Network Working Group
Internet-Draft
Expires: December 30, 2005

L. Eggert
S. Schuetz
S. Schmid
NEC

June 28, 2005

TCP Extensions for Immediate Retransmissions
draft-eggert-tcpm-tcp-retransmit-now-02

Abstract

This document classifies connectivity disruptions along an end-to-end path into four types and describes how standard TCP mechanisms can lead to inefficient or over-aggressive sending behavior when connectivity resumes. The proposed techniques for TCP mobility detection and response (LMDR) can improve behavior for some types of disruptions. This document describes another, complementary and orthogonal modification to TCP's retransmission scheme that improves performance for disruption types that TCP LMDR does not address. This extension is based on connectivity indicators, i.e., generic network events that may indicate that end-to-end connectivity has resumed. This document focuses on TCP modifications that use connectivity indicators to increase performance, but does not define the specifics of such connectivity indicators itself.

Table of Contents

1. Introduction

Depending on the specific path between two nodes in the Internet, disruptions in connectivity may be frequent. Host mobility and other factors can further increase the likelihood of connectivity disruptions. When hosts communicate with the Transmission Control Protocol (TCP) [RFC0793], their connections may either abort during periods of disrupted connectivity or exhibit significant performance reduction compared to permanently connected paths [SCHUETZ-THESIS][SCHUETZ-CCR]. This decrease in performance is mainly caused by TCP's retransmission behavior after connectivity returns.

While connection aborts due to TCP's "user timeout" [TCP-ILLUSTRATED] can be prevented with the TCP User Timeout Option [I-D.ietf-tcpm-tcp-uto], additional mechanisms are required to improve TCP performance when operating along paths with frequent connectivity disruptions.

This document describes a modification of TCP's retransmission scheme to improve performance in such scenarios. The basic idea is to trigger a speculative retransmission attempt when a TCP implementation receives an indication that connectivity to a previously unreachable peer node may have returned. [anchor2] These extensions increase TCP performance through more intelligent scheduling of retransmission attempts by using periods of connectivity more efficiently. They do not cause significant amounts of additional traffic (at most three empty ACKs in some situations, see Section 5.1) and do not change TCP's congestion control algorithms.

Note that this draft treats connectivity indicators as relatively abstract events and focuses on how TCP should react when receiving such triggers. Specific connectivity indicators will be described in separate drafts. Simple connectivity indicators (some of which have already been experimentally implemented [SCHUETZ-THESIS][SCHUETZ-CCR]) include link- and network-layer events occurring at the first and last hop of a path, i.e., on those links that are observable directly from the end hosts. Examples of such simple connectivity indicators are DHCP lease events [RFC2131], IPv6 router advertisements [RFC2460], or MobileIP [RFC3775] binding updates and HIP [I-D.ietf-hip-arch] readdressing events. It is also conceivable to use events inside the network, which are not directly observable by the end hosts, as connectivity triggers. However, it is currently unclear how to signal these events to the end hosts.

Section 3 discusses TCP performance over intermittently connected paths in more detail, classifies different types of disruptions and describes their implications on TCP. Section 4 presents examples for connectivity indicators and Section 5 describes the proposed

"immediate retransmission" extension to TCP. Section 6 discusses related work, Section 7 investigates security aspects of the proposed modification and Section 8 summarizes and concludes this document.

2. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Classification of Connectivity Disruptions

Connectivity disruptions occur in many different situations. They can be due to wireless interference, movement out of a wireless coverage area, switching between access networks, or simply due to unplugging an Ethernet cable. Depending on the situation in which they occur, the implications of connectivity disruptions are different and must be handled appropriately. This section attempts to classify different types of connectivity disruptions and discusses their implications and effects on TCP.

Two main properties affect how TCP reacts to a connectivity disruption: the duration of a connectivity disruption and whether the path characteristics have significantly changed after it ends. This draft distinguishes between "short" and "long" disconnections and "changed" and "unchanged" path characteristics.

Connectivity disruptions are "short" for a given TCP connection, if connectivity returns before the RTO first fires. In this case, standard TCP recovers lost data segments through Fast Retransmit and lost ACKs through successfully delivered later ACKs. Section 3.1 briefly describes this case.

