White Paper

radisys®

SCTP Fast Path Optimization for 3G/LTE Networks

By: Deepak Wadhwa, Chief Architect

Overview

The exploding growth of the internet and associated services over the last decade is fueling the need for ever increasing bandwidth. The number of intelligent handheld devices is growing exponentially and in turn the demand for high-speed data services while on the move is increasing tremendously. Current 3rd Generation (3G) mobile technology is able to cope with the huge increase in demand to some extent but is not suitable for satisfy the needs completely.

CONTENTS

Long Term Evolution (LTE), a whole new “4th Generation” mobile radio access network (RAN) technology, promises higher data rates—100Mbps in the downlink and 50Mbps in the uplink in LTE’s first phase—and will reduce the data and control plane latency with an aim at quenching the insatiable thirst for high-speed mobile data access. Additionally, LTE is designed to support interoperability with existing mobile network technologies such as GSM, GPRS and UMTS. LTE also supports scalable bandwidth, from 1.25MHz to 20MHz, which allows operators significant deployment flexibility and also can allow for more rapid roll-out due to spectrum flexibility. All of these features make LTE a very attractive technology for operators as well as subscribers, and many dozens of operators worldwide have committed to LTE roll-outs in the next two to five years.

All is not rosy, however, and the performance demands of LTE technology is leading to increasing signaling and data requirements which impose additional demand on the network. In this paper, we look at the need for and methods of optimizing the Stream Control Transmission Protocol (SCTP) to handle increased signaling loads in LTE and 3G networks.

Network Architecture

Figure 1 illustrates the LTE network architecture with the various interfaces between the network elements; GERAN and UTRAN networks are shown as well for completeness.

The functions of the various network elements are as follows:

- *eNodeB:* the base station in the LTE network, it provides Radio Resource Management functions, IP header compression, encryption of user data streams, selection of an MME, routing of user plane data to S-GW, scheduling and transmission of paging messages.

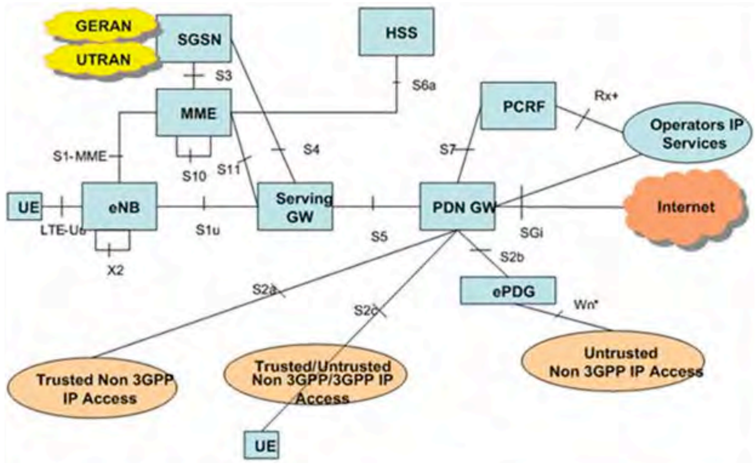

Figure 1. *LTE Network Architecture*

- *Mobility Management Entity (MME):* the primary control plane element in the LTE core network, also known as the Evolved Packet Core (EPC), the MME provides NAS signalling (Evolved Mobility Management (eMM), Evolved Session Management (ESM)) and security, AS security, tracking area list management, PDN GW and S-GW selection, handovers (intra- and inter-LTE), authentication and bearer management.
- *Serving Gateway (S-GW):* the primary data plane element in the EPC, the S-GW provides the local mobility anchor point for inter-eNodeB handover, downlink packet buffering and initiation of network-triggered service requests, lawful interception, accounting on user and QCI granularity, and Uplink (UL)/Downlink (DL) charging per User Equipment (UE).
- *Packet Data Network Gateway (P-GW):* acts as a breakout and data service gateway in the EPC; provides UE IP address allocation, packet filtering and PDN connectivity, UL and DL service-level charging, gating and rate enforcement.

Figure 2 highlights the control protocol stacks of these LTE network elements.

Stream Control Transmission Protocol (SCTP) is the key protocol utilized as transport for all signaling between the RAN and EPC. SCTP is also utilized as transport for the Diameter protocol in LTE. In the EPC, Diameter is used for communication between core network elements and the Home Subscriber Server (HSS) as well as the policy control and management infrastructure. Additionally, in 3G Femtocell (i.e., Home NodeB) deployments, SCTP is used as the transport protocol between femtocells and femtocell gateways as well as from femtocell gateways to the 3G core network.

