Integration of 3G Protocols into the Linux Kernel to Enable the Use of Generic Bearers

Nils Aschenbruck, Matthias Frank, Wolfgang Hansmann,
Peter Martini, Christoph Scholz, and Jens Tölle

University of Bonn - Institute of Computer Science IV
Roemerstr. 164, 53117 Bonn, Germany
{aschenbruck,matthew,hansmann,martini,scholz,toelle}@cs.uni-bonn.de

Abstract. The General Packet Radio Service (GPRS) is widely deployed in second and third generation mobile cellular networks. Special benefits of GPRS are mobility management as well as support of authentication, authorization, and accounting (AAA). However, the data rates of GPRS are low and the price is high, compared to wired networks or Wireless LAN. Furthermore, Wireless LAN hotspots are starting to sprout. While Wireless LAN in fact offers high data rates, it lacks a standard for billing and roaming.

One solution is to combine both technologies, GPRS and Wireless LAN. The resulting system would offer Wireless LAN's higher bandwidth, while keeping GPRS' sophisticated billing and roaming support. Wireless inter-system roaming supporting seamless handovers could be a benefit beyond.

In this paper we present the first step of combining Wireless LAN and GPRS, by integrating the GPRS protocol stack into the Linux kernel. In addition to the integration we present evaluation results and improvements, concerning the choice of GPRS parameters.

1 Introduction

In these days there is a demand to use the services of the Internet like Word Wide Web, E-Mail and file transfer everywhere. To satisfy these demands the General Packet Radio Service (GPRS) is widely deployed within second and third generation mobile networks. Special benefits are mobility management and authentication, authorization, and accounting (AAA) support. A handicap of GPRS is its limited maximum data-rate of 170kbit/s, compared to wired networks like Ethernet or ADSL, but also other wireless technologies like Wireless LAN and Bluetooth.

Especially with newer wireless technologies the frontier to higher data rates in the wireless area is crossed. These technologies present more bandwidth, compared to the cellular networks, but lack an integrated mobility management and billing system (AAA). Especially the latter will be decisive for the further success of the wireless technologies.

The idea is to use GPRS over faster wireless bearers (see figure 1) like Wireless LAN, to support bit-rates up to 54Mbit/s. This combination provides the

Z. Mammeri and P. Lorenz (Eds.): HSNMC 2004, LNCS 3079, pp. 533–544, 2004.

opportunity of using authentication, authorization and accounting (AAA) of 3G wireless packet data service over a faster bearer. The figure shows a general overview of the architecture. At the moment we focus on the implementation of the mobile station, so the network side implementation is strongly simplified, containing a proprietary SGSN and GGSN implementation.

A use case of this architecture might be for example Internet cafes, or hot spots in general, in which such a combination of 3G with Wireless LAN is used. Furthermore, it will be possible to realize roaming between different bearers (inter-system roaming). If available, Wireless LAN will be used; when there is no Wireless LAN available, "Standard-GPRS" of the second or third generation mobile networks can be used.

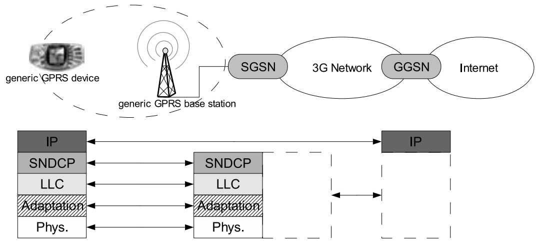

Fig. 1. General overview of the architecture

According to [1] roaming between Wireless LAN and the 3G network is an important service requirement. This ETSI standard mentions different interworking scenarios, from *common billing and customer support* (Scenario 1) over *seamless services* (interworking scenario 5) up to *access to 3GPP CS services* (Scenario 6).

In the literature there are several studies on combining Wireless LAN and GPRS. Two architectures are discussed, the loose and the tight coupling architecture ([2]). Both architectures support mobile terminals capable of accessing Wireless LAN as well as GPRS radio resources. The loosely coupled architecture connects Wireless LAN to the operator's IP network. In contrast to this, the tight coupling architecture realizes GPRS over Wireless LAN for a direct integration into the 3GPP network. Within this approach Wireless LAN has to be integrated deeply into the GPRS protocol stack.