Connectivity disruptions are "long" for a given TCP connection, if the RTO fires at least once before connectivity returns. In this case, TCP can be inefficient in its retransmission scheme, as described in Section 3.2.

Whether or not path characteristics change when connectivity returns is a second important factor for TCP's retransmission scheme. Standard TCP implicitly assumes that path characteristics remain unchanged for short disruptions by performing Fast Retransmit based on path parameters collected before the disruption. For long disruptions, standard TCP is more conservative and performs slow-start, re-probing the path characteristics from scratch.

These implicit assumptions can cause standard TCP to misbehave or perform inefficiently in some scenarios. Figure 1 illustrates the

standard TCP behavior.

```
                +----------------------+----------------------+
      Short     | Fast Retransmit      | Fast Retransmit      |
      Duration  | using collected path | using collected path |
      < RTO     | characteristics      | characteristics      |
                +----------------------+----------------------+
      Long      |                      |                      |
      Duration  | Slow-start           | Slow-start           |
      >= RTO    |                      |                      |
                +----------------------+----------------------+
                    Unchanged Path         Changed Path
                    Characteristics        Characteristics
```

Figure 1: Standard TCP behavior

3.1 Short Connectivity Disruptions

For short disruptions, standard TCP performs Fast Retransmit based on the previously collected path characteristics. When path characteristics remain unchanged after connectivity resumes, this is an effective and efficient mechanism.

On the other hand, when path characteristics do change after a short disruption, TCP may be too aggressive in its sending behavior. This can occur, for example, after MobileIP handovers. A TCP extension to inform a peer about such events and trigger the probing behavior is currently being proposed [I-D.swami-tcp-lmdr]. The LMDR extension involves forced invalidation of collected path characteristics and re-probing the new path from scratch.

3.2 Long Connectivity Disruptions

For long disruptions, standard TCP performs slow-start after connectivity returns. This is a conservative strategy that avoids overloading the new path. However, TCP's general exponential back-off retransmission strategy can time these slow-starts such that performance decreases.

This section describes this issue in detail. Later sections of this document propose a different method of scheduling retransmission attempts after connectivity returns, which results in a more efficient retransmission behavior.

When a long connectivity disruption occurs along the path between a host and its peer while the host is transmitting data, it stops receiving acknowledgments. After the retransmission timeout (RTO)

expires, the host attempts to retransmit the first unacknowledged segment. TCP implementations that follow the recommended RTO management proposed in [RFC2988] double the RTO after each retransmission attempt until it exceeds 60 seconds. This scheme causes a host to attempt to retransmit across established connections roughly once a minute. (More frequently during the first minute or two of the connectivity disruption, while the RTO is still being backed off.)

When the long connectivity disruption ends, standard TCP implementations still wait until the RTO expires before attempting retransmission. Figure 2 illustrates this behavior. Depending on when connectivity becomes available again, this can waste up to a minute of connection time for TCPs that implement the recommended RTO management described in [RFC2988]. For TCP implementations that do not implement [RFC2988], even more connection time may be lost. For example, Linux uses 120 seconds as the maximum RTO.

```
       Sequence
       number      X = Successfully transmitted segment
        ^          O = Lost segment
        |     :                     :              : X
        |     :                     :              :X
        |     OO O  O    O        O :              X
        |    X:                     :              :
        |   X :                     :<------------>:
        |  X  :                     :    Wasted    :
        | X   :                     :  connection  :
        |X    :                     :     time     :
        +-----:---------------------:--------------:-------->
              :                     :              :       Time
         Connectivity          Connectivity       TCP
            gone                  back        retransmit
```

Figure 2: Standard TCP behavior in the presence of disrupted connectivity

This retransmission behavior is not efficient, especially in scenarios where connected periods are short and connectivity disruptions are frequent [DRIVE-THRU]. Experiments show that TCP performance across a path with frequent disruptions is significantly worse, compared to a similar path without disruptions [SCHUETZ-THESIS][SCHUETZ-CCR].

