at the message generator and makes a single attempt to deliver it, with no feedback on success or failure. This class is included for completeness and to emphasise that it will be core NTLP functionality whatever else we do (it may also be the class that signaling applications use, for example, for refresh messages).

Class 1: Reliable delivery - the NTLP undertakes to get the message to the NTLP instance in the receiving signaling application node, or to signal an error to the message generator. This will provide recovery from network loss (due to congestion or corruption), but there are no guarantees that the receiving signaling application has started or finished processing the message (successfully or otherwise). This is the level of reliability provided by e.g. TCP for individual data segments.

Class 2a: Reliable execution - the NTLP delivers the message, and returns an acknowledgement indicating how the message has been processed at the signaling application level (e.g. that a reservation has or has not actually been installed). Most sensible layering designs would regard this type of acknowledgement as living in the signaling application protocol (NSLP), since the semantics of 'success' and 'failure' are likely to be very application specific. Class 2a is mentioned here to highlight that there may well need to be acknowledgement at the signaling application level anyway, regardless of what functionality the NTLP provides.

Class 2b: Hard state - the NTLP delivers the message which installs the state, and the signaling application is then allowed to assume that no further update messages are needed: the state will be removed when explicitly torn down, and the NTLP will reliably detect loss of a peer. Such functionality was indeed present in early versions of [3] (see e.g. the 'Last_Refresh' flag in [7]). However, it has been discussed (ad nauseam, literally) on the NSIS mailing list and agreed that, even if such design approaches were reasonable, they would be implemented in the signaling layer protocols without explicit NTLP support; the NTLP will provide at most 'hints' about possible neighbour state changes rather than reliable state change detection.

On the assumption that class 2a/2b should not be provided by the NTLP acting alone, the question from this section is therefore:

"Should the NTLP provide class 1 service (reliable message delivery), in addition to unreliable delivery, given that application specific acknowledgements will be handled by signaling application protocols (NSLPs)?"

Note that we are also assuming that the selection of 'class 1' is done by the generating NSLP instance on a per-message basis - i.e. it is not a global NTLP configuration setting per node, nor does an NSLP have to send all its message types the same way. Even for a given NSLP and message type, the appropriate reliability class might depend on local conditions.

3 Practical and Theoretical Performance Results

This section gathers together the available 'objective' information about how much of a problem a purely unreliable message delivery service is likely to be.

There is actually a disappointingly small amount of such information about 'vanilla' RSVP, presumably because of its limited deployment. So this section also includes a small discussion of how other signaling protocols have evolved to cope with running over lossy networks (section 3.4).

3.1 Message Loss Rates

3.1.1 Raw Packet Loss Rates

There is a moderate amount of literature on this subject [8,9,10,11], which attempts to both characterize loss patterns and quantify them on the basis of real measurements. Unfortunately, one implication of the work is that packet loss patterns can have very complex statistical behaviour, and attempting to quantify loss as a single probability 'p' applying independently at the packet level is almost certainly over-simplified. In particular, there is very substantial variation between flows, between different destinations, and over different time periods (especially distinguishing between quiet periods and a 'busy hour'). However, a crude quantitative summary is that while a very high proportion of flows suffer losses of around $p<0.01$, a significant number (several %) suffer losses in the region $0.01<p<0.05$, and loss rates of $p>0.10$ are not uncommon, especially for some regions. An overall mean value of $p=0.02$-0.03 was apparently typical in 1995 [9], falling to $p<0.01$ 5 years later [10](but still with around 1% of flows experiencing $p>0.10$). Another way of putting this is that few flows experience loss, but if they do a figure of $p=0.03$-0.05 is typical and seems to be stable over time.

There are also ongoing 'live' Internet measurement activities; collections of pointers are at [12,13], and some particular sites are [14,15,16]. These latter sites tend to measure loss statistics for low rate ping probes, and the results for this may be more applicable to signaling traffic than TCP measurements. One site reports long term loss rates of the order of p=0.04 but without much background information; the IEPM site [14] reports lower averages but still p around 0.03-0.04 in several parts of the network. (IEPM is also measuring network performance between 'well-connected' academic and research sites rather than the Internet as a whole.)

