Carrier Cloud Telecoms – Exploring the Challenges of Deploying Virtualisation and SDN in Telecoms Networks

Background

The introduction and deployment of Cloud-based concepts into the IT datacentre industry has been very rapid over recent years. In parallel, the associated ecosystem and business models have changed fundamentally. IT Cloud economy-of-scale benefits and the overall value proposition derived from shared datacentre resources and "pay-as-you-go" business models are also potentially valid for telecom networks.

However, in the telecoms domain, network elements are traditionally assembled in a heterogeneous way with many different equipment architectures and vendors right through the network. This network architecture has evolved by necessity, owing to carrier-grade requirements such as real-time performance and high availability. Mainstream industry adoption and deployment of Cloud Telecoms will be hindered, or even blocked, until carrier grade telecom requirements have been fully proven on the key technologies underpinning the Cloud Telecoms concept, namely Network Functions Virtualisation (NFV), Software Defined Networking (SDN) and deployment on industry-standard, high-volume servers.

Virtualisation promises freedom of deployment for various functionalities in an open, standardised environment. When introduced in a telecoms context, the critical success factors are performance, latency and standardized management interfaces.

SDN is an approach to building networks that accomplishes the following: separates the control and data planes; provides a global view of the network to a centralized controller; and enables external applications to program the network. It also promises freedom of connectivity on a number of different levels in the network infrastructure. For example, the same management concepts can be applied to connectivity functions (e.g., on network, element, blade and chip levels). SDN can be seen as a complementary technology to virtualisation and is potentially well suited for a network-enabled cloud and improving network resource utilization on the link level.

The more network functions that can be migrated to industry-standard, high-volume server environments, the bigger the business case becomes for Datacentre concepts to be more widely deployed in the network. However, many network functions have extreme characteristics and performance requirements. These need to be addressed on industry-standard servers with intelligent load balancing, power management and high-speed packet processing.

This paper describes the Carrier Cloud Telecoms initiative[1], a joint program between Intel and Tieto*, which explores the key R&D challenges and possible approaches central to cloud realization in the telecoms domain. The initiative includes a virtualised proof of concept implementation of 3G core network and LTE evolved packet core (EPC) network functions in an open, scalable multicore environment based on Intel® architecture. Thanks to cloud-based technologies, and improvement gains from Intel® multi-core technology and software optimizations developed by Intel and Tieto, the potential exists to open the telecoms network to allow for new innovative service designs that promote ecosystem cooperation and best-of-breed solutions. In addition, the potential exists to reduce provisioning costs (CapEx) required for telecoms workloads and applications, and significantly lower operational costs (OpEx). A truly virtualised cloud-based network architecture can allow equipment providers and mobile network operators to:

- Efficiently operate, maintain and upgrade network resources, while speeding up functional and service deployments.
- Lower CapEx by reducing the need for specialised equipment and tool chains.
- Innovate to deliver new and enhanced services and revenue streams, while also allowing existing investments to be re-used.

The rest of this paper describes the virtualised proof of concept developed by Intel and Tieto, provides performance results, and evaluates both the benefits and ongoing challenges to Carrier Cloud based telecom deployments.

www.intel.com

www.tieto.com

Demonstration Setup Description

The virtualised proof of concept uses traffic generators and server platforms that are all based on industry-standard, high-volume hardware, specifically Intel® Xeon® processor E5-2600 v2 product family. There are also two switches in use, a high-capacity, OpenFlow*-enabled Intel® Ethernet Switch FM6764 (10GbE/40GbE L2/L3/L4) and a 1GbE switch, which is used for the management interfaces.

The software execution environment is based on a standard Linux* distribution (Debian), with the target hardware being an Intel Xeon processor E5-2600 v2 product family based platform. For the virtualisation parts of the demonstration, KVM (Kernel-based Virtual Machine) and OpenStack* software are used, and for SDN, OpenFlow* is used to configure Open vSwitch*. SR-IOV (Single Root I/O Virtualisation), Intel® Virtualization Technology (Intel® VT) for Directed I/O (Intel® VT-d) and the Intel® Data Plane Development Kit (Intel® DPDK) are used to accelerate packet processing performance in the virtualised environment.

In the Carrier Cloud Telecoms initiative, Intel and Tieto are currently working to integrate the Intel® Multi-Buffer Crypto for IPSec Library to accelerate on-core IPSec processing. Performance benchmarking and measurements will be provided as part of the demonstration once completed.

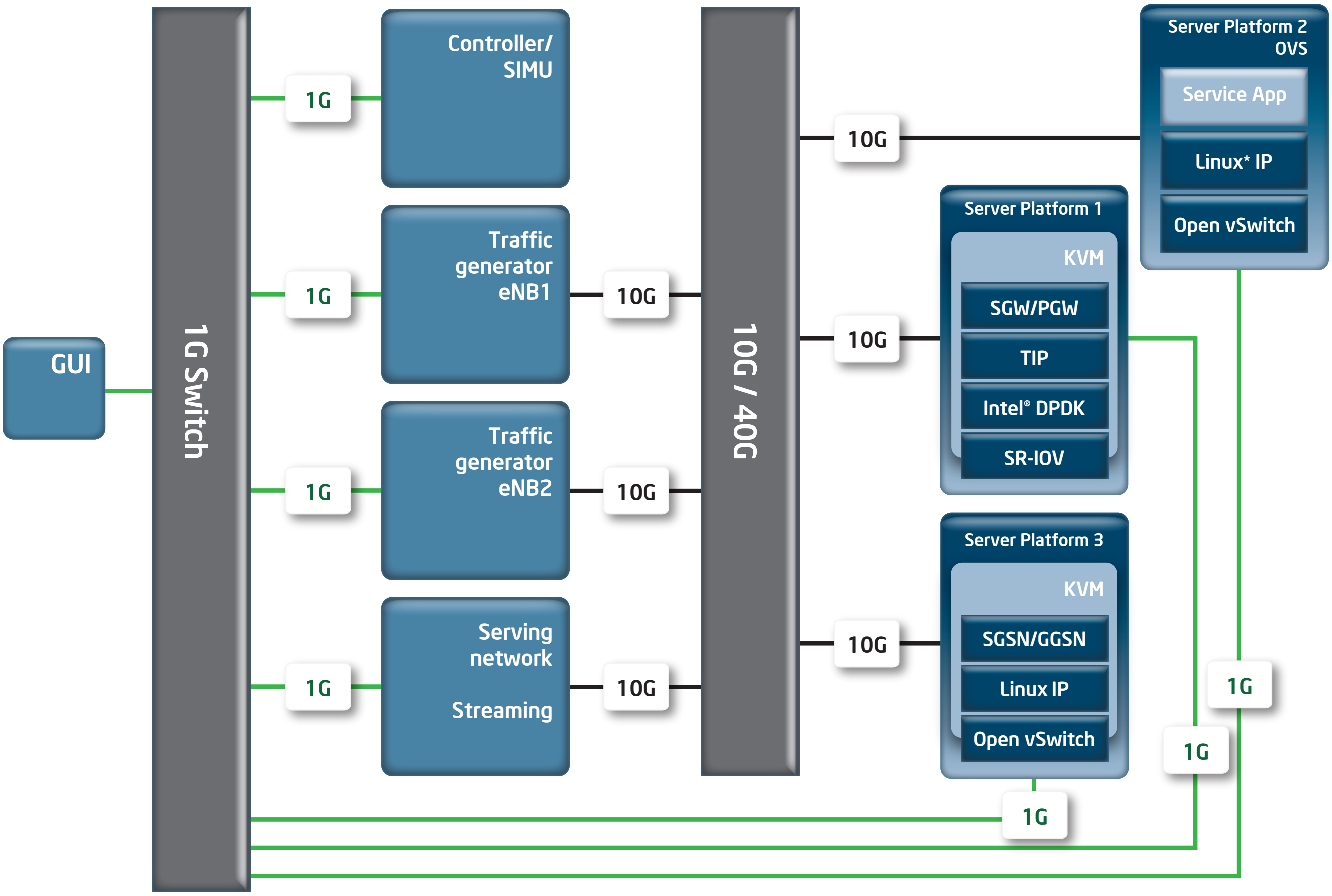


Figure 1. Demonstration Deployment Setup
Intel® DPDK: Intel® Data Plane Development Kit