In the ideal case, TCP would attempt a retransmission as soon as connectivity to its peer was re-established. Figure 3 illustrates the ideal behavior.

```
          Sequence
          number      X = Successfully transmitted segment
           ^          O = Lost segment
           |     :                     : X            :
           |     :                     :X             :
           |     OO O  O    O        O X              :
           |    X:                     :              :
           |   X :                     :<------------>:
           |  X  :                     :  Efficiency  :
           | X   :                     :  improvement :
           |X    :                     :              :
           +-----:---------------------:--------------:-------->
                 :                     :              :       Time
            Connectivity          Connectivity      Next
               gone             back = immediate  scheduled
                                 TCP retransmit   retransmit
```

Figure 3: Ideal TCP behavior in the presence of disrupted connectivity

The ideal behavior is difficult to achieve for arbitrary connectivity disruptions. One obviously problematic approach would use higher-frequency retransmission attempts to enable earlier detection of whether connectivity has returned. This can generate significant amounts of extra traffic. Other proposals attempt to trigger faster retransmissions by retransmitting buffered or newly-crafted segments from inside the network [SCOTT][I-D.dawkins-trigtran-linkup][RFC3819]. Section 6 compares these approaches to the "immediate retransmission" extension.

Note that scenarios exist where path characteristics remain unchanged after long connectivity disruptions. In this case, even an intelligently scheduled slow-start is inefficient, because TCP could safely resume transmitting at the old rate instead of slow-starting. Although originally developed to avoid line-rate bursts, techniques for the well-known "slow-start after idle" case [I-D.ietf-tcpimpl-restart] may be useful to further improve performance after a disruption ends. This document does not currently describe this additional optimization.

Also note that on asymmetric paths, TCP's standard retransmission behavior can be over-aggressive in some scenarios. If the forward and reverse paths have significantly different RTTs, a long disruption for the host that sends along the lower-delay path may still be a short disruption for its peer host, because of its correspondingly longer RTO. When the sender resumes and slow-starts, the peer performs fast retransmit, which can lead to line-rate bursts. An LMDR-like mechanism that resets collected path

characteristics and forces a slow-start re-probe could be effective in these cases [I-D.swami-tcp-lmdr].

The following sections of this document focus on mechanisms to improve TCP's efficiency when resuming after long disruptions. They use connectivity indicators to schedule retransmission attempts more effectively than TCP's standard exponential back-off scheme. To improve behavior after short disruptions, the mechanisms in [I-D.swami-tcp-lmdr] are effective. Eventually, the TCP-option-based mechanism described there may be extended to incorporate the retransmission option described in Section 5.2 below.

4. Examples of Connectivity Indicators

This section describes examples of connectivity indicators, which the retransmission mechanism described in the next section acts upon. This document does not define the specifics of such connectivity indicators but merely discusses them to illustrate the operation of the "immediate retransmission" extension.

Connectivity indicators signal TCP when connectivity to a previously unreachable peer may have returned. They depend on the specifics of a node and its environment, for example network-layer mechanisms such as DHCP [RFC2131], MobileIP [RFC3775] or HIP [I-D.ietf-hip-arch]. The IETF's Detection of Network Attachment (DNA) working group currently investigates the specifics of providing such connectivity indicators [I-D.ietf-dna-goals].

One example of a connectivity indicator is "next hop reachable." This indicator could occur if a combination of the following conditions is true, depending on host specifics:

- o Network-layer connectivity along the path to the destination returned, e.g., the outbound interface has an IP address and a next-hop router is known, maybe due to DHCP [RFC2131] or IPv6 router advertisements [RFC2460].

- o Link-layer connectivity of the link to the next-hop router along the path to the destination returned (e.g., link-layer "link up").

- o Other local conditions that affect reachability of the destination are satisfied (e.g., IKE exchanges [RFC2409], MobileIP binding updates [RFC3775] or HIP readdressing [I-D.nikander-hip-mm] have completed).

The "next hop reachable" connectivity indicator only depends on locally determinable information (e.g., state of directly-connected links, etc.) and does not require network cooperation. It can signal

TCP to restart active connections across intermittently connected links where disruptions occur on the first or last hop. This simple indicator has the potential to improve TCP performance in many cases, because connectivity disruptions at the first or last hop are arguably the most common cause of connectivity disruptions in today's Internet.

A second, more general example of a connectivity indicator would be "end-to-end connectivity returned." If hosts have the ability to detect or be notified of connectivity changes inside the network (i.e., not only at the first or last hop), a more general retransmission mechanism could act on those pieces of information. This can improve TCP performance across intermittently connected paths where disruptions occur at arbitrary links along the path, even inside the network. However, providing this more general connectivity indicator is problematic due to its dependence on remote information and its related issues, such as trust.

