Priority Based Delivery of PR-SCTP Messages in a Syslog Context

Mohammad Rajiullah, Anna Brunstrom, and Stefan Lindskog

Department of Computer Science, Karlstad University
SE-651 88 Karlstad, Sweden
{mohammad.rajiullah,anna.brunstrom,stefan.lindskog}@kau.se

Abstract. Unquestionably, syslog provides the most popular and easily manageable computer system logging environment. In a computer network, syslog messages are used for several purposes such as for optimizing system performance, logging user's actions and investigating malicious activities. Due to all these essential utilities, a competent transport service for syslog messages becomes important. Most of the current syslog implementations use either the unreliable UDP protocol or the more costly reliable TCP protocol. Neither of these protocols can provide both timeliness and reliability, while transporting inherently prioritized syslog messages in a congested network. In this paper, we both propose and evaluate the use of PR-SCTP, an existing partial reliability extension of the SCTP transport protocol, as a candidate transport service for the next generation syslog standard. In our emulation based experimental results, PR-SCTP shows better performance than TCP in terms of average delay for message transfer. Furthermore, PR-SCTP exhibits less average packet loss than UDP. In both cases, PR-SCTP exploits priority properties of syslog messages during loss recovery.

Keywords: Syslog, PR-SCTP, performance evaluations, transport service.

1 Introduction

An important task of network, system or security administrators is the analysis of system or device generated log files. These files contain entries about specific events that occurred in a system or a network. Logs within a computer network are generated from various applications and operating systems on servers, clients or other networking devices. Log files are normally used for several functions, for example in optimizing system and network performance, recording the actions of users, and providing data important for the investigation of security related events. Generally, logging systems are used in large organizations where the number of computer systems can range in thousands. Such logging system must thus provide not only a high degree of reliability but also timeliness while transporting log messages within the network. Traditionally, log messages have been collected and compiled by using the syslog protocol [1]. This protocol allows a machine or device to send any event notification message across an IP network to a logger, commonly known as a syslog server.

The widely used syslog protocol does not specify any mechanism to provide reliability and is normally run over the unreliable transport protocol User Datagram Protocol (UDP) [2]. Hence, messages can be dropped unnoticed or may be maliciously intercepted and altered. A standard for reliable syslog transport has also been proposed [3]. Here reliability in log delivery is provided using the connection oriented transport protocol Transmission Control Protocol (TCP) [4]. However, TCP has its inherent difficulties to provide both reliability and timeliness for transporting log messages in a lossy or congested network scenario.

In this paper, we first identify several problems in the current syslog standard and then propose the use of PR-SCTP [5], an extension of Stream Control Transmission Protocol (SCTP) [6], as a candidate for the underlying transport service to ensure both reliable and timely delivery of syslog messages. PR-SCTP uses the message based abstraction of SCTP and chooses transport policy on a per message basis. In PR-SCTP, a sender application can apply a range of policies to define the retransmission limit for each transmitted message. In other words, the transport service can control the reliability level on a per message granularity. It guarantees maximum bandwidth usage for some prioritized messages with overall lower delay performance for all messages. We show a first set of evaluations of PR-SCTP performance in comparison with both TCP and UDP. The results show that in a congested network scenario, PR-SCTP can ensure improved delay performance while still providing reliable delivery of high priority messages.

The remainder of the paper is organized as follows. The syslog protocol and the related transport services and problems are described in further detail in section 2. In section 3, we describe several features of SCTP in relation to the syslog protocol and suggest PR-SCTP as a transport alternative for syslog. In section 4, we detail the experiment of the performance evaluation and discuss the results for comparative analysis. Finally, we give some concluding remarks and some indication of future work in section 5.

2 Background

In this section, we discuss the syslog protocol in general and then we mention the major challenges for its existing transport services.

2.1 Syslog

The syslog protocol was designed simply to transport event notification messages from an originator to a collector. In addition, a relay forwards messages, received from the originator or other relays to collectors or other relays. As the number of systems under surveillance increases, it becomes common to move to a centralized logging of syslog messages. This allows an administrator to monitor the log files at one location rather than trying to monitor the log files on a large number of systems.