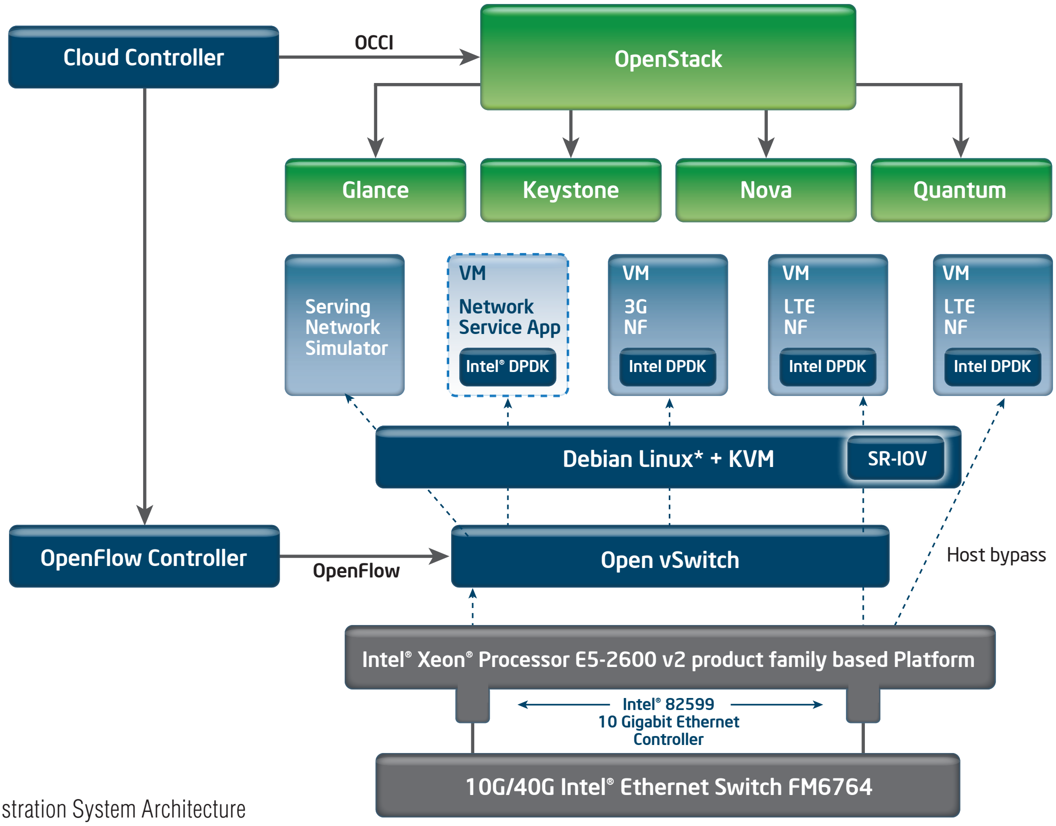


Figure 2. Demonstration System Architecture

Demonstration Use Case Descriptions

The following scenarios are part of the Carrier Cloud Telecoms demonstration:

- Dynamic provisioning of 4G/LTE traffic and resources in a virtualised SDN environment
  - VLAN-based load distribution enabled by OpenFlow
  - Multiple fault-isolated KVM virtual machines (VM) allowing for multiple logically separated tenants in the same physical system
  - Dynamic provisioning and management of VMs via OpenStack
  - Event and fault management of virtual resources via industry-standard, operation and maintenance (O&M) northbound APIs (notifications and alarms)
  - Utilization of SR-IOV hypervisor bypass technology on Intel® processors along with Intel VT-d for improved performance and reduced latency
  - Use of Open vSwitch for the inter-VM interconnect and load distribution
- High-performance and energy-efficient packet processing and protocol distribution using the Intel DPDK and the Tieto IP stack (TIP)
  - High-speed software load balancing and distribution of SCTP and GTP-U bearer traffic
  - Advanced power management schema based on an inter-VM control plane solution
- 4G/LTE to 3G video stream handover scenario
  - Seamless 3GPP handover from the LTE to 3G network using Open vSwitch and OpenFlow in line with 3GPP procedures
  - Quality of service (QoS) implementation
- LTE Network Functions - PGW handover scenario using SDN
  - Consolidation of traffic streams at low system loads
  - Software controlled seamless traffic handover of internet traffic at the Packet Data Network Gateway (PGW)
  - Utilization of OpenFlow API and Open vSwitch to re-program traffic flows during handovers

Intel® Architecture and Intel® Data Plane Development Kit (Intel® DPDK)

Packet processing performance has improved tremendously on Intel processor-based platforms due to the combination of software advances and Intel® microarchitecture enhancements.

Intel architecture-based platforms, combined with high-performance packet processing software, like the Intel DPDK, provide the capability to consolidate telecoms workloads onto a single platform. Moreover, Intel is committed to delivering ever-increasing performance and power-efficiency through an industry-leading beat-rate of manufacturing process improvements and microarchitecture advancements via a predictable "Tick-Tock" model. This predictability, matched with optimized software solutions, helps to unleash the full potential of Intel architecture-based platforms.

The Intel DPDK is a set of optimized software libraries and drivers that enable high-performance data plane packet processing on network elements based on Intel architecture. For more information on the Intel DPDK, see www.intel.com/go/dpdk.

Network Functions Virtualisation

Network Functions Virtualisation (NFV) is today a major topic of industry attention and interest, and has resulted in the creation of industry collaboration groups such as the ETSI NFV Industry Specification Group (ISG), of which both Intel and Tieto are participants. The Carrier Cloud Telecoms demonstration uses virtual network functions, such that each of the LTE EPC and 3G Core Network applications is executing within a VM. The demonstration uses OpenStack and KVM, which provide the overall virtualisation infrastructure. KVM supports native virtualisation on processors with hardware virtualisation extensions. Deploying virtualisation in the network can help to support multi-tenancy deployments, a key requirement for future network evolution. Virtualisation separates the underlying cloud infrastructure 'enabling' host framework from the tenants, and it will also provide inter-tenant separation when multiple tenants are occupying the same physical hardware.

The key benefits of using virtualisation are:

- Flexibility of deployment, helping to simplify scalability of application software on industry standard hardware.
- Reduced equipment costs through consolidation of equipment and energy efficient power management implementations.
- Reduced time to market (TTM) by minimising the procure-design-integrate-deploy innovation cycle. Expenditures and time required to innovate, validate and deploy hardware-based functionalities are much less for software-centric development, allowing for more innovation in overall lifecycle.

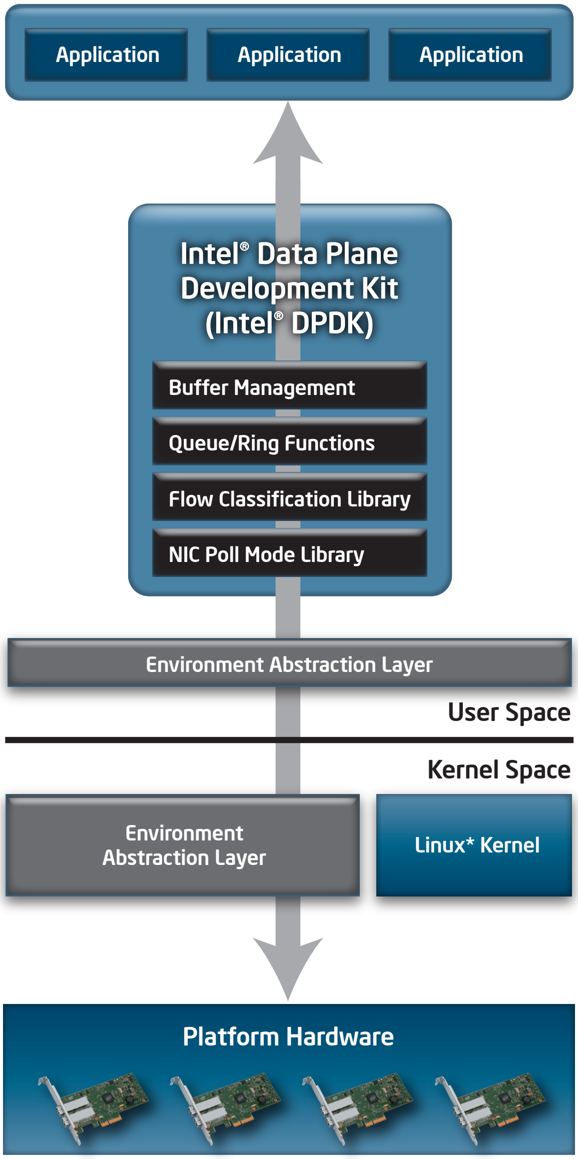


Figure 3. Intel® Data Plane Development Kit

- Reuse of application software, which encourages more innovation internally and in the broader industry. This allows new services and revenue streams to be developed quickly and at much lower risk. It also allows these new services to be targeted geographically for greater flexibility.
- Increased support for fault isolation and security in the network.