Other approaches, such as [3], [4], and [5], rely on the loosely coupled architecture. In these papers the challenge of roaming that supports seamless services is realized by using new ideas based on Mobile IP. In contrast to this, our approach is based on the tight coupling architecture and within this enables *access to 3GPP CS services* (ETSI interworking scenario 6). In this architecture the *seamless services support* (ETSI interworking scenario 5), that are by the way the new "dimension in evolving the 3GPP-WLAN interworking system"[4], is integrated anyway, because of the combination with the GPRS mobility management and AAA. The disadvantages of the tight coupling architecture, mentioned in [3], are integrational and not conceptual.

To realize a combination of GPRS with other bearers, for example Wireless LAN, an integration of the GPRS protocol stack into the Linux kernel is our first task. The stack has to be integrated into the Linux kernel network architecture.

In general, an integration of GPRS in an operating system has the benefit to use GPRS over generic network technologies (bearers) like Wireless LAN, Bluetooth, Ethernet or anything that is supported by Linux.

The first task performed was the architectural design (see section 2). The GPRS protocol stack, according to the ETSI standards ([6], [7], [8], [9], [10]), was adapted for the integration into the Linux kernel. Finally, the stack was integrated into the Linux network architecture (see section 3). Having implemented the GPRS protocol stack into the Linux kernel, an evaluation was performed (see section 4).

2 Architectural Design

In this section the designed architecture for the mobile station side is described. In this context several general restrictions were made:

- The implementation, at least in the first release, is restricted to a GPRS class C device; there is no circuit switched speech service.
- SMS (Short Message Service Support) and SS (Supplementary Services Support) are omitted.
- Quality of service, data compression, and ciphering support are not implemented yet, but provision for these features is already made.
- We focus on the mobile station side.

The general overview in figure 1 results from these restrictions. We focused on the mobile station side. Thus, we mainly implemented the interface between mobile station and SGSN. Furthermore, there is no base station implementation, yet. It was integrated in the SGSN (see figure 1).

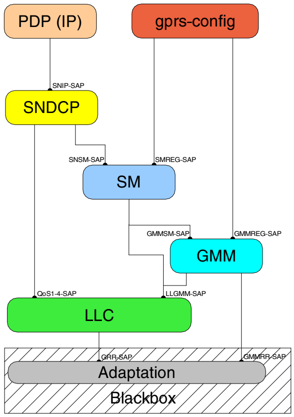

Fig. 2. Adapted GPRS Protocol Stack

The architectural design for the mobile station is presented in figure 2. According to the restrictions, Session Management (SM) is the only layer of the Connection Management that is seen as essential to realize. In addition to the protocol stack layers a control instance *gprs-config* is added in the figure. This instance can be a control program or daemon that takes care of attaching to the network and PDP context management.

The aim is to use GPRS over generic bearers. Below Logical Link Control (LLC) there are bearer specific layers. Thus, LLC is the lowest layer of the GPRS that is realized in our architecture.

There is an adaptation layer below LLC, that adapts to GPRS GSM hardware (device driver) or to a generic bearer, like Ethernet or Wireless LAN. Interesting aspects for example handover strategies between different bearers will be considered in the future (cf. section 5). At the moment there is a simple, proprietary

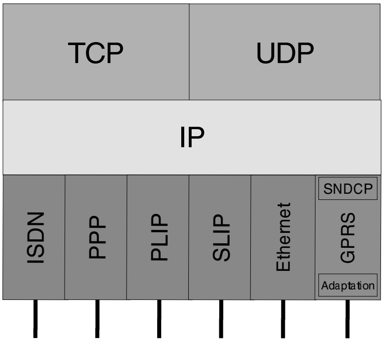

Fig. 3. Position of the GPRS stack in the structure of Linux network architecture

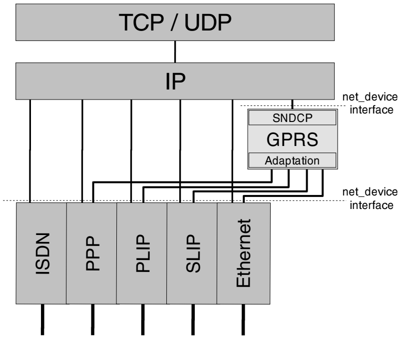

Fig. 4. Position of the GPRS stack in the structure of Linux network architecture layers using a generic bearer

protocol that does the basic adaptation work, containing address conversion, protocol identification and segmentation, without any further extensions.