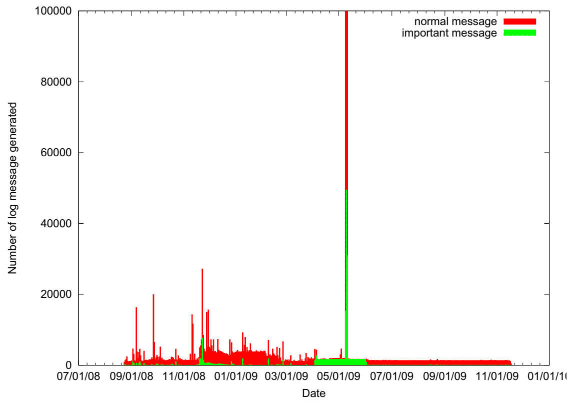


Fig. 1. Number of Syslog messages received at the server between August 2008 and November 2009

Syslog is intended to be very simple. Each syslog message has only three parts. The first part specifies the priority of a syslog message. It represents, as a numeric value, the facility and severity of the corresponding event that generated this message. To produce a priority value, the facility value of a message is first multiplied by 8 and then added to the severity value of the message. For example, a user level message ($facility = 1$) with a severity of alert ($severity = 1$) would produce a priority value of 9. Like in [7], messages with severity level of emergency, alert, critical, or error can be marked as important messages and the rest as normal messages. This marking can be used to prioritize the messages if needed. Figure 1 shows the statistics of syslog messages generated in a research network at the Computer Science Department at our university. For the sake of visualizing important messages, which are small in number, we restrict the number of normal messages up to one hundred thousand. It represents a common scenario where the number of normal messages surpasses the number of important messages by several orders of magnitude. The highest peak shows a sign of several mishaps in the network.

The second part of the message contains a timestamp along with the host name or IP address of the source of the log. The third part, finally, is the actual message content and is human readable. Any application can generate syslog compliant messages and send them across a network. Since, each different application and operating system was developed independently; there is little uniformity in the content of messages. Also, message transmission can happen without any explicit knowledge about a receiver, and on the other hand a syslog server or a collector cannot ask a specific device to generate logs. Due to all this simplicity, syslog has been widely deployed and supported. The specification of the syslog protocol was standardized in March 2009 in RFC 5424 [8]. The specification is based on a layered architecture in which message content is separated from message transport.

2.2 Transport Service for Syslog

Although it is practically simple to implement a syslog environment, it has several drawbacks while considering both reliability and timeliness in transporting log messages. This is due to the fact that most syslog implementations are based on UDP, as standardized in RFC 5426 [9]. Since, UDP is connectionless and unreliable, a collector does not send back any acknowledgement when a message is received. As a consequence, if a UDP packet is lost or damaged due to network congestion, resource unavailability, or interception or alteration by an intruder, then this will not be noticed. Moreover, UDP lacks a congestion control mechanism. If the network path is not over provisioned, with UDP, voluminous syslog traffic may aggravate the congestion level and harm the fairness of other flows sharing the same communication path.

Several syslog implementations such as syslog-ng [10] support TCP in addition to UDP. TCP is a reliable connection oriented protocol. Using flow control, sequence numbers, acknowledgements, and adaptive timers, TCP guarantees the reliable, in-order delivery of a stream of bytes. Besides, it provides congestion control. This mechanism limits an application from overwhelming the network. Considering all these advantages, IETF has standardized TCP as an alternative transport protocol for delivering syslog messages.

As pointed out by recent work [7], TCP has several shortcomings to provide the intended reliability syslog messages demand. First of all, TCP provides reliable delivery of the data transfer and in the same time it maintains strict order of data transmission. However, syslog messages may need reliable transmission but they are semantically independent. These messages do not demand sequence maintenance. So, partial ordering is highly desirable as the strict sequencing in TCP can introduce unnecessary delays to the overall message delivery services. This happens when a transmitted TCP segment is lost in the network and a subsequent segment arrives out of order. This subsequent segment is ceased at the transport layer until the lost segment is retransmitted and arrives at the receiver. This problem is called head of line (HOL) blocking [11].

