Abstract—Selected Internet Protocol (IP) Traffic Offload (SIPTO) is a feature defined in the 3GPP standards. In this paper we propose a Converged Gateway (CGW) which enables the SIPTO functionality to be performed locally, before data reaches the edge of the Evolved Packet Core (EPC) network. This feature relieves the EPC from high-bandwidth flows that could otherwise cause significant congestion. This local SIPTO functionality within the CGW takes into account the uplink-initiated IP Flow type, applies a local user policy rule (optionally provisioned from the mobile core network), and then executes the offloading decision. The CGW will manage the local SIPTO IP Flows from their inception until their end. We also show two methods for populating the local policies based upon Domain Name Server (DNS) queries and responses.

Keywords-component; formatting; Local Selected IP Traffic Offload; LTE; EPC; Converged Gateway; Local IP Flow Mobility

I. INTRODUCTION

The existing 3GPP Standards-based Selected Internet Protocol (IP) Traffic Offload (SIPTO) solution requires data to be routed into the Evolved Packet Core (EPC) before being offloaded by the Serving Gateway (SGW) [1][2]. While the currently defined standard SIPTO solution is a benefit, it does not completely relieve the core network of processing the data. A solution that completely offloads this data from the EPC would be most beneficial. It is well established that there is a bandwidth crunch in many operator’s core network [3][4]. Currently, the operators can manage this congestion by adding capacity, changing the usage habits of their customers by charging policies, by implementing SIPTO, or some combination of the above. In this paper, we propose a more efficient solution to offload selected traffic before it enters the EPC network.

Most data sessions are uplink initiated. This means that an end-user device sends an uplink packet towards an application server to start a session with that server. For example, when browsing a web page, or performing a file transfer, the end-user begins a session by contacting a server. Despite some exceptions, such as Voice-over-IP (VoIP), we believe that uplink-initiated data sessions comprise a majority of the data traffic generated by end-user devices so the solution described herein covers that case.

Uplink-initiated data sessions that are destined for external applications servers currently travel into the EPC network. Some of these IP Flows require significant amounts of bandwidth, unnecessarily stressing the EPC network backhaul and creating congestion. In standards-based SIPTO, the SGW can take a decision to offload IP traffic directly to the public Internet or route it to a different Packet Data Network (PDN) Gateway (PGW) that is “more efficient,” meaning it is less costly to use, closer to the SGW, or has more capacity.

It is proposed herein that the Converged Gateway (CGW) will perform a local SIPTO (L-SIPTO) function. When an uplink-initiated IP Flow starts, the CGW will be able to determine the application server to which the end-user device is establishing a connection. Depending on the application server, examples include YouTube or NetFlix, the CGW will bypass the EPC and route the uplink data directly to the application server. The downlink data associated with that IP Flow will also be routed directly from the application server to the CGW, bypassing the EPC network.

The L-SIPTO feature can be applied to currently existing or future end-user devices without the need for any changes and requires no application download to the end-user device.

It should be noted that the term HeNB, eNB and Home eNB are identical from the standpoint of L-SIPTO and are used interchangeably in this paper.

II. CONVERGED GATEWAY

A. Introduction

The concept of L-SIPTO requires an edge node, in this case defined as a CGW. The CGW can be introduced within the EPC or deployed locally, outside the EPC, depending on the use case, as shown in Figure 1. In the case of L-SIPTO, the CGW would be located outside the EPC. It should be noted that while Figure 1 shows the CGW, eNode Bs and WiFi Access Points (APs) as separate boxes, a physical realization with some combination of these boxes in a single unit is a natural evolution. The CGW will provide L-SIPTO analyzing the traffic that passes through it, originated by end-user devices using the cellular and WiFi spectrum managed by the CGW.
While the L-SIPTO feature is very beneficial, the CGW has other prominent benefits [5][6] including:

- Provides for improved capacity by performing Local IP Flow Mobility (L-IFOM) between the WiFi and cellular accesses
- Allows for local control and management of the cellular and WiFi spectrum
- Supports simultaneous use of L-SIPTO and L-IFOM

These benefits are gained with no change to any existing EPC elements and requiring no change to any existing protocols used by any of these EPC elements. Furthermore, the CGW supports all procedures defined in the current standards. It does, however, require that other network components be provisioned to know about the existence of the CGW.