Virtualisation does have its challenges, however. From the Cloud Telecoms perspective, ensuring that performance latencies caused by virtualisation and hypervisors are kept to a minimum is paramount.

Executing programs in a VM is normally quite efficient due to virtualisation features built into newer CPUs, e.g. the Intel Xeon processor E5-2600 v2 product family with Intel VT-d. This applies as long as the execution of instructions is contained within the VM. However, operations that require execution to exit from the VM and continue in the hypervisor can have considerable performance overhead. In classic virtualisation, hardware is emulated in the hypervisor and all hardware access requires an exit from the VM into the hypervisor to simulate the hardware access from the VM environment. Such a switch of context has a considerable performance overhead.

Para-virtualisation means that less of the hardware is simulated, which results in better VM performance; this is one of several features that can be used to improve performance of virtualisation. In the Carrier Cloud Telecoms initiative, SR-IOV, Intel VT-d and the Intel DPDK have been used to allow the Tieto IP-stack within the VM to access the physical hardware directly from within the VM to achieve network performance comparable to native, non-virtualised performance. While SR-IOV allows an IP-stack within a VM direct access to parts of the physical hardware, it does not allow the IP-stack to configure the hardware, i.e., the configuration needs to be handled from outside the VM by the hypervisor. While virtualisation provides efficient compartmentalization and abstraction of physical hardware, it also makes infrastructure performance monitoring and management more difficult. In the Carrier Cloud Telecoms demonstration, basic hardware management has been implemented using OpenStack; however, fine-grained control of the hardware needs a control channel between IP-stack and hypervisor. In the demonstration, a specialized inter-VM power management algorithm has been implemented to minimize latencies, which is a good example of a more fine-grained control channel.

Performance Benchmarking

The purpose of the performance measurements within the Carrier Cloud Telecoms initiative is to demonstrate high performance and scalability on multiple cores with the protocols used in LTE EPC. To demonstrate flexibility and cost efficiency, only generalized hardware such as Intel Xeon processor E5-2600 v2 product family based CPUs and Intel® Ethernet Gigabit Server Adapters (NICs) are used. The primary protocols of LTE EPC are SCTP and GTP-U, and these are used by nodes, such as the MME and SGW nodes for signalling and user-plane data, respectively. The performance demonstration shows performance and scalability of these protocols.

The results presented are thus applicable to any node using protocols and traffic patterns similar to those presented (like eNodeB). Performance has been measured in a simplified network with two nodes connected back-to-back. In the simulation, one node generates traffic while the other relays back duplex traffic. A fully-featured IP-stack is used (Tieto IP Stack) together with simulated SGW and MME functionality. The GTP protocol is not fully featured but performance-wise is representative of a full implementation. Tests have been made in both a virtualised and a non-virtualised setup. Virtualisation adds complexity, and despite improved hardware support for virtualisation, some overhead is still expected. Intel processors support extended page tables that reduce virtualisation overhead. NICs are being accessed directly through Intel VT-d, which adds an input/output translation look-aside buffer (IOTLB) to reduce the overhead from accessing the NICs. Also, the setup uses huge-pages to reduce the load on the translation look-aside buffer (TLB) caches.

The performance measurement system has four physical cores with Intel® Hyper-Threading Technology (Intel® HT Technology), which creates eight logical cores. Performance results are based on an architecture where there is one load distribution core and three “worker cores”, upon which the traffic is distributed. The load distributor runs on core 0. When running multiple application instances in one VM, core 1 is used for the TX multiplexer required to map transmit traffic from the worker cores into the single transmit queue available in the virtualised environment. When running one application per VM, the load distributor and the TX multiplexer are not required. The application worker cores are running on the logical cores 2, 4 and 6. The additional logical threads created by Intel HT Technology are not used for the application worker cores.

The maximum throughput is measured by increasing the traffic generation rate until the resulting throughput does not increase. To measure the scalability of performance when increasing the number of cores, we have also made measurements with a fixed (high) traffic rate and by increasing the number of cores. The traffic is always equally loaded between the involved cores.

Performance measurements have been taken for both Host and Virtualised environments.

For all the figures below, the throughput is in units of:

- Kpacket is 1000 message signal units (MSUs)
- Mbit is 1.000.000 bits
- Packet size is GTP-U payload

Throughput in the graphs is one-way only (even though the traffic is two-way). All traffic is sent from the generator to the target test application, which echoes the traffic back. The generator measures the throughput, and it only counts the number of packets received. This means that all packets are both received and transmitted by the target test application but are only counted once. The total throughput (both receive and send) is the double of the throughput values shown in the graphs. For all tests, Intel HT Technology is enabled, although the additional logical cores are not used for applications.

The load distribution component is software-based functionality that distributes ingress traffic to IP-stack slices based on a (static) set of rules. A dedicated core receives ingress traffic from the NIC ports and distributes traffic to worker cores. The Intel DPDK ring functionality is used for passing ingress traffic from the load distributor core to the worker cores.

The software load distributor uses a static set of rules, and the supported rules are (in deceasing order of precedence):

1. IP address range
2. Port range match
3. VLAN tags
4. IPSec selectors
5. GTP-U endpoint ID match.

The load distributor also supports exception traffic such as ARP and ICMP. Handling of these packets is defined with special rules; for example, each ARP packet can be directed to a specific core running an ARP proxy.

The use of the load distributor enables efficient traffic distribution on a multicore environment. However, for very high throughput wireless traffic (GTP-U) distribution (40GbE and higher), the software-based load distribution mechanism can start to experience performance-bottlenecking (essentially IO-bound). Thus, the architecture of the load distributor will need to be enhanced and potentially complemented by hardware-based load distribution for extremely high throughput scenarios where GTP-U traffic needs to be distributed over the available cores in the system.

The proof of concept performance using the Tieto IP stack and Intel DPDK was measured for both host and virtualised environment. In summary, the obtained performance results are listed below. These numbers are given for two-way traffic (i.e., packets are counted both as received and transmitted).

Sustainable SGW throughput:

- Generally host and VM performances are identical
- Performance with and without power management is identical
- GTP 10 bytes MSU 1 core (host) 8.000.000 packet/s
- GTP 10 bytes MSU 1 core (VM) 8.000.000 packet/s
- GTP 64 bytes MSU 1 core (host) 7.600.000 packet/s
- GTP 64 bytes MSU 1 core (VM) 7.600.000 packet/s
- GTP Packet Mix (50% 64byte, 50% 1400 Byte) 1 core (VM) 14.000.000 packet/s bi-directional (at 70% core utilization)
- SCTP 100 bytes MSU 1 core (host) 2.580.000 MSU/s
- SCTP 100 bytes MSU 1 core (VM) 2.580.000 MSU/s

Host and Virtualised Performance Benchmarks

The Carrier Cloud Telecoms initiative has baselined formal performance numbers for GTP-U based bearer traffic in both a virtualised and host OS environment as part of the demonstration. The system used is an Intel® Core™ i7-3820 processor @ 3.60GHz with 10 Mbyte cache and support for Intel VT-d, and has Intel C-state management enabled in the BIOS. The systems are configured for a host OS or a virtualised environment, depending on the actual tests. These findings are presented on the next page.