Besides, the congestion control algorithm in TCP can cause additional delay if there is not enough space in the receiver window. In this case, a sender must wait until sufficient space is freed. This waiting time becomes quite undesirable and fatal when some messages need to be delivered instantly. In this sense, this strict reliability may not always be desirable [8]. The induced delay can block any syslog originator. According to [8], in Unix/Linux, a syslog originator or relay runs inside a high priority system process, that means if that process is blocked, the system may even face a deadlock situation to some extent.

Additionally, TCP handles application data as a byte stream. This is often an inconvenience for the transportation of syslog messages, which are mostly independent in nature. So, in this case each application must add special markings to make sure that the receiver can easily perceive the particular message boundary.

Lastly, an attacker may try to overwhelm the transport receiver by message flooding and cause denial of service (DoS) in the network. Hence, the transport protocol should provide features that minimize this type of threat [8]. TCP is known to be relatively vulnerable to this DoS attack although some solutions can be found in [12].

In addition to TCP, secure syslog transport over TLS is standardized in RFC 5425 [13]. Also, there is some ongoing work on specifying the use of DTLS for transportation of syslog messages [14].

As mentioned above, none of the currently standardized transports offer any possibility to prioritize syslog traffic beyond the choice of an unreliable or reliable transport. In [7], the authors proposed an application layer based prioritized retransmission mechanism to provide reliable delivery of syslog messages. However, their work indicates limited possibilities of prioritizations. In addition, they skipped many important details of flow and congestion control in their quite simplistic description. Also the added complexity in the application layer may be considered excessive for an application designer. Due to the limitations in the current solutions, we propose the use of PR-SCTP as a transport service for the syslog protocol.

3 PR-SCTP as a Transport Service for Syslog Messages

In this section, we discuss PR-SCTP in detail. Since it is an extension of SCTP and accommodates all its features, we start our discussion with SCTP. Similar to TCP and UDP, SCTP provides transport layer functions on top of a connectionless packet service such as IP. SCTP was primarily designed to overcome TCP's shortcomings as a telephony signaling transport in IP networks. Later it was noticed that SCTP is also useful in diverse application areas other than signaling transport [15]. SCTP is now a mature general purpose transport protocol with implementation on various platforms, such as AIX, FreeBSD, HP-UX, Linux, Mac OS X, and Solaris/OpenSolaris. A separate kernel driver provides SCTP functionalities in Windows XP and Vista. Like TCP, it provides a reliable, connection-oriented, and flow-controlled transport service to various applications. Several advanced features are available in SCTP, which can help to deal with the shortcomings of TCP.

SCTP supports multi-streaming which is used to ameliorate the HOL blocking effect that results from TCP's strict ordering requirement. An SCTP association consists of multiple individual streams, which are logical unidirectional paths between the sender and the receiver. Data sequencing is only preserved within any particular stream, but not between streams. In other words, if a particular transmission unit is lost within a single stream, only that stream is blocked until the sender retransmits the missing data. However, other streams can allow their data to flow from the sender to the receiver in the usual fashion. Thus, HOL blocking is limited to a particular stream. SCTP also supports unordered delivery [6] where messages can be instantly delivered to the higher layer in whatever order they arrive at the receiver. This is an essential feature for the transportation of time sensitive logs in syslog.

Besides, syslog originators send small (less than 1024 bytes) messages [1] to the syslog servers. The byte oriented nature of TCP is thus inconvenient to transport these messages. In comparison, SCTP is a message oriented protocol where whole messages can be delivered in its entirety as long as there is space in the receivers window. SCTP based syslog will not need any explicit message delimiters, which simplifies application level message parsing.

Moreover, as previously discussed, an attacker may try to flood a syslog server with continuous spoofed connection setup requests to consume all its resources and cause DoS in the network. SCTP uses a simple but powerful technique to eliminate the risk of SYN flooding as a DoS attack. It utilizes a four-way handshake with a cookie mechanism [6] during association initialization to avoid maintaining state information in the server side for incomplete sessions. This is an efficient yet simple technique to allow only legitimate users to access servers resources. An authentication extension of SCTP [16] supports further security enhancements. It adds functionality for sender authentication and message integrity.