What level of 'p' we aim to cope with in NSIS is of course a value judgment about how widely usable we would like NSIS signaling to be - do we only care about operation in well-dimensioned networks, or do we want functionality also even in 'network meltdown' situations. A personal preference would be that:

"Signaling protocols should suffer only marginal performance degradation in environments where source-destination packet loss rates are in the region 3-5%; and the protocols should still function somehow even if packet loss rates are >10%, although accepting that user level applications will also probably function poorly in such environments."

3.1.2 Impact of Fragmentation

The signaling message loss rate is the same as the packet loss rate only if signaling messages fit into single network layer packets. Crudely, in the absence of any reliability support, fragmentation into F fragments expands the message loss rate from p to 1-(1-p)^F. As an example, for an application generating a 2kbyte signaling message that had hit a link with around a 576byte MTU, we would be wanting 'reasonable' performance in the face of a 11-18% message loss rate, and some continued functioning in the face of a 35% message loss rate.

(This calculation may be pessimistic if packet loss is really dominated by losing bursts of sequential packets. But there is general acceptance (see [17]) that fragmentation without reliability is bad news for overall network performance, and it isn't clear how else to quantify this effect.)

3.1.3 Distribution of 'p' Over the Path

The above values for 'p' refer to end-to-end packet loss rates. However, in the case of NSIS, signaling messages are exchanged between adjacent NSIS-aware peers, which will generally be just a subset of the complete path. Therefore, the values of p given above

will not necessarily be appropriate for use in calculations of the effect of packet loss on signaling responsiveness.

However, in fact it is implied in several discussions of Internet packet loss that the dominant contribution for p comes from a single 'bottleneck' link (or a very small number of them); for example, this would be consistent with the high variability of p between different paths. In other words, we can use the above values of p unchanged:
- trigger messages, which have to be propagated along enough of the path to include the bottleneck, will have the corresponding transaction fail with probability p
- refresh messages over the affected bottleneck link will be lost with probability p, and this will be the dominant contribution to premature session termination.

3.2 Trigger Message Delivery

The main problem caused by packet loss is delayed or lost execution of trigger-induced state changes:
- failure of a trigger at state initialization or modification (e.g. after a route change) will cause some session failure for at least one further refresh period;
- failure of a trigger at state termination will lead to incorrect state persisting in the network for at least one cleanup period (usually some number of refresh periods).

An analysis specifically of RSVP flow setup is given in [18], which gives a rather thorough derivation of formulae for the probability of failing to set up a reservation during the first refresh period, and the expected number of refresh periods required; some simulation results are also given. The results given are the intuitively reasonable ones, for example that only around (1-p)^2 of sessions will be set up successfully by the first round of messages (the exponent 2 arises because RSVP requires both a PATH and RESV message). For our 'typical' environments, this corresponds to a success rate of 90-94% at p=0.03-0.05 (66-78% with fragmentation); at p=0.10 the figures become 81% (42%). Such success rates would probably be considered unacceptable for many applications, which is the origin of all the RSVP extensions to improve startup behaviour, such as [3]. (Of course, they only apply to flow paths which experience such loss rates, which may be only a small proportion of the total; however, that proportion might well include the whole busy hour every weekday, for example.)

A more abstract analysis of soft-state protocols in general in provided in [19]. The model (using queuing theory) is not directly based on RSVP, but is applicable to the NSIS problem space. The authors introduce a metric for the 'level of consistency' in the

system, and show how adding NACK feedback improves this consistency even at low-moderate loss rates (from 90% to nearly 100% at p=0.05 for a system parametrisation typical of voice calls), and maintains good values even at very high values of p.

Of course, none of this proves either way whether reliability is required in the signaling protocol. Potential users have to make up their own minds based on their impression of the figures.

3.3 Refresh Message Delivery

The effect of using unreliable refresh message delivery is that the network must be prepared to retain state during a cleanup period longer than a single refresh period to allow for lost refreshes. The cleanup period is measured as some number K of refresh periods. To remove state before this cleanup period requires an explicit trigger (a teardown).

If K successive refreshes are lost the session will also be lost. Assuming that the session has been successfully initialized, the probability that this has happened by the Nth refresh period is roughly 1-(1-p^K)^(N-K).

(A more exact answer to within O(p^N) is given by the expression

$$1 - a^N \frac{1-(1-a)/(1-p)}{1-K(1-a)/a}$$ where a is near 1 and satisfies $a=1-(1-p)(p/a)^K$.)

