UNIVERSITY OF ALBERTA

Caching in LTE networks using Software-Defined Networking

Project Supervisor:

Prof. Gurpreet Nanda

Acknowledgement

I would like to express my sincere gratitude to Prof. Gurpreet Nanda for making this project-work possible. He taught me many things, such as how to proceed in a scientific way.

I am highly indebted to him for his guidance and supervision as well as for providing necessary information regarding the same and also for the support in completing it. I can’t say thank you enough for his tremendous support and help. I would like to express my gratitude to my faculty members as well for extending their support.

Harmandeep Kaur

Abstract

This report provides an overview of the current LTE architecture and the proposed solutions to integrate Software Defined Network (SDN) technology. Mobile operators are looking for a technology which helps them in increasing the network capacity, in order to fulfill the increasing demand for data and helps them from customers perspective as well with low network costs. The integration of SDN into mobile networks become Software Defined Mobile Network (SDMN). Firstly, it talks about the integration of SDN in LTE networks which results in the replacement in the current mobile backhaul network with SDN based network.

Moving Further, it describes the designed solution for the same aim which utilizes Software-Defined Networking Technology. Also, presents the results which are based on the performance analyzation. In the end, the report illustrates about the relocation of the cache on the basis of which benefits from both mobile operators and users points of view could be analyzed.

Contents:

1) Overview:

Future mobile network architectures should mould themselves in such a way that they are able to cope with the future demands which are as following:

High Bandwidth;

Services according to the requirements;

Low energy consumption;

High-Level security;

Proper utilization of spectrum.

Specifically, it is expected that in the future, there will be an increment in the number of mobile users. The result of this leads to high consumption of data which means higher consumption of Bandwidth and Resources. This in turn increases the operators' cost in order to maintain their network capable of providing internet access to all their users.

Moreover, users are not willing to pay more for these services, which leads to decrement in profit for the operators and it has been shown in a recent study as well.

Also, in order to keep up with the future demands, mobile networks architectures should be in a way that they are optimizing the resources which leads to adding new components or technologies. However, mobile backhaul networks contain exceptionally complex and inflexible segments and furthermore mobile administrators don't have adaptability to "blend and match" abilities from various vendors. Also, Mobile Operators are searching for an answer which gives sought administrations: higher network capacity and augmenting the use of accessible assets to the clients and gives benefit to them by decreasing the Operative Expenses (OPEX).

On these grounds, Software defined networking (SDN) is one of the promising technologies that are relied upon to understand these restrictions in current mobile systems. SDN gives the required upgrades in adaptability, versatility, and execution to adjust the mobile system to stay aware of the expected development. Caching is the solution embraced for this issue which has been as of now utilized by Internet Service Providers (ISP) in wired systems keeping in mind the end goal to enhance their customers' experience and lower the utilization of bandwidth. In this paradigm, every administrator has the adaptability to build up his own networking ideas, optimize his system/network, and address particular needs of his endorsers.

The introduction of Software defined networking (SDN) into Long−Term Evolution (LTE) mobile networks gets the economic preferred standpoint in two ways: Initially, SDMN requires cheap equipments, for example, commodity servers and switches rather than costly mobile backhaul gateway devices. Second, the acquaintance of SDN innovation with mobile networks permits entering new components in the mobile network ecosystem such as independent software vendors (ISV), cloud providers, and Internet service providers (ISP) that will change the business model of mobile networks.

“A cache server is a dedicated network server or a service which acts like a server that saves web pages or other internet related content locally”. At the point when a demand lands for a data which is not put away in cache server, the cache server will recover that specific data from the first web server and stores a duplicate of it for further demands. By doing so, the cache server both decreases request on an enterprise’s bandwidth and furthermore accelerate the access to information. With the assistance of cache server, the contents are presently served from an area which is near clients, so along the packets need to travel less which thus diminishes the utilization of network assets. It likewise helps in lessening peering costs. From the client perspective, caching grants to enhance the quality of experience (QoE), since the content is stored nearer to the client which diminishes the inertness and furthermore helps in expanding the downloading speed.

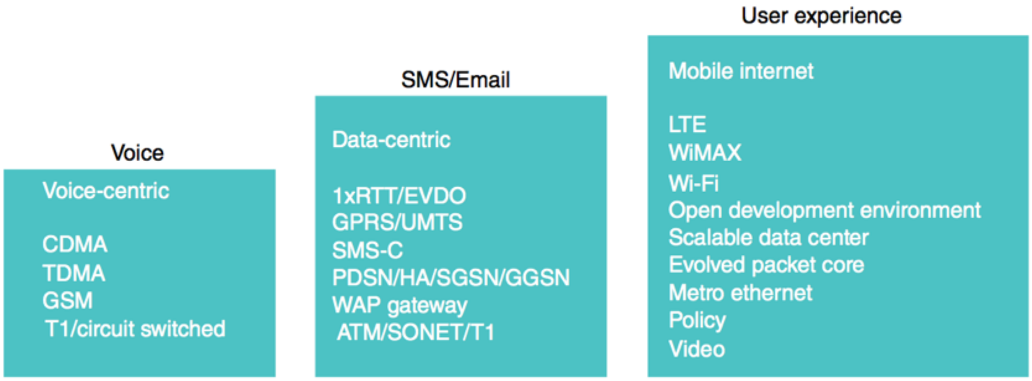
2) Mobile Network:

i) Overview:

In 1990, landlines were the huge moneymaker for the service providers (SP's). As the interest for mobile services has transformed from the earliest starting point of the commercially accessible GSM in the mid 1990s to the development to a packet−based engineering in GPRS, to a more hearty service in UMTS, and to the more develop plans in LTE and past, the supporting centre design has additionally changed. The underlying requests and drivers for a mobile network have likewise changed throughout the years. While voice was the essential administration amid the underlying designs, information(data) and video have overwhelmingly predominated voice.

ii) The Evolution of the Mobile Networks:

The network has made some amazing progress from the times of the 56K rented lines and T1s alongside the vast, awkward mobile telephone in the mid 1990s. Today, as the mobile endpoints turn out to be increasingly various, the measure of information required to serve these endpoints turns into an issue alongside dealing with the network access of these mobile devices. The Internet of Things is adding to the expansion in mobile traffic drove by the move to more quick witted mobile gadgets, rise of wearable gadgets, and the expansion in machine−to−machine (M2M) associations. Today, the new data centre engineering touts virtualization, coordination (orchestration), and versatility(scalability).



Sharing Resources:

Back in the 1980s, since voice was the fundamental driver for all SP business, the fledgling data network, including the mobile information network, was designed according to the current voice network. Voice was carried over committed time-division multiplexed (TDM) lines intended to firmly pack in the recently digitized voice traffic into 64-kilobit channels.

Today, the thoughts of sharing register and capacity assets are not new ideas. They are an old idea connected to a more present specialized engineering. The mobile SP’s in the mid 1990s utilized the current innovation to share what assets they had between clients keeping in mind the end goal to effectively convey voice services. For example: In the United States, a provider could multiplex 28 T1s into a single T3 and further multiplex many T3s into larger optical circuits.

As fresher multiplexing innovations were presented, the consequence of sharing those assets was that the cost of a solitary T1 kept on dropping. Asynchronous Transfer Mode (ATM) was likewise used to share high data transmission associations among numerous clients. ATM gave the higher transfer speed and quality of service (QOS) required for developing networks that were transporting voice, video, and information (data).

Furthermore, once the foundation was introduced, new clients were rapidly and effortlessly added with relative to before. As the network transitioned to a greater extent a packet-based design and voice and information were joined onto a similar network, sharing data transfer capacity now enhanced considerably all the more, empowering the ability to share more assets. Circuits were no longer utilized per client however could now convey different clients and applications, additionally moving the design to a common space. Where before, in the circuit-switched world, a committed circuit must be set up and kept up during a call, the new packet-based network could share transmission capacity on a per-packet level. As we moved to the packet-based network, other innovation added to the proficiency of the overall architecture. For instance, silence suppression on voice trunks was one element that would not transmit packets if an end client were not talking, along these lines expanding the sharable data transfer capacity (bandwidth) of the network.

However sharing transport assets, the mobile packet core was likewise on the move. Where before, devoted assets must be introduced and kept up for each particular capacity of the mobile bundle centre, now these capacities can be shared on a similar processing platform.

Orchestration:

Computing orchestration may be defined as "the automated arrangement, coordination, and management of complex computer systems, middleware, and services." In layman's terms, orchestration is the higher−level coordination of hardware and software components, stitched together to support a particular service or application, and the active management and monitoring of that network. In the beginning of communication, a telephone call was started by picking up the receiver and telling the administrator the number that you needed to interface. The administrator then physically associated a wire to the following hop of the circuit. Since most calls were neighbourhood or local, the administrator associated the two guests together. At the point when the call was finished, the administrator needed to detach the circuit.

In the beginning of commercial mobile administrations, mobile SP's held an imposing business model on the network, the devices associated with the network, and the applications that kept running over the network. Improvement of new applications took months, if not years, to make or create. At the point when another application was prepared to be brought into the network, the coordination (orchestration) of that implementation was influenced by a few components. Computing technology was still being developed and Communication was moderate, network visibility was restricted, and the inspiration to quickly actualize the administration was deficient. Rivalry was not yet a driver for upgrades in this procedure.

Today, with the multiplication of shoddy memory, vast database abilities, quick network interchanges, virtualized processing, and standard associations, the situation above can be automated and finished in a small amount of the time. The idea is still the same. A focal figure directs what should be done at every area in the network. Once the fulfillment has been recognized, the administration or application can now be put into administration.

Scalability:

As mobility services developed in popularity, the SP's expected to address scalability issues. These issues included extending points of presence (POPs) to encourage extra gear to end more clients and extra circuits for core limit alongside figuring out where to interface these new core connections. The result of the progressions with network configuration gave sufficient data transfer capacity in the centre, so QOS was not tended to around there. The entrance into the network from the edge was every one of that was expected to address QOS. As data usage grew, providing the customer with a good experience had to be addressed. Because of the high cost of core transfer speed, it rapidly got to be cost restrictive to continue adding enough ability to deal with peak data load in the core. Network displaying advanced to incorporate a more mind boggling QOS system, which addressed the increased cost of adding additional bandwidth. SP's need to guarantee that signalling traffic takes the most brief way so as to protect both voice calls and information sessions as clients are moving, which require the voice call or information session to changing, starting with one base station then onto the next.

Scalability configuration usually involves a trade−off of most extreme size of the network components (the quantity of connections, hubs, and conventions or protocols) and least accessibility. SP's need to guarantee that the network will stay stable keeping in mind the end goal to give the best client encounter additionally need to grow the network. This will keep on being one of the fundamental difficulties with the advancing mobility networks.

iii) Current Mobile Networks and their limitations:

The mobile communication was introduced in the 1980s. The first generation of mobile networks supports only the voice call services and the connectivity speed up to 56 kbps. However, the mobile network innovation accomplished a colossal advancement amid the most recent four decades. Today's mobile network bolster different system administrations, for example, amended mobile Web access, Internet Protocol (IP) telephony, gaming services, high− definition mobile television (TV), videoconferencing, 3D television, cloud computing, and high−speed broadband connectivity up to several Gbps.

It is challenging to fulfill all these prerequisites by utilizing present−day mobile network engineering. Present−day mobile networks are confronting different impediments, and they can be arranged as beneath:

- ***Scalability Constraints:*** The fast augmentation of mobile traffic utilization is anticipated because of new bandwidth−hungry mobile administrations, for example, online streaming, video calls, and high−definition mobile TV. The current static over-provisioned mobile networks are unbendable and expensive to scale to stay aware of the expanding traffic demand.
- ***Hand-typed network configuration:*** Most of the network administration frameworks are physically intensive, and prepared administrators are required to

accomplish even moderate security. However, these manual arrangements are inclines to misconfiguration blunders.

- ***Mind boggling and costly network devices:*** Some of the mobile backhaul gadgets need to deal with broad measure of work. For example, Packet Data Network Gateway (PDN GW) is in charge of numerous imperative Data plane (DP) functions, for example, traffic observing, quality−of−service (QoS) administration access control, and parental controls in LTE networks. In this way, the gadgets are unpredictable and costly.
- ***Rigidity:*** The standardization procedure for mobile networks is a long−lasting procedure. It requires numerous months or years to present new administrations. Besides, the usage of new administration likewise takes weeks or months because of the physically intensive service activation, delivery, and confirmation.
- ***Complex network administration:*** Significant skill and platform assets are required to deal with the present mobile network. Much of the time, backhaul devices are missing of regular control interfaces. Hence, straightforward assignments, for example, configuration or strategy authorization additionally require a lot of exertion.

3) LTE Mobile Networks:

In media transmission, "Long haul Evolution(LTE) is a standard for high-speed wireless communication for data terminals, in light of GSM/EDGE and UMTS/ HSPA innovations." This standard is produced by the 3GPP (Third Generation Partnership Project) and is indicated in its Release 8 document series. The gathering of media communications organizations that created it, known as the Third Generation Partnership Project (3GPP), picked the name because LTE evolved out of the current 3G innovation and did not entirely supplant it, similar to the case with the third era of GSM-based remote innovation (known as UMTS).

LTE systems can transmit information 10 times speedier than 3G systems. They're about three times speedier than the propelled High-Speed Packet Access+ (HSPA+) systems.

i) The Motivation for LTE:

- Users demand for higher data rates and quality of service.
- Packet Switch optimized framework.
- Users continued demand for cost reduction (CAPEX and OPEX).

- Low unpredictability.
- Requirement for evasion of pointless discontinuity of technologies for paired and unpaired band operation.
- Need to ensure the continuity of competitiveness of the 3G system for the future.

ii) LTE Overview:

LTE (LongTerm Evolution) or the E-UTRAN (Evolved Universal Terrestrial Access Network) introduced in 3GPP R8, is the access part of the Evolved Packet System (EPS).

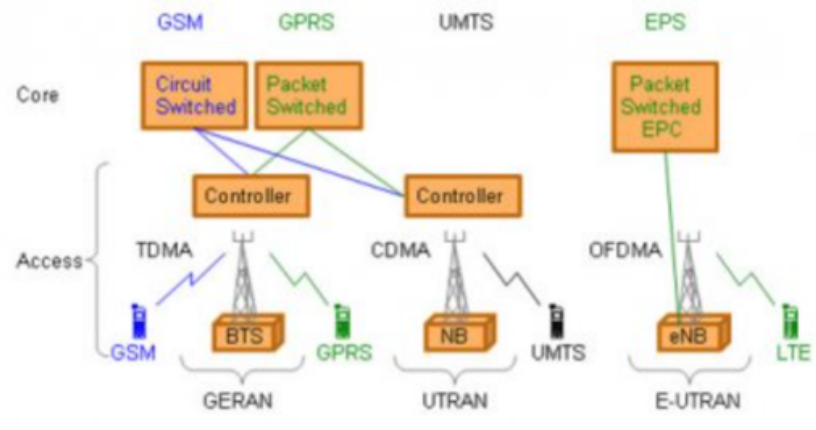
iii) Main Requirements:

- High spectral efficiency.
- High peak data rates.
- Short round trip time.
- Flexibility in Frequency and Bandwidth.

LTE innovation essentially empowers suppliers to push more information over a similar measure of radio spectrum. That implies they require less cellphone towers to serve a similar scope territory and can oblige more clients at the same time without compromising speed.

iv) Network Solutions from GSM to LTE:

GSM was created keeping in mind the end goal to convey constant services, in a circuit switched way as appeared in the figure beneath, however the issue is that information administrations are just conceivable over a circuit switched modem association with low data rates.



In this way, need is to move towards IP based packet switched administration and it was accomplished with the advancement of GSM to GPRS with the utilization of a similar air interface and access technique: Time Division Multiple Access (TDMA).

v) Evolution of WCDMA:

A new technology WCDMA (Wideband Code Division Multiple Access) was developed in order to get higher data rates in UMTS (Universal Mobile Terrestrial System). The access network in UMTS emulates a "circuit switched connection" for "real time services" and a "packet switched connection" for "data-com services."

In UMTS the IP address is assigned to the UE when a data-com administration is built up and discharged when the administration is discharged. Therefore, Incoming data-com services are still relying upon the circuit switched core for paging. In this technique, NodeB is responsible for

- FEC
- Modulation
- Spreading

Advantages:

- WCDMA does not require inter-base station synchronization which implies there is no GPS dependency feature in WCDMA.
- In WCDMA, deployment endeavours are lessened.

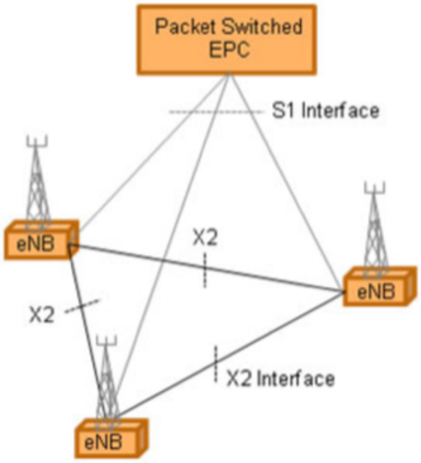
Drawbacks:

- In this technique, interference of one BS to other BS is high.
- Cell-search methodology is more challenging in WCDMA.
- Implementation of handover process is extremely complicated.

vi) Evolution of EPS (Evolved Packet System):

The biggest advantage of this system is that it is purely IP based strategy in which IP Protocol is in charge of conveying both real time services and data-com services. The IP address is allocated when the mobile is switched on and released when switched off. The new access solution, LTE, depends on Orthogonal Frequency Division Multiple Access (OFDMA) and in combination with higher order modulation (up to 64QAM), large bandwidths (up to 20 MHz) and spatial multiplexing in the downlink (up to 4x4) high data rates can be accomplished.

The LTE access network is basically a network of base stations, evolved NodeB (eNB). Additionally, there is no centralized intelligent controller and the eNBs' are normally inter-connected by means of the X2-interface and towards the core network by the S1-interface as shown in the figure beneath:



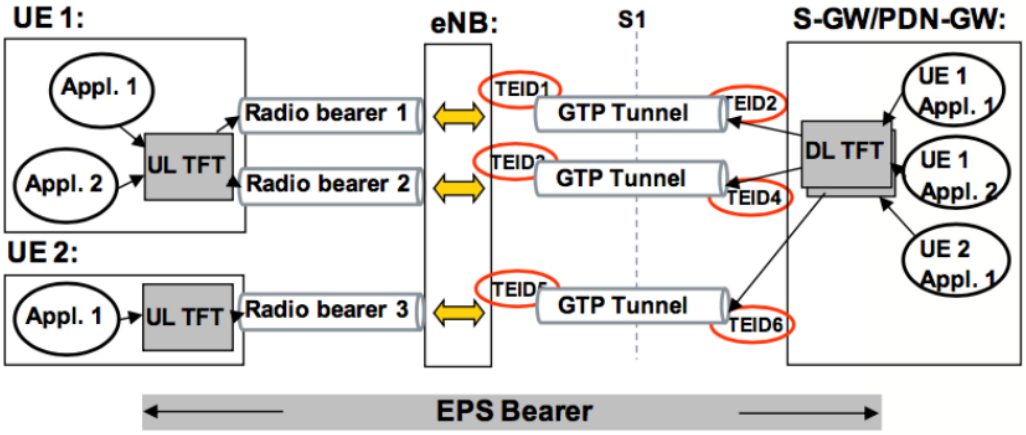
Base-stations have the control in the LTE architecture instead of offering it to any outsider controller or a centralized controller and there are many key-points behind this kind of distribution:

- One of the greatest inconvenience in WCDMA which is delay in handover, has been overcame in this system with the assistance of distribution feature and furthermore, it helps in accelerate the connection set-up and reduce the time required for a handover. For instance: set-up time is extremely crucial for an end user in real time data sessions, particularly in on-line gaming. Moreover, the time for handover is likewise critical in situations where end-clients tend to end calls if the handover takes too long.

- Another huge unfavourable position with the distributed arrangement is that the MAC protocol layer, which is in charge of scheduling, is represented only in the User Equipment (UE) and in the base station which prompts to quick communication and makes decision making procedure between eNB and UE faster as well.

Mobile networks consists of two layers: Physical and Logical layers. The Physical layer is made of network switches (L2), routers (L3) and physical links along with different topologies and technologies. The Logical layer consists of network elements (e.g. eNodeB, MME, S/P-GW, HSS, etc) that perform the connection of user end devices, mobility and transport of information from mobile devices across the entire mobile network.

The access network mainly consists of eNodeBs which is responsible for providing radio access to the User Equipment (UE). The physical layer (L2 and L3) is responsible for providing connectivity and transport functionality to the logical layers and logical layer helps in implementing the mobile specific control functions.