TCP's firm reliability implementation may block a sender during transmission. This may for instance happen when a receiver becomes unable to receive any more messages, such as the corresponding application layer being slow. In this case, one solution may be to discard some of the messages where a syslog sender application would otherwise be blocked. We can think of several policies to do this job. According to RFC 5424 [8], when messages need to discarded, prioritization should be considered according to the severity value of a message. And if any message is abandoned, this should be reported to the receiver. PR-SCTP can be very useful to implement such an idea in the next version of syslog release.

In PR-SCTP, a sender can choose (re)transmission behavior on a per message basis. When a message is abandoned, the sender notifies its receiver to move the Cumulative ACK point forward, which simply tells the receiver not to expect that particular message from it any more. It serves two purposes; one is that the sender's transport layer can now drop some messages according to the particular partially reliable semantics used and secondly, this incident can also be notified to the receiver. Timed reliability is an example of this kind of service. In such case, an application defines a specific lifetime for every message. PR-SCTP tries to reliably transmit a message during its lifetime and upon expiration it simply abandons the message and notifies the receiver without considering whether the message is successfully transmitted or not. In this way, bandwidth wastage is reduced which can be used to transmit unexpired messages. Besides, under all circumstances, an unexpired message is a candidate for retransmission if it is lost. This is how PR-SCTP provides partial reliability.

The overall benefits of using PR-SCTP are summarized as follows:

- A single PR-SCTP association can carry both reliable and unreliable messages according to an application specific partial reliability policy,
- Compared to TCP, PR-SCTP avoids the HOL blocking problem,
- PR-SCTP allows message boundaries to be preserved during transportation,
- PR-SCTP specifies several security benefits through the four way handshake mechanism and the authentication extension.

Moreover, the PR-SCTP extension allows a receiver to be completely unaware about the sender's particular reliability policy. In this work, we adopt PR-SCTP as the transport service for syslog messages and evaluate its performance.

4 Performance Evaluation

In this section, we test and evaluate the performance of PR-SCTP in comparison to both UDP and TCP in an emulation based experiment setting.

4.1 Experiment Setup

For the investigation of PR-SCTP, we adopt a single bottleneck emulation based experiment setup. All the experiments are carried out in a local LAN test bed. Figure 2 shows the setup we use. Both end machines are configured with FreeBSD 8.1 PRERELEASE. Our custom made application implements all the necessary SCTP API and flags to enable the PR-SCTP extension. Traffic is routed through a middle box running also a FreeBSD kernel. The Dummynet traffic shaper [17] is used in this machine to introduce artificial and random packet loss, delays and bandwidth limitations in the network. End-to-end delay in the network is set to 40 ms and bandwidth is set to 10 Mbps. In this experiment, we only consider a single flow to understand the manifestation of various PR-SCTP properties under various network settings. In every experiment, the client machine initiates a connection and the server machine sends 10,000 syslog messages in response. Messages were generated according to a Poisson arrival process. Relating to our discussion in Section II, the server sends both important and normal messages. We use several distributions of important messages starting from 1% to 20%.

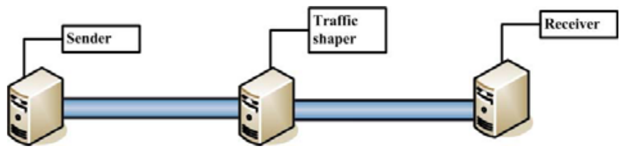


Fig. 2. Client-server based network used in the experiment

In these experiments, we use a timed reliability based PR-SCTP policy. Our intention is that, even under heavy congestion in the network, important messages should go through whereas normal messages have a smaller chance. We use 5000 ms TTL (time to live) for an important message and 100 ms TTL for a normal message. In a network with a 40 ms end-to-end delay and under a light congestion scenario, a 100 ms TTL is considered to be sufficient for a single retransmission opportunity in response to packet loss. We use the same application settings on top of various transport services such as TCP, UDP and PR-SCTP. We perform each experiment with 30 repetitions to allow for 95% confidence intervals. All the experiment related parameters are given in Table 1.