To make this concrete, the likelihood of a premature cleanup for a 3 minute session, K=3 and 30 second refreshes is <0.05% for p=0.05, quadrupling for a 10 minute call. Fragmentation would be an unusual requirement for refreshes (assuming that the receiving node is prepared to retain per-flow state instead), but for completeness the rates rise to 2% and 9% respectively in that case.

It is certainly not the intention of this section to argue that soft-state refresh messages should be delivered reliably (or, in reality, maintaining a high delivery probability regardless of network behaviour for user traffic). An equally reasonable approach is simply to increase the value of K to 4 or 5. However, unless the refresh period is reduced (increasing signaling load), this will likewise increase the cleanup period and hence the importance of reliable teardown delivery.

3.4 Experience from Other Protocols

RSVP is not the only soft-state protocol; other examples are PIM [20] and SAP [21]; ROHC [22] also uses soft state mechanisms in one of its modes of operation. Neither PIM nor SAP contain any mechanisms for feedback and retransmission (which are of course hard to provide in the multicast environment in any case); the updated PIM specification [23] does contain some additional reliability mechanisms, and in any case, PIM is less dependent on the prompt delivery of trigger messages at initialization than protocols such as RSVP. ROHC is able to function without feedback, but this mode of operation is usually reserved for unidirectional links; feedback is used in other modes to indicate that particular decompression state has been established or as negative acknowledgements to indicate that it is invalid and must be refreshed. When feedback is used, the hardness of the state becomes discretionary for the decompressor, which can use NACKs to signal that state refresh is required.

In the unicast routing area, the original protocols (RIP, EGP) were soft-state protocols based on periodically repeated advertisements. For other than trivial networks, they have been replaced by protocols (OSPF, BGP) with much better resilience to packet loss (among of course a very large number of other extensions in functionality). It seems clear that the protocol designers preferred to avoid having to worry about detecting and recovering from message loss at the same time as specifying the parts of the protocol specific to the routing application, and in each case, retransmission is provided as a fairly self-contained lower protocol (sub-)layer. However, the end result (that BGP in particular is essentially a hard-state protocol) may also not be the best guidance for NSIS protocol development. A similar evolution has taken place in the AAA environment, from the UDP-based RADIUS [24] which relies on a fairly simple application layer retransmission strategy to DIAMETER [25] which uses a fully reliable lower transport layer. The need for and justification of using of a separate reliable transport is discussed (somewhat inconclusively) in [26] and [27].

This set of comparisons does not prove that reliability (of any sort) is needed in a new signaling protocol. However, it does probably strongly imply that the problem of packet loss in the Internet cannot be ignored as 'too rare to bother about' during protocol design, however tempting that may be.

4 Reliability Architectural Options

Even accepting that some form of reliability is needed, there are still several options for how to provide it.

4.1 Uni-Directional Staged Refresh Timers

One option is simply to forget about using feedback at all, and use exponentially backed-off refreshes to minimize session initiation latency. This is one of the components of the RSVP extensions in [3], and similar techniques are used in some other protocols such as CRTP [28].

The design rationale and benefits of the approach for the RSVP extensions are discussed in more detail in the original paper that proposed them [29]; however, the approach provides most benefit when coupled with feedback messages (MESSAGE_ID_ACK), and the authors of that paper regard the particular solution eventually designed as something that could be done much better if backwards compatibility was not a requirement (see [30] and [31] for this and further discussion).

Particular issues are
- the complex interactions between staged refresh timer management and other events taking place within the signaling application (section 2.1 of [31]);
- the fact that for short flows, using an initial rapid refresh is a non-trivial increase in network load. (This is much less of an issue in the MPLS environment, for which [3] is ideal.)

4.2 Network Engineering

If the network can be engineered so that signaling messages are not lost even when other (data) packets, a lot of the reliability problem goes away. In a context where the purpose of signaling is to guarantee loss-free data transport (i.e. QoS) to applications, this is a logically reasonable position, and was a background assumption in RSVP design: just use the same mechanism to provide QoS for signaling.

The NSIS environment is different. Some signaling will be in support of loss-tolerant flows, either real-time flows which can repair lost packets [32,33], or non-real-time flows using retransmission. The purpose of the signaling could be to guarantee the throughput in some remote part of the network (while accepting a degree of local packet loss), or to maintain a middlebox configuration. In addition, each reservation has a cost (maybe a monetary cost) to maintain, reducing the attraction of signaling for signaling flows; configuring a non-signaled mechanism for prioritizing signaling traffic opens up an avenue for abuse of the network by other traffic.