Moving to the backhaul network, it comprises of all the system switches which helps in accumulating the traffic from both the systems: access network and core network. Next is Mobility Management Entity (MME) which is the key control for any LTE access network. It is responsible for tracking and paging method including retransmissions, and also for idle mode of User Equipment (UE). It is responsible for validating client towards HSS. And if the user is in roaming area then MME is the one which is responsible for ending S6a interface. MME functions include:

- Non Access Stratum (NAS) signalling and security;
- PDN GW and Serving GW selection;
- Selecting handovers with change in MME;
- Roaming (S6a towards home HSS);
- Authentication;
- Authorization;
- Selection of appropriate eNodeB;
- Tracking Area list Management.

Next key element is Serving Gateway (SGW) which is responsible for terminating the interface towards E-UTARN. It is additionally in charge of handovers with neighbouring eNodeB's. It also take care about mobility interface to other networks for example 2G/3G. SGW duties include:

- Monitoring and maintaining context information related to User Equipment (UE) when it is in its idle state;
- It likewise helps in replicating of the user traffic in case of LI;
- Responsible for packet routing and forwarding;
- Packet Marking on transport level in both uplink and downlink;
- Responsible for inter-operator charging.

Further, Packet Data Network Gateway (PDN GW) is the following essential component in mobile systems which helps in ending the SGI Interface towards PDN. Number of PGW relies on the quantity of PDNs, for example if the client is accessing numerous PDNs, their might be more than one PGW for that specific User Equipment (UE). PGW is responsible to act as an "anchor" of mobility between 3GPP and non-3GPP technologies. PGW gives connectivity from the User Equipment (UE) to external PDN by being the “point of entry or exit” of traffic for the User Equipment (UE). Its obligations include:

- Manages policy enforcement;
- Per User based Packet Filtration (for instance: deep packet inspection);

- Allocates Ip Address per User Equipment (UE);
- Does Packet Screening;
- Does Accounting per User Equipment (UE) and bearer.

In addition, User Plane Entity (UPE) is likewise there which is in charge of the following functions:

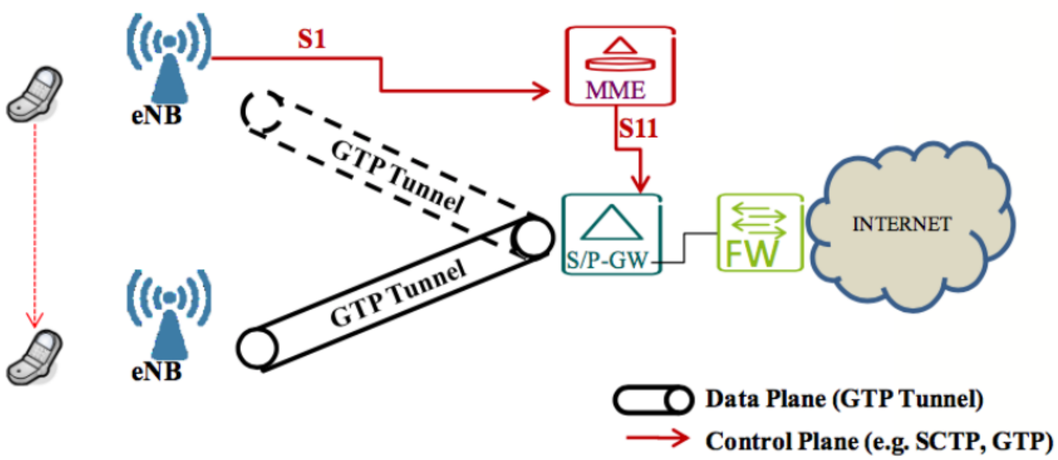
- It is responsible for IP Header Compression and encryption of user information streams;
- UPE is the one which does the termination of U-plane packets for paging reasons;
- It is in charge of the switching of U-plane for support of UE mobility.

Mobility is a critical functionality in mobile networks and it ought to be such innovation which conveys a reliable and low latency handover. Mobility in LTE networks is implemented through various techniques which relies on whether the new target eNodeB is under a different Tracking Area ID (TAI) associated to a different Mobility Management Element (MME) or the new Tracking Area ID (TAI) is managed by the same Mobility Management Element (MME). Moreover, the handover process depends on the S1-MME interface between logical elements (e.g. eNodeB, MME and SGW).

A fundamental issue in IP protocol from the mobility perspective is that the IP address recognizes the node and fixes its location to a specific IP subnet. The common solution in mobile networks consists of utilizing tunnelling UE IP packets in GTP tunnels that are established between the eNodeB and the S/P Gateway. "A GTP tunnel uniquely identifies traffic flows that receive a common QoS treatment between a UE and a P-GW." The Traffic Flow Templates (TFT) are used for mapping traffic to an EPS bearer.

vii) Mobility Control Process:

When an UE moves to a new eNodeB, the GTP tunnel has to be recreated between the new eNodeB and the S/P-Gateway while the inner data flow keeps using the original UE IP address. The GTP tunnel endpoint identifier (TEID) unambiguously recognizes the tunnel endpoint of a user data packet, isolates and distinguishes the users and furthermore isolates the bearers of a specific user.



The Handover process is used to hand over a UE from a source eNodeB to a target eNodeB when the MME is unchanged. They are intended to reduce interruption time compared to the circuit-switched handover process in 2G networks. This process is initiated and managed through the S1 interface as shown in Fig 2. MME is aware of the mobility process and communicates with the S/P-GW to recreate the GTP tunnel between the new eNodeB and the S/P-GW. In this process, first a Handover Request is issued to which Handover Request Acknowledge is received, after which MME Status is being shared.

viii) Limitations of Current LTE Networks:

The main driver behind the design of the current LTE networks was the requirement for supporting all-IP communication paradigm. In particular, the main focus was on supporting IP-based multimedia services via the introduction of the IP Multimedia Subsystem (IMS).

However, the emergence of more advanced mobile services, such as Mobile Social Networking (MSN), Mobile Cloud Computing (MCC), and Internet of Things (IoT), imposes very tight constraints on end-to-end and handover latency.

The main limitations of current LTE networks are as following:

- ***Inefficiency of Content Routing:***

The current approach says that all IP traffic passes via the MCN, however this is very inefficient because of the localized services. This introduces unnecessary delays and consumes insufficient network resources. This problem becomes bigger when connections are established via Packet Data Network (PDN) Gateways (P-GWs). Moreover, the use of “tunnelling" which is the main element of basic LTE Network hinders the routing optimization.

- ***Poor support for multiple radio access technologies:***

There is no mechanism to support efficient use of radio resources when multiple RATs, using both licensed and license-exempt spectrum, are present. Also, there is no coordination among different RATs which can be supported due to LTE requirement to forward data via the P-GW. This may be too slow in case, e.g., of MCC applications, which typically require low latency and demand considerable amount of radio resources to guarantee good QoE to MUs.

- ***Significant Signalling Overhead:***

For efficient support of IP multimedia services in LTE networks, PDN connections should be persistent, which in turn imposes significant signalling overhead for Internet of Things (IoT) and mobile internet services. So, services which need to send small packets with long period of inactivity within the same connection greatly suffer from huge signalling overhead (e.g. MSN and MCC applications). Also, services which rely on data from a large number of devices in different locations, suffer from the same signalling overhead issues, for e.g. sensor networks or monitoring services.

- ***Scalability limitations for IoT services:***

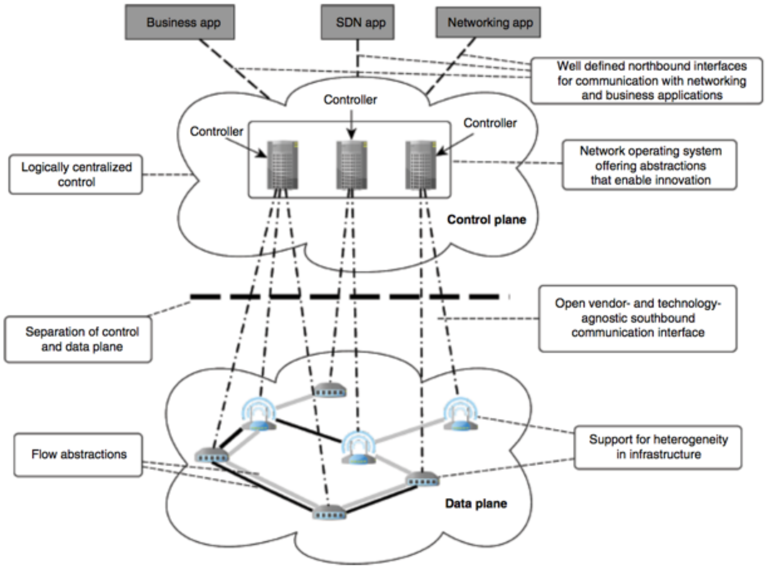
Internet of Things (IoT) are designed for a large number of resource-constrained devices with intermittent connectivity and current LTE network is not enough scalable for supporting IoT services. However, the current LTE attachment strategy, that demands each device maintaining at least one connection to the P-GW, is inefficient and not scalable in case of IoT and smart grid devices, such as sensors, actuators, and smart meters.

4) Software Defined Networking:

i) Introduction:

Software defined networking (SDN) is an idea that has recently reignited the interest of network researchers for programmable systems and moved the consideration of the systems administration group to this point by promising to make the way of planning networks and the way of managing systems more creative and rearranged compared with the well−established however firm current approach. The tight coupling between a systems's control plane (where the decisions of handling traffic are made) and data plane (where the actual forwarding of traffic takes place) gives rise to various challenges related to its management and evolution. Network administrators need to physically change high−level approaches into low−level arrangement commands, a procedure that for complex systems can be truly testing and error prone.

Introducing new functionality to the network, like intrusion detection systems (IDS) and load balancers, usually requires altering the system's infrastructure and directly affects its logic, while conveying new protocols can be a moderate procedure requesting years of standardization and testing to guarantee interoperability among the executions gave by different vendors.


Business app
SDN app
Networking app
Well defined northbound interfaces for communication with networking and business applications
Controller
Controller
Controller
Logically centralized control
Network operating system offering abstractions that enable innovation
Control plane
Separation of control and data plane
Open vendor- and technology-agnostic southbound communication interface
Flow abstractions
Support for heterogeneity in infrastructure
Data plane

The possibility of programmable systems has been proposed as a way to remedy this situation by promoting advancement in system administration and the organization of system administrations through programmability of the basic system elements utilizing some kind of an open network API. This prompts to adaptable systems ready to work as indicated by the client's needs in a direct analogy to how programming languages are being utilized to reconstruct computers so as to play out various tasks without the requirement of continuous modification of the underlying hardware.

SDN is a generally new worldview of a programmable system that progressions the way that systems are designed and managed by presenting a reflection that decouples the control from the information plane, as showed in Figure 3.1. In this approach, a software control program, alluded to as the CONTROLLER, has a diagram of the entire system and is in charge of the basic leadership, while the equipment (switches, switches, and so on) is just in charge of forwarding packets into their goals according to the controller's directions, commonly an arrangement of packet handling rules.

The division of the logically centralized control from the basic data plane has rapidly turned into the focus of the distinctive research interest for the systems administration group since it extraordinarily arrange network management and development in various ways. New conventions and applications can be tried and conveyed over the system without influencing irrelevant system activity; additional infrastructure can be presented without much botheration; and middle-boxes can be effectively incorporated into the software control, permitting new potential solutions to be proposed for issues that have long been in the spotlight, such as dealing with the profoundly complex core of cellular systems.

ii) SDN History and Evolution:

While the term programmable is utilized to sum up the idea of the simplified system administration and reconfiguration, it is imperative to understand that truly it epitomizes a wide number of thoughts proposed after some time, each having an alternate concentration (e.g., control or data plane programmability) and diverse methods for accomplishing their objectives.

History of Programmable Networks:

The idea of programmable systems dates its birthplace back in the mid−1990s, right when the Internet was beginning to encounter far reaching achievement. Until that moment, the use of computer systems was restricted to few administrations like email and record exchanges. However, it rapidly got to be distinctly clear that a major obstacle toward this course was the high complexity of managing the system infrastructure. Network devices were utilized as black boxes intended to support particular protocols necessary for the operation of the system, without ensuring seller interoperability. In this manner, altering the control logic of such devices was impossible, extremely confining system development. To cure this circumstance, different endeavours concentrated on discovering novel answers for making more open, extensible, and programmable systems were performed.

Two of the most critical early thoughts proposing methods for isolating the control software from the fundamental equipment (hardware) and giving open interfaces to administration and control were from the open signalling (OpenSig) working group and from the active networking initiative.

OpenSig:

The OpenSig working group showed up in 1995 and concentrated on applying the idea of programmability in ATM systems. The primary thought was the division of the control and data plane of systems, with the signalling between the planes performed through an open interface. Subsequently, it is conceivable to control and program ATM switches remotely, basically transforming the entire system into a distributed platform, greatly simplifying the process of deploying new services.

The thoughts pushed by the OpenSig community for OpenSig interfaces went about as inspiration for further research. Toward this course, the "Tempest framework," in view of the OpenSig philosophy, permitted numerous switch controllers to deal with various allotments of the switch at the same time and thus to run different control architectures over the same physical ATM network. This approach gave more flexibility to network administrators, as they were no longer compelled to characterize a single unified control design fulfilling the control prerequisites of all future system administrations.

Active Networking:

The active networking initiative showed up in the mid−1990s and was mainly supported by DARPA Like OpenSig, its fundamental objective was the formation of programmable systems that would advance network developments. The fundamental thought behind active networking is that assets of system nodes are exposed through a system API, permitting system administrators to effectively control the nodes as they wishing by executing self-assertive (arbitrary) code. In this manner, in spite of the static functionality offered by OpenSig systems, active networking permitted the fast deployment of customized services and the dynamic setup of systems at run−time. The general engineering of active systems (networks) characterizes a three−layer stack on dynamic nodes. At the base layer sits an operating system (NodeOS) which is in charge of multiplexing the nodes' communication, memory, and computational assets among the packet flows traversing the node.

At the following layer exist at least one execution situations giving a model to composing active networking applications, including ANTS and PLAN. At last, at the top layer are the active applications themselves, that is, the code created by system administrators.Two programming models fall inside the work of the active networking community:

The capsule model, in which the code to be executed is incorporated into regular data packets, and

The programmable switch/switch model, in which the code to be executed at network nodes is built up through out−of−band mechanism.

Out of the two, the capsule model came to be the most creative and most nearly connected with active networking. The reason is that it offered a drastically unique way to deal with system administration, giving a basic technique for putting in new data plane functionality across network paths. However, both models had a critical effect and left an imperative legacy, since huge numbers of the ideas met in SDN (separation of the control and data plane, network APIs, and so on) come straightforwardly from the endeavours of the active networking community.

iii) Evolution of Programmable Networks to SDN:

In spite of the fact that the key ideas communicated by these early methodologies imagined programmable systems that would permit development and would make open systems administration conditions, none of the proposed innovations was met with widespread achievement. One of the principle explanations behind this disappointment was the absence of convincing issues that these methodologies figured out how to settle. While the execution of different applications like content distribution and network management seemed to profit by system programmability, there was no genuine need that would turn the shift to the new worldview into a need, prompting to the commercialization of these early thoughts.

The new paradigm pushed as one of its advantages, the adaptability or flexibility it would provide to end clients to program the system, despite the fact that in reality the use case of end user programmers was truly uncommon. A last purpose behind the disappointment of early programmable systems was that they centred around proposing creative structures, programming models, and platforms, giving very little consideration to viable issues like the execution or performance and the security they offered.

However, these endeavours were truly huge, since they characterized surprisingly key ideas that changed the way that systems are seen and distinguished new research zones of high potential. Indeed, even their weaknesses were of high importance, since they uncovered numerous lacks that ought to be tended to if the new paradigm was to be effective one day. With everything taken into account, these early endeavours were the foundations that formed the route to the all the more encouraging and now broadly acknowledged worldview of SDN.

Shift to the SDN Paradigm:

The main years of the 2000s saw real changes in the field of networking. New technologies like ADSL developed, giving high-speed Internet access to customers. The mass reception of high-speed Internet and of all the new administrations that went with it had huge impacts for networks, which saw their size and scope increment alongside traffic volumes. Industrial stakeholders like ISPs and network operators began emphasizing on system unwavering quality, execution or performance, and QoS and required better methodologies in performing vital network configuration and management functions like routing, which at the time were primitive, best case scenario.

Moreover, new patterns in the capacity and management of data like the presence of cloud computing and the production of expansive data centres made evident the requirement for virtualized situations, joined by network virtualization as a way to support their mechanized provisioning, automation and orchestration. Every one of these issues moved the consideration of the networking community and the industry to this point afresh. This move was reinforced by the change of servers that turned out to be considerably superior to the control processors in switches, streamlining the assignment of moving the control capacities outside network devices. A consequence of this technological move was the rise of new enhanced system programmability endeavours, with the most conspicuous case being SDN.

In ForCES, two logical entities could be recognized:

The forwarding element (FE), which worked in the information plane and was in charge of per-packet preparing and dealing with, and

The control element (CE), which was in charge of the logic of system devices, that is, for the usage of management protocols, for control protocol handling, and so on. The thought behind ForCES was that by permitting the forwarding and control planes to advance independently and by giving a standard method for interconnection, it was conceivable to create diverse sorts of FEs' (universally useful or specific) that could be joined with third-party control, permitting more prominent adaptability or flexibility for development.

Another approach focusing on the clean division of the CE and FE of system devices was the 4D project. However, as opposed to past methodologies, the 4D project imagined an architecture in light of four planes: a decision plane in charge of making a system design or network configuration, a dissemination plane in charge of conveying data identified with the perspective of the system to the decision plane, a discovery plane permitting system devices to find their prompt neighbours, and a data plane in charge of sending activity or forwarding traffic.

A final project worth saying amid the pre−SDN period is SANE/Ethane. Ethane was a joint endeavour made in 2007 by researchers in the universities of Stanford and Berkeley to make another system design for the enterprise. Ethane embraced the primary thoughts communicated in 4D for a centralized control engineering, growing it to incorporate security.

A convincing element of Ethane was that its flow−based switches could be incrementally deployed close by traditional Ethernet switches and with no change to end hosts required, permitting the wide spread selection of the architecture. Ethane was actualized in both software and hardware and was deployed at the campus of Stanford University for a time of a couple of months. The Ethane project was extremely critical, as the encounters picked up by its design, execution, and deployment established the framework for what might before long get to be SDN.

iv) The Emergence of SDN:

In the second half of the 2000s, financing organizations and analysts began indicating enthusiasm for network experimentation at scale. This intrigue was for the most part roused by the need to send new protocols and administrations, focusing on better execution and QoS in huge enterprise networks and the Internet, and was further reinforced by the achievement of experimental infrastructures like PlanetLab and by the rise of different activities like the US National Science Foundation's Global Environment for Network Innovations (GENI).

One essential prerequisite of such infrastructure−based efforts was the requirement for system programmability, which would simplify network administration and network service implementation and would permit numerous tests to be run at the same time at a similar infrastructure, each utilizing an alternate arrangement of forwarding rules.

The OpenFlow protocol was proposed as a method for researchers to run experimental protocols in regular systems administration situations. Correspondingly to past methodologies like ForCES, OpenFlow took after the rule of decoupling the control and forwarding plane and standardized the data trades between the two utilizing a basic correspondence protocol.

The arrangement proposed by OpenFlow, which gave architectural support for programming the system, prompted to the production of the term SDN to encapsulate all the networks following similar architectural principles. The key thought behind SDNs contrasted with the ordinary systems administration paradigm is the production of horizontally integrated systems through the detachment of the control and the data plane while giving an undeniably advanced arrangement of deliberations.

SDN was less of another thought, as it was the promising result of the refined information and experience obtained through a large number of ideas. What SDN figured out how to do any other way contrasted with these thoughts is that it coordinated the most imperative network programmability ideas into a design that rose at the correct time and had convincing use cases for an extraordinary number of interested groups.

v) SDN Building Blocks:

As already mentioned, the SDN approach permits the management of system services through the reflection of lower−level functionality. Rather than managing low−level details of network devices with respect to the way that packets and flows are managed, network administrators now just need to utilize the abstractions accessible in the SDN architecture.

At the bottom layer, we can observe the data plane, where the network infrastructure (switches, routers, wireless access points, etc.) lies. In the context of SDN, these devices have been stripped of all control logic (e.g., routing algorithms like BGP) just executing an arrangement of sending operations for controlling system data packets and flows, giving a unique open interface to the correspondence with the upper layers. In the SDN wording, these devices are usually referred to as network switches.