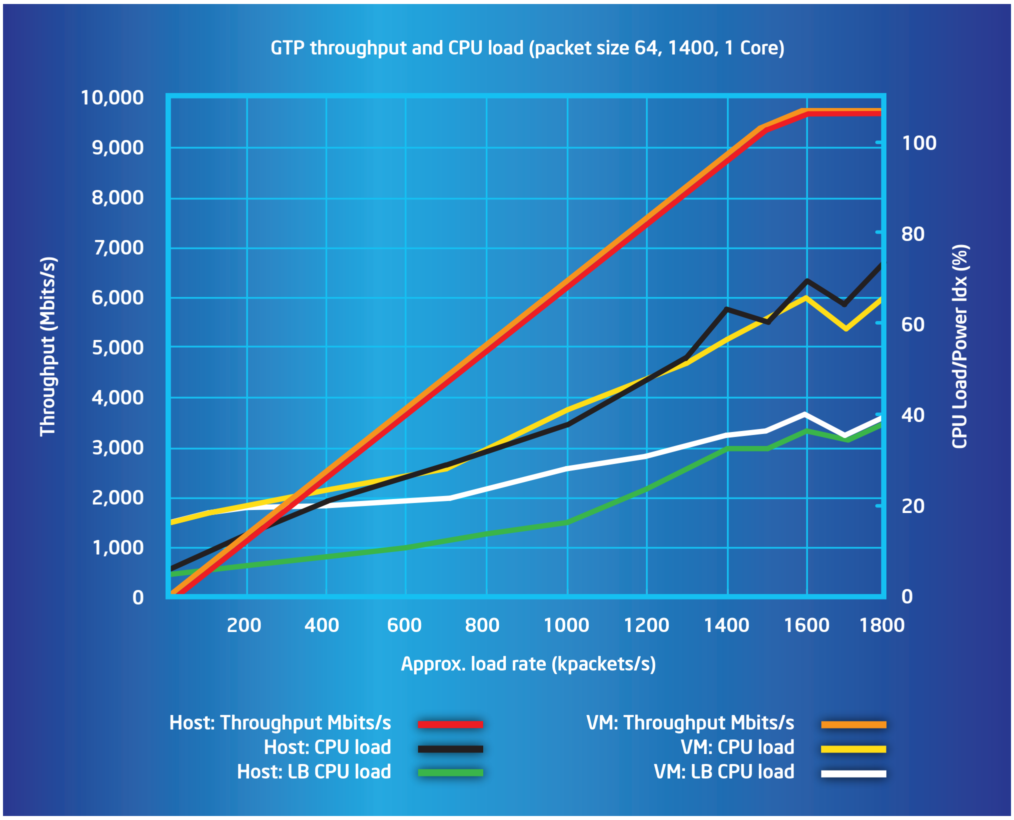


Figure 4. Internet Traffic Distribution

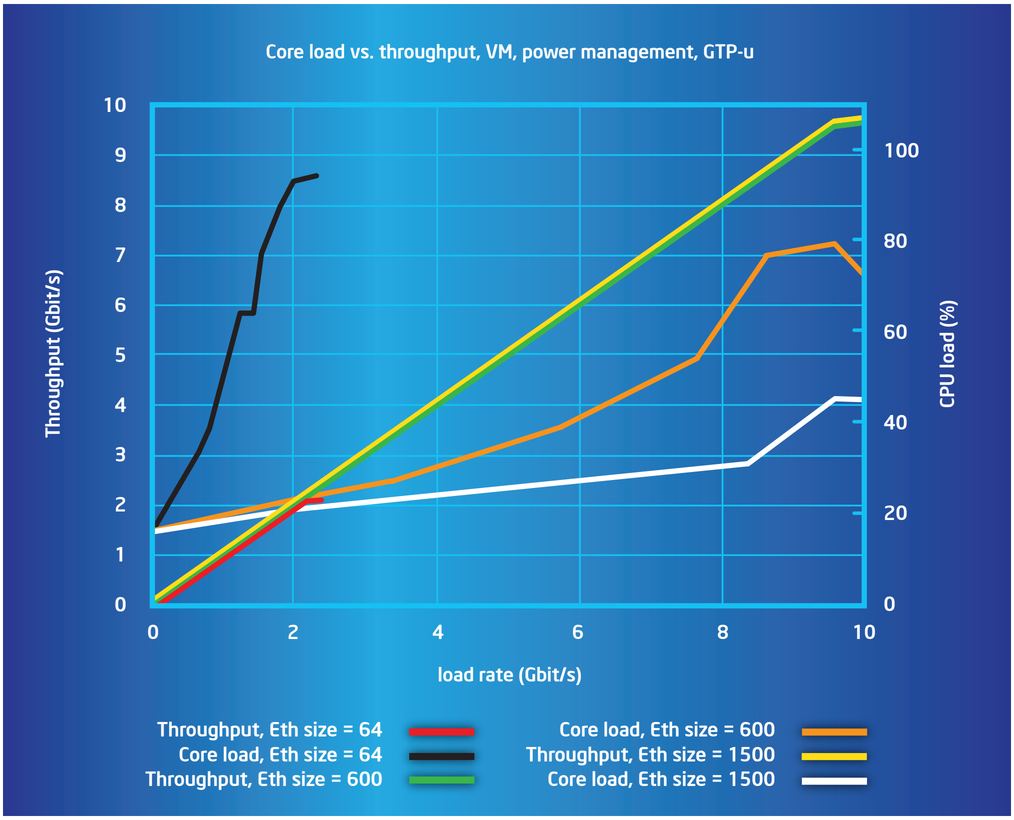


Figure 5. Tieto IP Stack and Intel® Data Plane Development Kit (Intel® DPDK) GTP-u Performance in a Virtualised Environment With Power Management

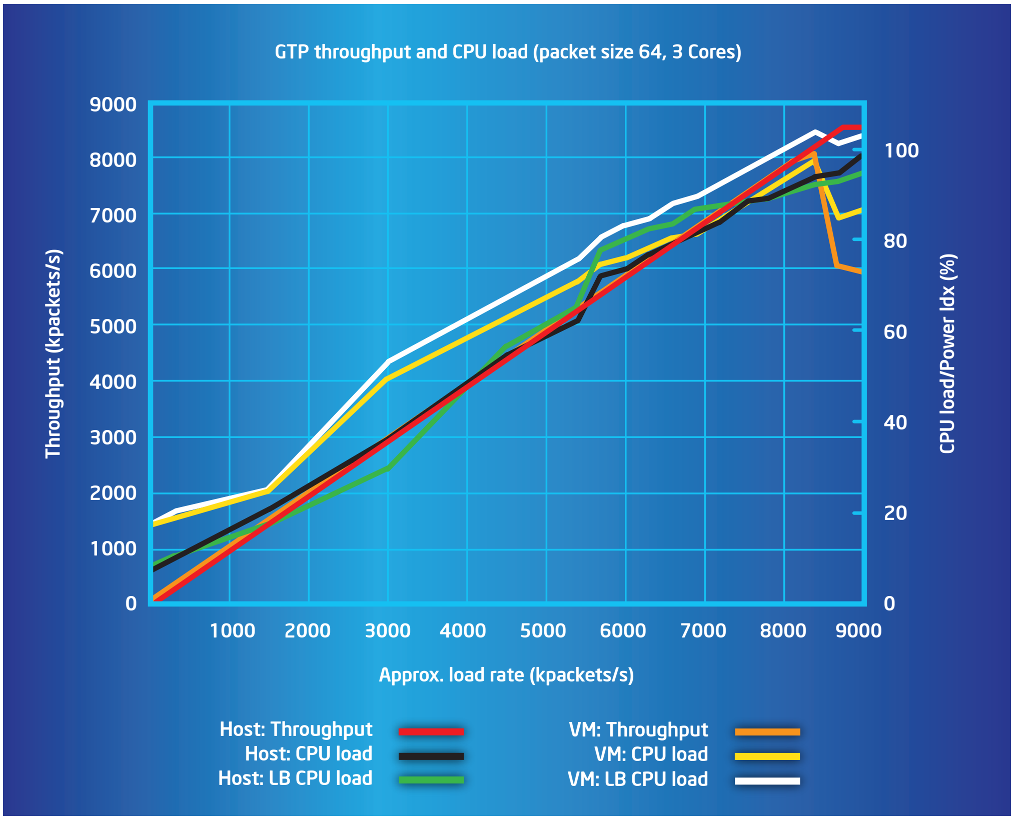


Figure 6. Maximum GTP Throughput With Three Cores in VM and Non-VM (Host) Environment