We should not rule out engineering the network to minimize loss of signaling traffic; however, we should not depend on it to make

signaling work in the first place, especially considering the barrier this would place in the way of initial deployment.

4.3 Upper-Layer Feedback

Another option is that one could have the NTLP provide only an unacknowledged service, and initiate any necessary retransmissions from the signaling application (possibly based on end-to-end feedback only). There are some attractions to this approach, especially given that applications will often have feedback messages anyway, and indeed it is modeled in some detail in [18].

The following issues would remain with such an approach (the most serious ones at the end):

- Handling both transport and application state within the signaling application is still a source of complication, which is probably unnecessary.
- Compared to the NTLP, the signaling application is insulated from knowledge about network performance, and is much less able to make accurate judgements about sensible retransmissions timers or rates. In particular, any signaling application would know only about timing information for its own messages, whereas the NTLP naturally would have a wider view.
- Relying on end-to-end feedback (e.g. using an RSVP RESV as an implicit acknowledgement for a PATH) forces the management of per-flow state to get messages back through the network, or forces the endpoints to establish a separate (secure) relationship to exchange such feedback. This would hurt applications which process per-flow messages but which only need to store per-class state at interior nodes.
- Handling retransmission within the signaling application is very inefficient given the decision to handle fragmentation in the NTLP, since only complete messages (rather than fragments) would be retransmitted. (There were good, independent reasons to handle fragmentation in the NTLP, and this should not be seen as an excuse to re-open that argument.)
- Application layer feedback (if it exists) probably has different semantics from transport layer feedback, because it reports the result of much more processing (e.g. executing admission control algorithms, policy/AAA control checks, even user interaction). For the same reason, very different timeouts should probably apply. In other words, an application should not expect feedback at the application layer for several seconds, but if the reason for lack of feedback was a lost message, several seconds is much too long to wait to retransmit it.
- In particular, end-to-end application layer RTT estimation will have to be much more cautious than hop-by-hop NTLP RTT estimation. This is at least partly because in some cases the signaling

application could have a hard time working out where the 'end' really is (if there is some chain of proxies before an NSIS unaware flow endpoint). Therefore, the NTLP will be much more prompt in recovering from message losses.

My conclusion from this is that, in an ideal world for signaling application designers, the NTLP would provide the (optional-to-use) functionality of sending a message 'reliably' - that is, doing an optimal job of retransmission (at the right time and only if necessary) to make sure it arrived at the next node, or giving up and reporting an error.

In other words, this functionality appears to be clearly useful and correctly located in the NTLP rather than somewhere else. The remaining question is whether it can actually be provided in a cost-effective way.

5 NTLP Design Implications

The following sections describe some of the implications of reliability for the NTLP design. They indicate some of the attributes of what might be considered an 'appropriate' reliability service for signaling messages in the NSIS context, and some possible constraints on how it should be provided by the NLTP.

5.1 Intermediate NSIS Nodes

It's a consequence of the multi-application scope of NSIS that the signaling path between two NSLP peers may cross other NSIS nodes with no interest in that signaling application (or its messages), except possibly to do some message translation or enforce a routing policy. This situation is shown in Figure 1. Messages for NSLP A need to be sent reliably from NE1 to NE4, and go through NE2 and NE3 on the way.

There are good arguments that the reliability aspects of NTLP operation between NE1 and NE4 should not be forced to be processed fully at NE2 and NE3. One reason is that this represents a processing and state management burden on NE2 and NE3 which they do not benefit from; another is that an acknowledgement generated by (for example) NE2 to NE1 actually implies nothing about successful delivery to NE4, and requires NE1 to trust that NE2 and NE3 will correctly carry out any necessary retransmissions (in the face of node failures, implementation bugs, and so on).