Moving to the following layer, we can observe the control plane, where an entity referred to as the "Controller" lies. This entity encapsulates the networking logic and is in charge of giving an automatic interface to the system, which is utilized to execute new functionality and perform different management tasks. An essential perspective that recognizes SDN from past programmable system endeavours is that it has presented the idea of the network operating system abstraction. This reflection accepts a consistently concentrated control display, in which the applications see the system as a single system. As such, the network operating system goes about as a middle layer in charge of keeping up a predictable perspective of system state, which is then exploited by control logic to give different networking services for topological discovery, routing, management of mobility, and statistics.

At the highest point of the SDN stack lies the application layer, which incorporates all the applications that exploit the services gave by the controller so as to perform network−related assignments, similar to load adjusting or balancing, network virtualization, and so forth. A standout amongst the most critical elements of SDN is the openness it gives to third−party developers through the deliberations it characterizes for the simple improvement and deployment of new applications in different arranged conditions from data centres and WANs to wireless and cellular networks.

Additionally, the SDN design dispenses with the requirement for devoted middle-boxes like firewalls and IDS in the network topology, as it is presently feasible for their functionality to be implemented as programming applications that monitor and modify the system state through the network operating system services.

Finally, the communication of the controller to the data plane and the application layer can be achieved through well−defined interfaces (APIs). We can distinguish two main APIs in the SDN architecture:

(i) a southbound API for the communication between the controller and the network infrastructure and

(ii) A northbound API defining an interface between the network applications and the controller.

This is similar to the way communication is achieved among the hardware, the operating system, and the user space in most computer systems.

SDN Switches:

In the conventional networking paradigm, the network infrastructure is considered the most integral part of the network. All the functionality which would be required for the operation of a network would be encapsulated be network devices. For instance, a device such as a router need to provide the proper hardware as well as sophisticated software. Hardware is for a ternary content like an addressable memory (TCAM) for quickly forwarding packets and software for executing distributed routing protocols like BGP. This has been changed by the three-layered SDN architecture by decoupling the control from the forwarding operations, re-arranging the management of network devices. Hardware is responsible for storing the forwarding tables (for instance application-specific integrated circuits (ASICs) with a TCAM), therefore all forwarding tables hold the hardware, however are stripped of their logic. Packets should be forwarded by installing new forwarding rules through a unique interface and this is managed by the controller as well. For the proper functioning of the controller, each time a packet arrives to a switch, it’s forwarding table is consulted and the packet is forwarded accordingly.

Active queue management (AQM) and scheduling configuration are operations that are still considered some portion of the data plane even on account of SDN switches. However, there is no inherent issue keeping these functions from turning out to be a part of the control plane by presenting some kind of abstraction permitting the control of low−level conduct in switching devices.

Then again, while moving all control operations to an intelligently concentrated controller has the benefit of simpler network management, it can likewise raise versatility issues if physical execution of the controller is additionally centralized.

Issue of SDN switches is that the forwarding rules utilized as a part of the instance of SDN are more mind boggling than those of conventional networks, utilizing wildcards for forwarding packets, considering numerous fields of the packet like source and destination addresses, ports, application, and so forth. Subsequently, the switching hardware can't cope with the management of packets and flows. All together for the forwarding operation to be quick, ASICs utilizing TCAM are required.

Only a limited number of forwarding entries for flow-based forwarding schemes can be supported in each switch because of the expensiveness of such specialized hardware and moreover, it is power consuming as well. This in turn hinders the network scalability.

The wireless data plane should be upgraded keeping in mind the end goal to offer more helpful abstractions similarly to what occurred with the data plane of fixed networks. While the data plane abstractions offered by protocols like OpenFlow support the idea of decoupling the control from the data plane, they cannot be stretched out to the wireless and mobile field unless the underlying hardware (e.g., switches in backhaul cellular networks and wireless access points) begins giving similarly refined and helpful deliberations.

SDN Controllers:

As of now said, one of the core thoughts of the SDN theory is the presence of a network operating system present between the system infrastructure and the application layer. This network operating system is in charge of organizing and dealing with the assets of the entire system or network and for revealing a dynamic perspective of all components to the applications executed on top of it. This thought is equivalent to the one followed in a normal computer system, where the working system lies between the hardware equipment and the client space and is in charge of dealing with the hardware equipment assets and giving regular services to client/user programs. Similarly, network executives and engineers/developers are currently given a homogeneous situation which is simpler to program and arrange and also much like a common computer program designer would.

SDN model offers logically centralized control and the generalized network abstraction which in turn makes the model applicable to a wider range of applications and heterogeneous network technologies compared to the conventional networking paradigm. For example, if we consider an environment which is heterogenous and is composed of a fixed and a wireless network consisted of a large number of related network devices (routers, switches, wireless access points, middle-boxes, etc.).

Each network device would require individual low−level configuration by the network administrator if it belongs to the traditional networking paradigm and this is necessary for the proper functioning of that particular network device. Moreover, each device would have its own specific management and configuration prerequisites because each device targets a diverse networking technology which in turn implies that additional exertion would be required by the administrator to make the entire system work as planned.

Then again, with the coherently concentrated control of SDN, the administrator would not need to stress over low−level particulars. Rather, the system administration would be performed by characterizing an appropriate high−level strategy, leaving the system operating system in charge for communicating with and arranging the operation of system devices.

vi) Centralization of Control in SDN:

The SDN design indicates that the system infrastructure is logically controlled by a central element which is in-charge of administration and arrangement implementation.

A physically centralized control configuration streamlines the controller implementation. All switches are controlled by the same physical element, implying that the system is not subject to consistency−related issues, with every one of the applications seeing a similar system state (which originates from a similar controller). In spite of its advantages, this approach experiences a similar shortcoming that every centralized systems do, which is, the controller goes about as a solitary purpose of disappointment for the entire system. An approach to beat this is by associating various controllers to a switch, permitting a backup controller to assume control in case of a disappointment.

For this situation, all controllers need a predictable perspective of the network; generally, applications may neglect to work appropriately. Additionally, the centralized approach can raise versatility/scalability worries, since all system devices should be under the supervision of a same element.

One approach that further sums up utilizing different controllers over the network is to keep up a coherently centralized yet physically decentralized control plane. For this situation, every controller is in charge of overseeing some portion of the system, however all controllers impart and keep up a common perspective of the system.

Consequently, applications see the controller as a solitary substance, while as a general rule control operations are performed by a disseminated network. The upside of this approach, aside from not having a solitary purpose of disappointment, is the expansion in execution/performance and adaptability/scalability, since just a piece of the system should be overseen by every individual controller segment. Some well-known controllers that have a place with this class are Onyx and HyperFlow.

One potential downside of decentralized control is once more identified with the consistency of the system state among controller segments. Since the condition of the system is disseminated, it is conceivable that applications served by various controllers may have an alternate perspective of the system, which may make them work disgracefully.

A hybrid arrangement that tries to envelop both adaptability and consistency is to utilize two layers of controllers like the Kandoo controller does. The base layer is made out of a gathering of controllers that don't know about the entire system state. These controllers just run control operations that require knowing the condition of a solitary switch. Then again, the top layer is an intelligently centralized controller in charge of performing network-wide operations that require learning of the entire system state. The possibility of coherent decentralization comes specifically from the early period of programmable systems. Their has been recommendations for SDN intermediary controllers like FlowVisor that permit numerous controllers to have a similar forwarding plane. The inspiration for this thought was to empower the synchronous arrangement of experimental and enterprise networks over a similar framework without influencing each other.

A standout amongst the most successive concerns raised by SDN cynics is the capacity of SDN systems to scale and be responsive in instances of high system/ network load. This worry comes mostly from the way that in the new paradigm, control moves out of system devices and goes in a solitary element which is in charge of dealing with the entire system activity. Execution investigations of SDN controller usage have revealed that even physically centralized controllers can perform truly well, having low reaction times.

Another critical execution concern brought up for the situation of a physically decentralized control plane is the way that controllers are put inside the system, as the system execution can be incredibly influenced by the number and the physical area of controllers, and additionally by the calculations utilized for their coordination. So as to address this, different arrangements have been proposed, from review the situation of controllers as an improvement issue to building up associations of this issue to the fields of neighbourhood calculations and circulated figuring for creating effective controller coordination protocols.

A last concern brought up for the situation of physically distributed SDN controllers is identified with the consistency of the system state kept up at every controller when performing strategy updates because of simultaneousness issues that may happen by the error−prone, appropriated nature of the logical controller. The arrangements of such an issue can be like those of value-based databases, with the controller being extended with a value-based interface characterizing semantics for either totally committing a strategy update or aborting.

vii) Management Of Traffic:

Another essential plan issue of SDN controllers is identified with the way that traffic is handled. The choices about traffic management can directly affect the execution of the system, particularly in instances of extensive systems made out of many switches and with high traffic loads. Further, the issues related to traffic management are separated into two categories: control granularity and strategy authorization.

Control Granularity:

The control granularity connected over network traffic denotes to how fine or coarse grained the controller investigation operations ought to be in connection to the packets crossing the system. In ordinary systems, every packet arriving at a switch is inspected independently, and a directing choice is made as to where the packet ought to be sent relying upon the data it conveys (e.g., destination address).

While this approach usually works for ordinary systems, the same can't be said for SDN. For this situation, the per−packet approach gets to be distinctly impractical to execute over any sizeable network, since all packets would need to go through the controller that would need to build a route for every one of them separately. Because of the execution issues raised by the per−packet approach, most SDN controllers take after a flow−based approach, where every packet is allocated some stream (flow) as indicated by a particular property (e.g., the packet's source and destination address and the application it is connected with).

The controller sets up another flow by looking at the very first packet arriving for that particular flow and designing the switches in like manner. So as to further off−load the controller, an additional coarse−grained approach is implemented which depends upon collection of flow match as opposed to utilizing singular flows.

The fundamental trade−off while analyzing the level of granularity is the load in the controller versus the QoS offered to network applications. The more fine grained the control, the higher the QoS. In the per−packet approach, the controller can simply settle on the best choices for routing every individual packet, thus prompting to enhanced QoS.

On the flip side, upholding control over a collection of flows implies that the controller choices for sending packets don't completely adjust to the condition of the system/network. For this situation, packets may be sent through an imperfect course, prompting to tainted QoS.

Policy Enforcement:

The second issue in the management of traffic is identified with the way that system/network approaches are connected by the controller over system/network devices.

One approach, trailed by systems like Ethane, is to have a receptive control model, where the switching device counsels the controller each time a choice for another flow should be made. For this situation, the arrangement for each flow is built up to the switches just when a genuine request emerges, making system management more adaptable.

A potential drawback of this approach is the corruption of execution, because of the time required for the principal packet of the flow to go to the controller for examination. This execution drop could be noteworthy, particularly in instances of controllers that are physically situated far from the switch.

An option strategy authorization approach is utilize a proactive control model. For this situation, the controller populates the flow tables early for any activity that could experience the switches and after that pushes the rules to all the switches of the system/network.

Utilizing this approach, a switch no longer needs to demand directions by the controller to set up another flow and rather can play out a straightforward query at the table stored in the TCAM of the device. The pros of proactive control is that it wipes out the idleness actuated by consulting the controller for every flow.

viii) SDN Programming Interfaces:

As already mentioned, the communication of the controller with alternate layers is accomplished through a southbound API for the controller–switch connections and through a northbound API for the controller–application communications.

Southbound Communication:

The southbound correspondence is critical for the deceiving of the behaviour of SDN switches by the controller. It is the way SDN endeavours to "program" the system. The most protruded case of a standardized southbound API is OpenFlow.

Most undertakings identified with SDN expect that the communication of the controller with the switches is OpenFlow based, and in this manner, it is imperative to make a point by point introduction of the OpenFlow approach.

However, it ought to be clarified that OpenFlow is only one (somewhat mainstream) out of numerous conceivable executions of controller–switch collaborations. Different choices, for instance, DevoFlow, additionally exist, endeavouring to understand execution issues that OpenFlow faces.

ix) Overview of OpenFlow:

Following the SDN rule of decoupling the control and information planes, OpenFlow gives an standardized method for managing traffic in switches and of trading data between the switches and the controller, as Figure 3.2 represents. The OpenFlow switch is made out of two legitimate segments.

The principal part contains one or more flow tables which is responsible of keeping up the data required by the switch with a specific end goal to forward packets. The second segment is an OpenFlow customer, which is basically a basic API permitting the communication of the switch with the controller.

The flow tables comprise of flow entries, each of which characterizes an arrangement of principles deciding how the packets belonging to that specific flow will be managed by the switch (i.e., how they will be processed and forwarded).

Each entry in the flow table has three fields:

(i) A packet header characterizing the flow;

(ii) An activity deciding how the packet ought to be processed and

(iii) Statistics: which monitor data like the quantity of packets and bytes of each flow and the time since a packet of the flow was last sent.

Once a packet arrives at the OpenFlow switch, its header is inspected, and the packet is coordinated to the flow that has the most comparable packet header field. In the event that a coordinating flow is found, the activity characterized in the activity field is performed.

These activities incorporate the sending of the packet to a specific port with a specific end goal to be directed through the system, the sending of the packet keeping in mind the end goal to be inspected by the controller, or the denial of the packet. In the event that the packet can't be coordinated to any flow, it is dealt with as per the activity characterized in a table−miss flow entry.

The trading of data between the switch and the controller occurs by sending messages through a secure channel uniformly characterized by the OpenFlow protocol. Along these lines, the controller can control the flows found in the flow table of the switch (i.e., add, update, or delete a flow entry) either proactively or responsively as examined in the fundamental controller standards.

Since the controller can speak with the switch utilizing the OpenFlow protocol, there is no longer a requirement for system administrators to connect specifically with the switch.

An especially convincing aspect of OpenFlow is that the packet header field can be a wild card, implying that the coordinating to the header of the packet does not need to be precise. The thought behind this approach is that different system/ network devices like routers, switches, and middle-boxes have a comparable forwarding behaviour, varying just as far as which header fields they use for coordinating and the activities they perform.

OpenFlow permits the utilization of any subset of these header fields for applying rules on traffic flows, implying that it thoughtfully brings together a wide range of sorts of system/network devices. For example, a router could be copied by a flow entry utilizing a packet header playing out a match just on the IP address, while a firewall would be imitated through a packet header field containing extra data like the source and destination IP addresses and port numbers and in addition the transport protocol being utilized.

Northbound API:

One of the fundamental thoughts pushed in the SDN paradigm is the presence of a system/network operating system, lying between the system infrastructure and the high−level services and applications, correspondingly to how a computer operating system lies between the equipment (hardware) and the client space.

Accepting such a centralized coordination substance and in view of the fundamental operating system standards, a plainly characterized interface ought to likewise exist in the SDN design for the connection of the controller with applications. This interface ought to permit the applications to get to the basic equipment (hardware), deal with the system assets, and permit their interaction with different applications without having any learning of low−level system data.

Rather than the southbound correspondence, where the connections between the switches and the controller are very much characterized through an standardized open interface (i.e., OpenFlow), there is presently no acknowledged standard for the collaboration of the controller with applications. Accordingly, every controller model needs to give its own particular strategies to performing controller–application communication.

For instance, consider a power administration and a firewall application. The power administration application needs to reroute activity utilizing as few connections as conceivable so as to deactivate idle switches, while the firewall may require these additional switches to route traffic as they best fit the firewall rules. Leaving the programmer to manage these contentions could turn into an exceptionally mind To tackle this issue, numerous thoughts have been proposed, upholding the utilization of high−level net work programming language in charge of making an interpretation of strategies to low−level stream requirements, which therefore will be utilized by the controller to deal with the SDN switches.

x) Need for SDN:

- The “consumerization of IT”: Now-a-days, trend is Bring Your Own Device (BYOD) which requires a network which is both flexible and secure, and traditional networks does not fulfills this trend.
- “Big data” means more bandwidth: Handling today’s mega datasets requires massive parallel processing that is fuelling a constant demand for additional capacity and any-to-any connectivity.
- Rise of cloud services: Users expectations are increasing with the advancement in technology, Users expect on-demand access to applications, infrastructure, and other IT resources with the rise of cloud services. This leads to need of SDN.
- Changing traffic patterns: Applications that commonly access geographically distributed databases and servers through public and private clouds require extremely flexible traffic management and access to bandwidth on demand.

In trying to meet the networking requirements posed by evolving computing trends, network designers find themselves constrained by the limitations of current networks and they are as following:

- Complexity that leads to stasis: Adding or moving devices and implementing network-wide policies are complex, time-consuming, and primarily manual endeavours that risk service disruption, discouraging network changes.
- Inability to scale: The time-honoured approach of link oversubscription to provision scalability is not effective with the dynamic traffic patterns in virtualized networks—a problem that is even more pronounced in service provider networks with large-scale parallel processing algorithms and associated datasets across an entire computing pool.
- Vendor dependence: Lengthy vendor equipment product cycles and a lack of standard, open interfaces limit the ability of network operators to tailor the network to their individual environments.

Typical Use Cases:

- Security
- Switching
- Routing
- Traffic Engineering

- QoS
- Network Access Control
- Load Balancing
- Monitoring
- Network Taps
- Cut-Trough Applications
- Network Virtualization
- Multi-Tenancy
- Campus Slicing

xi) SDN Application Domains:

In an extensive variety of networking areas, in order to illustrate the applicability of SDN we briefly present two trademark cases in which SDN could turn out to be useful: data centres and cellular networks.

It is obviously that the list of SDN applications is not constrained to these areas but also reached out numerous others, for instance: enterprise networks/systems, WLANs, and heterogeneous networks, optical networks and the Internet of Things

Data Centre Networks:

A standout amongst the most critical prerequisites for data centre networks is to discover approaches to scale keeping in mind the end goal to support huge number of servers and millions of virtual machines. However, accomplishing such scalability can be a herculean task from a network point of view. First of all, there is a requirement for more complex and costly forwarding devices because as we know that if the number of servers increases thus in turn, size of forwarding tables increases. In addition, since data centres are expected to constantly accomplish large amounts of execution/performance, due to which traffic management and policy enforcement can turn out to be imperative and basic issues.

If we consider the traditional data centres, in order to accomplish previously mentioned prerequisites, the design and configuration of the underlying network should be very precise. However, mostly there are two ways for performing the same operation: manually performed by characterizing the favoured routes for traffic and by putting middle boxes at strategic choke points on the physical network.

As we know that SDN offers decoupling the control from the data plane which acts as a favourable circumstance to network management because with this property forwarding devices turn out to be significantly less complex and therefore less expensive. Also, all control logic is assigned to one logically centralized entity

This permits the dynamic administration of flows, the load balancing of traffic, and the allocation of resources in a way that best adjusts the operation of the data centre to the requirements of running applications, which in turn prompts to expanded performance. At last, putting middle boxes in the network is not required anymore, since policy enforcement can now be accomplished through the controller entity.

Cellular Networks:

The market of cellular mobile networks is maybe a standout amongst the most gainful in telecommunications. The fast increment in the quantity of cellular devices (e.g., smartphones and tablets) during the past decade has pushed the current cell systems as far as possible.

Recently, there has been huge enthusiasm for incorporating the SDN standards in current cellular architectures like the 3G Universal Mobile Telecommunications System (UMTS) and the 4G Long−Term Evolution (LTE).

The core of the network has a centralized data flow along with all traffic passing through specific equipment which acts as a primary impediments of current cellular network architectures due to the reason that it packs various system capacities from routing to access control and billing (e.g., packet gateway in LTE) and also it prompts the expansion of the infrastructural cost because of the concerns like complexity of the devices and versatility of the network/system.

Applying the SDN standards to cellular networks guarantees to solve some of these insufficiencies. Most importantly, decoupling the control from the data plane and presenting a centralized controller that has a total perspective of the whole network permit network equipment to become simpler and accordingly lessen the general infrastructural cost. In addition, operations like routing, real−time monitoring, mobility management, access control, and policy enforcement can be assigned to various coordinating controllers, making the system more adaptable and easier to manage. Besides, the operations of load and interference management would be simplified with the utilization of a centralized controller which acts as a conceptual base station as in this scenario there will be no requirement for the direct communication coordination of the base stations. As an alternative, in this case the controller is responsible for all the decisions for the entire system/network and simply instructs the data plane (i.e., the base stations) on how to operate. One last favourable aspect is that the utilization of SDN facilitates the acquaintance of virtual operators to the telecommunication market, prompting to expanded competitiveness. Moreover, all providers become responsible for managing the flows of their own subscribers through their own controllers because to the virtualization of the switching equipment and also there is no requirement to pay large sums for obtaining their own infrastructure.

5) Software Defined Networking for Wireless:

i) Introduction:

The current years have seen the appearance of a digital, mobile, and connected society in which digital devices are used for innumerable applications. Individuals utilize wireless devices such as laptops, smartphones, or tablets for communicating: audio or video, online shopping; data storing purposes; entertainment purpose: Netflix, Youtube; internet surfing etc . These situations, which are a reality today, importantly affect the systems/networks.