3 Realization in Linux

This section describes the practical integration of the protocol stack into the Linux kernel network architecture on the mobile station side.

The relevant layers for the implementation are realized in one kernel module each (see also section 2). So there are five modules, one for each protocol layer SNDCP, SM, GMM, LLC and one for the adaptation to a generic bearer or the driver.

In the Linux kernel the bottom interface of IP is the net_device interface (see also [11]). Linux regards the GPRS protocol stack as a logical network device, so the GPRS protocol stack should be integrated by implementing the net_device interface. The structural position in the network architecture can also be seen in figure 3.

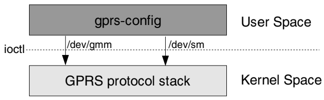

Fig. 5. User space control using character devices

At the bottom of the protocol stack there is the adaptation layer, which can be a real GSM or UMTS service. In this case, a device driver to the radio resource in the GPRS hardware has to be realized. The GPRS stack may also be used over generic bearers, such as Ethernet or Wireless LAN. If so, GPRS can also be a user of layers that are below the net_device interface (see figure 4). Thus, the adaptation layer has to access the net_device interface as a user, because it should abstract from the kind of bearer. This has the advantage that any network device that is supported by Linux can be used as bearer below the GPRS protocol stack.

In summary one can say, that on the one hand GPRS is positioned below IP and therefore implementing the net_device interface of the Linux network architecture (see figure 3) and on the other hand may also itself use the net_device interface when for example Ethernet is used as a bearer for GPRS (see figure 4).

The GPRS protocol stack supports unacknowledged as well as acknowledged data transmission. The transmission mode is one parameter that can be set for a connection. GPRS supports several PDP-Contexts to be able to have several connections with different parameters. The individual PDP-Contexts are realized as different network devices. A network device is created dynamically and set up, according to the activation of a PDP-Context.

The interface to the control instance in the user space *gprs-config* is realized via ioctl calls on dedicated device files. Therefore, our architecture provides two character devices, one for GMM and one for SM (see figure 5). This approach is similar to the realization of the Linux kernel tunnel device ([12]).

The Linux kernel network architecture supports flow control in an indirect way. There is no explicit congestion notification mechanism between the layers, but there is a memory limit for each socket [13]. A new socket buffer for a socket can only be allocated, when the memory of the used socket buffers is less than the maximum memory count. The standard value is 64kbyte (/proc/sys/net/core/-wmem_default).

In acknowledged mode LLC has a transmission window with a standard maximum count of 16 packets [7]. To support flow control the transmit window size has to be harmonized with the maximum socket memory count. The LLC standard [7] allows a transmit window size up to 255. For our realization we assume a value of 255. This value allows more than five connections at the same time sending at full load.

$$\frac{|\text{transmit window}| \cdot |\text{standard LLC MTU}|}{|\text{max. socket memory count}|} = \frac{255 \cdot 1503}{65535} \simeq 5.8$$

4 Evaluation

We evaluated our implementation by Round Trip Time (RTT) and throughput measurements. For the evaluation the testbed in figure 6 was used. In public GPRS networks the measurements that use IP are only possible between mobile station and GGSN. However, we started with the implementation of the mobile station. Thus, the network side is strongly simplified. Using this simplified network implementation, it is possible to perform IP measurements between Mobile Station and SGSN. This has the advantage of not having effected the measurements by the backbone network between SGSN and GGSN.

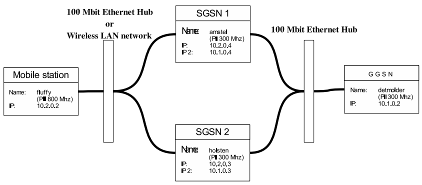

Fig. 6. Testbed for evaluation

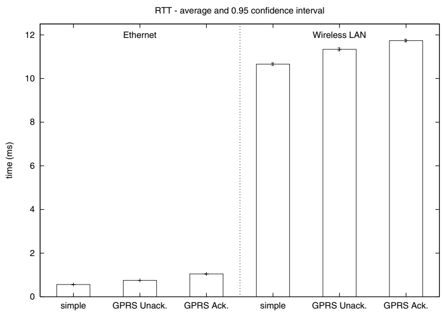

Fig. 7. RTT results with standard parameters (1600 byte packet size)