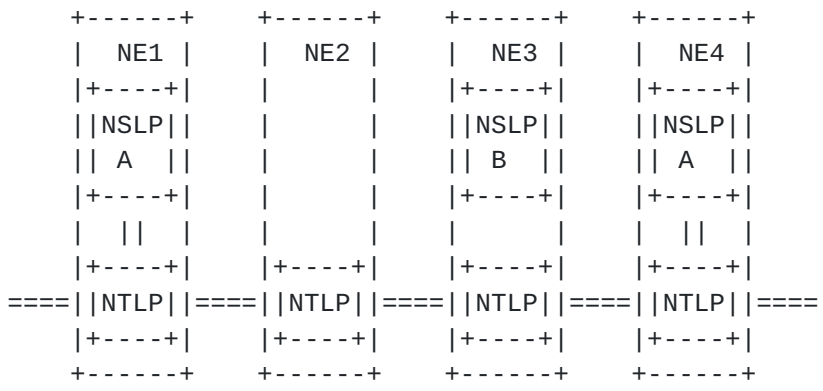

Figure 1: Signaling with Heterogeneous NSLPs

It would be preferable if acknowledgements were generated only at NE4 and forwarded transparently to NE1 (intermediate nodes could still generate negative acknowledgements to speed up retransmission of lost messages, and this might be a useful function in some specialized environments). The implication of this is that the NTLP would have to work in terms of messages that can be independently processed at intermediate nodes, without terminating the complete transport protocol within which they run.

5.2 Multiplexing, Head of Line Blocking, and Message Ordering

Compared to ordinary bulk data transmission, signaling messages (especially triggers) may have some fairly short 'useful' lifespan, beyond which delivering them makes no sense. The reliability functions of the NTLP should respect this.

Where messages for multiple applications and/or sessions are multiplexed over a single reliable link, messages for one application/session might be held up due to losses of messages for entirely unrelated applications/sessions. Ideally, the NTLP design should avoid this, and allow independent delivery of unrelated messages. This can either be done with multiple independent associations, or with multiple streams within a single association (sharing congestion control and RTT estimators, for example), as is possible with SCTP.

A related issue is where a message has been retransmitted several times (unsuccessfully), and as a result the application has generated an updated message for the same application/session which is blocked behind it. Further retransmissions of the original message are a waste of time. The question of how persistent to make local retransmissions has been discussed very intensively in the context of TCP operation over link layers using ARQ, and the results can be found in [34]; broadly, the conclusion is that fairly high

persistence is appropriate even if upper layers are also retransmitting. The argument is complicated by the fact that TCP reacts badly to re-ordering and high RTT variance (at least one of which must be caused by ARQ); putting the bulk of retransmission responsibility in the lower layer and insisting that upper layers are reordering tolerant would make the performance tradeoffs much less complex.

What does seem to be clear is that, in the NSIS context, the NTLP probably need not enforce ordering between messages (the receiving signaling application can do this if and when it wants), but it ideally would provide feedback at the sender about the fact that a message has been discarded as impossible to deliver. (If nothing else, many messages will be genuinely impossible to deliver, e.g. because there is no peer to deliver them to, and this certainly has to be reported.)

5.3 Congestion Control

It is assumed that any protocol implementing a retransmission strategy would have to do so in a congestion sensitive way. Any other approach would probably not be credible.

5.4 ACKs, NACKs, and Protocol State

There are several variant methods techniques to achieve reliable message delivery. The sender can retransmit on not receiving a per-message ACK in a given period; it can retransmit on receiving a per-message NACK; and it can set up some protocol state (a transport layer session) with its peer, within which combinations or more advanced variants can be used (e.g. acknowledgements for ranges of sequence numbers).

All of these have different trade-offs. A pure ACK approach can be lightweight at the receiver but requires RTT tracking at the sender; a pure NACK approach requires more synchronization or is less effective at spotting all message losses (e.g. trigger losses). Setting up a transport layer session has a cost in setup latency, but this cost can be shared over all signaling exchanges between two NTLP peers; it is also generally easier to protect against DoS attacks in a session based approach. The choice between these approaches is really a matter of NTLP detailed design.

5.5 Specific Link Layers

There are well known and exhaustively analysed issues in running certain transport protocols over certain types of link layer (specifically, TCP over wireless links, as discussed in [35]). Some

of these problems are intrinsic to attempting to achieve certain functionality - for example, to have retransmissions necessarily implies the overhead of header fields for message identification - whereas others may be artifacts of a particular protocol design approach or constraint. In any case, NTLP design work would have to assess the possibility of using variant approaches in different environments (e.g. as mentioned in [31]), or exploiting the work done in optimizing standard protocols for operation over such links (as in, for example, [36]).

6 Security Considerations

Adding any functionality to the NTLP means intrinsically that there is a greater number of threats it can be sensitive to, but also the additional functionality may provide protection against some security threats.