To begin with, because of the cloud services, we can more and more regularly synchronized our information among the terminals. However, because of this regular synchronization, the bandwidth of the wireless channel would be consumed more and it also reminds peer−to−peer applications as it makes important data flows. In this manner, handovers are more continuous, and concern multimedia data that can endure losses better than delays and application known as file transfer which is completely reliable upon TCP connections and it can also tolerate delays as long as the connections remain.

Operators consequently require an approach to deal with their wireless network all in all productively and in a transformative way. Besides, several optimization possibilities would likewise require coordination and cooperation between operators, users, and infrastructures, which is hard to accomplish in practice.

A network ought to have the capacity to increment or diminishing its ability on request, providing the desired quality of service (QoS) to users when conceivable and staying away from unsettling influences to close networks when possible. This requires a bandwidth management entity fit for analyzing not just the request and the user service−level agreements (SLAs) additionally the condition of the distinctive access points (AP’s) or the wireless channel status in order to take proper choices.

This management element might assemble its vision of the system by gathering data with the help of various network devices into order to perform a global optimization so that it can recommend connection options to each user and for each application. This paradigm is referred to as network virtualization or software defined networking (SDN).

ii) Wireless SDN:

The software defined network concept has initially been imagined with data centres and fixed networks in mind. However, the wireless world could greatly benefit from such a framework. In fact, as we know that most of the wireless technologies have very limited resources, also we know that Cellular networks has an ever-increasing mobile traffic for which it needs a methodology responsible for taking care of the increasing traffic.

Mobile operators attempt to envision answers for off−load data to WLANs in urban territories, which can hypothetically suit more flexible flows on account of random access. In any case, remote LANs additionally get to be distinctly congested, as the unlicensed frequency bands are used by various innovations whose traffic faces an increment as well. Additionally, the arrangement of WLAN AP’s in urban territories is denser and denser and clumsy, making interference mitigation troublesome.

Furthermore, the mobility of users additionally affects the accessible assets. Users ought to have the capacity to keep their communications and connections open while voyaging, which requires the administrator to foresee client portability and to save a piece of the assets for clients going from cell to cell. A few handover systems are conceivable, yet this procedure regularly requires committing a piece of the assets to portability management.

These issues are altogether identified with the rare limit offered by wireless channels contrasted with the request of the applications, which is regularly scaled on the wired association performance. A strategy was carried out in the past in order to solve this problem, which can be defined as a process which works on modulation and coding due to which it helps in the improvement of the spectral efficiency. In any case, this procedure has a farthest point, and it is presently normal to peruse that we are achieving Shannon Capacity Limit. The most hopeful forecasts place today's innovation inside 20% of this point of confinement, and a definitive endeavours to enhance signal over impedance in addition to noise ratio which are relied upon to be exceptionally troublesome.

Operators as of now react to the absence of bandwidth on specific systems by off−loading traffic to different systems. For example, in the access systems, a few mobile administrators have an answer for off−load traffic from the 3G system to an accomplice Wi−Fi network when a client is in range, which nevertheless causes disengagement and execution issues for mobile clients, as handover is not legitimately dealt with particularly in respect to security. Confirmation and encryption keys frequently should be restored, which causes postpones that are regularly incompatible with high mobility. Additionally, off−loading should be precisely managed, as it could easily cause over-burden. The most significant cases of such circumstances originate from different domains. For instance: electrical systems utilize off−loading widely, and the absence of coordination between administrators has been one of the reasons for the escalade in electrical disappointment of November 4, 2006, that prompted to a power outage in occidental Europe.

A third answer for the shortage of the wireless spectrum comprises in improving the spatial effectiveness of wireless communications. Conveying the clients nearer to there serving base station would permit to decrease both mobile terminal and base station transmission powers, in this way creating less impedance on close cells. Femtocells in cell systems work along these lines, despite the fact that their objective is more to enhance scope than to diminish impedances. All things considered, a generalization of this rule brings its heap of issues. On the off chance that every client sends a smaller than normal plug−and−play base station, there will be no plausibility for the administrator to control its position, and worldwide arranging systems are accordingly unthinkable. There is a slight possibility that wireless spectrum−related issues will show up in a few zones as opposed to being settled, particularly when distinctive administrators take a shot at a similar band of frequencies (e.g., Wi−Fi, UMTS).

In September 2013, the ONF published a brief paper entitled “OpenFlow−Enabled Mobile and Wireless Networks” . This report depicts a few results of wireless SDN, concentrating on radio access network execution improvement. They list a few key difficulties for executing wireless SDN before concentrating on two noteworthy issues: wireless channel asset administration through intercell obstruction diminishment and versatile movement administration through roaming and off−loading. This report likewise reminds that the wireless channel has some specificity that makes the SDN idea intense to actualize. That is the reason this study has not been broadly contemplated yet. However, the arrangement of issues it brings is likewise what makes it interesting to study.

Concerning status report, the issue is considerably more troublesome. To start with, the wireless channel state changes much of the time, particularly in an indoor situation. Blurring and shadowing, for instance, can make a connection vanish all of a sudden, frequent updates on link state that should be considered in the routing protocols. A controller consequently needs to assess more than the basic channel or device load; it additionally needs to secure data on the connection strength, for instance. This variety is partial because of the varieties in the physical condition (door closing, individuals passing, and so forth.), additionally to the presence of close APs' that have their own particular traffic pattern and that don't really have a place with a similar administrator. Finding these potential interferers is challenging.

iii) Implementations:

OpenRoads and OpenRadio

OpenRoads is the adjustment of OpenFlow to wireless systems. It depends on OpenFlow to separate control path and data path through the FlowVisor open API. Concerning the wired case with OpenFlow, in OpenRoads, the network OS constitutes the interface between the infrastructure and the applications that observe and control the system.

The OpenRoads project fabricated a demonstration platform made out of Wi-Fi and WiMAX AP's that has been utilized for scholastic courses. Essentially, this comprise in multiplexing at the recipient the same traffic originating from various base stations in the meantime and from various systems, for example, Wi-Fi and WiMAX. Parcel duplication encourages the handover amongst technologies and expands the QoS. Some other mobility administrator implementations succeed to enhance the handover procedure with a detailed decreased packet loss rate and represent close to twelve lines of code.

In the present version, cutting in OpenRoads is actualized by making virtual interfaces on the same AP and allotting various service set identifier (SSID) to every interface. Each SSID can be considered as an isolated cut and might be managed by an alternate controller. Despite the fact that the controller can apply distinctive arrangements to various clients, it is restricted by the physical imperatives and the equipment capacities of today's APs. All cuts ought to utilize a similar channel and power settings, for instance, which limits disengagement (isolation). This shows it is on a very basic level distinctive to run SDN on customary wired switches than on remote APs.

iv) Wireless SDN Opportunities :

The capability of SDN and the OpenFlow approaches has as of now been shown into various illustrations concerning the system infrastructure:

Multi-network Planning:

WLAN benchmarks, and in addition most unlicensed advances, have a predetermined number of autonomous channels. Wi−Fi, for instance, has just three non overlapping channels in the 2.4 GHz band and eight in the 5.2 GHz band. There will always be a limit on the quantity of terminals that can work with no interference in a given geographic territory.

SDN could give assistance with respect to such wireless system arranging. It is undoubtedly conceivable to make zone−specific and operator−independent controllers, which can even be distributed processes, that would be fit for accumulating statistics originating from the AP's and could choose channel portions and transmission force of the access points all together which hence limit interdependencies and impedances, which is similar to the Control and Provisioning of Wireless Access Points (CAPWAP) protocol. CAPWAP, which has been executed in some Wi−Fi AP models, could incorporate easily in worldwide multi-innovation SDN, with CAPWAP filling in as a checking and control interface for Wi−Fi and compatible APs.

Handovers and Off−Loading:

Today's devices are exceedingly versatile, sufficiently ergonomic to permit clients to connect with online administrations while moving, and they are likewise equipped with different wireless interfaces. A state−of−the−art smart phone can possibly be associated with the Internet through a cellular access (LTE, UMTS, and so forth.), Wi−Fi, and even Bluetooth LE if a perfect gateway is around. A device could in this manner select the best network AP(s) to get to remote administrations in light of a mix of criteria that incorporate connection quality (throughput, SINR, and so forth.), stability, billing, QoS abilities, and additionally the administrators' inclinations with respect to, for instance, off−loading.

Established "flat" handovers over AP's are regular in cell networks and are very much taken care of by administrators through mobility expectation and asset soft reservation. As Wi−Fi limitation is moderately exact today in urban territories, a cell administrator may expect a slight change in mobility forecast.

Vertical handover crosswise over advancements happens when a cell administrator favours off−loading to Wi−Fi systems. Off−loading today experiences the short scope of Wi−Fi AP's, which makes the connection exceptionally unpredictable, and from the long verification delay.

Cross−operator handovers may happen when a mobile client incidentally or forever leaves the scope zone of an administrator he used to begin an association, for instance, leaving the home Wi−Fi network while going outside, leaving the cell administrator when taking a lift, and roaming to another phone administrator.

For this situation, SDN could convey to the client data on the accessible systems, their stability, their coverage, etcetera to encourage a neighbourhood choice process.

Dead Zone Coverage:

One interesting case in cell scope is the dead zone. This is basic in some provincial zones or extensive woods or likewise happens when there is a disaster.

At times, a client situated in a dead zone is not secured by an AP but rather is still inside the scope of another client. Both clients can convey by means of specially appointed systems, however the "uncovered" client won't have the capacity to be come to by the AP since the AP does not realize that the client is reachable through ad hoc. On account of the idea of wireless SDN, we can utilize the topology discovery abilities at the controller to help backhauling clients in some dead zone through ad hoc system by means of neighbouring clients. Obviously, this infers a few reports by the end client to it's covering AP to help it manufacture an exact perspective of the topology of the clients covered.

Security:

The monitoring capacity of OpenFlow and OpenRoads plays a vital role as it provides a clear vision of status of network to an entity which is responsible for detecting intrusions or abnormal behaviour. A process which decides if the current traffic matches the expected values for this date and time and this calculation is based on the analyzation of the network load and the packet distributions per protocol which is then compared to statistics. There are circumstances which may indicate an interruption or the presence of inside computers taking an interest in a botnet, for instance, control traffic can also be inspected.

CDN and Caching:

Wireless SDN could utilize content−centric paradigm (CDN) in which the base station could assemble and store data in order to convey the same in a timely way so that application which are too sensitive can be delayed. The controller is responsible for distinguishing the users based on their needs so that all the users can be associated with respect to its closest base station which can hold the required data of the users. For instance, there is a technical department in an office, it is very obvious that a large number of individuals require the same data from the technical server and that with very short delays as well. So, instead of forwarding the query of each individual to the technical server, it is a good approach to gather and store data in base stations and deliver it to the users as soon as requested. In this way, the load on the main technical server decreases as now the overall load is divided between the base stations and the main server. This is a good technique which is now-a-days recommended for LTE networks due to an increase in overall traffic.

6) LTE architecture integration with SDN:

The LTE mobile network architecture defines a logical network running on the top of a legacy L3 transport network. It is usually divided into two parts: the backhaul and the core networks. The backhaul network is the part of the network connecting the access network where the eNodeB's (LTE base stations) are located to the core network where the other logical components of the LTE network are located.

Caches are servers added in the networks, that intercept the requests of the clients in order to serve them from local storage as much as possible. When a request arrives and if that particular content is not in the cache, cache server retrieves it from the main web server and stores a copy of the same for further requests. This improves quality of experience (QoE) and also reduces the load on the web server.

i) In-Network caching application for mobile networks:

Implementation of technique named Caching in LTE networks helps in reducing congestion and load on the servers by caching the web content. Th best known caching technique is HTTP based caching which is applied in most of the networks with the help of caches and proxies. However, the implementation of caches should comply some recommendations and they are as beneath:

Desired Properties: These are the fundamental elements in order to implement an efficient caching system:

Fast Access: In order to ensure low latency.

Scalability: Cache must be scalable as it is responsible to support the load.

Adaptiveness: Should not be rigid and not be such that the user cannot amends it according to the change in demands.

Stability: Disruption in services is not what customers are looking for. Also, operators don't want such service which is disrupted. So scalability must be in the cache network.

Robustness: A system which is not a single point of failure otherwise customers cannot receive expected services.

Load balancing: For scalability and stability services.

Simplicity: For system administration.

Transparency: This one is for users so that they need not to care about it.

Advantages and drawbacks of web caching:

The first and most significant favourable position is that caching lessens the network traffic and the bandwidth consumption. The requests are served from a local copy, thus they don't go over the entire system to get the content from Internet. Thus, caching reduces the network load and the peering costs since less volume is going in and out of the network. Another favourable position is that serving the substance from local copies enhances the clients' understanding by decreasing the idleness. Also, the cache provides lower round-trip time (RTT) for a local cache as it allows retrieving the content from a close location as compared to the round-trip time (RTT) for the original web server. Moreover, in caching, the connection does not go across multiple links through the uncertain Internet which increase reliability.

Caching lessens the load on the servers by load-balancing the requests on the different caches which is an another positive point behind caching technique. In case, the web server goes down, cache can still server the requests with the contents stored in local copy, therefore it can be stated that Cache can also upgrade the robustness of the service.

On the flip side, caching can have drawbacks too, for instance if content is not stored yet in the cache then their will be a cache-miss which in turn increases the latency of the network. Caches can also be a bottleneck or a single point of failure if the design is not done carefully. Finally, the caches modify the content retrieval information.

ii) Cache-ability analysis in a LTE network:

The performance of the caching systems depends upon the type of content on which caching has to performed. So the cache-able content should be somewhat with which the bandwidth consumption would be reduced. To be cache-able, an object must fulfill following requirements:

Set-cookie must be null.

Content length must not be equal to 0.

Last-modified must not be set to 0.

Vary must be null.

Cache control must not be "private" or "no-cache".

Cache has been impacted by the type of information, for instance information may be images, audio, video, text or some application, impacts cache-ability. The images have high revisit rate which in turn helps in reducing the latency and also bandwidth consumption, so it is easy to cache images. A piece of content with a high revalidation rate should not be cached since the cache will probably send a request to the web server to revalidate the content. For instance, an application contents which have higher revalidation rate. Caching for the contents with high revalidation rate can only spare bandwidth. The text contents have a high rate of revisit too, but a low data volume. Thus, it reduces the latency however, there is no effect on the bandwidth consumption.

Audio contents are the worst contents for caching because of their very high revalidation rate. However, video contents are good target since they have a huge volume of data which spare bandwidth.

In short, if the administrator wants to spare bandwidth the video content would be the first choice but if the administrator is more inclined towards latency reduction then images, text and application contents would be a good choice.

Cache Strategy:

The cache line can be in three states for each piece of information as beneath:

- The full cache line is stored and the document is served when requested.
- Only the header is stored and cache will retrieve the requested data and then stored it as indicated in the header. This procedure is performed only in the case

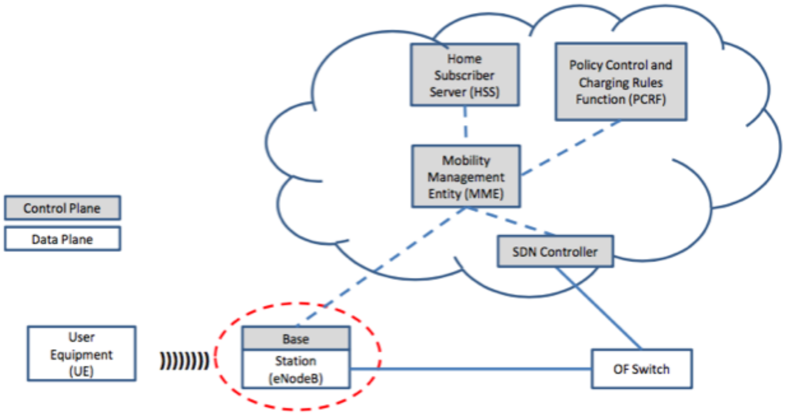
when it takes a shorter time to fetch requested data from a closet cache as compared to fetch it directly from the web server.

- The third scenario is when nothing is stored and a broadcast request has been sent from cache. Broadcast the request means each and every cache will receive it and it any cache has the requested document it will respond to it with header field. Then same as above is repeated and line is created.

Therefore, it can be concluded that Caching is a strategy of storing local copy of contents from the web and then serving the requests with the help of the stored copy. It is an efficient way to spare bandwidth and network resources as well improving the user's experience by reducing the latency.

iii) SDN architecture with LTE:

There are many features which can be utilized as extensions for the current SDN implementations. For SDN architecture with LTE aspects, we need to translate the policies which are based on the subscriber's information for which the basic requirement is to get the user's location, the network topology, etc. The user's location can be used for general services like roaming, echo cancellation according to the requirement. The SDN controller uses Subscriber Information Base (SIB) for the mapping of the information of subscriber with user IP address. In the end, SDN controller uses these policies to translate policies based on users to policies based on packets.


Home Subscriber Server (HSS)
Policy Control and Charging Rules Function (PCRF)
Mobility Management Entity (MME)
Control Plane
Data Plane
SDN Controller
User Equipment (UE)
Base Station (eNodeB)
OF Switch

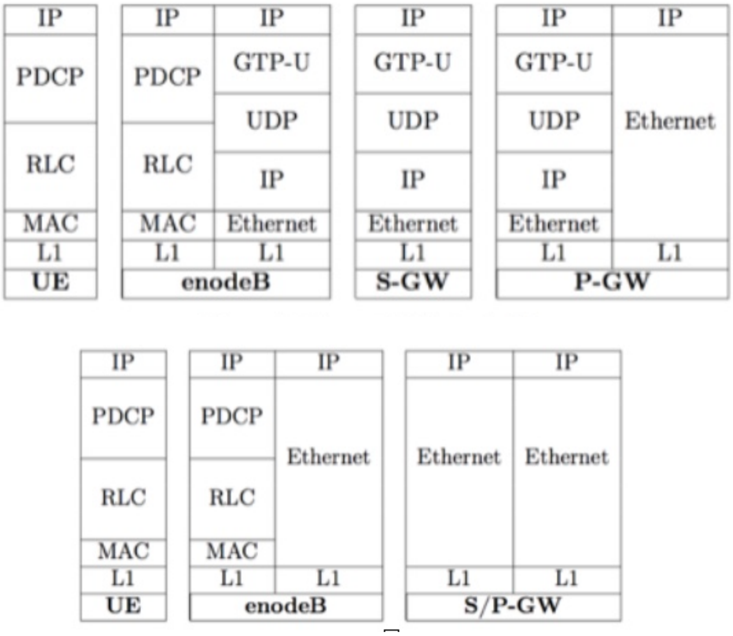
Second extension would be the offloading of the Controller with the help of the switches, as they can handle the events that may be difficult for the controller to handle or controller would need excess of resources. As we know that these agents have some of the control over the environment, so they can figure out some of the issues independently and faster than the controller.

Next extension is all about the patterns and actions of the switches and the last one is about the virtualization of the networks which is again based on the subscriber's parameters.

Moving Further, it is stated that caching should be done only in the backhaul network which can performed only on a legacy IP network. However, in current backhaul networks of the traditional LTE architecture, the traffic is encapsulated in GTP tunnels. Because of these tunnels, caching in backhaul network cannot be performed. SDN permits to remove the GTP tunnels. Also, a SDN based packet network is used in the place of the S/P-GW as it is also removed and SDN controller can directly communicates with the MME in order to set up the rules on the SDN switches. And MME handles the handovers with a notification which is being sent to SDN Controller from MME in order to change rules on the SDN switches.

iv) SDN-based caching validation in mobile networks:

This chapter presents the implementation of the prototype and the tests we performed to validate the concept. In this section, we first present the environment of the tests, the implementation and the results of these validation tests.