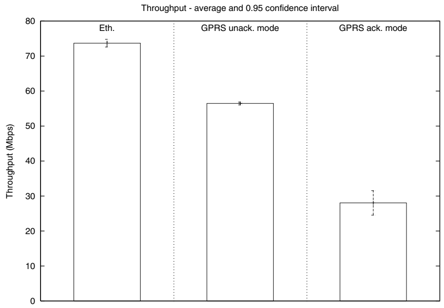

Fig. 8. Throughput results with standard parameters for Ethernet

The RTT was measured with the *ping* tool. The throughput was measured by using *netperf* [14] tool, a TCP bulk traffic generator. We evaluated both GPRS over Wireless LAN (11 Mbit/s) and GPRS over Ethernet (100Mbit/s, full duplex). Furthermore, during the evaluation GPRS data transmission was done for acknowledged as well as for unacknowledged traffic.

4.1 Results

The results of the evaluation using standard parameters are shown in figure 7 and 8. The RTT measurements show an overhead of up to 140% compared to the simple bearers (Ethernet respectively Wireless LAN) not using the GPRS protocol stack. The throughput measurements show that the throughput when using GPRS over the bearers is only 38% of the throughput using the simple bearer.

Both measurement results are caused by non optimized parameters. For example the packet size as well as several other parameters should be adjusted to the new context. The next section will explain the more sophisticated choice of the parameters.

4.2 Improvements

According to the ETSI standards [7] LLC supports a standard maximal transfer unit size of 500 bytes for unacknowledged and 1503 bytes for acknowledged transfer. Other values may be negotiated between mobile station and network. If the packets are too large, SNDCP will apply segmentation.

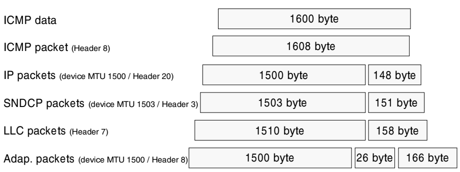

Fig. 9. Segmentation of a 1600 byte ICMP packet using standard values

When using GPRS in acknowledged mode over a generic bearer (e.g. Ethernet) an MTU of 1503 bytes is not a good choice. A device that implements the net_device interface has a device MTU of 1500 bytes by default, so all packets that are larger than 1500 bytes are segmented. But when using GPRS in acknowledged mode with default parameters it is segmented twice; at first by IP to fit to the standard MTU size of 1500 bytes, then again by the adaptation layer caused by the additional headers of the GPRS protocol stack. Thus, a 1600 byte packet for example is segmented into three packets: one of 1500 bytes and two small ones (26 and 166 bytes), see figure 9. Especially for Wireless LAN small fragments impact performance badly. Wireless LAN uses CSMA/CA; the smaller the packets are, the longer the average time a packet has to wait. To prevent segmentation in the adaptation layer the LLC MTU should be set to 1485 instead of 1503:

$$\text{max. MTU size bearer} - (\text{Header size LLC ack} + \text{Header size Adaptation layer})$$
$$= \text{optimal LLC MTU size} = 1500 - (7 + 8) = 1485$$

Setting the LLC MTU to 1485, there is no more segmentation necessary below SNDCP. A similar calculation for the acknowledged mode results in an LLC MTU of 1486.

The standard device MTU for the GPRS devices should also be improved. There are two possibilities. A large value may be chosen for the device MTU, so that SNDCP fragments large IP packets to up to 16 SNDCP packets. The IP header is saved in this case. The maximum in the acknowledged mode is 23712.

$$(\text{optimal LLC MTU size} - \text{Header size SNDCP ack})$$
$$\cdot \text{max. SNDCP fragment count}$$
$$= (1485 - 3) \cdot 16 = 23712$$

The alternative is to let IP do the whole fragmentation work by setting the device MTU to 1482.

$$\text{optimal LLC MTU size} - \text{Header size SNDCP ack} = 1485 - 3 = 1482$$

Using 1482 as device MTU, fragmentation work will be done by IP and there is no further segmentation in the GPRS protocol stack.

The acknowledged mode may further be improved. In standard mode each acknowledgement is sent in a separate packet. The ETSI LLC standard [7] supports an extension to piggyback the acknowledgements. According to it the acknowledgments of LLC may be sent as part of the next LLC data packet.