Connectivity indicators are generally asymmetric, i.e., they may occur on one peer host but not the other. As discussed above, a local event at one host may trigger the "immediate retransmission" mechanism, while the other host is unable to detect this event across the network. Symmetric connectivity indicators are a special case and always occur concurrently at both communicating hosts. Examples for such symmetric connectivity indicators are handshake events such as IKE exchanges or HIP readdressing. Symmetric indicators are an important special case, because the retransmission procedure required in response to a symmetric indicator is simpler than that for an asymmetric one. The next section will describe this in detail.

5. TCP Immediate Retransmission Extension

This section describes the main contribution of this document, i.e., TCP extensions for immediate retransmission in response to connectivity indicators. The basic idea behind the "immediate retransmission" extension is to allow TCP to resume stalled connections as soon as it receives an indicator that connectivity to previously unreachable peers may have returned.

This document does not specify how TCP determines which connections a specific connectivity indicator affects, i.e., for which connections it should initiate retransmission attempts. This is a property of individual connectivity indicators. For example, the "next hop reachable" indicator described in the previous section affects connections to all destinations routed through that hop.

It is important to note that this retransmission extension does not modify TCP's basic congestion control, fairness properties or slow-

start algorithms. The only difference in TCP behavior is the timing of retransmission events and, in some cases, a minor, fixed increase in the number of initially retransmitted segments. The "immediate retransmission" extension increases performance through better utilization of connected periods, not through sending traffic at a faster rate or modifying TCP's congestion control mechanisms.

Hosts that implement the "immediate retransmission" TCP extension MUST implement the following retransmission mechanism whenever they receive a connectivity indicator:

When receiving a symmetric or asymmetric connectivity indicator, conforming TCP implementations MUST immediately initiate the standard retransmission procedure for connections that the connectivity indicator affects - just as if the RTO for those connections had expired.

If the connectivity indicator is symmetric, i.e., all peers receive it concurrently; this simple change is sufficient to kick-start the relevant TCP connections.

If the connectivity indicator is asymmetric, this simple extension is not always sufficient, because only one peer has received the indicator. In case the host receiving the connectivity indicator has no (or too little) unacknowledged data awaiting retransmission, it will not emit enough segments to cause its peer node, which may have unacknowledged data as well, to attempt retransmission. Transmission would thus only resume in one direction, which is ineffective for two-way communication.

To avoid this issue, conforming TCP implementation MUST perform a different retransmission procedure in response to an asymmetric connectivity indicator. The following sections describe two alternative TCP modifications that aim to improve retransmission behavior after receiving an asymmetric connectivity indicator. Section 5.1 describes the first variant. As described in an earlier revision of this document, this variant generates duplicate ACKs to activate the peer's fast retransmit algorithm. Section 5.2 describes the second variant, based on an explicit, new TCP "immediate retransmission" option.

5.1 Variant Based on Fast Retransmit

This variant of improving TCP retransmission scheme based on connectivity indicators uses duplicate ACKs. Conforming TCPs MUST send at least four segments that all acknowledge the last segment received from a peer for all connections that the connectivity indicator affects. These triple-duplicate ACKs will activate the

peers' fast retransmit algorithms and cause them to immediately restart communication in the reverse direction, i.e., before their next scheduled retransmission.

In this variant, if a TCP connection that a connectivity indicator affects has four or more unacknowledged data segments in the retransmission queue, it SHOULD piggyback the triple-duplicate ACK to the regular retransmissions of those data segments. In this case, the "immediate retransmission" TCP extension does not require additional messages, compared to standard TCP.

For connections where the retransmission queue contains only three or less unacknowledged data segments, TCP implementations supporting the "immediate retransmission" TCP extension MUST send additional pure ACKs until they have sent a complete triple-duplicate ACK. In the worst case, when the retransmission queue is empty, this scheme requires four additional ACKs, compared to standard TCP.

After the peer's fast retransmit algorithm sends the assumed missing segment, TCP performs either fast recovery or a slow-start [RFC2581], depending on the length of the connectivity disruption. If the connectivity indicator occurs before the RTO, i.e., for very short disruptions, TCP has not yet lost its ACK clock and can thus perform fast recovery. After longer disruptions, TCP falls back to slow-start to restart the ACK clock, just as it does at the beginning of a connection.