First Solution to join SDN and LTE:

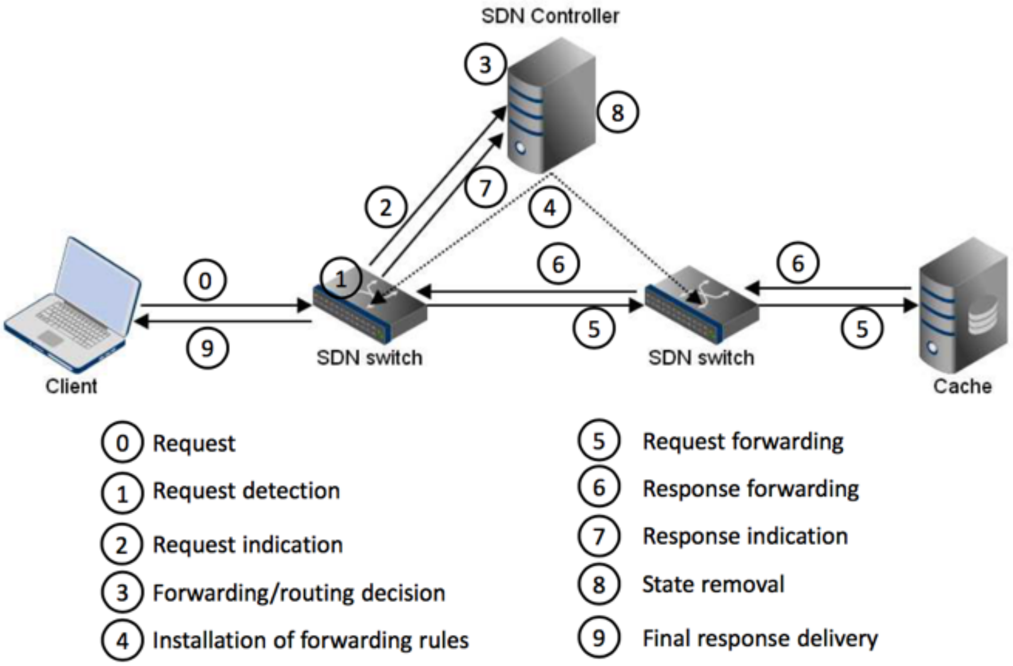
A first answer to join SDN and CCN was proposed by N. Blefari-Melazzi et al. Their outline is based on an another IP transport protocol and is not interoperable with legacy systems. There are some systems which supports only TCP or UDP and to interface with these particular systems/networks, Gateways are utilized.

Infrastructure:

The infrastructure comprises of two types of nodes as listed beneath:

End nodes and Border nodes.

End nodes are the ones which are corresponds to Users and Border nodes are the nodes which are responsible for the forwarding of user requests and their acknowledgements. Border nodes takes into account the information about the caching nodes which they named as internal nodes.



In this scenario, the customer will first issue a request and send it to the nearest border node. The border node will forward it in view of the name of the content, and will fill its Pending Interest Table (PIT) with the name of the content and the port from where the request was originated. Next step is that the request will reach the border nodes or internal nodes. There, it will either be a cache hit or a cache miss. Cache hit will occur if the requested content is in cache and cache miss is the condition in which the requested content is not present in the cache itself. In case of "Cache hit," requested content will be sent back to the user. However, in the case of "Cache miss," the same request will be forwarded to the main controller for the contents.

Operation:

They defined a system which helps in the process of mapping and they named it as "Name Resolution System (NRS)" which maps the element name with the redefined port fields of the packet so the requested element can be effortlessly recognized. Nodes are used for storing the response which was received from Name Resolution System (NRS) . For the proper functioning as requirements, it is mandatory that a single request for content should be happened to the NRS, and it should be until the response timeout expires. Then, the combination of the ports may be used again.

The ingress border node will send the approaching client's demand to the controller that will inquiry the NRS to get the identifier of the object. At that point the controller will set the guidelines to forward either to the correct cache or to the egress switch if the substance is not stored. IP packets were used for sending all the requests and responses for transfer between nodes.

However, there is a limit on the content which can be cached and it is because of mapping that only 232 pieces of information can be requested at one time.

Besides, this arrangement utilizes a custom transport protocol. So different arrangements have been proposed to comprehend these issues.

7) Implementation of the Prototype:

This chapter defines the implementation of the prototype and the tests I have performed to validate the concepts. Firstly, the architecture has been given in which I have defined three cases and after that performance analysis on the basis of the same cases.

I have used SDN Controller Node which can be defined as a node that performs SDN controller tasks, which are creating and managing network flows, applies definitions of applications. And it contains two different modules which are as beneath:

Sensor mode module and Controller server module

Sensor mote module: It runs on a sensor mote and it utilizes ActiveMessageC in order to communicate with the SDN-enabled sensor node. In order to do the basic task of forwarding and receiving messages, it utilizes SerialActiveMessageC. It is also responsible for forwarding messages to the controller server module and it receives messages so that it could be sent to the network from the controller server module. For the adjustment and management of these messages, portion of TinySDNControllerC comes into action.

Controller server module: It contains the control plane logic, i.e., hosts controller applications and manages network flows and topology information.

i) Architecture:

The architecture is composed of several elements: including caches, proxies, a SDN controller and a cache controller. Figure below presents the design and the tests were performed with two configurations:

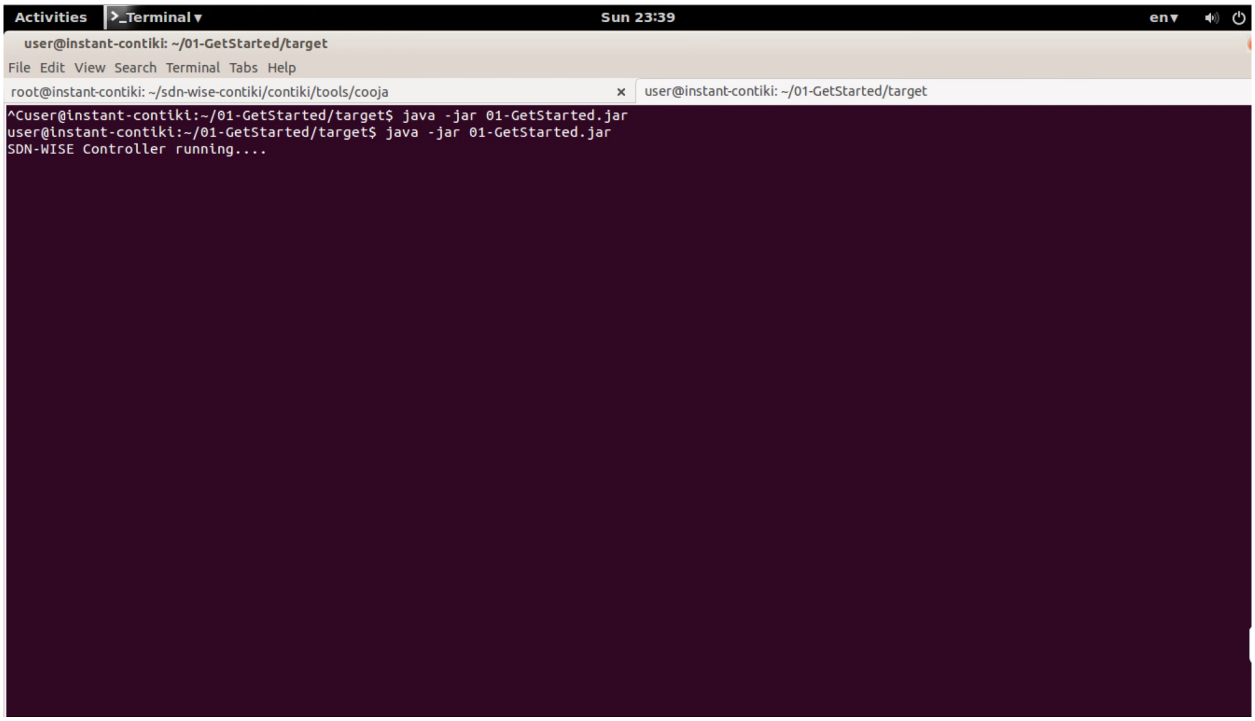
- Without any caching; directly fetching the content from Main Controller
- With caching in the topology

In the first scenario, there will not be any Cache Server, the request has been sent directly to the Main Controller and the Main Controller will revert to the following request by the user.

However, in the second scenario, the placed a Cache Controller in between the users and the Main Controller, which acts as a self mini main controller and will handle the requests of the users. It will revert to the users' requests which in turns increases the performance by increasing the throughput.

First scenario is without cache server:

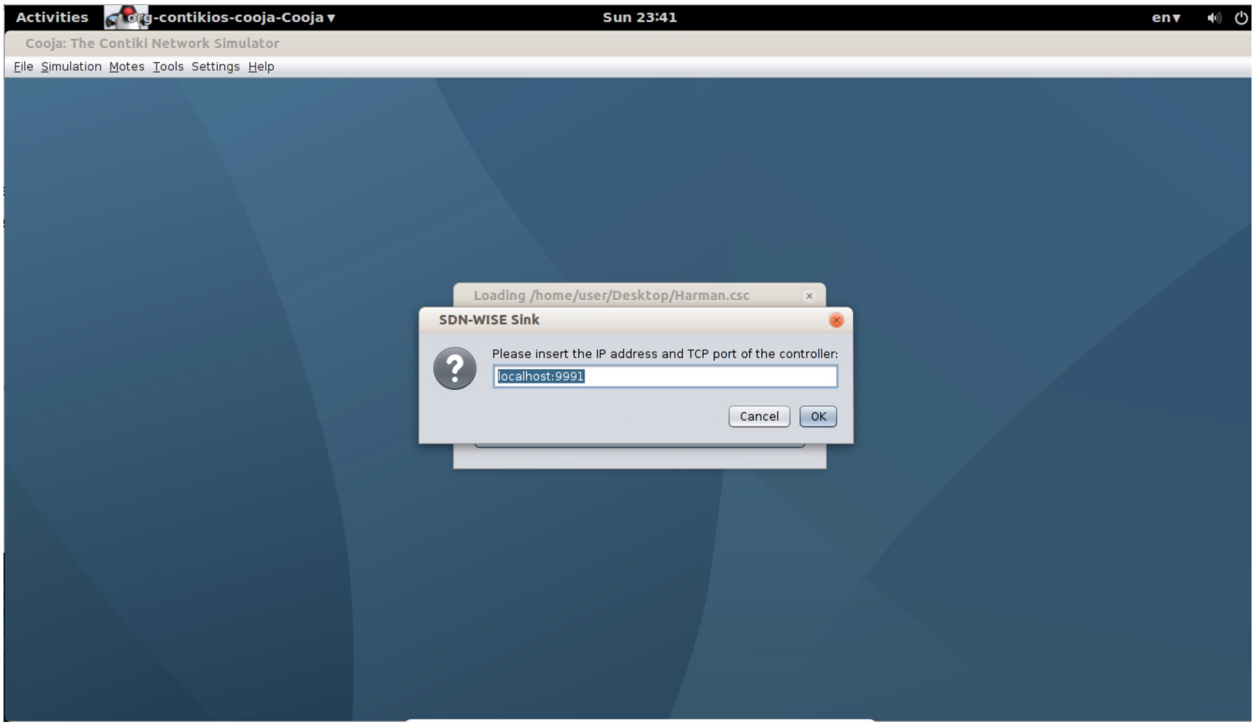
SDN Controller Terminal is as shown below:



Activities
>_Terminal ▾
Sun 23:39
en ▾
user@instant-contiki: ~/01-GetStarted/target
File Edit View Search Terminal Tabs Help
root@instant-contiki: ~/sdn-wise-contiki/contiki/tools/cooja
user@instant-contiki: ~/01-GetStarted/target

```
^Cuser@instant-contiki:~/01-GetStarted/target$ java -jar 01-GetStarted.jar
user@instant-contiki:~/01-GetStarted/target$ java -jar 01-GetStarted.jar
SDN-WISE Controller running....
```

Controller IP Address and TCP port is as following:



Activities
org-contikios-cooja-Cooja ▾
Sun 23:41
en ▾
Cooja: The Contiki Network Simulator
File Simulation Motes Tools Settings Help
Loading /home/user/Desktop/Harman.csc
SDN-WISE Sink
Cancel
OK

Please insert the IP address and TCP port of the controller:

localhost:9991

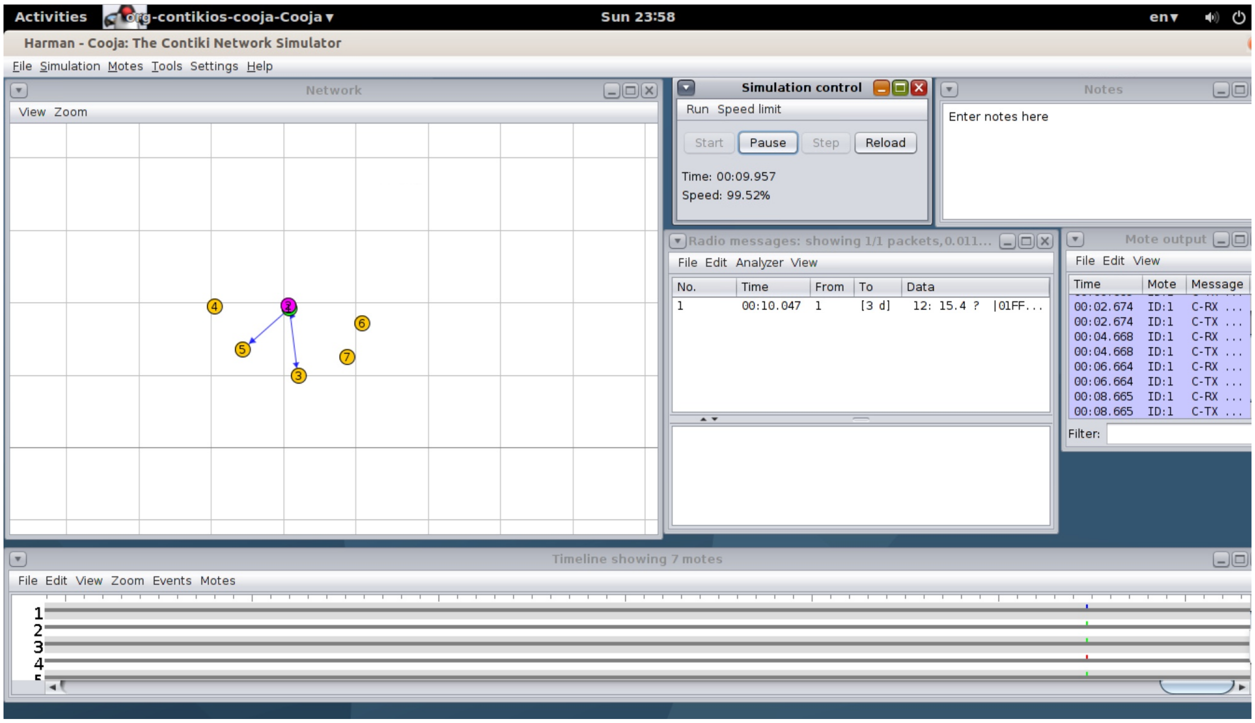
ii) Next is the topology:

As I’m using five hosts and the following scenario is without using Cache Server:

In this case, if any hosts wants to communicate, then the request directly goes to the Main Controller instead of going through Cache Server. And the Main Controller would respond to the following requests of the hosts.

However, this increases load on the Main Controller as it has to receive and revert every request with required information.

Moreover, in the following scenarios we can see that this load on the Main Controller would leads to decreasing throughput of the Main Controller. Thus, it can be concluded that if the Main Controller is the only one in the topology responsible for the entire functioning then by somehow it might leads to decrease in performance.


Activities
org-contikios-cooja-Cooja ▾
Sun 23:58
en▾
Harman - Cooja: The Contiki Network Simulator
File Simulation Motes Tools Settings Help
Network
View Zoom
1
3
4
5
6
7
Simulation control
Run Speed limit
Start
Pause
Step
Reload
Notes
Radio messages: showing 1/1 packets,0.011...
File Edit Analyzer View
Mote output
File Edit View
Filter:
Timeline showing 7 motes
File Edit View Zoom Events Motes
1
2
3
4

Time: 00:09.957
Speed: 99.52%

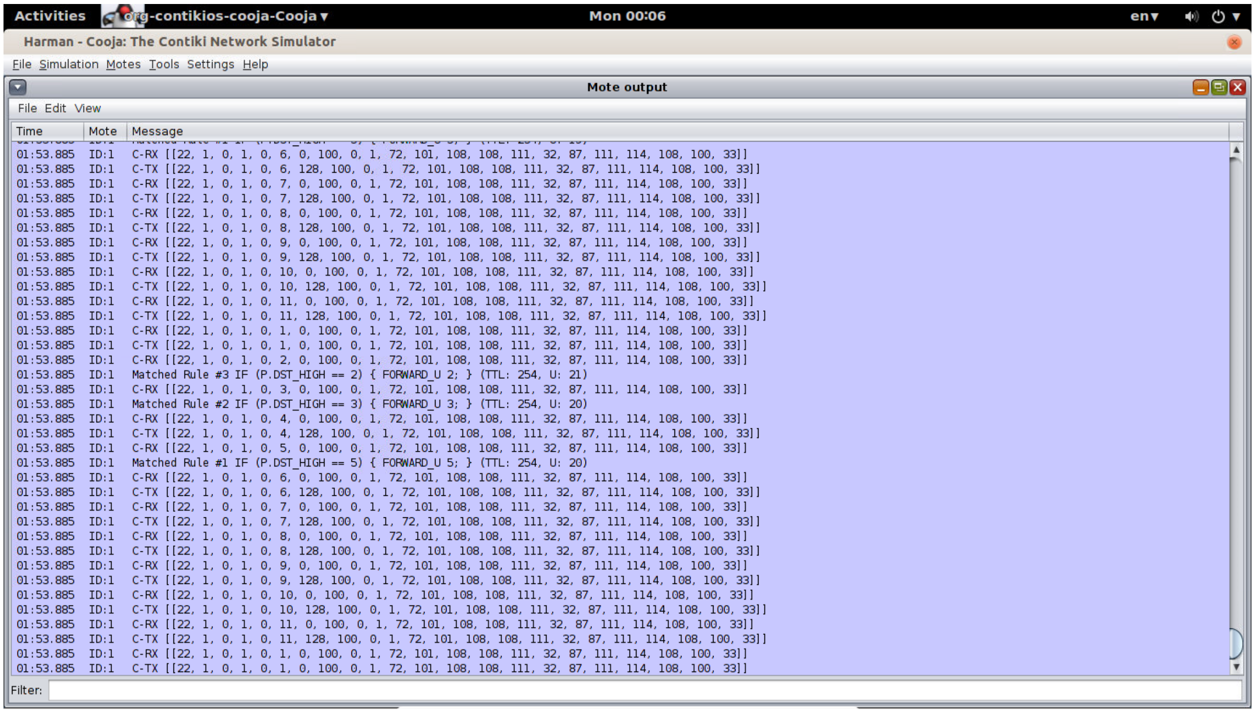
Enter notes here

No.	Time	From	To	Data
1	00:10.047	1	[3 d]	12: 15.4 ? |01FF...

Time	Mote	Message
00:02.674	ID:1	C-RX ...
00:02.674	ID:1	C-TX ...
00:04.668	ID:1	C-RX ...
00:04.668	ID:1	C-TX ...
00:06.664	ID:1	C-RX ...
00:06.664	ID:1	C-TX ...
00:08.665	ID:1	C-RX ...
00:08.665	ID:1	C-TX ...

iii) Node Output without Cache:

This is the information about the output on every node which is our users. As in the topology without Cache, it does not mean that the users or nodes can't communicate, the following screenshot shows the output on nodes in case which has no Cache Server.