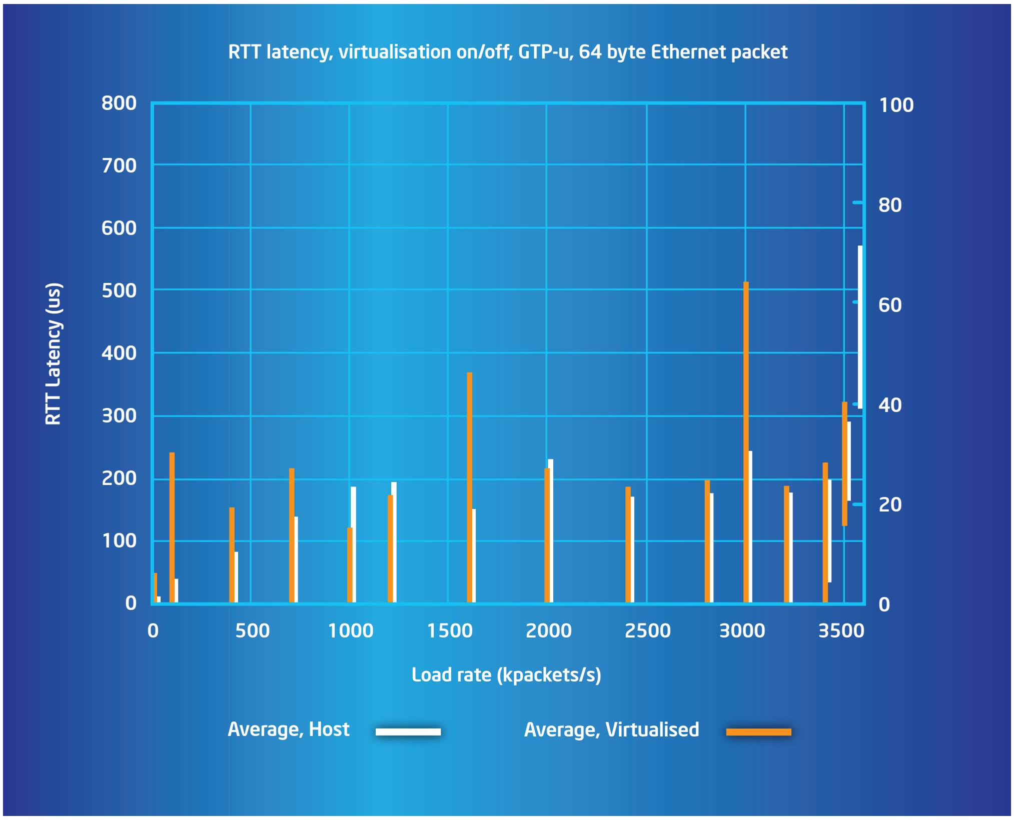


Figure 7. RTT Latency With and Without Virtualisation (Host)

Power Management in a Virtualised Environment

In the Carrier Cloud Telecoms architecture running on Linux, a specialized power management algorithm replaces the standard Linux power management (basically developed for laptops) because it tends to be inefficient, especially when the workload increases. The standard Linux implementation makes decisions based on CPU load history, whereas the approach taken in the Carrier Cloud Telecoms initiative is based on the detected future workload. Low-level power management APIs are only available in a non-virtualised environment and virtualised machines cannot perform power management (e.g., manage CPU frequencies). To overcome this, the initiative developed a power management agent that is part of the control plane of the cloud virtualisation infrastructure and not merely within the virtualisation environment.

The Carrier Cloud Telecoms demonstration scales the number of resources in the cloud depending on individual CPU loads. Load, as measured on a single processing unit (e.g., a core), not only depends on the number of instructions executed by that individual core, but also on the resource contention amongst other cores (i.e., execution units (physical and logical), cache, memory bandwidth, I/O devices, etc). This means other tenants may actually increase the load on other unrelated tenants if they share physical resources in some way. As more physical hardware topology and physical hardware load is hidden by virtualisation, it becomes more difficult to manage the cloud.

Power management in previous phases of the initiative was implemented using frequency management through the Linux userspace “frequency governor” API and CPU load/C-state transitions triggered by an adaptive algorithm that for brief periods of time suspends the application. The Linux userspace frequency governor API is not available in current hypervisors; therefore, the current phase of the program and the associated demonstration implemented a control channel from the VM to a frequency management agent in the hypervisor. The frequency target calculations are implemented by the IP-stack within the VM based on current load levels and lengths of ingress packet queues (i.e., attributes that are intrinsic to the IP-stack). The resulting frequency targets are communicated to the frequency management agent, which propagates it through the Linux userspace API.

CPU load throttling and C-state transitions, triggered through adaptive IP-stack and application suspension in the VM, need to be treated slightly differently with current hypervisors. This is because the suspension is implemented by starting a timer with the current suspend interval and then waiting for that timer to expire. Using timers from a VM with current hypervisors will trigger an exit from the VM into the hypervisor, which results in timers having higher overhead than when used natively (e.g., from a host Linux OS). The consequence is that short suspends are not possible, and power management from a VM has a slight overhead compared to the native implementation shown in previous phases of this program.

The challenge has been mitigated by placing a lower limit on the sleep value, but it inevitably causes a small shift of the load curve upwards compared to the non-virtualised environment. The overhead will typically manifest itself at low to medium loads, with the worst-case overhead occurring at very low loads (just above idle), and may be as large as 100%. The overhead decreases as throughput is increased, and at around 30% throughput, the difference is negligible.

The conclusion is that virtualisation is an excellent capability, but there are use cases and situations where knowledge and access to the physical hardware may still be necessary; at least until inter-VM software switching performance improves. Also, as implemented in the demonstration, the underlying cloud infrastructure has to provide control plane mechanisms that allow the guest to manage low-level features like power management.

OpenStack

IaaS computing resources are pooled to serve the needs of applications that require different physical and virtual resources. In the cloud, these resources are dynamically assigned and reassigned on demand. Examples of computing resources include storage, processing (compute), memory, network bandwidth and virtual machines.

In addition, a traditional IT cloud enforces location independence, therefore, the application generally has no control or knowledge over the exact location of the provided resources. This model is not sufficient for the telecoms cloud; instead, a more specific mechanism for placement is needed.

The Carrier Cloud Telecoms demonstration uses OpenStack to implement the management of virtualised resources as part of the Cloud Management Infrastructure. The benefits of using OpenStack include:

- Uses open source software
- Simplifies the management of complex virtualised systems

- Allows heterogeneous VM environments (KVM, Xen*, VMware*)
- Supports a well-defined IaaS interface with Amazon* Elastic Compute Cloud (EC2) compatibility
- Implements a REST base remote management interface

At a high level, OpenStack is built from five components:

- Nova - cloud infrastructure "runtime"
- Glance - image management and OCCI management interface
- Swift - object store
- Keystone - user management and authentication
- Quantum – network infrastructure

The Nova component is composed of:

- Nova API - OpenStack IaaS API
- Nova compute – manages the VM lifecycle
- Nova volume - disk management
- Nova scheduler - schedules cloud resources for Nova API calls
- Nova network - handles host machine network configurations

The demonstration uses these components as follows:

1. The VM images are stored in the Glance image store
2. Each user is defined in Keystone, and prior to the OCCI call, the user needs to authenticate via Keystone
3. VMs are started and stopped remotely via OCCI - Internally OCCI calls the Nova API
4. The Nova API and Nova compute are used for running the VMs, with the VM lifecycle being fully managed by Nova compute

Quantum is used to configure a private network between the OpenStack management node and virtual machines. Unfortunately, Quantum cannot be used at present to configure SR-IOV networks. Therefore, in the demonstration, the hypervisor bypass was configured manually to the host system.

Given the fact that OpenStack is primarily developed for the IT cloud realization, there are some modifications and extensions that will be necessary when orchestrating a telecoms cloud using this technology. For example, the VM placement mechanism, which is part of the Nova scheduler, is currently too limited to support the complex Telecom requirements, like placement with high availability (2N or N+1) or explicit assignment of computing resources such as pinning a VM to specific set of CPU cores. In the future, Telco-specific extensions need to be added to OpenStack to support these and other requirements. These extensions are being considered as part of future phases of the Carrier Cloud Telecoms program. Note that from a standards perspective it remains to be seen which Cloud Orchestration path will find greatest acceptance in the industry.

Software Defined Networking

It is important in the Telecoms cloud to maximize resource utilization but to do it in an energy efficient manner, which is why the elasticity (aka workload distribution) of the computing resources is very important. In the demonstration, this resource elasticity is realized in both the SGW and PGW.

A good example to consider is how a software-defined network handles UE packets when the PGW they are attached to is removed. In the telecoms cloud, this scenario is a realistic one since resources are elastic, and the system is capable of shutting down elements and resources on demand. Once the PGW is removed, there is no network element in place to receive the UE's incoming packets.

This is where software-defined networks and OpenFlow can help. Since it is not possible to control the incoming packet from the serving network, it is necessary to modify the traffic flows. In the Carrier Cloud Telecoms demonstration, this is achieved using the OpenFlow protocol and Open vSwitch.

The SDN setup in the demonstration has four levels of hierarchy: blade, CPU, VM and core. From the SDN perspective, only core is relevant since all PGW VMs have the same IP address for outgoing IP packets. When a packet arrives to the edge of the cloud, the PGWs need to do core identification (i.e., determine which PGW instance the subscriber is currently assigned to). This is implemented using VLANs. Now, assume a subscriber has an active tunnel in the PGW running on core 2 with VLAN tag 2. The system decides to halt the PGW on core 2; therefore, all the traffic from that PGW needs to be handed over to another PGW. Assume that the handover target is PGW on core 1 with VLAN tag 1. The handover is implemented using Open vSwitch and OpenFlow as follows:

1. The system implements PGW pseudo-handover and creates a new tunnel for the subscriber to the PGW on core 1.
2. Once the pseudo-handover completes the PGW on core 1, it uses OpenFlow to request Open vSwitch to modify the VLAN tag of the user to 1 instead of 2. The user is identified using the subscriber IP address.