Table 1. Experiment related parameters

Parameter	Values
Message size:	250 Bytes
Number of msg sent:	10000
Send rate:	1 Mbps
Imp msg:	1%, 3%, 5%, 10%, 20%
Packet loss rate:	.5%, 1%, 1.5%, 2%, 3%, 4% and 5%
One way Delay:	40 ms
Queue size:	20 packets
Bandwidth:	10 Mbps(Up and Down)
Evaluated protocol:	TCP, UDP and PR-SCTP
TTL for imp msg:	5000 ms
TTL for norm msg:	100 ms
Number of repetitions:	30
Operating system:	FreeBSD 8.1 PRERELEASE

We measure several performance metrics such as average packet delay and average packet loss rate for both important (imp msg) and normal messages (norm msg).

4.2 Experimental Results

In this section, we present the results from our measurements. We start with delay performance of both important and normal messages under varying network conditions. The delay is measured as the time difference between the generation of a syslog message at the sender and the reception of the corresponding message at the receiver. It is crucial for the syslog protocol and the overall optimization of the system that messages go through the network in a timely manner. We evaluate the delay performance for PR-SCTP, TCP and UDP for various important message distributions. Before we go to the details of the evaluation, we will first have a look at the delay behavior in the network for each message for both PR-SCTP and TCP, as displayed in Figure 3. Figure 3 suggests that when the packet loss rate in the network is as high as 3%, TCP cannot keep up with the application demand and delay keeps rising indefinitely, whereas PR-SCTP shows quite good response with the application demand by prioritizing its traffic. This result motivates for further investigation of PR-SCTP properties.

Figure 4a shows the numerical results from several individual experiments with different transport services. Each data point in the graphs is an average result of 30 experimental runs. Along with each data point, a 95% confidence interval is also shown. Since PR-SCTP supports prioritization, we present the delay graph for both important and normal messages separately in this figure. PR-SCTP shows much faster response compared to that of TCP for both normal and important messages. It is worth noticing that graphs for both normal

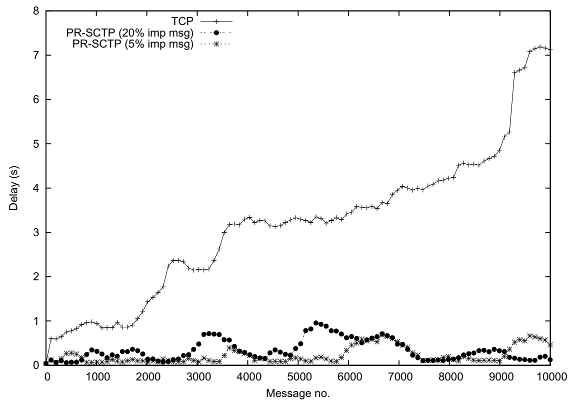
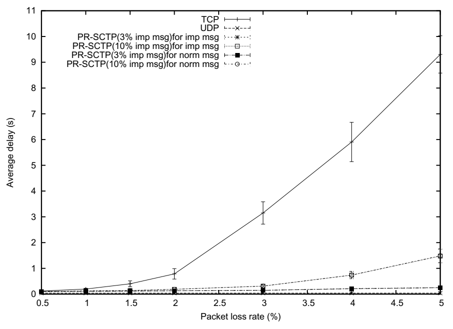


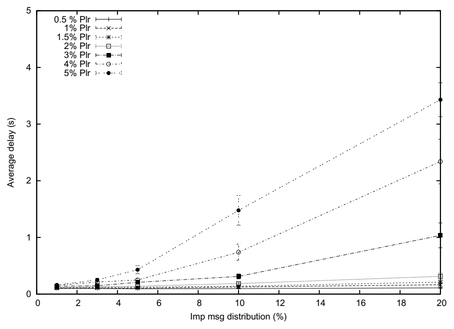
Fig. 3. Per message delay for 3% packet loss rate