Activities
org-contikios-cooja-Cooja ▾
Mon 00:06
en ▾
Harman - Cooja: The Contiki Network Simulator
File Simulation Motes Tools Settings Help
Mote output
File Edit View
Filter:

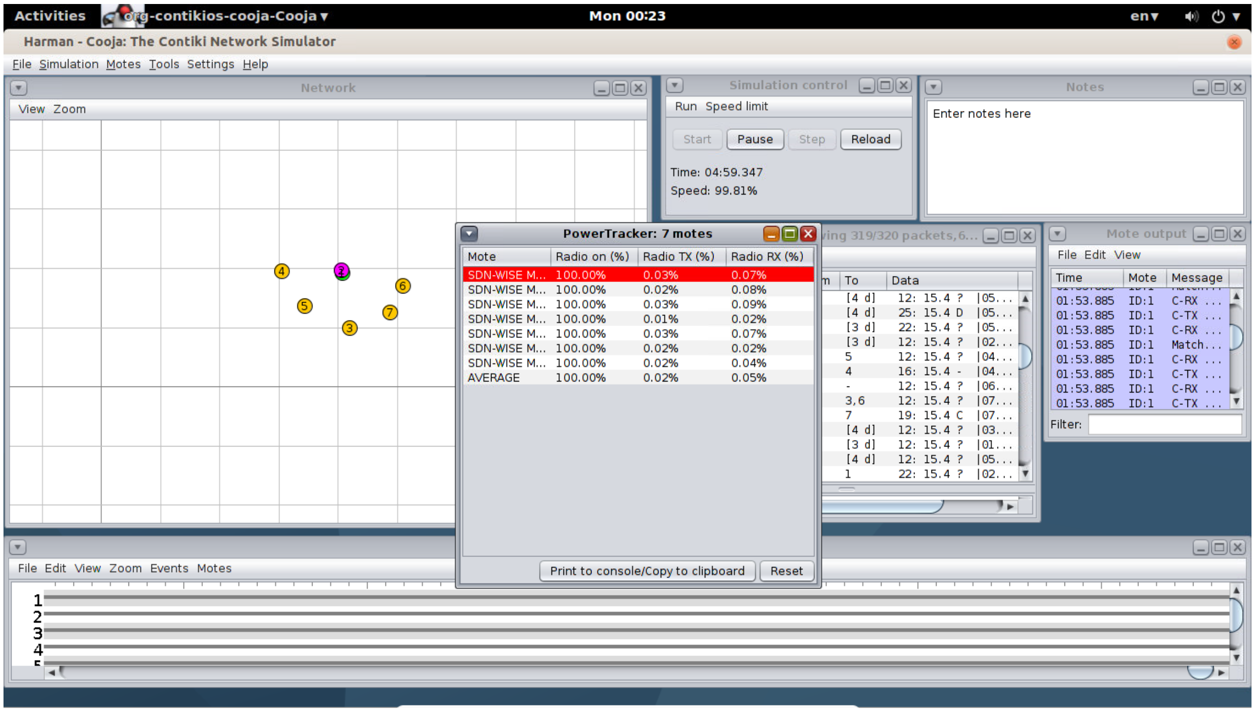
```
Time       Mote   Message
01:53.885  ID:1   C-RX [[22, 1, 0, 1, 0, 6, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-TX [[22, 1, 0, 1, 0, 6, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-RX [[22, 1, 0, 1, 0, 7, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-TX [[22, 1, 0, 1, 0, 7, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-RX [[22, 1, 0, 1, 0, 8, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-TX [[22, 1, 0, 1, 0, 8, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-RX [[22, 1, 0, 1, 0, 9, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-TX [[22, 1, 0, 1, 0, 9, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-RX [[22, 1, 0, 1, 0, 10, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-TX [[22, 1, 0, 1, 0, 10, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-RX [[22, 1, 0, 1, 0, 11, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-TX [[22, 1, 0, 1, 0, 11, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-RX [[22, 1, 0, 1, 0, 1, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-TX [[22, 1, 0, 1, 0, 1, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-RX [[22, 1, 0, 1, 0, 2, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   Matched Rule #3 IF (P.DST_HIGH == 2) { FORWARD_U 2; } (TTL: 254, U: 21)
01:53.885  ID:1   C-RX [[22, 1, 0, 1, 0, 3, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   Matched Rule #2 IF (P.DST_HIGH == 3) { FORWARD_U 3; } (TTL: 254, U: 20)
01:53.885  ID:1   C-RX [[22, 1, 0, 1, 0, 4, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-TX [[22, 1, 0, 1, 0, 4, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-RX [[22, 1, 0, 1, 0, 5, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   Matched Rule #1 IF (P.DST_HIGH == 5) { FORWARD_U 5; } (TTL: 254, U: 20)
01:53.885  ID:1   C-RX [[22, 1, 0, 1, 0, 6, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-TX [[22, 1, 0, 1, 0, 6, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-RX [[22, 1, 0, 1, 0, 7, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-TX [[22, 1, 0, 1, 0, 7, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-RX [[22, 1, 0, 1, 0, 8, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-TX [[22, 1, 0, 1, 0, 8, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-RX [[22, 1, 0, 1, 0, 9, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-TX [[22, 1, 0, 1, 0, 9, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-RX [[22, 1, 0, 1, 0, 10, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-TX [[22, 1, 0, 1, 0, 10, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-RX [[22, 1, 0, 1, 0, 11, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-TX [[22, 1, 0, 1, 0, 11, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-RX [[22, 1, 0, 1, 0, 1, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:53.885  ID:1   C-TX [[22, 1, 0, 1, 0, 1, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
```

In the above output, we can easily notice the time which has been taken by the Main Controller for providing required information to the users. Message is the filed which shows that whats the status of the output is. For instance, if the required information has been received or transmitted.

In the above Case, the information only has to be received by the users which is shown in the Message files with RX which is for receiving.

iv) Power Tracker:

For power consumption, it is recommended to always go for Contiki's own internal power profiler, which is COOJA simulates nodes which helps in estimating the energy and power consumption by the users or nodes. It also gives the user the idea about the average duty cycles of all simulated nodes.


Activities
org-contikios-cooja-Cooja ▾
Mon 00:23
en▾
Harman - Cooja: The Contiki Network Simulator
File Simulation Motes Tools Settings Help
Network
View Zoom
4
1
6
5
7
3
Simulation control
Run Speed limit
Start
Pause
Step
Reload
Notes
Enter notes here
ving 319/320 packets,6...
Mote output
File Edit View
Filter:
Print to console/Copy to clipboard
Reset
File Edit View Zoom Events Motes
1
2
3
4

Time: 04:59.347
Speed: 99.81%

PowerTracker: 7 motes

Mote	Radio on (%)	Radio TX (%)	Radio RX (%)
SDN-WISE M...	100.00%	0.03%	0.07%
SDN-WISE M...	100.00%	0.02%	0.08%
SDN-WISE M...	100.00%	0.03%	0.09%
SDN-WISE M...	100.00%	0.01%	0.02%
SDN-WISE M...	100.00%	0.03%	0.07%
SDN-WISE M...	100.00%	0.02%	0.02%
SDN-WISE M...	100.00%	0.02%	0.04%
AVERAGE	100.00%	0.02%	0.05%

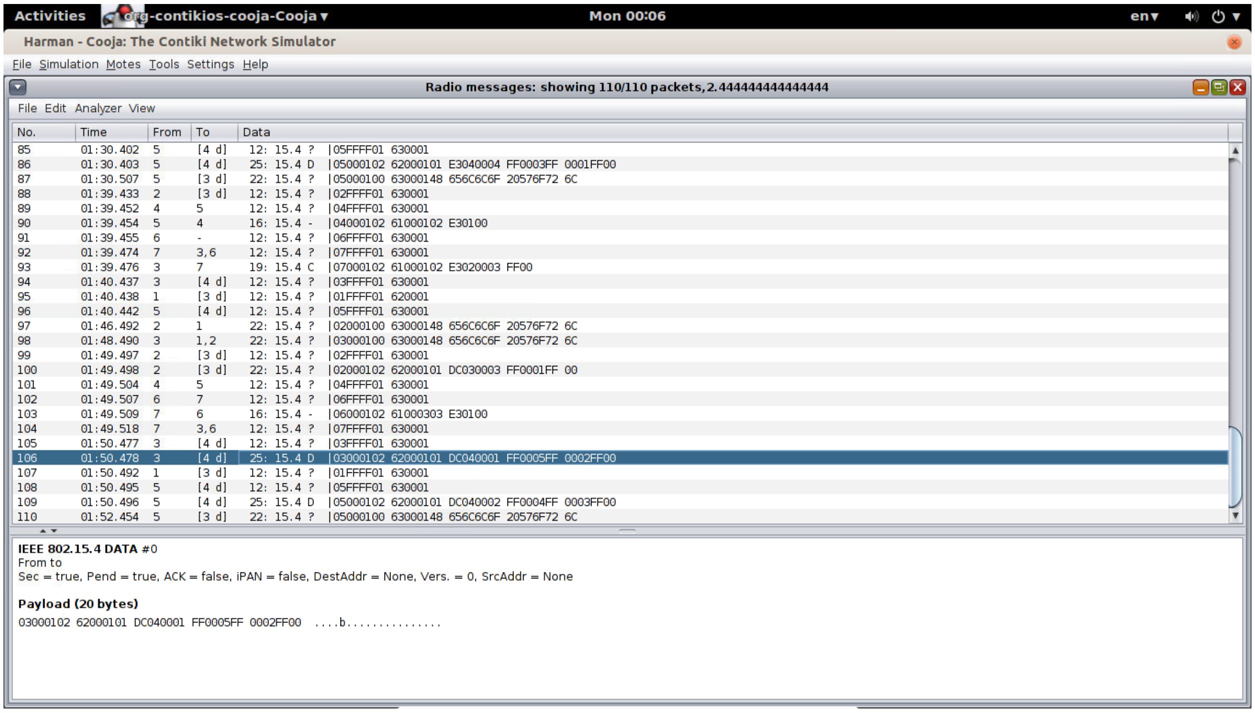
To	Data
[4 d]	12: 15.4 ? |05...
[4 d]	25: 15.4 D |05...
[3 d]	22: 15.4 ? |05...
[3 d]	12: 15.4 ? |02...
5	12: 15.4 ? |04...
4	16: 15.4 - |04...
-	12: 15.4 ? |06...
3,6	12: 15.4 ? |07...
7	19: 15.4 C |07...
[4 d]	12: 15.4 ? |03...
[3 d]	12: 15.4 ? |01...
[4 d]	12: 15.4 ? |05...
1	22: 15.4 ? |02...

Time	Mote	Message
01:53.885	ID:1	C-RX ...
01:53.885	ID:1	C-TX ...
01:53.885	ID:1	C-RX ...
01:53.885	ID:1	Match...
01:53.885	ID:1	C-RX ...
01:53.885	ID:1	C-TX ...
01:53.885	ID:1	C-RX ...
01:53.885	ID:1	C-TX ...

As presented in the above snapshot, the scenario is the same without Cache server, we are tracking the consumption of power of all the nodes. It has been illustrated for the Transmitting power and Receiving power. The results have been given in percentage and at the end of the results, average Transmitting power and Receiving power is also calculated and shown in percentage for both the ends: Transmitting and Receiving.

v) Radio Message without Cache:

Radio Traffic can be defined as “the animation which is responsible for showing the communication among the nodes.”


Activities
org-contikios-cooja-Cooja ▾
Mon 00:06
en▾
Harman - Cooja: The Contiki Network Simulator
File Simulation Motes Tools Settings Help
Radio messages: showing 110/110 packets,2.444444444444444
File Edit Analyzer View

No.	Time	From	To	Data
85	01:30.402	5	[4 d]	12: 15.4 ? |05FFFF01 630001
86	01:30.403	5	[4 d]	25: 15.4 D |05000102 62000101 E3040004 FF0003FF 0001FF00
87	01:30.507	5	[3 d]	22: 15.4 ? |05000100 63000148 656C6C6F 20576F72 6C
88	01:39.433	2	[3 d]	12: 15.4 ? |02FFFF01 630001
89	01:39.452	4	5	12: 15.4 ? |04FFFF01 630001
90	01:39.454	5	4	16: 15.4 - |04000102 61000102 E30100
91	01:39.455	6	-	12: 15.4 ? |06FFFF01 630001
92	01:39.474	7	3,6	12: 15.4 ? |07FFFF01 630001
93	01:39.476	3	7	19: 15.4 C |07000102 61000102 E3020003 FF00
94	01:40.437	3	[4 d]	12: 15.4 ? |03FFFF01 630001
95	01:40.438	1	[3 d]	12: 15.4 ? |01FFFF01 620001
96	01:40.442	5	[4 d]	12: 15.4 ? |05FFFF01 630001
97	01:46.492	2	1	22: 15.4 ? |02000100 63000148 656C6C6F 20576F72 6C
98	01:48.490	3	1,2	22: 15.4 ? |03000100 63000148 656C6C6F 20576F72 6C
99	01:49.497	2	[3 d]	12: 15.4 ? |02FFFF01 630001
100	01:49.498	2	[3 d]	22: 15.4 ? |02000102 62000101 DC030003 FF0001FF 00
101	01:49.504	4	5	12: 15.4 ? |04FFFF01 630001
102	01:49.507	6	7	12: 15.4 ? |06FFFF01 630001
103	01:49.509	7	6	16: 15.4 - |06000102 61000303 E30100
104	01:49.518	7	3,6	12: 15.4 ? |07FFFF01 630001
105	01:50.477	3	[4 d]	12: 15.4 ? |03FFFF01 630001
106	01:50.478	3	[4 d]	25: 15.4 D |03000102 62000101 DC040001 FF0005FF 0002FF00
107	01:50.492	1	[3 d]	12: 15.4 ? |01FFFF01 630001
108	01:50.495	5	[4 d]	12: 15.4 ? |05FFFF01 630001
109	01:50.496	5	[4 d]	25: 15.4 D |05000102 62000101 DC040002 FF0004FF 0003FF00
110	01:52.454	5	[3 d]	22: 15.4 ? |05000100 63000148 656C6C6F 20576F72 6C

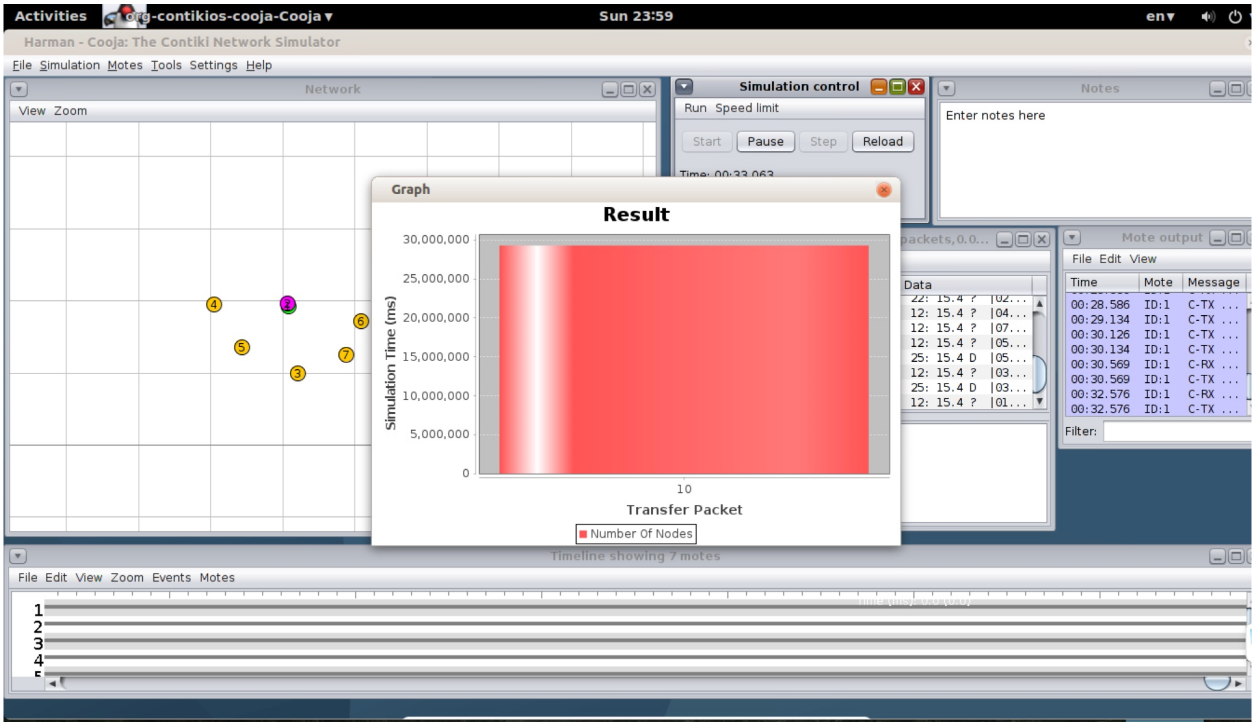
```
IEEE 802.15.4 DATA #0
From to
Sec = true, Pend = true, ACK = false, iPAN = false, DestAddr = None, Vers. = 0, SrcAddr = None

Payload (20 bytes)
03000102 62000101 DC040001 FF0005FF 0002FF00   ....b...............
```

Radio Message has a format which is shown as above in which ‘time’ is given which is as usual the time taken to perform the task of Transmitting request to the Main Controller and Receiving reply from the Main Controller. Next field is ‘from’ and ‘to’ which informs about the node from which node the request has been sent and to which node it has been sent. In case, the nodes are trying to communicate packets in between themselves, there is a message displayed ‘d’ which is that the packet has been dropped.

It is mandatory that the request has been only and only travelled from the users to the Main Controller, not in between the nodes or users. They should not be allowed to exchange information on any basis. That’s why the packet has been dropped as shown above.

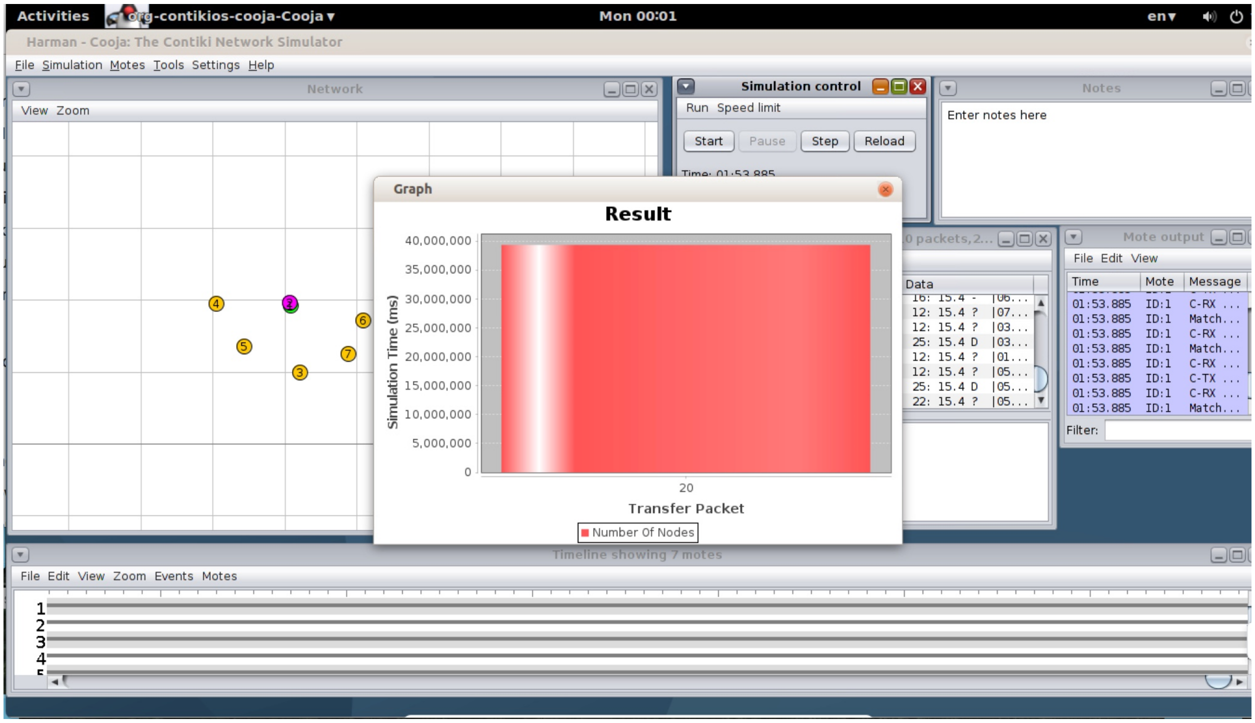
Radio Environment is the environment in which the coverage area of any particular node or user can be achieved by only clicking on it. Coverage area can be defined as the area up to which that particular node is permitted to access information. Beyond that the request generated from that particular node can’t be processed.

Next Task is to analyze the performance with packets, and I have done this analyzation with three different number of packets: 10 Packets, 20 Packets, 30 packets and the results are in the form of graph as shown below. Throughput is different for different number of packets.