If the PGW with active tunnels is shut down, a handover procedure is executed for each of the tunnels. In the handover, a new tunnel is established to the PGW, which is still active. While the handover executes, the old tunnel remains active (i.e., traffic continues to flow through it). Once the new tunnel is established, the PGW, which is being shut down, contacts the OpenFlow controller and requests that packets initially targeted for it are redirected to the new PGW. In order to avoid situations where the new PGW is not able to receive all the connections, the PGW selection is orchestrated by the Cloud Controller, which is aware of the PGW capacities and loads.

Handover from the LTE to 3G network is also realized using Open vSwitch and OpenFlow. Once the handover is initiated by the simulator, the attach procedure is implemented towards the 3G network to establish a new tunnel. The demo uses same IP address for both LTE and 3G users. After the tunnel has been created, the eNodeB requests the OpenFlow Controller to modify the incoming packet flow. The OpenFlow Controller uses OpenFlow to modify flow tables in the Open vSwitch in such a way that each incoming packet for the particular user gets sent to the GGSN instead of the PGW. This is realized by changing the gateway IP address.

Another problem related to the PGW handover is associated with the address resolution protocol (ARP) packets. Ethernet uses ARP to obtain the MAC address of the machine where a specific IP packet is being sent. In theory, the demonstration should have an ARP proxy that is aware of where each UE is connected. This would be quite a slow solution; therefore, the demonstration has a single ARP proxy that responds to each ARP request and returns the same MAC address each time. Then the PGW uses OpenFlow to define a correct MAC address for a particular UE IP address.

In addition, the load balancer needs to identify the target element within the VM. Each VM has a maximum of three element instances. The identification is implemented using VLANs. Each element within a VM has its own VLAN. OpenFlow is used to tag incoming packets with a correct VLAN tag so the core load balancer can distribute the packet to a correct element instance.

Based on this schema the following benefits were identified:

- Minimized downtime
- Reduced security risks
- Simplified and more flexible implementation of the PGW handover

Going forward, more work in this area is planned to determine how this schema can be used in more complex network scenarios. In addition, the whole question of migration of legacy systems needs to be dealt with.

The system also uses Open vSwitch to connect the VMs that implement the SGSN and GGSN applications. Open vSwitch is open source and provides flexible inter-VM communication. The challenge with Open vSwitch is the performance latencies it currently possesses. It requires optimizations and support from the latest high performance networking technologies such as the Intel DPDK. Open vSwitch optimisations with Intel DPDK are something that will be addressed in future phases of the Carrier Cloud Telecoms program.

IPSec

Traditionally, base stations have been placed at operator facilities that are physically secured. In LTE, however, base stations are increasingly being placed in less secure environments, and IPsec is employed to ensure confidentially and integrity for signalling and user plane traffic. Using cloud-based architectures, and thus shared infrastructures, also significantly increases the need for IPsec to provide security between nodes.

In the context of the Carrier Cloud Telecoms initiative, IPsec is used to secure the backhaul links between the eNodeB nodes and the SGW/MME nodes where both GTP-U and SCTP traffic is carried through IPsec tunnels. As illustrated in Figure 8 on the next page, adjacent eNodeB nodes may communicate with each other through an IPsec tunnel (the LTE X2 interface), and each eNodeB will communicate with at least one set of MME/SGW

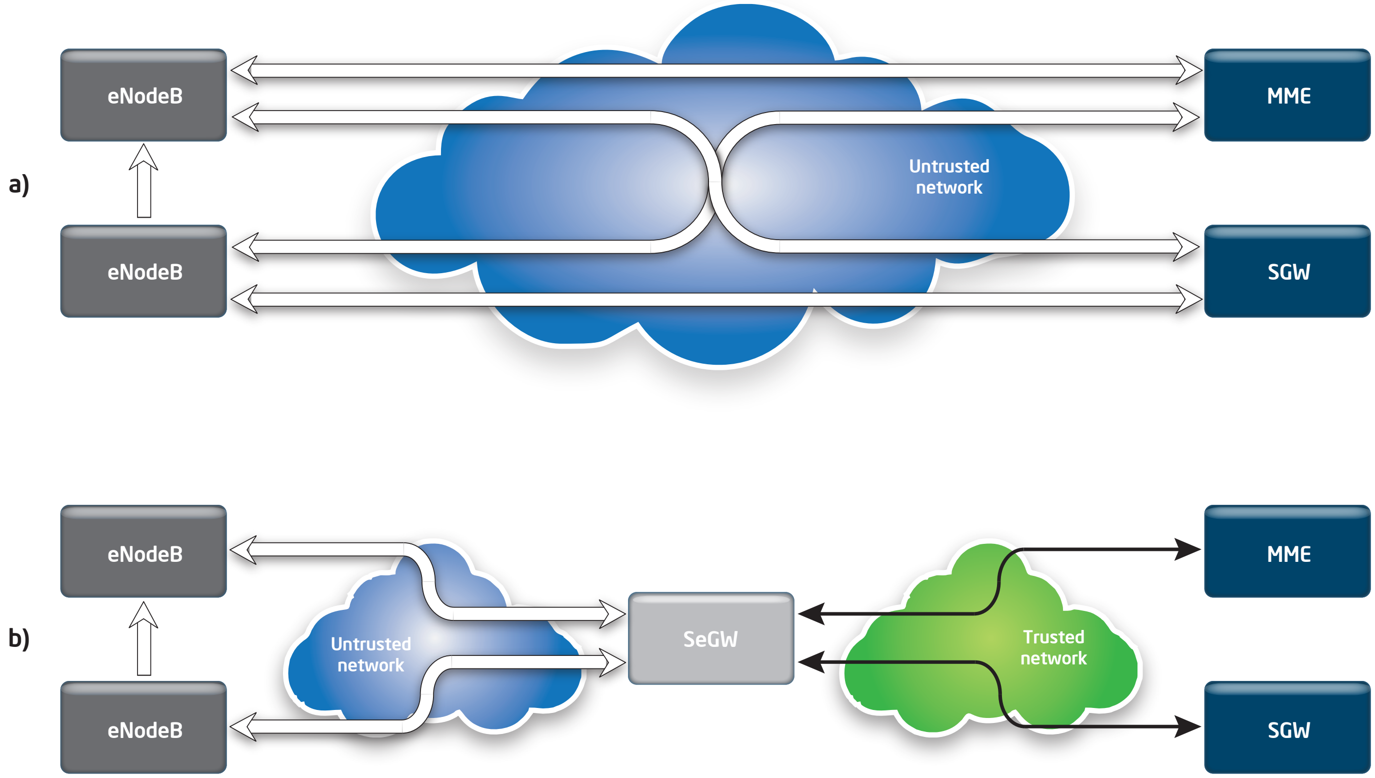


Figure 8. Two possible uses of IPsec tunnels between adjacent eNodeB's and core network nodes MME and SGW; **a)** direct connection from eNodeB to core network nodes and **b)** connection through a separate security gateway. In the latter case, the IPsec tunnels are potentially terminated at a different node than SCTP and GTP.

nodes through IPsec tunnel(s) (the LTE S1 interface). Since IPsec is essential for securing the backhaul network, IPsec performance demonstrations are very relevant when considering the Carrier Cloud Telecoms IP-stack functionality.

IPSec Acceleration Support

There are two possible ways to implement IPsec decryption:

Hardware based: IPSec acceleration is possible using support of an external accelerator, like the Intel® Communications Chipset 89xx series, which is part of the latest Intel® Platforms for Communications Infrastructure. The platform contains one or more Intel Communications Chipset 89xx series based devices, and each of these contains a number of dedicated crypto engines. These engines may be used for asynchronous look-aside offload of IPsec encryption/decryption.

Software based: Many recent Intel processors support Intel® Advanced Encryption Standard New Instructions (Intel® AES-NI), which improves the speed of AES encryption/decryption. For example, high-performance software crypto support, implemented via Intel AES-NI, is present in Intel® Xeon® processor E56xx series and E5-2600 v2 product family. In a multi-core Intel processor with these architecture features, each core will be able to use Intel AES-NI, and parallel IPsec encryption/decryption by each core is possible. In addition, software-accelerated IPSec performance can be enhanced by leveraging software acceleration libraries such as the Intel Multi-Buffer Crypto for IPSec Library.