and important messages coincide. The rational behind this is that only smaller portions of the total number of messages are important and treated specially in the event of loss. On the other hand, many of the normal messages are lost and are not recovered during heavy congestion. PR-SCTP abandons these messages and no delays affect the successive messages. Figure 4a also shows that when we have a small number of important messages, such as 3%, delay in PR-SCTP becomes almost as fast as UDP for transferring messages through the network. However, when the number of important messages becomes as high as 10%, average delay increases as the packet loss rate increases. In this case, a high packet loss rate also causes an increase in the number of retransmissions. Hence, as the number of important messages increases, PR-SCTP needs more time to deliver all the important messages in the event of heavy packet loss. This is particularly confirmed in the subsequent Figure 4b, where all the results of PR-SCTP for various distributions of important messages are presented in the same figure. Still PR-SCTP can sustain several orders of magnitude lower delay than TCP. According to Figure 3, TCP shows a stable trend of increasing delay because it does not have the benefit of prioritization and thus handles all message with equal importance. As a consequence, Figure 4a shows that it is substantially slower than PR-SCTP in various packet loss scenarios. Thus, the use of TCP for carrying important messages which needs special treatment in terms of both reliability and timeliness in a syslog environment may not be very wise.

Next, we evaluate average message loss rate for the transport services under evaluation. We do not expect that there will be any packet loss for TCP, but the result for TCP is included for the clarity of comparison. We use traces from both the sender and the receiver side to calculate the average loss rate. In this case, we divide the number of lost messages by the number of transmitted messages to derive the average loss rate. Figure 5a shows the loss rate of important messages for UDP, TCP and PR-SCTP with various distributions of important messages. This figure clearly shows that by giving high priority to an important message, PR-SCTP ensures TCP like reliability. Figure 5b, on the other hand, shows the

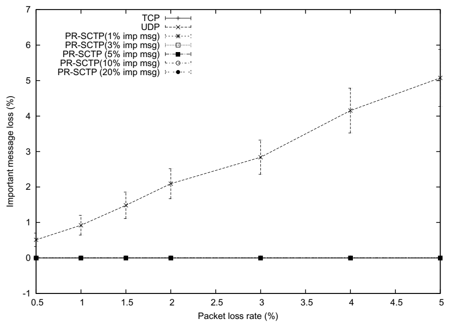


(a)

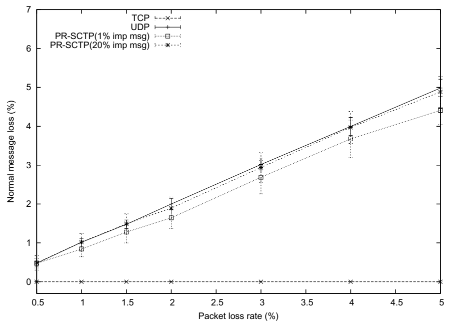


(b)

Fig. 4. (a) Average delay comparison between TCP, UDP and PR-SCTP(for both imp and normal message). (b) Average delay comparison for different loss rates and imp msg distributions for PR-SCTP.



(a)



(b)

Fig. 5. (a) Comparison of average loss percentage between TCP, UDP and PR-SCTP for important messages. (b) Comparison of average loss percentage between TCP, UDP and PR-SCTP for normal messages.

loss rate of normal messages for the same transport services. We avoid putting all the PR-SCTP results to make the figure less crowded and more readable. Since, the application level policy for PR-SCTP gives lower priority to a normal message, there is no statistically significant difference in the average loss rate between UDP and PR-SCTP for normal messages. However, still some difference is notable between the two results of PR-SCTP. When there is a large number of PR-SCTP messages with high priority, much of the bandwidth is spent for their retransmission during packet loss. However, this bandwidth is used to transmit some of the normal messages when we have a small number of high priority messages. This shows the bandwidth efficient behavior of PR-SCTP. According to both Figures 4 and 5, it is evident that PR-SCTP can provide both timeliness and reliability for the important messages in the evaluated scenario. For the lower percentage of important messages, PR-SCTP can be almost as fast as UDP and as reliable as TCP. In the syslog protocol, PR-SCTP can be very useful as a transport service where some log messages with high severity value enjoys a reliable service and some low priority messages are allowed to be dropped when congestion or blocking appears in the network. In our evaluation, we set the TTL value of important messages long enough so that even in highly congested network, important messages can get through. We intentionally do not set this value too high as that may cause a sender to block for a long time in some event such as buffer shortage at the receiver side. Since, the PR-SCTP policy is applied in the application layer, this type of optimization needs to be considered during application design.