There is a general move toward the deployment of small cell environments (e.g., femtocells and picocells) due to the ability to boost overall network coverage and capacity while increasing the quality of experience (QoE) for subscribers. Small cells are being deployed today as an overlay to existing 3G networks and are seen as a critical element of the LTE RAN. However, deploying small cells requires a huge increase in SCTP associations—way more than typically required in the network. In fact, this approach is pushing SCTP to the edge of its designated purpose and in general there is a lack of implementations which are adequate to meet these new requirements. Additionally, small cells increase the overall signaling load due to the signaling required to manage small cells, not to mention the additional signaling which results from frequent handovers.

From a design perspective, we have estimated that an MME or 3G Femtocell Gateway intended to service small cells must support the following key requirements:

- At least 1,000,000 SCTP packets per second
- At least 16,000 SCTP associations
- A high rate of association establishment and teardown

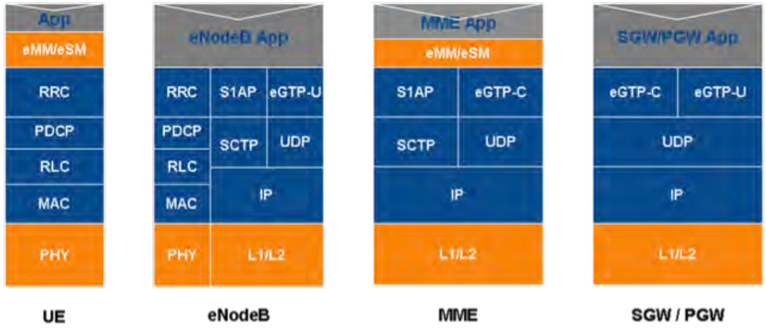

Figure 2. *LTE Control Plane Protocol Stacks*

Challenges in Current SCTP Implementations

The majority of current SCTP implementations in the marketplace are based in either the user space or kernel space running under some flavor of the Linux or Solaris Operating Systems (OS). These implementations have been adequate for traditional use of SCTP in SIGTRAN (i.e., SS7 over IP) networks for SCTP's original purpose: to carry SS7 signaling over IP networks.

However, in order to scale the performance of SCTP, the SCTP implementation needs to be able to take advantage of the new generation of multi-core processors. Although there are a few implementations of SCTP taking advantage of multi-core, they suffer from inefficiencies caused due to scheduling overheads, locking between threads and inefficient communication between threads running under traditional operating system environments. In addition, there is an added penalty for the additional buffer copies between SCTP and the SCTP application programming interfaces.

This paper describes how to achieve SCTP performance improvements on multi-core processors and provides a comparison against SCTP executing on other processing platforms.

SCTP Optimization in the Fast Path

This section of the paper explains an approach to optimizing SCTP by porting to the fast path on a next generation high performance packet processor, in this case the NetLogic XLR.

Figure 3 illustrates the overall architecture of the NetLogic XLR processor, which provides a few key features that are very useful for optimizing packet processing functions:

- Multiple hardware threads
- Security Acceleration Engine for IPsec
- Fast Messaging Network for communication between threads
- Support for fast path software which provides software development for a lightweight operating system, NetOS; this eliminates some of the overheads as mentioned earlier

In addition, SCTP fast path processing requires IP Layer 3 and IP security functions. In this case the IP L3 is provided by 6WINDGATE SDS software ported to and optimized for the XLR processor.

Please refer to *http://www.6wind.com/6WINDGate-software.html* for details on 6WIND software capabilities.

Figure 4 illustrates the Trillium fast path architecture which is an extension to the Trillium Advanced Portability Architecture (TAPA). TAPA has been the architecture of Trillium signaling software solutions for more than 20 years.

The fast path architecture splits the implementation of protocols into two parts. The part shown on the left of the diagram implements the control plane and management functions of the protocol, often referred to as the slow path. In general, the non-message or packet processing elements (i.e. management, control, statistics, exception cases, etc.) are moved to the slow path. This allows the elements in the fast path to operate in a "run to completion" mode, providing optimal performance while still meeting the overall control plane requirements of the protocol.