In the Carrier Cloud Telecoms initiative, Tieto integrated the Intel Multi-Buffer Crypto for IPSec libraries. Figure 9 shows characteristics of a single non-virtualised logical core (Intel Core i7-3820 processor @ 3.60 GHz) servicing GTP-u traffic through an IPsec tunnel. The IPsec tunnel is configured to use manually

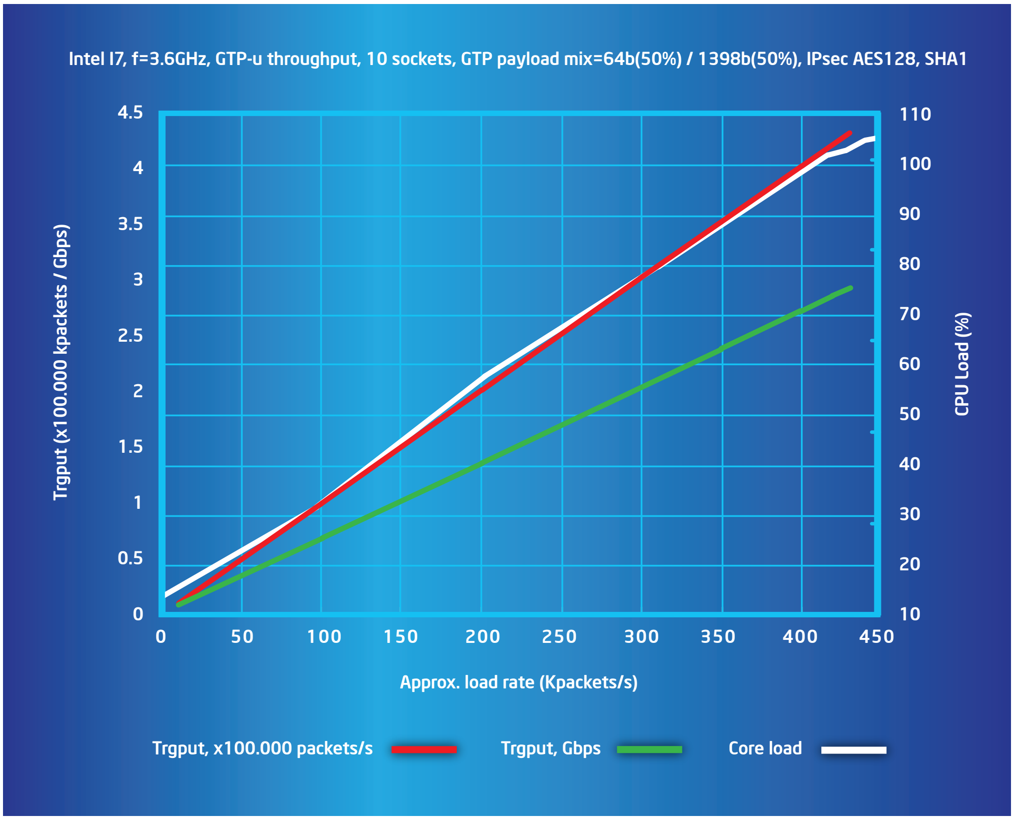


Figure 9. Single core IPSec performance using the Intel® Multi-Buffer Crypto for IPSec Library.

keyed AES-128 encryption and SHA1 authentication. Packet sizes are using Internet traffic distribution (50% each of 64 and 1398 byte GTP-u payload). IPsec performance achieved is as follows:

2.905 Gbps/s at 430k packets/s load rate[2]

These numbers are one-way only, i.e., actual IPSec throughput/ operations are twice the quoted numbers.

Cloud Telecoms Management

The NIST (National Institute of Standards and Technology) definition of Cloud Computing states that:

"Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model is composed of five essential characteristics, three service models (IaaS, PaaS and SaaS), and four deployment models."

This impacts not only cloud computing itself, but also the technology and processes governing how cloud computing is managed, including provisioning, configuration and administration processes.

Telecom Cloud is one very important aspect when speaking about the future evolution of the cloud; more and more mobile network elements will be running on cloud infrastructure, providing new opportunities for developing different kinds of flexible services for end-users. Telecom Cloud creates not only challenges for cloud management systems, but also opportunities for the new companies to provide (centralized) management systems and/ or services for the operators and end-users. From the operator perspective, this kind of flexibility offers the potential for reduced costs and faster time to market (and to money) for new services.

A challenge currently facing cloud management is the lack of widely agreed standardized interfaces between the cloud infrastructure and cloud management system. There are some standards-based cloud management interfaces such as:

- CIMI (Cloud Infrastructure Management Interface), developed by DMTF's Cloud Management Working Group (CMWG).
- OCCI (Open Cloud Computing Interface), developed by OCCI working group and the OpenStack Foundation.

Cloud Management in the Carrier Cloud Telecoms Initiative

An important and challenging aspect from the management point of view is the elasticity and dynamic behaviour of the cloud infrastructure, as in when virtual machines are started and stopped, along with changing traffic volume. Intelligent management of dynamic behaviour keeps the amount of reserved