The result of this modification is twofold. First, TCP connections receiving the connectivity indicator attempt retransmission of their unacknowledged segments before the next scheduled RTO. This increases utilization of connected periods. Second, TCP connections receiving the connectivity indicator use an existing TCP mechanism (triple-duplicate ACK) to signal their peer. Although the peer may not have received a connectivity indicator itself (e.g., the indicator was asymmetric), this causes it to attempt faster retransmission as well.

As mentioned above, the "immediate retransmission" scheme can generate up to four additional segments, compared to standard TCP. All additional segments are pure ACKs and hence small, resulting in a minor total overhead. Furthermore, measurements have shown that increasing TCP's initial window is not problematic [ALLMAN]; this may indicate that a minor increase in traffic at retransmission time may be tolerable as well.

5.2 Variant Based on Retransmission Option

Unlike the mechanism described in the previous section, the second

variant described in this section does not overload an existing TCP mechanism - i.e., fast retransmit - to improve retransmission after a connectivity disruption. Generating duplicate ACKs in the manner described in Section 5.1 was criticized by some working group participants as "an abuse of a well-defined TCP mechanism for an unrelated purpose." The variant described in this section uses a new TCP "Immediate Retransmission" Option to explicitly signal to the remote peer that should attempt a retransmit. It was originally suggested by Kacheong Poon [POON] and the specifics are currently under investigation; this section describes the current state of this variant.

When receiving a connectivity indicator, conforming TCP implementations MUST send a single segment to each affected peer. This segment MUST contain the TCP Immediate Retransmission Option and may either be a queued data retransmission or a pure ACK, if the connection has no data awaiting retransmission.

Upon reception of the TCP Immediate Retransmission Option, conforming TCPs MUST immediately attempt a retransmit. This could either be a fast retransmit for short disruptions or a slow-start for long disruptions.

Note that the same effect can be achieved by overloading the TCP LMDR option [I-D.swami-tcp-lmdr], i.e., sending it in response to a connectivity indicator, and triggering a retransmission attempt upon reception. Overloading LMDR in this way also has the benefit of being able to force a reset of accumulated path state followed by a slow-start. However, a flag in the option would be needed to trigger a retransmit without this reset, to cover all types of disruptions discussed in Section 3. Given the similar motivation and relatively orthogonal, complementary mechanisms proposed here and in [I-D.swami-tcp-lmdr], this may be attractive solution.

5.3 Discussion

One major drawback of using a retransmission option compared to the one based on fast retransmit is that it requires both communicating TCPs to implement this modification. Triggering a peer's fast retransmit with duplicate ACKs only requires the triggering local peer to support this extension - the triggered remote peer may run an unmodified TCP stack. Additionally, firewalls may block segments carrying unknown TCP options. Finally, TCP option space is becoming limited.

One major advantage of using an option, however, is that it avoids overloading an established TCP mechanism to produce side effects. Using an option also allows a more careful definition of the

semantics of the mechanism, e.g., control of whether and when it should reset accumulated path state or what transmission scheme to schedule retransmits with.

6. Related Work

Several other approaches try to improve TCP performance in the presence of connectivity disruptions [SCOTT][I-D.dawkins-trigtran-linkup][RFC3819]. They attempt to improve TCP startup after a connectivity disruption by retransmitting buffered or newly-crafted segments from inside the network.

These proposals can be problematic, because TCP is built on the assumption that segments older than the maximum segment lifetime (MSL) of 2 minutes [RFC0793] will never be received. When a connectivity disruption lasts longer than the MSL, either these proposals will become ineffective or they risk leaking buffered old segments onto new connections, violating TCP's semantics.

The "immediate retransmission" modification also improves performance over a path with frequent connectivity disruptions. The basic idea is to schedule an additional, speculative retransmission attempt when a TCP implementation receives an indication that connectivity to a peer node has returned. Unlike the other proposals, the "immediate retransmission" scheme uses regular retransmissions, i.e., retransmits data that is buffered at the end systems. Because that data has not entered the network yet, it is not subject to the problematic MSL rule. Consequently, the "immediate retransmission" scheme remains effective even for connectivity disruption longer than the MSL, without the risk of compromising connection integrity.