5 Conclusion

In this paper, we have discussed several problems in both TCP and UDP, the currently used transport services, while transporting syslog messages. We propose PR-SCTP, an existing partially reliable extension of SCTP, as the transport service for both timely and reliable retrieval of high priority syslog messages at the receiver. By means of emulation, we have shown various performance aspects in a simple scenario. Our initial evaluation shows that PR-SCTP can provide lower average delay and can more intelligently use bandwidth for transporting messages while ensuring reliability for high priority messages. Although PR-SCTP drops some low priority messages, this saves time and capacity to effectively deliver more important messages. This is basically a trading of reliability against timing during times of congestion. Hence, PR-SCTP can be the right choice as a transport service for the syslog protocol.

Our initial observation of PR-SCTP properties seems useful and attractive. However, we believe that more experiments in a wide range of settings with multiple flows as well as RTT and capacity variations are needed to understand the true potential of this transport service. We intend to do all these evaluations in our future work.

PR-SCTP handles syslog messages on a priority basis, assigned at the senders application layer. This layer exploits the basic feature of syslog messages such as their associated severity parameters to prioritize messages. For the sake of simplicity in our initial work, we only use two kinds of priorities but multi-level priorities should be an obvious choice for a real implementation. In addition, our partial reliability policy is based on TTL values, which is only an option not a limitation. This policy can be implemented in many ways such as based on the number of sender retransmissions or based on the size limitation at the sender's transmission queue. In our future work, we also aim to evaluate these possibilities.

In conclusion, our motivation for applying prioritization while transmitting traffic in a syslog context seems appropriate. Furthermore, our current transport service evaluation confirms the effectiveness of PR-SCTP in terms of providing an attractive trade-off between reliability and timeliness for transporting syslog messages.

References

1. Lonvick, C.: The BSD Syslog Protocol. RFC 3164 (August 2001)
2. Postel, J.: User Datagram Protocol. RFC 768 (August 1980)
3. New, D., Rose, M.: Reliable Delivery for syslog. RFC 3195 (November 2001)
4. Postel, J.: Transmission Control Protocol. RFC 793 (September 1981)
5. Stewart, R., et al.: Stream Control Transmission Protocol (SCTP) Partial Reliability Extension. RFC 3758 (May 2004)
6. Stewart, R.: Stream Control Transmission Protocol. RFC 4960 (September 2007)
7. Tsunoda, H., et al.: A Prioritized Retransmission Mechanism for Reliable and Efficient Delivery of Syslog Messages. In: Proceedings of Seventh Annual Communication and Services Research Conference, Washington, DC, USA, pp. 158–165 (2009)
8. Gerhards, R., et al.: The syslog Protocol. RFC 5424 (March 2009)
9. Okmianski, A.: Transmission of Syslog Messages over UDP. RFC 5426 (March 2009)
10. Syslog New Generation (Syslog-ng), `http://www.balabit.com/network-security/syslog-ng/` (visited September 20, 2010)
11. Marco, G.D., et al.: SCTP as a transport for SIP: a case study. In: 7th World Multiconference on Systemics, Cybernetics and Informatics (SCI), Orlando, FL, USA, July 2003, pp. 284–289 (2003)
12. Eddy, W.: TCP SYN Flooding Attacks and Common Mitigations. RFC 4987 (August 2007)
13. Miao, F., et al.: Transport Layer Security (TLS) Transport Mapping for Syslog. RFC 5425 (March 2009)
14. Salowey, J., et al.: Datagram Transport Layer Security (DTLS) Transport Mapping for Syslog, draft-ietf-syslog-dtls-06.txt(work in progress) (expires: January 9, 2011)
15. Fu, S., et al.: SCTP: State of the art in research, products, and technical challenges. Communications Magazine, IEEE 42(4), 64–76 (2004)
16. Tuxen, M., et al.: Authenticated Chunks for the Stream Control Transmission Protocol (SCTP). RFC 4895 (August 2007)
17. Rizzo, L.: Dummynet: A simple approach to the evaluation of network protocols. ACM SIGCOMM Computer Communication Review 27(1), 31–41 (1997)