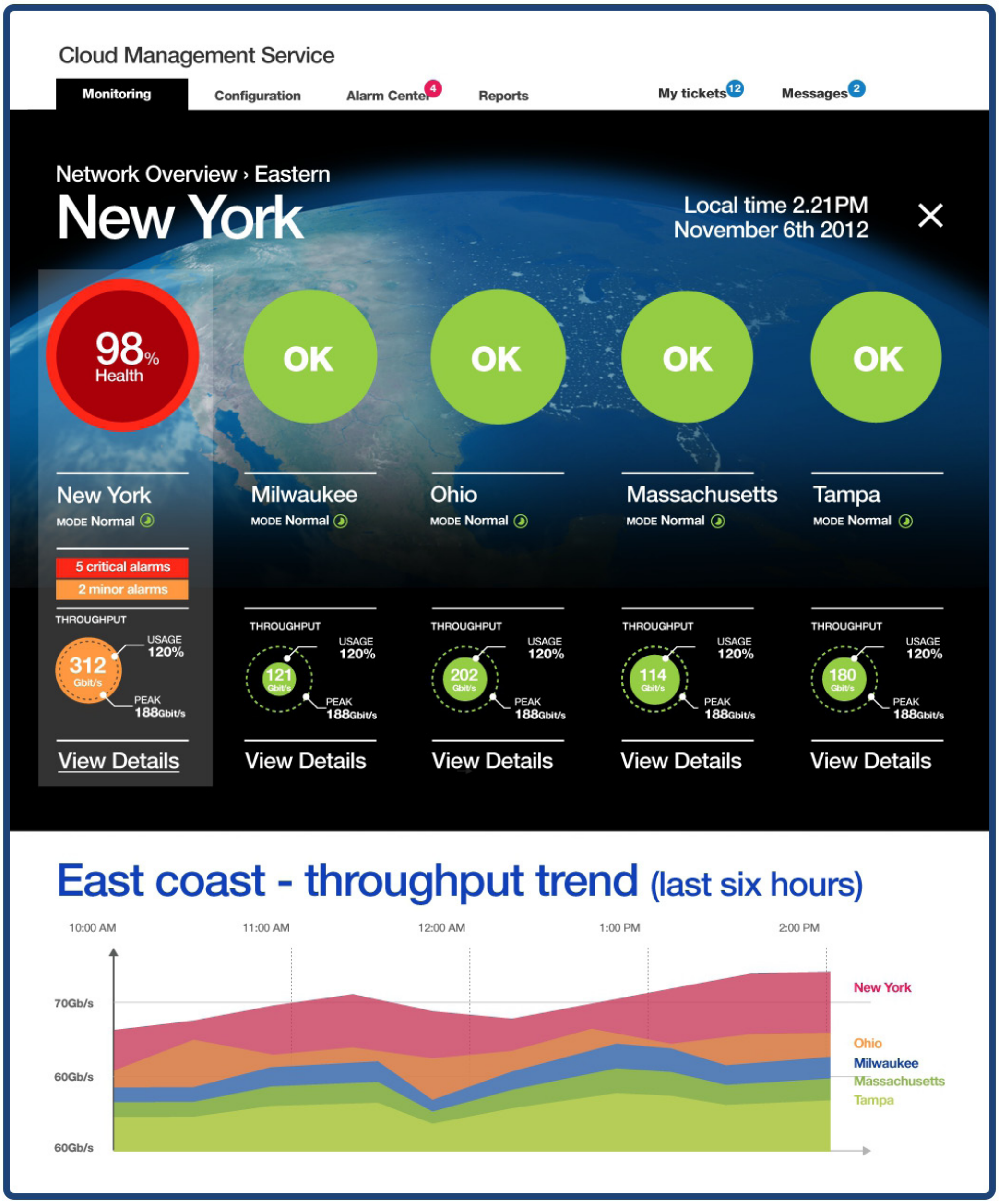
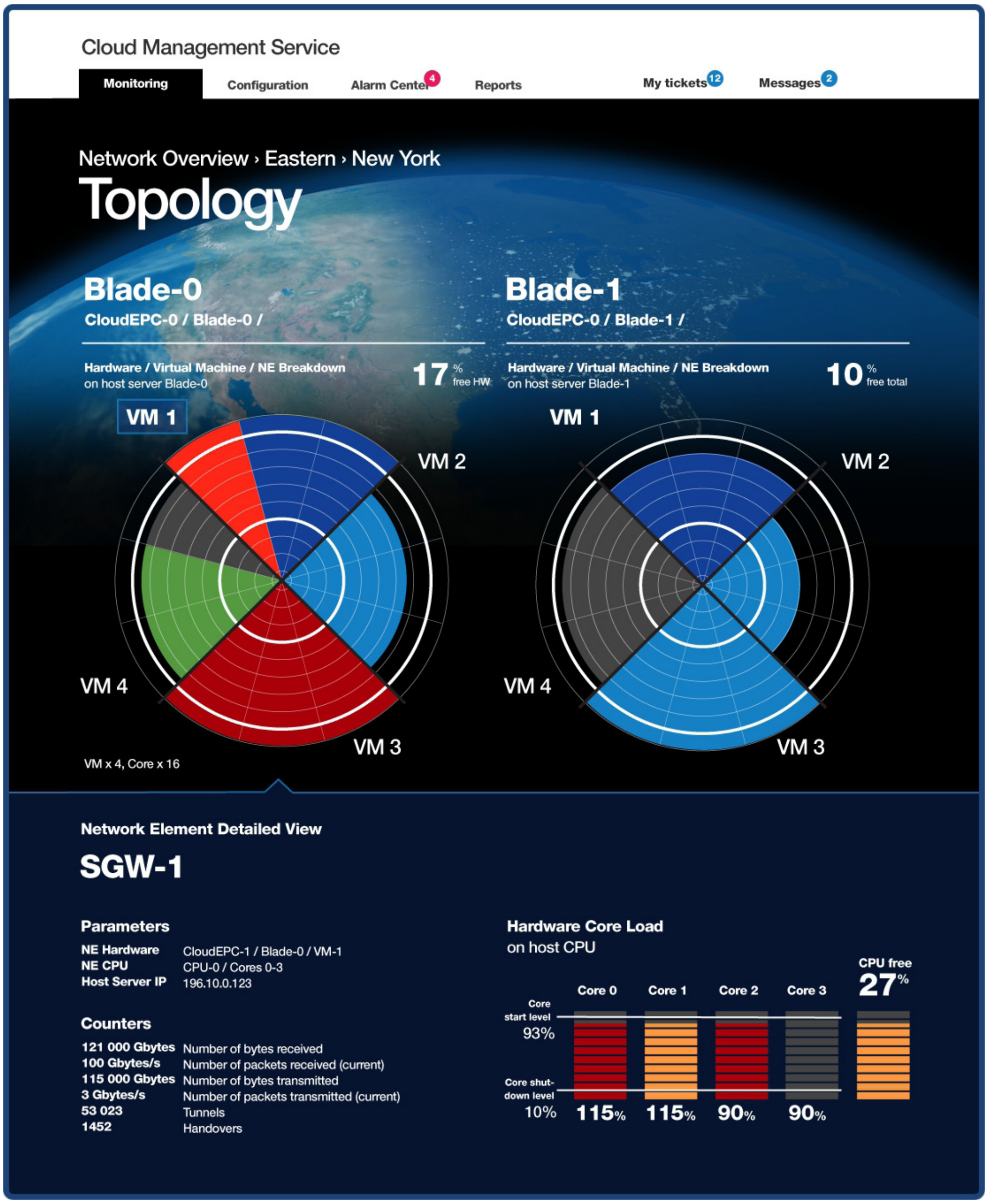


Figure 10. The Cloud Management System in the Carrier Cloud Telecoms Demonstration

and used hardware at an optimal level, so hardware is reserved and used only when needed. This dynamic behaviour is managed in the cloud management system by showing the state changes of the VMs and the network element instances in real-time and ensuring the topology of the cloud infrastructure is always up-to-date.

The Cloud Management system used within the Carrier Cloud demonstration provides a centralized management system for the telecom cloud, containing support for cloud infrastructure management: topology, monitoring (alarms and counters) and parameter management for some selected parameters used by the demonstration cloud infrastructure.



Parameters

NE Hardware	CloudEPC-1 / Blade-0 / VM-1
NE CPU	CPU-0 / Cores 0-3
Host Server IP	196.10.0.123

Counters

121 000 Gbytes	Number of bytes received
100 Gbytes/s	Number of packets received (current)
115 000 Gbytes	Number of bytes transmitted
3 Gbytes/s	Number of packets transmitted (current)
53 023	Tunnels
1452	Handovers

Figure 11. Topology Management in the Cloud Management System

In the demonstration, VMs and network elements are deployed, started and stopped dynamically. This kind of elasticity and dynamic behaviour creates requirements for the management system when the active topology, current state and state changes of the VMs and network elements must be presented clearly for end-users.

The Cloud Management system contains:

- A clearly visualized and easy to use Web user interface (UI).
- Real-time and automatic topology creation and update to support monitoring and parameter management.
- Real-time state monitoring of VMs and network elements.
- Real-time monitoring, including alarms and performance counters collected from cloud infrastructure.
- Configuration management, including support for a subset of cloud infrastructure parameters.
- A standard-based CIMI interface between the cloud infrastructure and cloud management system.

The elasticity and dynamic behaviour of the cloud infrastructure is visualized in the cloud management system by showing the state (active or inactive) information of each VM and network element instance in real-time. This means the end-user is able to see in real-time which VMs and/or network element instances are active at any given time, and can also see how traffic increases and decreases impact the behaviour of the cloud infrastructure. A standardised CIMI interface was used between the cloud infrastructure and cloud management system. It is important to note that the standardisation work of the CIMI interface is still ongoing, and it is not known which management interface will achieve widespread industry acceptance and deployment in the future.

Real Time Topology Management and VM State Monitoring

The cloud management system supports use cases related to topology, collecting alarms, collecting counter values and managing parameter values. The system creates, updates and visualizes the object topology automatically, and in real time. It also monitors and visualizes in real time the states, and changes in states of VMs and network elements. The object topology is updated during the demonstration; and as a result, the topology is always up-to-date in the cloud management system. The topology is presented in a Web UI for the end-user. The end user can also see in the Web UI which VMs and network elements are active ones and which are in-active. This provides the possibility for real-time monitoring of the usage of cloud infrastructure hardware.

Summary and Conclusions

A number of extremely useful technologies exist to address network resource utilization challenges in the telecoms network. These technologies offer the potential to evolve the network architecture to meet the challenges of the mobile data explosion, but also offer the potential to provide an environment to support the creation of new service and revenue streams that can take advantage of the phenomenal consumer and social dynamics now affecting the industry. These technologies have their origins in the IT world, and as such, they need to be tailored to the needs and requirements of the telecoms equipment industry.

This paper explored the maturity of available technologies and investigated their potential, performance and associated challenges as part of a comprehensive technical initiative and demonstration. This collaboration by Tieto and Intel demonstrates the potential for these technologies to be successfully deployed in Cloud Telecoms. However, widespread industry adoption, and the pace of adoption, will require a collaborative approach between service providers, network equipment providers, silicon vendors and ISVs.

For more information about Tieto, visit:

http://www.tieto.com/what-we-offer/rd-services

http://www.tieto.com/carriercloud

For more information about Intel® solutions for Telecoms, visit: www.intel.com/go/commsinfrastructure

[1] Two previous phases of the joint initiative have already been completed, which are known as “Cloud EPC” Phase 1 and Phase 2 respectively.

[2] Benchmark testing was performed in April 2013 at Tieto A/S, Aahave Parkvej 27, 8260 Viby, Denmark.

Printed in USA   MS/VC/0813   Order No. 328869-002US