Other transport-layer approaches such as the Explicit Link Failure Notification [HOLLAND] or TCP-F [CHANDRAN] use specific messages generated by intermediate routers to inform TCP senders about disrupted paths. The former extends the TCP state machine with a new "stand by" state during which the standard retransmission timers are disabled. In this state, TCP periodically probes the network to detect connectivity reestablishment. Depending on the frequency of the probes and the network environment, this can cause significant amounts of extra traffic. TCP-F completely suspends ongoing connections until receiving "route reestablishment notifications" that indicate peer reachability. Both proposals are primarily designed for ad hoc networks and rely on changes to intermediate routers, whereas the "immediate retransmission" extension only requires end system support.

ATCP [ATCP] uses a similar approach as the Explicit Link Failure Notification, but discovers link failures through ICMP Destination

Unreachable messages. Caceres and Iftode [CACERES] propose and evaluate a solution similar to the TCP "immediate" retransmission extension that improves performance during MobileIP hand-offs. Unlike the solution proposed in this paper, the hand-off mechanism targets connectivity disruptions of a few seconds.

7. Security Considerations

To protect against abuse of the TCP "immediate retransmission" extension, e.g., denial-of-service attacks by flooding TCP with connectivity indicators, a control mechanism that "rate-limits" these indicators may be effective. This document does not currently discuss the security aspects of connectivity indicators and the "immediate retransmission" extension to TCP.

8. Conclusion

This document described the "immediate retransmission" extension to TCP's standard retransmission scheme. The new extension improves performance across intermittently connected paths through additional, speculative retransmission attempts upon receiving external connectivity indicators. One example of such a connectivity indicator is "first hop router reachable." This document did not define the specifics of such connectivity indicators, although it described some examples to illustrate the operation of the "immediate retransmission" extension, which is its main contribution.

9. IANA Considerations

This section is to be interpreted according to [RFC2434].

This document does not define any new namespaces. It uses an 8-bit TCP option number maintained by IANA at http://www.iana.org/assignments/tcp-parameters.

10. Acknowledgments

The following people have helped to improve this document through thoughtful suggestions and feedback: Marcus Brunner, Wesley Eddy, Kacheong Poon, Juergen Quittek and Joe Touch.

The authors are partly funded by Ambient Networks, a research project supported by the European Commission under its Sixth Framework Program. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Ambient Networks project or the European Commission.

11. References

11.1 Normative References

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, September 1981.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2434] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 2434, October 1998.

[RFC2581] Allman, M., Paxson, V., and W. Stevens, "TCP Congestion Control", RFC 2581, April 1999.

[RFC2988] Paxson, V. and M. Allman, "Computing TCP's Retransmission Timer", RFC 2988, November 2000.

11.2 Informative References

[ALLMAN] Allman, M., Hayes, C., and S. Ostermann, "An Evaluation of TCP with Larger Initial Windows.", ACM Computer Communication Review, Vol. 28, No. 3, July 1998.

[ATCP] Liu, J. and S. Singh, "ATCP: TCP for Mobile Ad Hoc Networks", IEEE Journal on Selected Areas in Communication, Vol. 19, No. 7, July 2001.

[CACERES] Caceres, R. and L. Iftode, "Improving the Performance of Reliable Transport Protocols in Mobile Computing Environments", IEEE Journal on Selected Areas in Communication, Vol. 13, No. 5, 1995.

[CHANDRAN] Chandran, K., Raghunathan, S., Venkatesan, S., and R. Prakash, "A Feedback Based Scheme For Improving TCP Performance In Ad-Hoc Wireless Networks", IEEE Personal Communication Systems (PCS) Magazine: Special Issue on Ad Hoc Networks, Vol. 8, No. 1, February 2001.

[DRIVE-THRU] Ott, J. and D. Kutscher, "Drive-Thru Internet: IEEE 802.11b for Automobile Users", Proc. Infocom 2004, March 2004.

[HOLLAND] Holland, G. and N. Vaidya, "Analysis of TCP Performance

over Mobile Ad Hoc Networks", Proc. 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking, Seattle, WA, USA, 1999.

[I-D.dawkins-trigtran-linkup]
Dawkins, S., "End-to-end, Implicit 'Link-Up' Notification", draft-dawkins-trigtran-linkup-01 (work in progress), October 2003.