The RTT and throughput measurements have been repeated with the new LLC MTU values. The results are shown in figure 10 and 11. The RTT results for unacknowledged mode as well as throughput results for Wireless LAN have been omitted, but are comparable. Instead of these, the layers which do the fragmentation are outlined at the x-axis. The overhead of the RTT is reduced. The throughput for acknowledged mode has improved to 86% of simple Ethernet. Saving the IP overhead has advantages as long as there is no packet loss.

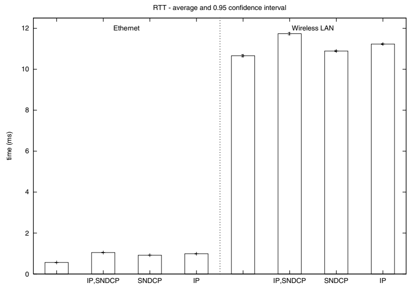

Fig. 10. RTT results (only acknowledged traffic) with new parameters; x-axis shows layer that does fragmentation (nothing means simple bearer without GPRS)

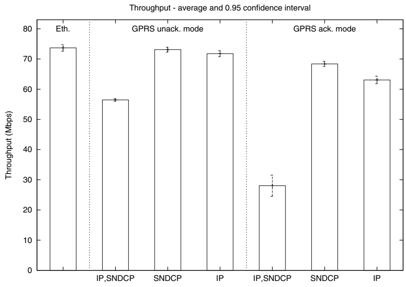

Fig. 11. Throughput results with new parameters; x-axis shows layer that does fragmentation (nothing means simple bearer without GPRS)

4.3 Losing Packets

Packet loss was simulated by a logical network device that randomly discards a fixed percentage of packets. Using GPRS in acknowledged mode, packet loss is noticed, and the packets are retransmitted. Thus, the acknowledged mode compensates the packet losses.

The results of the RTT measurements using the lossy device were as expected. The unacknowledged mode loses the packets, the acknowledged mode does not lose them, whereas the RTT depends on the retransmission timer. The GPRS specification does not support dynamic retransmission timers. This is a problem because the fixed timer has to be chosen before data transmission. The retransmission timer in LLC is by default set to five seconds; according to the ETSI standards [7] the smallest value allowed to be chosen is 0.1 second. This is too much time when having RTTs in Ethernet that are smaller than ten milliseconds.

The results of the TCP measurements using the lossy device with an average amount of losses of five percent are shown in figure 12. At the x-axis the layers which do the fragmentation and the retransmission timeout values are outlined. The throughput of pure Ethernet and the unacknowledged mode are both about 14Mbit/s, as long as in the unacknowledged mode each fragment is sent with an IP header. The idea of splitting large IP packets into several SNDCP packets is not successful, in case of packet loss. The whole IP packet has to be retransmitted on loss of any of the 16 fragments. The results for pure Ethernet and unacknowledged traffic are better than expected. The standard Linux implementation of TCP supports selective acknowledgments (SACK). This means, when the receiver misses a packet, a SACK is sent back to the sender. If the sender receives a SACK, it will retransmit the packet at once, without waiting for timer expiry. Anyway, TCP lowers its transmission rate in case of packet loss, because it assumes that data loss occurs due to congestion.

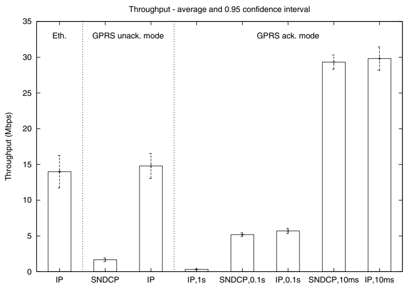

Fig. 12. TCP throughput using the lossy bearer; x-axis shows layer that does fragmentation (nothing means simple bearer without GPRS) and Retransmission Timer for acknowledged mode

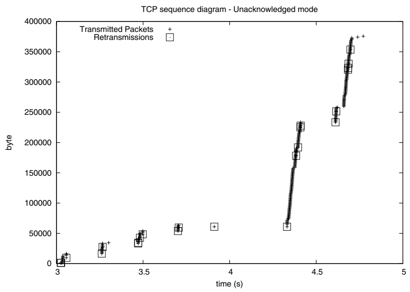

Fig. 13. TCP Time Sequence diagrams for unacknowledged mode

The throughput in the acknowledged mode using standard compliant values for the retransmission timer is not satisfying. When using a retransmission timer of ten milliseconds, results of 30Mbit/s are achieved. This is more than twice of the results using unacknowledged mode or pure Ethernet. To get a better impression of the reasons causing the different results we present some time sequence diagrams of TCP and LLC.