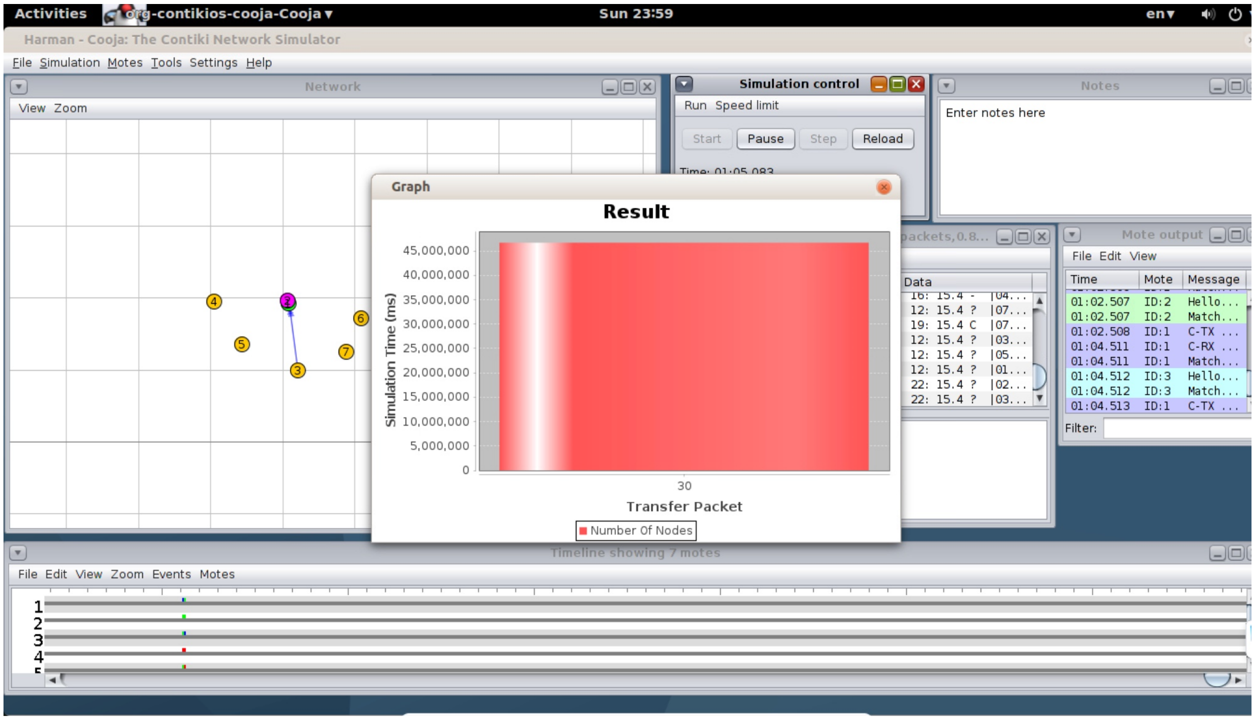
```
Data
22: 15.4 ?   |02...
12: 15.4 ?   |04...
12: 15.4 ?   |07...
12: 15.4 ?   |05...
25: 15.4 D   |05...
12: 15.4 ?   |03...
25: 15.4 D   |03...
12: 15.4 ?   |01...
```

```
Time       Mote   Message
00:28.586  ID:1   C-TX ...
00:29.134  ID:1   C-TX ...
00:30.126  ID:1   C-TX ...
00:30.134  ID:1   C-TX ...
00:30.569  ID:1   C-RX ...
00:30.569  ID:1   C-TX ...
00:32.576  ID:1   C-RX ...
00:32.576  ID:1   C-TX ...
```



```
Data
16: 15.4 -   |06...
12: 15.4 ?   |07...
12: 15.4 ?   |03...
25: 15.4 D   |03...
12: 15.4 ?   |01...
12: 15.4 ?   |05...
25: 15.4 D   |05...
22: 15.4 ?   |05...
```

```
Time       Mote   Message
01:53.885  ID:1   C-RX ...
01:53.885  ID:1   Match...
01:53.885  ID:1   C-RX ...
01:53.885  ID:1   Match...
01:53.885  ID:1   C-RX ...
01:53.885  ID:1   C-TX ...
01:53.885  ID:1   C-RX ...
01:53.885  ID:1   Match...
```



```
Data
16: 15.4 -  |04...
12: 15.4 ?  |07...
19: 15.4 C  |07...
12: 15.4 ?  |03...
12: 15.4 ?  |05...
12: 15.4 ?  |01...
22: 15.4 ?  |02...
22: 15.4 ?  |03...
```

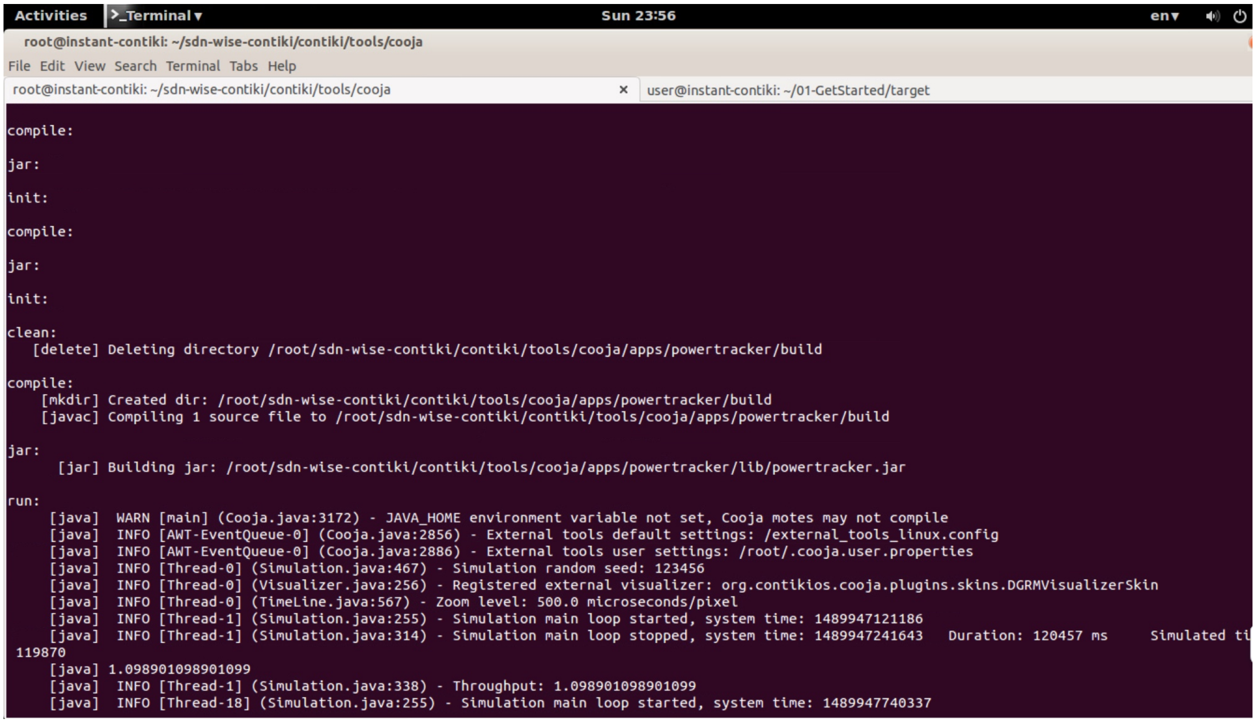
Time	Mote	Message
01:02.507	ID:2	Hello...
01:02.507	ID:2	Match...
01:02.508	ID:1	C-TX ...
01:04.511	ID:1	C-RX ...
01:04.511	ID:1	Match...
01:04.512	ID:3	Hello...
01:04.512	ID:3	Match...
01:04.513	ID:1	C-TX ...

In the graphs, transfer packets is on the x-axis and on y-axis, simulation time is there. Simulation Time can be defined as the estimated time for an end-to-end process. Stimulation Time Report can be the estimation of the execution for a single process and the given or maximum allowable execution time for that particular process.

Here, stimulation time is in milliseconds. On these basis, the overall throughput can be obtained which is the average for these different stimulations and different number of packets.

vi) Throughput:

Throughput can be defined as a measure of how many units of data or information being processed by a system in a given period of time. It depends upon the related measures of a system productivity which includes: response time, the period of time which has been taken by a system in order to receive a request and provide receipt to that particular request and the speed of the system for completing a particular task.


Activities
>_Terminal ▾
Sun 23:56
en ▾
root@instant-contiki: ~/sdn-wise-contiki/contiki/tools/cooja
File Edit View Search Terminal Tabs Help
root@instant-contiki: ~/sdn-wise-contiki/contiki/tools/cooja
user@instant-contiki: ~/01-GetStarted/target

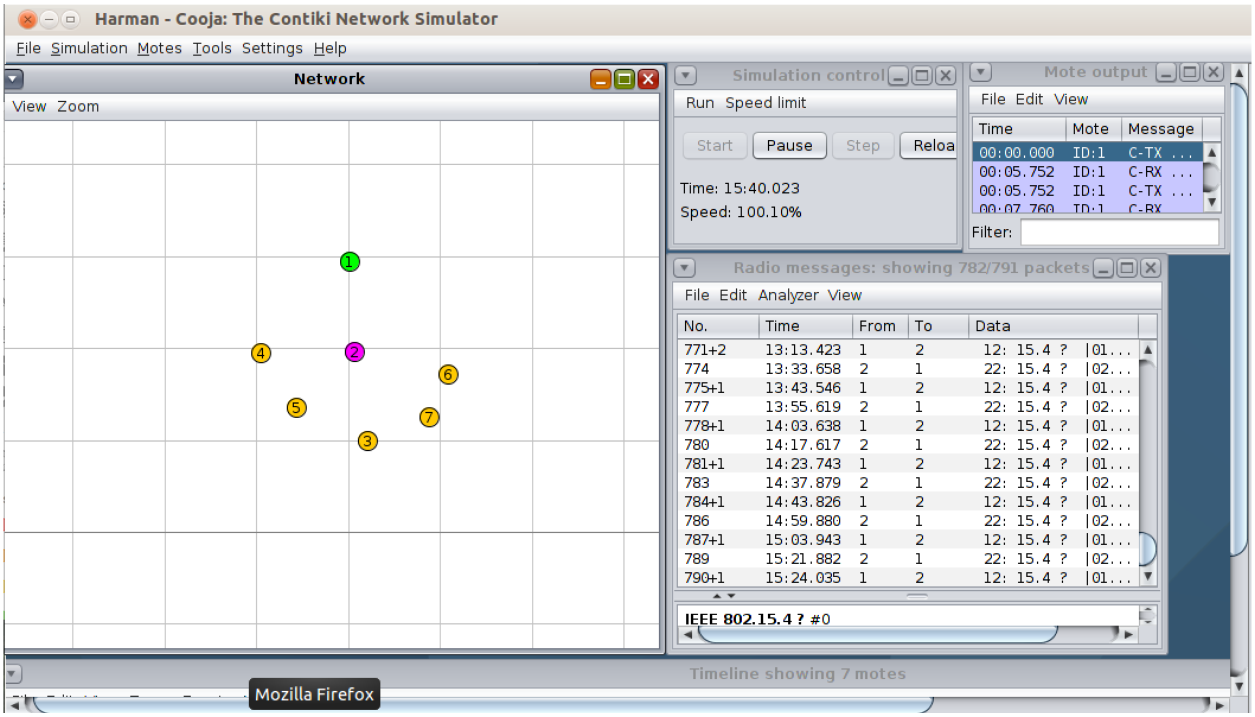
```
compile:

jar:

init:

compile:

jar:

init:

clean:
   [delete] Deleting directory /root/sdn-wise-contiki/contiki/tools/cooja/apps/powertracker/build

compile:
    [mkdir] Created dir: /root/sdn-wise-contiki/contiki/tools/cooja/apps/powertracker/build
    [javac] Compiling 1 source file to /root/sdn-wise-contiki/contiki/tools/cooja/apps/powertracker/build

jar:
      [jar] Building jar: /root/sdn-wise-contiki/contiki/tools/cooja/apps/powertracker/lib/powertracker.jar

run:
     [java]  WARN [main] (Cooja.java:3172) - JAVA_HOME environment variable not set, Cooja motes may not compile
     [java]  INFO [AWT-EventQueue-0] (Cooja.java:2856) - External tools default settings: /external_tools_linux.config
     [java]  INFO [AWT-EventQueue-0] (Cooja.java:2886) - External tools user settings: /root/.cooja.user.properties
     [java]  INFO [Thread-0] (Simulation.java:467) - Simulation random seed: 123456
     [java]  INFO [Thread-0] (Visualizer.java:256) - Registered external visualizer: org.contikios.cooja.plugins.skins.DGRMVisualizerSkin
     [java]  INFO [Thread-0] (TimeLine.java:567) - Zoom level: 500.0 microseconds/pixel
     [java]  INFO [Thread-1] (Simulation.java:255) - Simulation main loop started, system time: 1489947121186
     [java]  INFO [Thread-1] (Simulation.java:314) - Simulation main loop stopped, system time: 1489947241643   Duration: 120457 ms     Simulated ti
 119870
     [java] 1.098901098901099
     [java]  INFO [Thread-1] (Simulation.java:338) - Throughput: 1.098901098901099
     [java]  INFO [Thread-18] (Simulation.java:255) - Simulation main loop started, system time: 1489947740337
```

As shown in the above snapshot, the throughput for topology without Cache server is **“1.0989”**.

Moving Further, we performed the same analyzation on the same topology, with difference of Cache Server. Now we placed a Cache Server in between the hosts and Main Controller.

As Cache Server can be defined as the server for providing response to the users’ requests, which in turn decreases the load on the Main Controller and increase the performance and throughput of the system.

Topology with Cache Server:



Time: 15:40.023
Speed: 100.10%

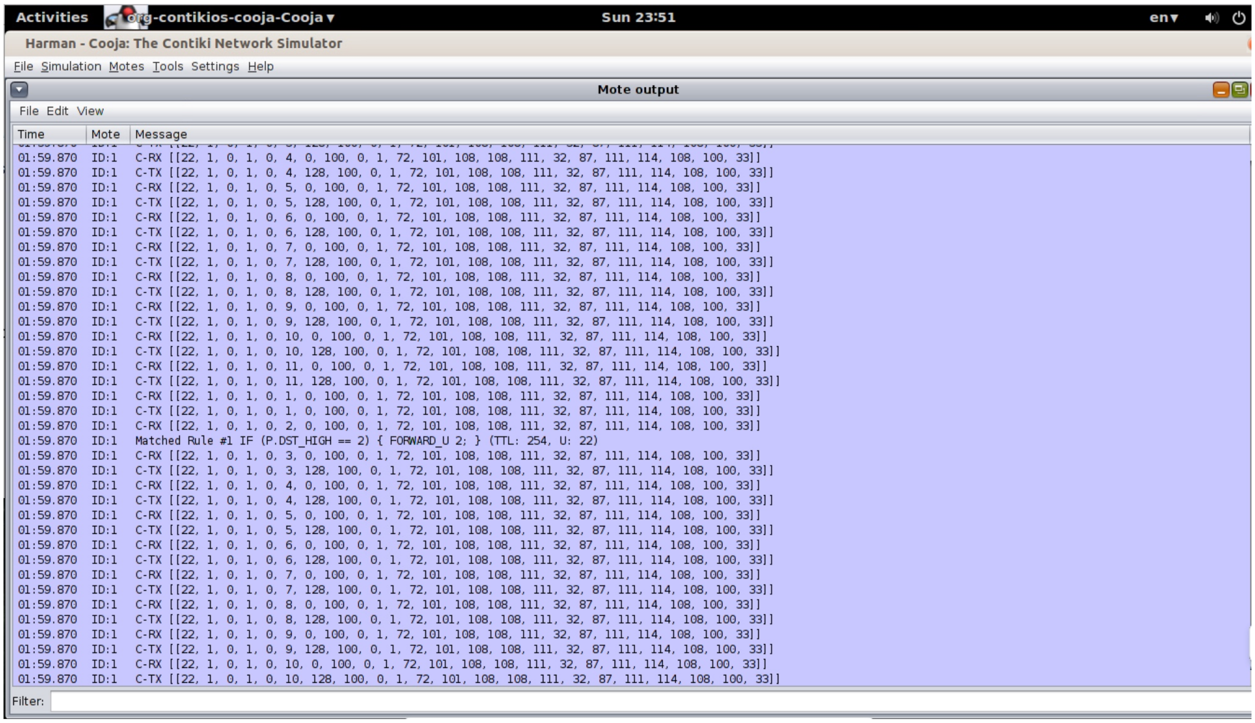
Time	Mote	Message
00:00.000	ID:1	C-TX ...
00:05.752	ID:1	C-RX ...
00:05.752	ID:1	C-TX ...
00:07.760	ID:1	C-RX

Filter:

No.	Time	From	To	Data
771+2	13:13.423	1	2	12: 15.4 ? |01...
774	13:33.658	2	1	22: 15.4 ? |02...
775+1	13:43.546	1	2	12: 15.4 ? |01...
777	13:55.619	2	1	22: 15.4 ? |02...
778+1	14:03.638	1	2	12: 15.4 ? |01...
780	14:17.617	2	1	22: 15.4 ? |02...
781+1	14:23.743	1	2	12: 15.4 ? |01...
783	14:37.879	2	1	22: 15.4 ? |02...
784+1	14:43.826	1	2	12: 15.4 ? |01...
786	14:59.880	2	1	22: 15.4 ? |02...
787+1	15:03.943	1	2	12: 15.4 ? |01...
789	15:21.882	2	1	22: 15.4 ? |02...
790+1	15:24.035	1	2	12: 15.4 ? |01...

IEEE 802.15.4 ? #0

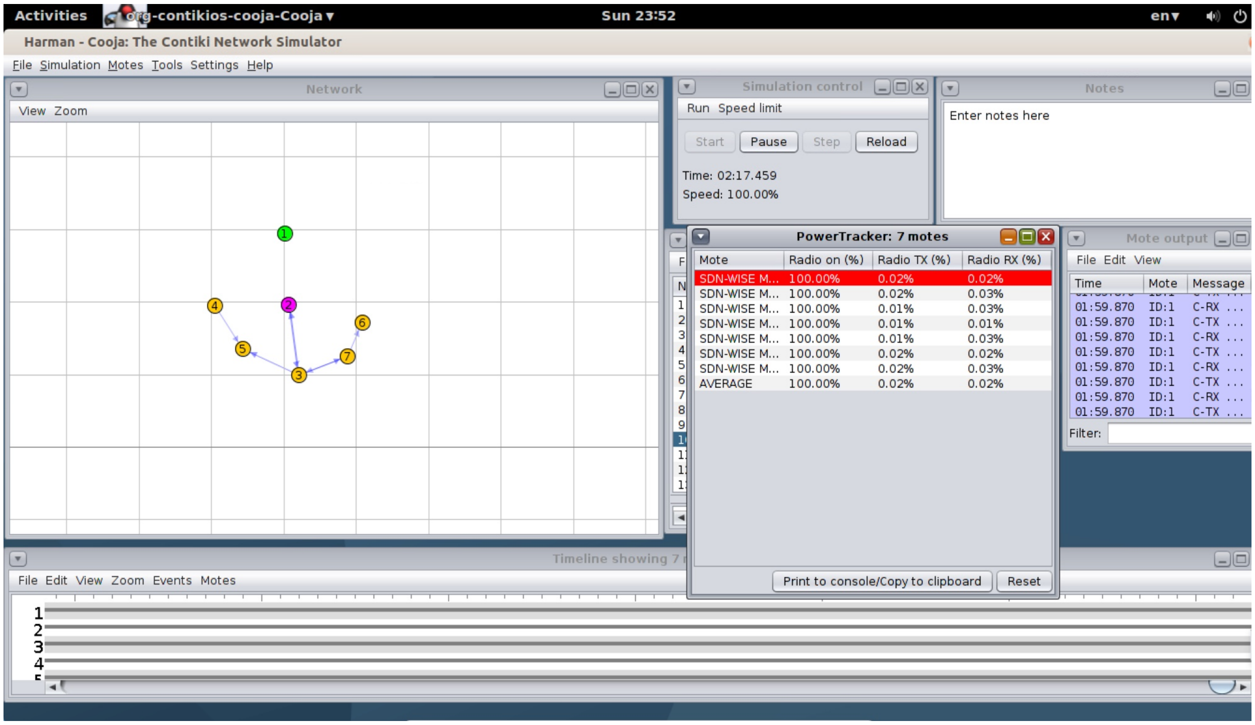
Node Output with Cache:


Activities
org-contikios-cooja-Cooja ▾
Sun 23:51
en ▾
Harman - Cooja: The Contiki Network Simulator
File Simulation Motes Tools Settings Help
Mote output
File Edit View

```
Time       Mote   Message
01:59.870  ID:1   C-RX [[22, 1, 0, 1, 0, 4, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-TX [[22, 1, 0, 1, 0, 4, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-RX [[22, 1, 0, 1, 0, 5, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-TX [[22, 1, 0, 1, 0, 5, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-RX [[22, 1, 0, 1, 0, 6, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-TX [[22, 1, 0, 1, 0, 6, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-RX [[22, 1, 0, 1, 0, 7, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-TX [[22, 1, 0, 1, 0, 7, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-RX [[22, 1, 0, 1, 0, 8, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-TX [[22, 1, 0, 1, 0, 8, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-RX [[22, 1, 0, 1, 0, 9, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-TX [[22, 1, 0, 1, 0, 9, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-RX [[22, 1, 0, 1, 0, 10, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-TX [[22, 1, 0, 1, 0, 10, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-RX [[22, 1, 0, 1, 0, 11, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-TX [[22, 1, 0, 1, 0, 11, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-RX [[22, 1, 0, 1, 0, 1, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-TX [[22, 1, 0, 1, 0, 1, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-RX [[22, 1, 0, 1, 0, 2, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   Matched Rule #1 IF (P.DST_HIGH == 2) { FORWARD_U 2; } (TTL: 254, U: 22)
01:59.870  ID:1   C-RX [[22, 1, 0, 1, 0, 3, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-TX [[22, 1, 0, 1, 0, 3, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-RX [[22, 1, 0, 1, 0, 4, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-TX [[22, 1, 0, 1, 0, 4, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-RX [[22, 1, 0, 1, 0, 5, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-TX [[22, 1, 0, 1, 0, 5, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-RX [[22, 1, 0, 1, 0, 6, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-TX [[22, 1, 0, 1, 0, 6, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-RX [[22, 1, 0, 1, 0, 7, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-TX [[22, 1, 0, 1, 0, 7, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-RX [[22, 1, 0, 1, 0, 8, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-TX [[22, 1, 0, 1, 0, 8, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-RX [[22, 1, 0, 1, 0, 9, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-TX [[22, 1, 0, 1, 0, 9, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-RX [[22, 1, 0, 1, 0, 10, 0, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
01:59.870  ID:1   C-TX [[22, 1, 0, 1, 0, 10, 128, 100, 0, 1, 72, 101, 108, 108, 111, 32, 87, 111, 114, 108, 100, 33]]
```