B. Placement within Long-Term Evolution Architecture

The introduction of a CGW between an eNode B and the SGW and Mobility Management Entity (MME) is transparent to both the eNode B and EPC. Despite this placement, there are no protocols or procedures [7] [8] [9] that require modification. The CGW and EPC-based network elements do require proper provisioning with keys, certificates, and Fully Qualified Domain Names (FQDN) to support this architecture. With no change to the existing procedures and protocols, the CGW benefits outweigh these configuration requirements.

In a metro or premise deployment environment, both of which will support L-SIPTO the CGW will be provisioned with the security keys needed to establish a security association to both eNode B and the Secure Gateway (SeGW) at the edge of the EPC. Two IP Security (IPSec) Security Associations (IPSec SA) [10] will replace the single IPSec SA that exists between the eNode B and SeGW.

Specifically, the first IPSec tunnel is between the eNode B and the CGW while the second IPSec SA is between the CGW and the SeGW. Both interfaces will have unique keys or certificates. This allows the CGW to act as a proxy for the signaling between the eNode B and EPC elements. Since it is a known entity, provisioned by the operator, it is not a security risk in and of itself.

For a non-EPC based configuration, when the eNode B has a signal to send to the EPC, it sends it through the IPSec SA to the CGW. The CGW terminates the IPSec encapsulating the signal, then encapsulates the signal with IPSec, and sends the signal to the SeGW. When the SeGW receives the signal, it will terminate the IPSec encapsulating the signal and will forward it into the EPC. When an element within the EPC has a signal to send to the eNode B, it occurs in a similar fashion. As these signals traverse the CGW, it is able to glean information about the eNode B and the connected end-user devices.

The CGW will act as a proxy for data that is sent between an end-user device and an application server in a similar
fashion. The eNode B and SGW each require a GPRS Tunneling Protocol (GTP) tunnel. The CGW acts as a GTP-proxy, terminating any GTP tunnel from the eNode B or SGW.

III. LOCAL SELECTED IP TRAFFIC OFFLOAD FUNCTION

IP Flows are examined and categorized at the CGW. Local user policies containing L-SIPTO rules for offloading are applied by the CGW to the categorized IP Flows. The characterization of IP Flows is done by examining its 5-tuple. The 5-tuple includes the source and destination IP addresses, the source and destination port numbers, and the IP protocol type [11].

The CGW performs L-SIPTO before an IP Flow reaches the SGW; in fact it is done before even reaching the Secure Gateway (SeGW) at the edge of the EPC network. Executing SIPTO locally at the CGW relieves the EPC of some traffic that is only being routed through the EPC to an application server. In Figure 2, the green path shows L-SIPTO routed data being sent directly from the CGW to the Application Server within the public Internet. If this IP Flow was, for example, a YouTube video, the EPC network would be unburdened from carrying this data. The red path depicts the non-L-SIPTO routed data traveling through the EPC Network and finally to the Application Server within the public Internet.

The CGW requires certain functionality to perform L-SIPTO. The CGW must be able to perform Packet Inspection (PI) which is the ability to read the 5-tuple of a packets header and it must be able to perform a Network Address Translation (NAT) function [12], similar to what the PGW performs when an IP Flow traverses through it, between an end-user device and application server. It should be noted that while these are described as distinct functions, to facilitate a cogent explanation, a deployment would most likely merge these functions.

A. Local Selected IP Traffic Offload Detailed Processing

In order to explain the L-SIPTO processing performed within the CGW an example is warranted. In Figure 4, the processing for an IP Flow that originates in the uplink direction is shown.

Figure 3. Local Selected IP Traffic Offloading Routing by Policy

In this example, it is assumed that the end-user device has connected to both the HeNB and WiFi AP and that the CGW knows both connections terminate at the same device. If the end-user device only has a WiFi or cellular connection, the process for L-SIPTO is identical. It is also assumed that the CGW has L-SIPTO, PI and NAT functionality as described in the previous section.

In this example, there are five steps shown which are described below. The intent is to describe the functionality performed within the CGW.

1. Step 1, an uplink packet reaches the L-SIPTO functionality. It can reach the CGW either via WiFi or cellular.

2. In Step 2, the CGW looks at the 5-tuple and realizes this is a new IP Flow. It routes the packet to PI which identifies the IP Flow based on its 5-tuple and returns this information to the L-SIPTO function in Step 3. After extracting the 5-tuple, the CGW consults the policy for this user for this type of data and realizes this is an IP Flow that should be sent via L-SIPTO. In Step 4, the CGW sends the packet to the NAT functionality within the CGW where the packet is translated by the NAT function (“NAT’ed”) and it is pushed towards the public Internet. This allows the Application Server to know to send any associated downlink packets to the client directly to the CGW.