In our case, an adversary may attempt a variety of denial of service attacks on the NTLP by forcing nodes to create state associated with managing reliability. An adversary may attempt to forge feedback messages (positive or negative acknowledgements) to modify retransmission behaviour. Such issues are common to transport protocols in general, and detailed discussions can be found in the security considerations sections of modern transport protocols such as SCTP [37] and DCCP [38]. The complexity and subtlety of these discussions implies that it would be best if possible to implement reliability functions in the NTLP by re-using as much as possible of existing transport protocol concepts.

7 Conclusions

The conclusion of this draft is that it is appropriate for the NTLP to provide a reliable message delivery service, which would be optional for signaling applications to use. The role of such a service would be limited to ensuring rapid delivery of messages to the nodes where they are to be processed in signaling applications, and not to provide any application-layer state synchronization service or hard-state support. Such a reliability service should if possible be implemented in a way which can be transparent to intermediate NSIS nodes which don't take part in the signaling application; it will probably require congestion control in the NTLP as a consequence.

References

1 Hancock, R., I. Freytsis, G. Karagiannis, J. Loughney, S. van den Bosch, "Next Steps in Signaling: Framework", draft-ietf-nsis-fw-03.txt (work in progress), June 2003

2 Braden, R. et al., "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", RFC 2205, September 1997

3 Berger, L., D. Gan, G. Swallow, P. Pan, F. Tommasi, S. Molendini, "RSVP Refresh Overhead Reduction Extensions", RFC2961, April 2001

4 Baugher, M., and S. Jarrar, "Test Results of the Commercial Internet Multimedia Trials", ACM SIGCOMM Computer Communication Review, January 1997

5 Pan, P. and H. Schulzrinne, "YESSIR: A Simple Reservation Mechanism for the Internet", In the Proceedings of NOSSDAV, Cambridge, UK, July 1998.

6 G. Feher, K. Nemeth, M. Maliosz, I. Cselenyi, J. Bergkvist, D. Ahlard, T. Engborg, "Boomerang: A Simple Protocol for Resource Reservation in IP Networks", IEEE RTAS, 1999

7 Berger, L., D. Gan, G. Swallow, "RSVP Refresh Reduction Extensions", (expired i-d), March 1999, available at http://www.watersprings.org/pub/id/draft-berger-rsvp-refresh-reduct-00.txt

8 Borella, M., D. Swider, S. Uludag, G. Brewster, "Internet packet loss: Measurements and implications for End-to-End QoS," in Proceedings of International Conference on Parallel Processing, August 1998

9 Paxson, V., "End-to-End Internet packet dynamics", ACM SIGCOMM'97, September 1997

10 Zhang, Y., V. Paxson, and S. Shenker, "The Stationarity of Internet Path Properties: Routing, Loss, and Throughput", ACIRI Technical Report, May 2000

11 Paxson., V. "Measurements and Analysis of End-to-End Internet Dynamics", PhD thesis, University of California, Berkeley, April 1997

12 Schulzrinne, H., "Internet Performance and Traffic Measurements", at http://www.cs.columbia.edu/~hgs/internet/performance.html

13 Floyd, S., "Measurement Studies of End-to-End Congestion Control in the Internet", at http://www.icir.org/floyd/ccmeasure.html

14 Internet End-to-End Performance Monitoring, "The PingER Project", at http://www-iepm.slac.stanford.edu/pinger/

15 "The Internet Traffic Report", at http://www.internettrafficreport.com/main.html

16 "Internet Average", at http://average.matrixnetsystems.com/

17 Kent, C. A., J. C. Mogul, "Fragmentation Considered Harmful", Proceedings of ACM SIGCOMM, pages 390-401, August 1987

18 Mathy, L., D. Hutchinson, S. Simpson, "Modelling and Improving Flow Establishment in RSVP", Protocols for High Speed Networks, August 1999

19 Raman, S., and S. McCanne, "A Model, Analysis, and Protocol Framework for Soft State-Based Communication", SIGCOMM Symposium on Communications Architectures and Protocols, August 1999

20 Estrin, D., D. Farinacci, A. Helmy, D. Thaler, S. Deering, M. Handley, V. Jacobson, C. Liu, P. Sharma, L. Wei, " Protocol Independent Multicast-Sparse Mode (PIM-SM): Protocol Specification", RFC2362, June 1998