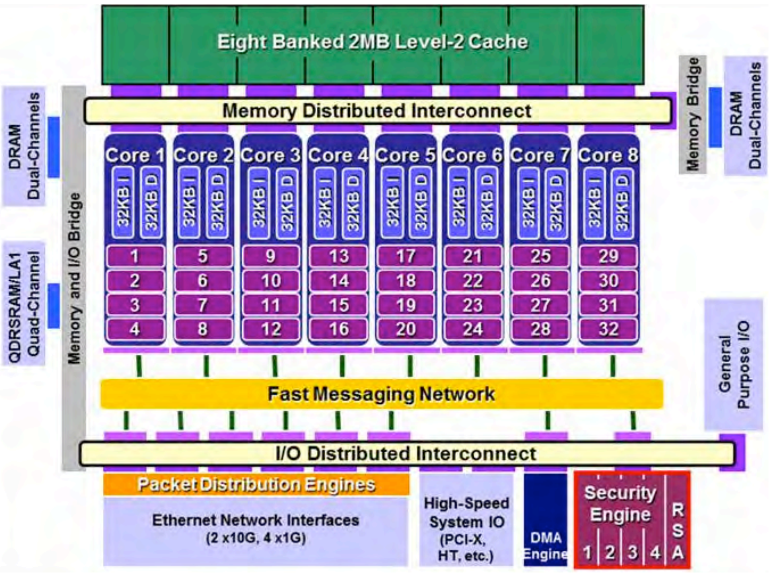

Figure 3. *NetLogic XLR Processor Architecture*

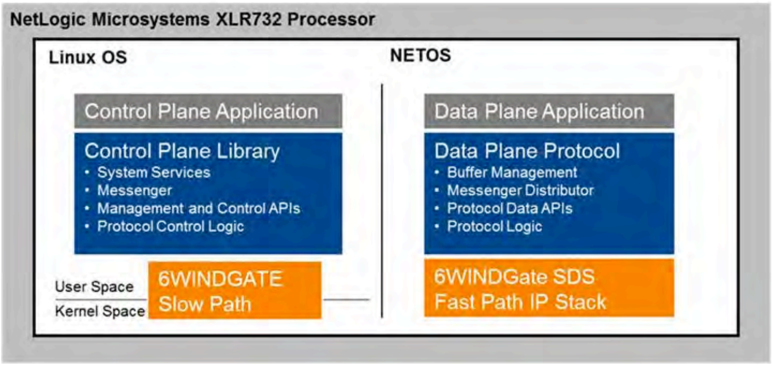

Figure 4. *Fast Path Architecture*

The slow path runs on a standard Linux OS and provides well-defined management and control APIs for the application. Through the management API, the application can configure protocol parameter and resources, control the protocol layer resources and collect statistics and status information. Through the control API, the application can, at run time, initiate protocol control operations. The slow path component of protocol implementation in the Linux space is also responsible for implementing exception scenarios of the protocol.

On the right side is the fast path software. The fast path of the software executes the core functionality of the protocol and in typical scenarios will process most of the ingress messages coming from peers. The fast path implementation utilizes the core functions and services provided by a thin executable operating environment, in this case NETOS, and optimized IP L3 layers, in this case provide by the 6WINDGATE SDS Fast Path software. Like the slow path elements, the fast path software also provides a well-defined API for the data application to process incoming messages.

The fast path runs on the NETOS thin executable to provide the best utilization of compute resources, whereas the control part software can handle control functions and runs under Linux. The communication between Control (slow path) and Data (fast path) parts is realized by utilizing shared memory and the inter-thread messaging framework provided by NETOS and Linux. Figure 5 provides a more detailed diagram of the SCTP Fast Path decomposition.

The main aim of a fast path architecture is to divide the functionality required between threads which can be pipelined together to achieve the overall objective of the protocol. The slow path processing functions are responsible for control and management requirements of the protocol.

For SCTP the fast path processing is divided into four different types of software threads:

- *SCTP Core Thread:* the primary function of this thread is to communicate with the control function and distribute the control commands to SCTP processing threads. The command set generally includes actions related to association or endpoint management. This function typically utilizes one thread.
- *IP and Distributor Thread:* these threads are responsible for performing Layer 3 IP/IPsec processing as well as determining the SCTP association to which a particular ingress message belongs.
- *SCTP Fast Path Thread(s):* these threads are responsible for implementing the core state machine of SCTP.