3. Any associated downlink packets from the Application Server will be sent directly to the CGW. The packets will be
“un-NAT’ed” and sent to the User Equipment (UE) over the wireless transport that was used for transport of the uplink packet.

After the above discussion, there are two obvious questions. First, how do the policies get populated? Second, what IP addresses are used as the IP packets are routed between the end-user device and the application server? We endeavor to answer both of these in the following two sections.

B. Policy Population

It is possible that the CGW could be provisioned with a list of those IP addresses whose traffic is to be offloaded via L-SIPTO. However, this would be difficult to maintain as application servers may have many different servers and may change IP addresses. Therefore, a more robust method is required. Two methods based on Domain Name Server (DNS) Queries and Responses [13] are described herein, each allowing the CGW to learn the IP addresses of those application servers whose traffic is to be offloaded:

1. Autonomous DNS query
2. DNS Interception

In either method, the CGW has a list of FQDN or domain names of the traffic that should be offloaded, similar to that shown in Table 1. This table links FQDN and/or domain names, used in DNS Queries, to users based on their International Mobility Subscriber Identity (IMSI) [14]. Additionally, the table can be constructed such that all the local users on this network would have the same policy, such as all local users YouTube traffic will get routed via L-SIPTO (“L-SIPTO’ed”).

<table>
<thead>
<tr>
<th>IMSI</th>
<th>FQDN</th>
</tr>
</thead>
<tbody>
<tr>
<td>01234567890123</td>
<td>youtube.com</td>
</tr>
<tr>
<td>01234567890124</td>
<td><a href="http://www.youtube.com">www.youtube.com</a></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>01234567890124</td>
<td>cnn.com</td>
</tr>
<tr>
<td>0000000000000000</td>
<td>youtube.com</td>
</tr>
<tr>
<td>0000000000000000</td>
<td><a href="http://www.youtube.com">www.youtube.com</a></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

TABLE I. POLICY TABLE

<table>
<thead>
<tr>
<th>Name</th>
<th>IP Address Minimum</th>
<th>IP Address Maximum</th>
<th>Time to Live</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www.youtube.com">www.youtube.com</a></td>
<td>173.192.43.3</td>
<td>173.192.43.11</td>
<td>4 minutes</td>
</tr>
<tr>
<td><a href="http://www.cnn.com">www.cnn.com</a></td>
<td>157.166.266.18</td>
<td>157.166.266.26</td>
<td>5 minutes</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Notice that the Time-to-Live [15] provided in the DNS Response is saved. This tells the CGW how long the IP addresses is valid. Also note that the CGW stores a range of IP addresses. This feature allows for a large range of IP addresses to be stored without requiring a large table within the CGW.

It is possible that the CGW may route some IP Flows from a single application through the EPC while routing the remainder of the IP Flows from the same application directly to the application server. A prototype developed to demonstrate this concept has successfully allowed for a mix between L-SIPTO and non-L-SIPTO traffic for the application session. This is possible in the case where an application server uses several IP addresses to deliver...
content over several IP Flows. This prototype and its implementation are described in detail in Section IV-B.

It may also be necessary for the CGW to process the Name Server (NS) responses and issue DNS Queries on those names to provide a more robust L-SIPTO feature.

C. IP Addressing Considerations

For IP packets (and for items such as TCP handshakes [16]) sent by the client to the CGW via either the cellular or WiFi transports, the source IP address will be the EPC-assigned IP address. If the CGW decides to route an IP Flow to the EPC, the source IP address will be set by the CGW to the EPC-assigned IP address, regardless of how the client delivered the packet to the CGW. This is consistent with the L-IFOM functionality of the CGW.

If the CGW decides to route an IP Flow to the application server directly, via L-SIPTO, it will perform a NAT on the uplink packet so that the Application Server can deliver downlink packets for the same IP Flow directly to the CGW.

The IP addressing for uplink packets is shown in Figure 5.

For downlink packets that are received by the CGW via the EPC, it will forward them to the client using either of the accesses.

For downlink packets that are received by the CGW directly from the Application Server, it will forward them to the client using either of the accesses. The source IP address will be the Application Server while the destination address will be either the EPC-assigned IP address (if via HeNB) or the CGW-assigned local IP address (if via WiFi AP).