21 Handley, M., C. Perkins, E. Whelan "Session Announcement Protocol", RFC2974, October 2000

22 Bormann, C., C. Burmeister, M. Degermark, H. Fukushima, H. Hannu, L-E. Jonsson, R. Hakenberg, T. Koren, K. Le, Z. Liu, A. Martensson, A. Miyazaki, K. Svanbro, T. Wiebke, T. Yoshimura, H. Zheng, "RObust Header Compression (ROHC): Framework and four profiles: RTP, UDP, ESP, and uncompressed", RFC 3095, July 2001

23 Fenner, W., M. Handley, H. Holbrook, I. Kouvelas, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", draft-ietf-pim-sm-v2-new-07.txt (work in progress), March 2003

24 Rigney, C., S. Willens, A. Rubens, W. Simpson, "Remote Authentication Dial In User Service (RADIUS)", RFC 2865, June 2000

25 Calhoun, P., J. Loughney, E. Guttman, G. Zorn, J. Arkko, "Diameter Base Protocol", draft-ietf-aaa-diameter-17.txt (work in progress), December 2002

26 Aboba, B., P. Calhoun, S. Glass, T. Hiller, P. McCann, H. Shiino, G. Zorn, G. Dommety, C. Perkins, B. Patil, D. Mitton, S. Manning, M. Beadles, P. Walsh, X. Chen, S. Sivalingham, A. Hameed, M. Munson, S. Jacobs, B. Lim, B. Hirschman, R. Hsu, Y. Xu, E. Campbell, S. Baba, E. Jaques, "Criteria for Evaluating AAA Protocols for Network Access", RFC 2989, November 2000

27 Mitton, D., M. St.Johns, S. Barkley, D. Nelson, B. Patil, M. Stevens, B. Wolff, "Authentication, Authorization, and Accounting: Protocol Evaluation", RFC 3127, June 2001

28 Casner, S., and V. Jacobson, "Compressing IP/UDP/RTP Headers for Low-Speed Serial Links", RFC 2508, February 1999

29 Pan, P., and H. Schulzrinne, "Staged Refresh Timers for RSVP", Proceedings of Global Internet, November 1997

30 http://www1.ietf.org/mail-archive/working-groups/nsis/current/msg02483.html

31 Pan, P., H. Schulzrinne, "An Evaluation on RSVP Transport Mechanism", draft-pan-nsis-rsvp-transport-01.txt (work in progress), July 2003

32 Li, A., F. Liu, J. Villasenor, J.H. Park, D.S. Park, Y.L. Lee, J. Rosenberg, H. Schulzrinne, "An RTP Payload Format for Generic FEC with Uneven Level Protection", draft-ietf-avt-ulp-07.txt (work in progress), November 2002

33 Liebl, G., M. Wagner, J. Pandel, W. Weng, "An RTP Payload Format for Erasure-Resilient Transmission of Progressive Multimedia Streams", draft-ietf-avt-uxp-05.txt (work in progress), March 2003

34 Fairhurst, G., L. Wood "Advice to link designers on link Automatic Repeat reQuest (ARQ)", RFC 3366, August 2002

35 Dawkins, S., G. Montenegro, M. Kojo, V. Magret, N. Vaidya, "End-to-end Performance Implications of Links with Errors", RFC 3155, August 2001

36 Inamura, H., G. Montenegro, R. Ludwig, A. Gurtov, F. Khafizov, "TCP over Second (2.5G) and Third (3G) Generation Wireless Networks", RFC 3481, February 2003

37 Stewart, R., Q. Xie, K. Morneault, C. Sharp, H. Schwarzbauer, T. Taylor, I. Rytina, M. Kalla, L. Zhang, V. Paxson, "Stream Control Transmission Protocol", RFC 2960, October 2000

38 Kohler, E., M. Handley, S. Floyd, J. Padhye, "Datagram Congestion Control Protocol (DCCP)", draft-ietf-dccp-spec-04.txt (work in progress), June 2003

Acknowledgments

Andrew McDonald and Hannes Tschofenig provided some valuable feedback on this draft during preparation. Abbie Surtees verified the mathematics, and Mark West explained RFCs 3095 and 3366 (in so far as this is possible). In addition, due thanks should be given to the members of the NSIS working group as a whole, whose >200 email messages on the subject have formed part of the input for this work.

Author's Address

Robert Hancock
Roke Manor Research
Old Salisbury Lane
Romsey
SO51 0ZN
United Kingdom
email: robert.hancock@roke.co.uk

Intellectual Property Considerations

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

Full Copyright Statement