[I-D.ietf-dna-goals]
Choi, J., "Goals of Detecting Network Attachment in IPv6", draft-ietf-dna-goals-04 (work in progress), December 2004.

[I-D.ietf-hip-arch]
Moskowitz, R., "Host Identity Protocol Architecture", draft-ietf-hip-arch-02 (work in progress), January 2005.

[I-D.ietf-tcpimpl-restart]
Hughes, A., Touch, J., and J. Heidemann, "Issues in TCP Slow-Start Restart After Idle", draft-ietf-tcpimpl-restart-00 (work in progress), March 1998.

[I-D.ietf-tcpm-tcp-uto]
Eggert, L. and F. Gont, "TCP User Timeout Option", draft-ietf-tcpm-tcp-uto-00 (work in progress), May 2005.

[I-D.nikander-hip-mm]
Nikander, P., "End-Host Mobility and Multi-Homing with Host Identity Protocol", draft-nikander-hip-mm-02 (work in progress), July 2004.

[I-D.swami-tcp-lmdr]
Swami, Y. and K. Le, "Lightweight Mobility Detection and Response (LMDR) Algorithm for TCP", draft-swami-tcp-lmdr-05 (work in progress), February 2005.

[POON] Poon, K., "Personal Communication", August 2004.

[RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, March 1997.

[RFC2409] Harkins, D. and D. Carrel, "The Internet Key Exchange (IKE)", RFC 2409, November 1998.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

[RFC3775] Johnson, D., Perkins, C., and J. Arkko, "Mobility Support in IPv6", RFC 3775, June 2004.

[RFC3819] Karn, P., Bormann, C., Fairhurst, G., Grossman, D., Ludwig, R., Mahdavi, J., Montenegro, G., Touch, J., and L. Wood, "Advice for Internet Subnetwork Designers", BCP 89, RFC 3819, July 2004.

[SCHUETZ-CCR] Schuetz, S., Eggert, L., Schmid, S., and M. Brunner, "Protocol Enhancements for Intermittently Connected Hosts", To appear: ACM Computer Communication Review, Vol. 35, No. 3, July 2005.

[SCHUETZ-THESIS] Schuetz, S., "Network Support for Intermittently Connected Mobile Nodes", Diploma Thesis, University of Mannheim, Germany, June 2004.

[SCOTT] Scott, J. and G. Mapp, "Link layer-based TCP optimisation for disconnecting networks", ACM Computer Communication Review, Vol. 33, No. 5, October 2003.

[TCP-ILLUSTRATED] Stevens, W., "TCP/IP Illustrated, Volume 1: The Protocols", Addison-Wesley , 1994.

Editorial Comments

[anchor2] LE: The authors have seen the idea of triggering retransmits based on connectivity events of directly-connected links attributed to Phil Karn, but were unable to locate a specific reference. Pointers to a citable reference are highly appreciated!

Authors' Addresses

Lars Eggert
NEC Network Laboratories
Kurfuerstenanlage 36
Heidelberg 69115
Germany

Phone: +49 6221 90511 43
Fax: +49 6221 90511 55
Email: lars.eggert@netlab.nec.de
URI: http://www.netlab.nec.de/

Simon Schuetz
NEC Network Laboratories
Kurfuerstenanlage 36
Heidelberg 69115
Germany

Phone: +49 6221 90511 65
Fax: +49 6221 90511 55
Email: simon.schuetz@netlab.nec.de
URI: http://www.netlab.nec.de/

Stefan Schmid
NEC Network Laboratories
Kurfuerstenanlage 36
Heidelberg 69115
Germany

Phone: +49 6221 90511 54
Fax: +49 6221 90511 55
Email: stefan.schmid@netlab.nec.de
URI: http://www.netlab.nec.de/

Appendix A. Document Revision History

Revision	Comments
00	Initial version.
01	Updated terminology according to [SCHUETZ-CCR]. Added "retransmission option" variant as Section 5.2.

02	Added classification of connectivity disruptions and describe the design space in relation to [I-D.swami-tcp-lmdr] and [I-D.ietf-tcpimpl-restart]. Updated references.

Intellectual Property Statement

Disclaimer of Validity

Copyright Statement

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.