For downlink packets, the IP addressing is as shown in Figure 6.

IV. IMPLEMENTATION ASPECTS

The key requirement for the implementation is to build a standalone platform that permits interoperation with the various existing HeNB, WiFi AP and EPC elements. InterDigital has prototyped [17] and demonstrated such a device [18].

A. Architecture

In order to satisfy the above requirement the decision was made to implement the CGW as a separate network element placed between the HeNB and the SGW. This network element is perceived as the EPC by the HeNB and as an HeNB by the EPC. Figure 7 presents a simplified and high level view of the architecture that permits the accomplishment of this behavior. For the sake of clarity the components such as the IPSec are not represented.
by the proxy when it enters the CGW. In the case of the uplink packets the Segregator has the choice to forward them through the HeNB proxy that will encapsulate and send them to the EPC or the Segregator may directly route them through the NAT function (SIPTO) toward the public Internet. In the case of downlink traffic the Segregator may either send the user packets directly through the WiFi AP or forward them through the EPC proxy that will encapsulate and send them to the HeNB. The user packets may also arrive un-encapsulated directly through WiFi in the uplink or through the NAT function (SIPTO) in the downlink. In these cases the Segregator has the same choices as with the encapsulated packets. Note that, as stated earlier, the decision to route packets directly through the NAT function (SIPTO) has to be performed on the first uplink packet of a flow and this decision must remain for all the remaining packets of the flow.

B. Implementation

The implementation is based on a 64bit Ubuntu Linux platform running on an off-the-shelf personal computer (PC). The implementation relies heavily on various Linux operating system services. The core of the implementation is executing in user space. It communicates with the kernel by using User Datagram Protocol (UDP) sockets, raw sockets, the Netfilter queue library (based on the NETLINK sockets) as well as the TUN (network TUNnel) interfaces. The packet routing is dictated by the Segregator module using the policy routing based on packet tagging. Figure 8 depicts the flow of an uplink user flow packet that is routed through both the EPC and HeNB proxies. An encapsulated packet is received by the GTP server in the EPC proxy; it is then de-encapsulated and sent out using a raw socket. After this, the packet is intercepted using the Netfilter queue by the Segregator. The Segregator inspects the packet, marks it with the appropriate routing value and returns it to the kernel. The kernel routes the packet using the appropriate routing table by inspecting the mark value. The packet is routed to the TUN interface that corresponds to the HeNB proxy entry point. As a final step, the HeNB Proxy encapsulates the packet and sends it out through a client UDP socket.

In the downlink direction a very similar process takes place and is omitted for the sake of brevity.

The user space implementation relies on the following Linux kernel services in order to route the packets appropriately:

1. Policy routing: Different routing tables are used based on certain criteria. In this case the criterion is packet marking.
2. Iptables: The iptables kernel support and tool set permit the kernel to manipulate the packets that are processed by the kernel. In our implementation the iptables key role consists in intercepting selected packets.
3. Netfilter queue library: This library permits retrieval of selected packets into user space for inspection and marking.

![Figure 8. Uplink packet through Converged Gateway](image)

4. TUN tunneling device: This is a virtual network device that permits the packets to be routed into the user space.
5. NAT: The Network Address Translation permits the packets that follow the L-SIPTO policy, to be sent out directly toward the Internet, to have their UE IP address replaced by the IP address of the CGW.
6. Conntrack-tools: This connection tracking support gives the capability to the CGW to follow all the flows established by the UE.

V. CONCLUSION

In this paper we propose a CGW which enables SIPTO to be performed locally, before reaching the edge of the EPC network. Currently, uplink-initiated IP Flows that require significant amounts of bandwidth continue to travel through the core network. Applying policy-based SIPTO rules locally at the CGW relieves the core network of unnecessary, high-bandwidth traffic. While the L-SIPTO function has been shown for a Long-Term Evolution-based EPC, the same methods and functionality are feasible for a 3G-based network. There are several areas for future study which include user mobility outside the area managed by the CGW; performing L-SIPTO on already existing IP Flows; as well as meeting the Lawful Interception (LI) requirements for IP Flows that are routed by the CGW using L-SIPTO.

REFERENCES

[2] 3GPP TR 23.859, “LIPA Mobility and SIPTO at the Local Network (Release 11),” v0.4.0.