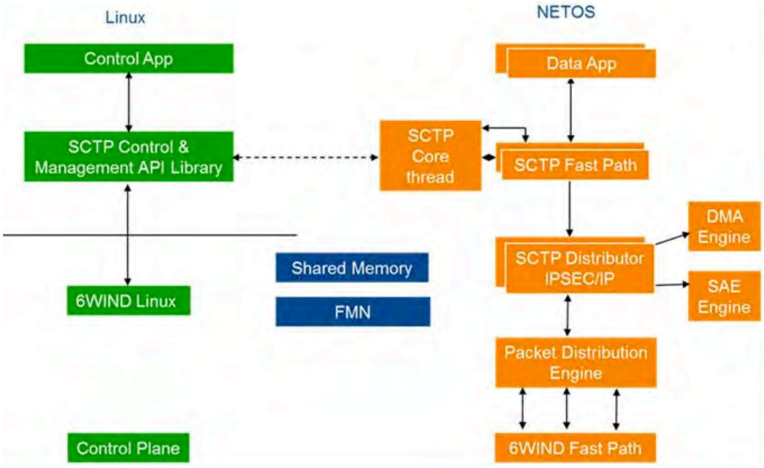

Figure 5. *SCTP Architecture*

- *Application Thread:* these threads are utilized to execute SCTP application functions.

SCTP Ingress Packet Flow

This section describes the packet flow received from the network (i.e, ingress). First, an Ethernet packet is received by an Ethernet port on the XLR processor. Next, the Ethernet packet is passed to one of the IP and distributor threads for further processing.

The IP and distributor thread performs Layer 3 IP processing, including security processing when required. This thread is also responsible for communicating with the Security Acceleration Engine (SAE) for decryption of the packet in case IPsec is used on the flow. After the IP processing is completed, if the type of packet is SCTP, the SCTP association is identified that is using the address parameters received in the message. The message is then delivered to the specific SCTP processing thread which is responsible for processing all messages belonging to that association. This architecture avoids locking in the SCTP processing thread which in turn provides highly increased SCTP processing efficiency.

SCTP Egress Packet Flow

This section describes the flow for packets going out into the network (i.e., egress). First, an SCTP application thread sends the packet to the SCTP processing thread. The SCTP thread processes the packet and prepares it for transmission. It then sends the packet to its associated IP processing thread which performs IP Encryption, if needed, and adds the IP header. Finally, the IP processing thread sends the packet to the network.

SCTP Performance Improvements

Table 1 specifies the SCTP performance comparison on different platforms.

The first platform is a Dual Intel Harpertown x86-based ATCA blade running 8 cores. The second platform is an XLR processor running SCTP on Linux OS.

Finally, the third platform is an XLR processor running SCTP optimized for the fast path utilizing the fast path architecture detailed above. The optimized SCTP Fast Path shows a minimum 10X improvement over the competing implementations.

The XLR processor has 8 cores with 4 hardware threads in each core. The SCTP thread allocation in the fast path is specified in Figure 6 as follows:

- Core 0 running Linux
- Core 1,2 running Layer 3 and distributor functions
- Core 3,4,5 running SCTP Fast Path implementation
- Core 6, 7 running SCTP Application functions

Note: SCTP core allocation can be changed for different processing requirements to achieve the best utilization of compute resources.

Processing Environment	SCTP Message Throughput
Dual Harpertown x86-based ATCA blade	100K
XLR running SCTP on Linux OS	30K
XLR running SCTP on Fast Path	1029K

Table 1. *SCTP Performance Comparison*

Figure 6. *SCTP Thread Allocation*

Conclusions

Signaling performance in existing 3G and LTE networks is a key emerging issue in overall network performance. The ability to efficiently support a constantly escalating number of connected devices and, in turn, the migration to small cells, requires innovative hardware and highly optimized software. Network Equipment Providers are no longer able to rely on generic implementations of key protocols to achieve these performance gains and optimized solutions, such as Trillium SCTP fast path, will become the predominate approach to dealing with key platform performance issues.

In this paper, we covered the example of an SCTP implementation and how it can be optimized to efficiently provide signaling transfer in wireless networks. This optimization model is extensible to optimize additional protocols in the fast path to achieve better efficiency and throughput. The architectural tenants and overall design approach provide a framework for fast path optimization to provide a key technological advance in the telecommunications marketplace.

References

[1] 3GPP TS 24.301: “Non-Access-Stratum (NAS) protocol for Evolved Packet System; Stage 3”

[2] 3GPP TS 24.302:“Access to the 3GPP Evolved Packet Core (EPC) via Non-3GPP Access Networks”

[3] 3GPP TS 36.401: “Evolved Universal Terrestrial Radio Access Network (E-UTRAN); Architecture Description”

Corporate Headquarters

5435 NE Dawson Creek Drive
Hillsboro, OR 97124 USA
503-615-1100 | Fax 503-615-1121
Toll-Free: 800-950-0044
www.radisys.com | info@radisys.com

September 2011