When using the simple bearer or the GPRS device in unacknowledged mode, TCP performs frequent retransmission, see figure 13. TCP has to take care of the reliable data transmission on its own. Several delays can be seen in the time sequence diagram because of TCP retransmission timeouts. The whole data transmission takes about two seconds.

When using the acknowledged mode with a retransmission timer of one second, there are a lot of TCP retransmissions, see figure 14. The LLC retransmission timer is too large, so that the two acknowledgment schemes interfere each other, because LLC does not prevent TCP retransmissions, but interferes TCP retransmissions by doing its own ones. Both time sequence diagrams contain a lot of retransmissions. In the LLC diagram there are always blocks of retransmission because LLC uses the "go back n" retransmission scheme in the current implementation. When there is a retransmission timeout all packets that were sent after the one that was lost are retransmitted as well. Compared to the unacknowledged mode a further reason for the reduced throughput is that when using acknowledged mode no TCP SACKs are possible. LLC guarantees that the packets are transmitted in the correct order, so TCP does not recognize a missing packet, because LLC buffers out of order packets. The TCP retransmissions are caused by timer expiration.

When the LLC retransmission timer is set to ten milliseconds the cooperation behavior of the two acknowledgment schemes works better, see figure 15. After some transient time the dynamic retransmission timer of TCP reaches a steady

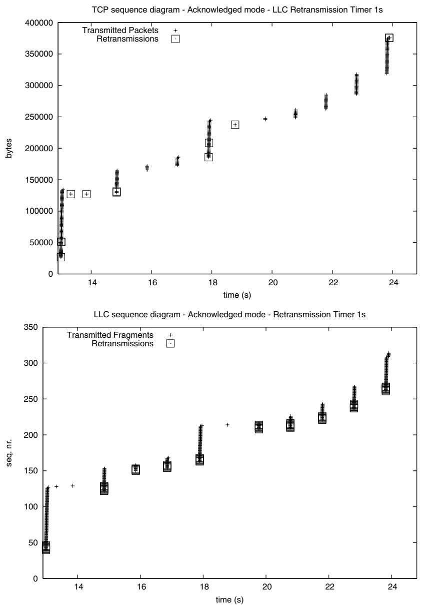

Fig. 14. TCP and LLC Time Sequence diagrams for acknowledged mode with retransmission of 1s

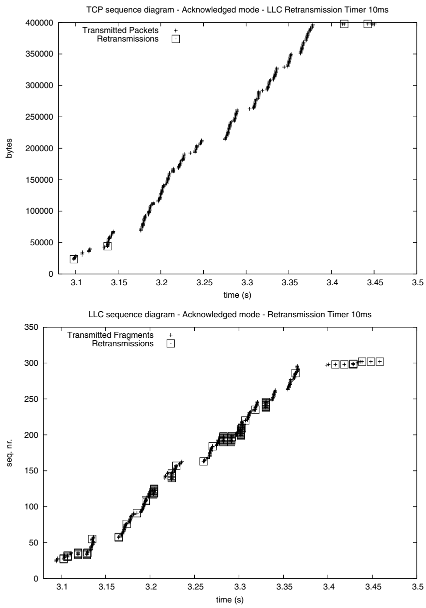

Fig. 15. TCP and LLC Time Sequence diagrams for acknowledged mode with retransmission of 10ms

state. After this, there are no more retransmissions necessary for TCP during the data transmission. The whole data transmission only takes less than a half second. One can realize the differences in transmission times, when considering the different x-axes scalings in the figures. This explains the difference in the throughput measurements, presented in figure 12. LLC has to do the retransmissions, but does it in time, caused by the smaller LLC retransmission timer. The retransmissions cause little jumps in the time sequence diagram of TCP. Again there are retransmission bursts caused by “go back n”.

In the current implementation the only possibility for LLC to recognize packet loss is the expiry of its retransmission timer. The GPRS standard [7] also describes SACKs for LLC. This has not been implemented yet, but may be done in the future. Further improvements may be reached by using a different retransmission scheme than “go back n”.

5 Conclusion and Future Work

With our implementation of the GPRS protocol stack in Linux it is possible to use 3G over generic bearers. Our evaluation shows that it is not enough just to implement the ETSI standard and use it with its standard parameters. When using bearers like Wireless LAN and Ethernet these parameters result in suboptimal performance. Parameters like MTU and retransmission timer have to be set to different, non-standard-conform values to achieve a smaller RTT and higher throughput.

In the future we want to enhance our architecture by removing some of the limitations (see section 2). One idea is to separate our combination of SGSN and base transceiver station. In this context, the adaptation layer has to be redesigned. Furthermore, it allows to perform *cell updates* between different access points respectively bearers. Therefore, it is necessary to compare the link qualities of different bearers, for example Wireless LAN and Bluetooth. In the literature there are some approaches (e.g. [15] and [16]) that deal with the evaluation of the different bearers. These approaches can be seen as a base for our future work.

In summary, the aim is to extend the implementation particularly with regard to the lower layers and handovers between different bearers, but also data compression, and ciphering support.

References

1. ETSI: Universal Mobile Telecommunications System (UMTS); Feasibility study on 3GPP system to Wireless Local Area Network (WLAN) interworking. 3GPP TS 22.934 version 6.2.0 Release 6 (09-2003)
2. Salkintzis, A., Fors, C., Pazhyannur, R.: WLAN-GPRS Integration for Next-Genereation Mobile Data Networks. IEEE Wireless Communication (10-2002)
3. Buddhikot, M., Chandranmenon, G., Han, S., Lee, Y., Miller, S., Salgarelli, L.: Design and Implementation of WLAN/CDMA2000 Interworking Architecture. IEEE Communication Magazine (11-2003)
4. Ahmavaara, K., Haverinen, H., Pichna, R.: Interworking Architecture Between 3GPP and WLAN Systems. IEEE Communication Magazine (11-2003)
5. Floroiu, J., Ruppelt, R., Sisalem, D., Stephanopoli, J.: Seamless Handover in Terrestrial Radio Access Networks: A Case Study. IEEE Communication Magazine (11-2003)
6. ETSI: Technical Specification Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); General Packet Radio Service (GPRS); Service description; Stage 1. 3GPP TS 22.060 version 5.2.0 Release 5 (06-2002)
7. ETSI: Digital cellular telecommunications system (Phase 2+); Mobile Staion (MS) - Serving GPRS Support Node (MS-SGSN); Logical Link Control (LLC) Layer Specification. 3GPP TS 44.064 version 5.1.0 Release 5 (03-2002)
8. ETSI: Digital cellular telecommunications system (Phase 2+); Mobile Staion (MS) - Serving GPRS Support Node (SGSN); Subnetwork Dependent Convergence Protocol (SNDCP). 3GPP TS 44.065 version 5.0.0 Release 5 (06-2002)
9. ETSI: Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); Mobile radio interface signalling layer 3; General Aspects. 3GPP TS 24.007 version 5.1.0 Release 5 (09-2002)
10. ETSI: Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS);Mobile radio interface signalling layer 3 specification; Core network protocols; stage 3. 3GPP TS 24.008 version 5.6.0 Release 5 (12-2002)

11. Wehrle, K., Pählke, F., Ritter, H., Müller, D., Bechler, M.: Linux Netzwerkarchitektur - Design und Implementierung von Netzwerkprotokollen im Linux-Kern - zu Kern 2.4. Addison-Wesley (2002)
12. Documentation, L.K.: Universal TUN/TAP device driver. Documentation: [src]/Documentation/networking/tuntap.txt, Sources: [src]/drivers/net/tun.c (2003)
13. Eisen, H.: The sk_tunnel kernel framework - flow control. http://www.baty.hanse.de/linux-x25/sk_tunnel/linux-kongress/ (2000)
14. Jones, R.: The Public Netperf Homepage. http://www.netperf.org/netperf/NetperfPage.html (2003)
15. Matusz, P., Machan, P., Wozniak, J.: Analysis of Profitability of Inter-system Handovers between IEEE 802.11b and UMTS. 28th Conference on Local Computer Networks (LCN 2003) (2003) pp. 210–217
16. Albrecht, M., Frank, M., Martini, P., Schetelig, M., Vilavaara, A., Wenzel, A.: IP Services over Bluetooth: Leading the Way to a New Mobility. Proc. of the 24th Annual Conference on Local Computer Networks (LCN 1999) (1999)