Filter:

Power Tracker with Cache:


Activities
org-contikios-cooja-Cooja ▾
Sun 23:52
en ▾
Harman - Cooja: The Contiki Network Simulator
File Simulation Motes Tools Settings Help
Network
View Zoom
1
2
3
4
5
6
7
Simulation control
Run Speed limit
Start
Pause
Step
Reload
Notes
Enter notes here
Mote output
File Edit View
Filter:
Timeline showing 7
File Edit View Zoom Events Motes
Print to console/Copy to clipboard
Reset

Time: 02:17.459
Speed: 100.00%

PowerTracker: 7 motes

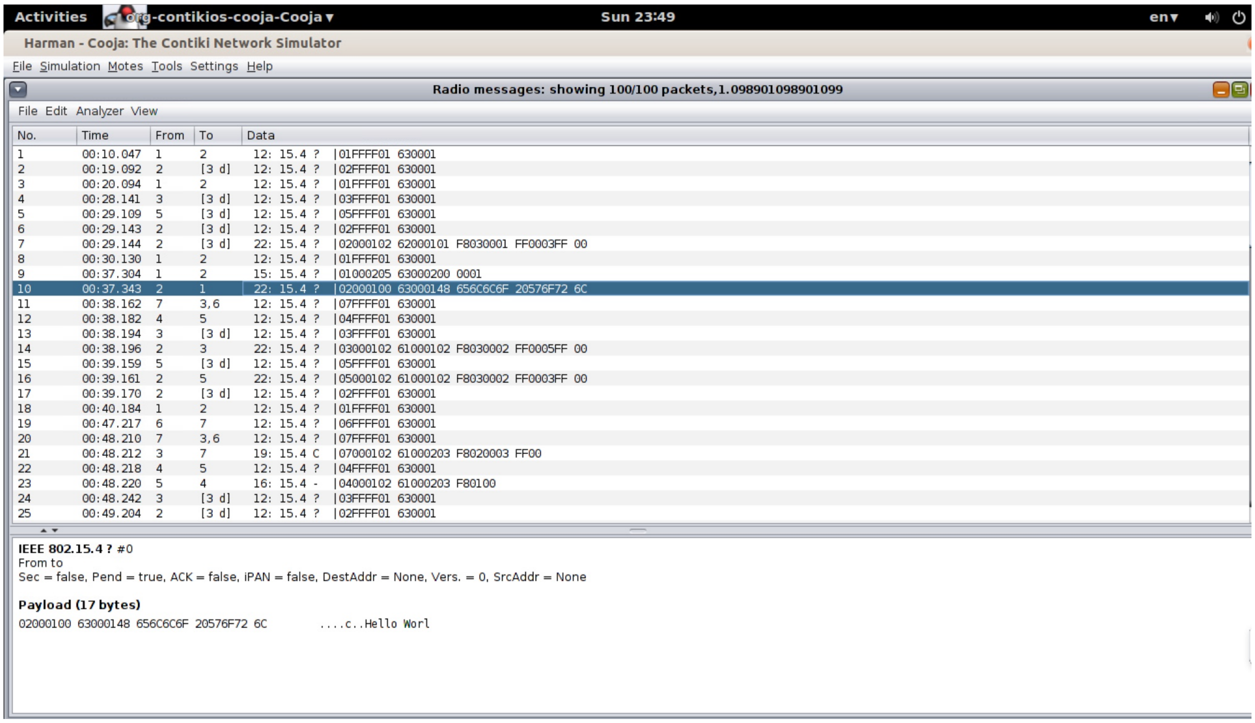
Mote	Radio on (%)	Radio TX (%)	Radio RX (%)
SDN-WISE M...	100.00%	0.02%	0.02%
SDN-WISE M...	100.00%	0.02%	0.03%
SDN-WISE M...	100.00%	0.01%	0.03%
SDN-WISE M...	100.00%	0.01%	0.01%
SDN-WISE M...	100.00%	0.01%	0.03%
SDN-WISE M...	100.00%	0.02%	0.02%
SDN-WISE M...	100.00%	0.02%	0.03%
AVERAGE	100.00%	0.02%	0.02%

```
Time       Mote  Message
01:59.870  ID:1  C-RX ...
01:59.870  ID:1  C-TX ...
01:59.870  ID:1  C-RX ...
01:59.870  ID:1  C-TX ...
01:59.870  ID:1  C-RX ...
01:59.870  ID:1  C-TX ...
01:59.870  ID:1  C-RX ...
01:59.870  ID:1  C-TX ...
```

Transmitted Power and Receiving Power can be extracted with Power Tracker. This has been performed for the topology with Cache Server.

Radio Message with Cache:

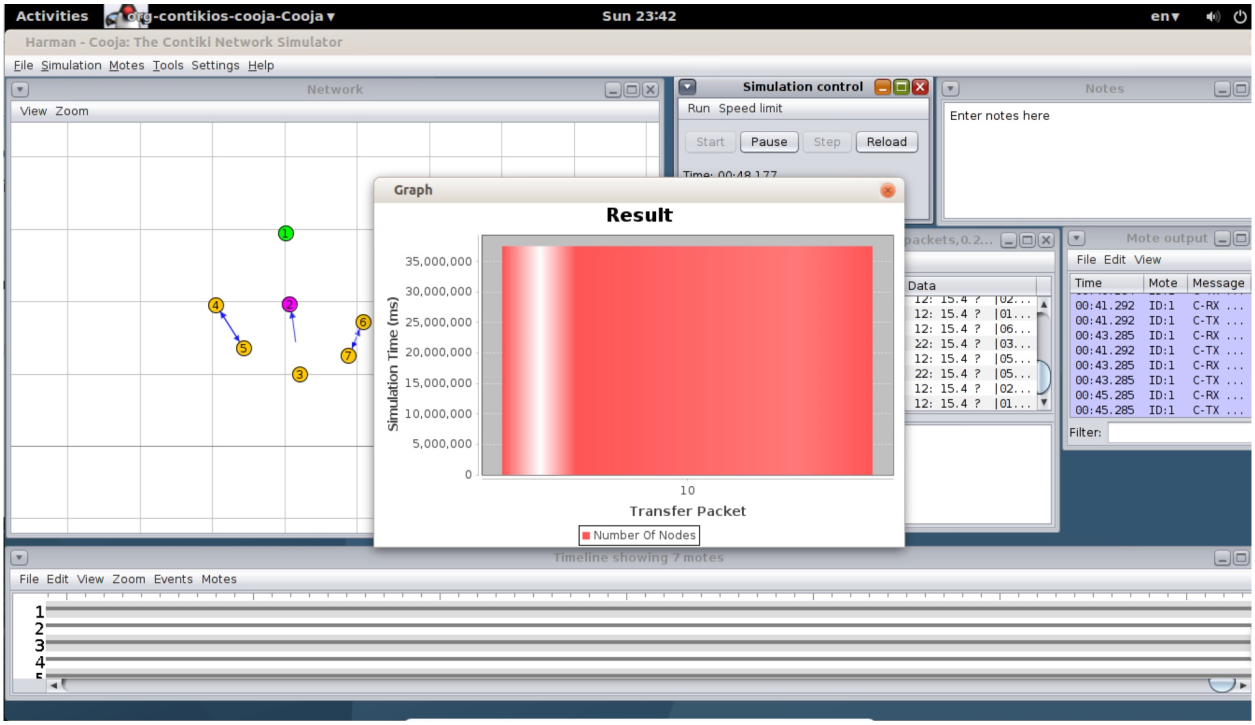
Radio Environment is the environment in which the coverage area of any particular node or user can be achieved by only clicking on it. Coverage area can be defined as the area up to which that particular node is permitted to access information. Beyond that the request generated from that particular node can’t be processed.


Activities
org-contikios-cooja-Cooja ▾
Sun 23:49
en ▾
Harman - Cooja: The Contiki Network Simulator
File Simulation Motes Tools Settings Help
Radio messages: showing 100/100 packets,1.098901098901099
File Edit Analyzer View

No.	Time	From	To	Data
1	00:10.047	1	2	12: 15.4 ? |01FFFF01 630001
2	00:19.092	2	[3 d]	12: 15.4 ? |02FFFF01 630001
3	00:20.094	1	2	12: 15.4 ? |01FFFF01 630001
4	00:28.141	3	[3 d]	12: 15.4 ? |03FFFF01 630001
5	00:29.109	5	[3 d]	12: 15.4 ? |05FFFF01 630001
6	00:29.143	2	[3 d]	12: 15.4 ? |02FFFF01 630001
7	00:29.144	2	[3 d]	22: 15.4 ? |02000102 62000101 F8030001 FF0003FF 00
8	00:30.130	1	2	12: 15.4 ? |01FFFF01 630001
9	00:37.304	1	2	15: 15.4 ? |01000205 63000200 0001
10	00:37.343	2	1	22: 15.4 ? |02000100 63000148 656C6C6F 20576F72 6C
11	00:38.162	7	3,6	12: 15.4 ? |07FFFF01 630001
12	00:38.182	4	5	12: 15.4 ? |04FFFF01 630001
13	00:38.194	3	[3 d]	12: 15.4 ? |03FFFF01 630001
14	00:38.196	2	3	22: 15.4 ? |03000102 61000102 F8030002 FF0005FF 00
15	00:39.159	5	[3 d]	12: 15.4 ? |05FFFF01 630001
16	00:39.161	2	5	22: 15.4 ? |05000102 61000102 F8030002 FF0003FF 00
17	00:39.170	2	[3 d]	12: 15.4 ? |02FFFF01 630001
18	00:40.184	1	2	12: 15.4 ? |01FFFF01 630001
19	00:47.217	6	7	12: 15.4 ? |06FFFF01 630001
20	00:48.210	7	3,6	12: 15.4 ? |07FFFF01 630001
21	00:48.212	3	7	19: 15.4 C |07000102 61000203 F8020003 FF00
22	00:48.218	4	5	12: 15.4 ? |04FFFF01 630001
23	00:48.220	5	4	16: 15.4 - |04000102 61000203 F80100
24	00:48.242	3	[3 d]	12: 15.4 ? |03FFFF01 630001
25	00:49.204	2	[3 d]	12: 15.4 ? |02FFFF01 630001

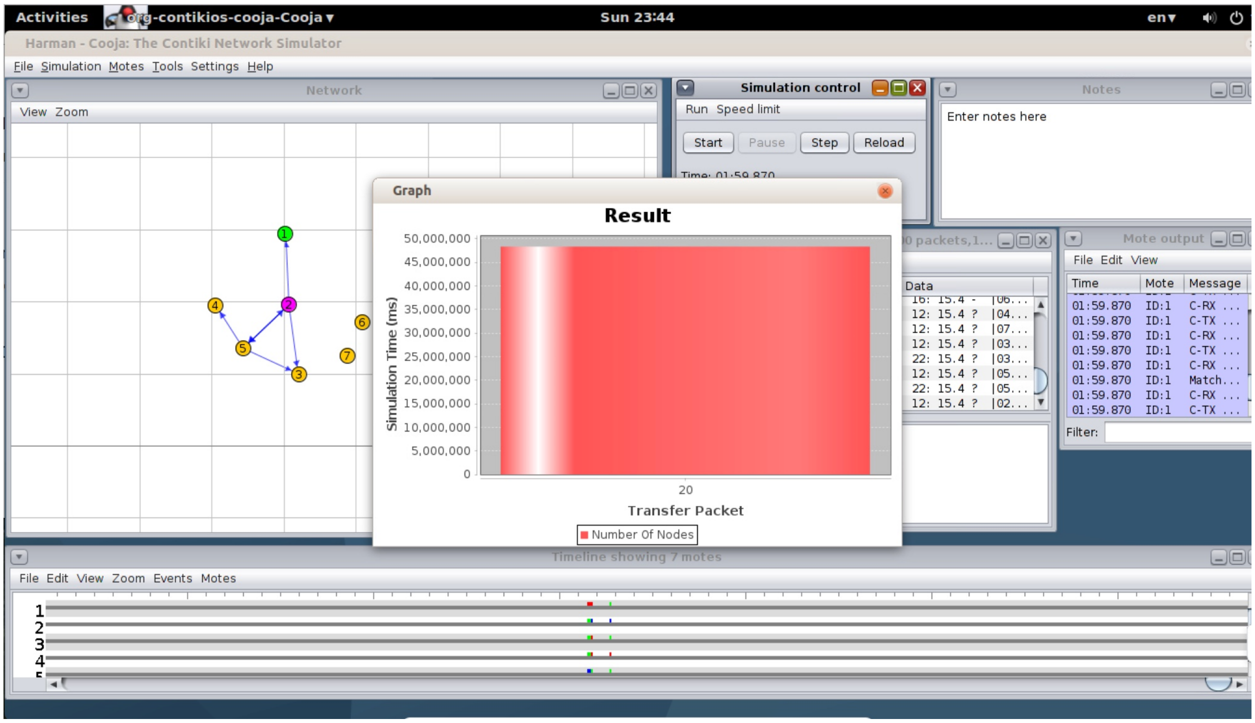
```
IEEE 802.15.4 ? #0
From to
Sec = false, Pend = true, ACK = false, iPAN = false, DestAddr = None, Vers. = 0, SrcAddr = None

Payload (17 bytes)
02000100 63000148 656C6C6F 20576F72 6C        ....c..Hello Worl
```

Next Task is same as we did without Cache, analyzation the performance with different packets: 10, 20 and 30 packets.



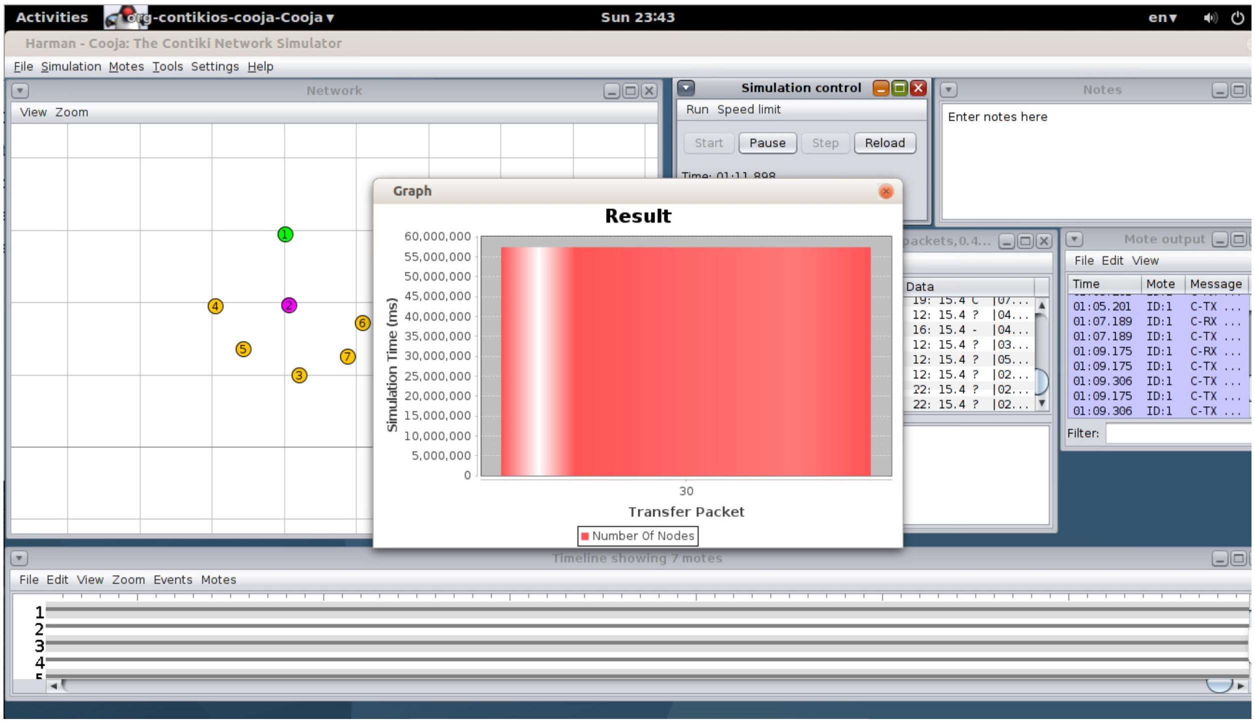
```
Data
12: 15.4 ?  |02...
12: 15.4 ?  |01...
12: 15.4 ?  |06...
22: 15.4 ?  |03...
12: 15.4 ?  |05...
22: 15.4 ?  |05...
12: 15.4 ?  |02...
12: 15.4 ?  |01...
```

Time	Mote	Message
00:41.292	ID:1	C-RX ...
00:41.292	ID:1	C-TX ...
00:43.285	ID:1	C-RX ...
00:41.292	ID:1	C-TX ...
00:43.285	ID:1	C-RX ...
00:43.285	ID:1	C-TX ...
00:45.285	ID:1	C-RX ...
00:45.285	ID:1	C-TX ...



```
16: 15.4 -  |06...
12: 15.4 ?  |04...
12: 15.4 ?  |07...
12: 15.4 ?  |03...
22: 15.4 ?  |03...
12: 15.4 ?  |05...
22: 15.4 ?  |05...
12: 15.4 ?  |02...
```

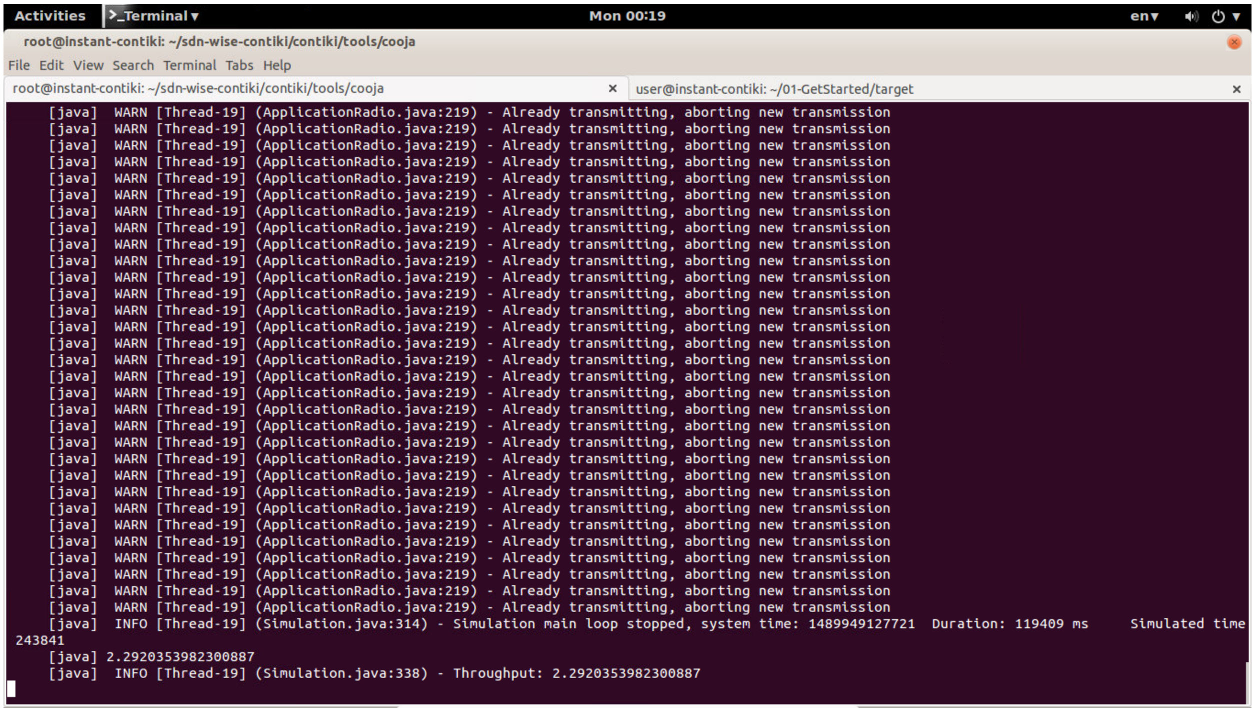
```
Time       Mote  Message
01:59.870  ID:1  C-RX ...
01:59.870  ID:1  C-TX ...
01:59.870  ID:1  C-RX ...
01:59.870  ID:1  C-TX ...
01:59.870  ID:1  C-RX ...
01:59.870  ID:1  Match...
01:59.870  ID:1  C-RX ...
01:59.870  ID:1  C-TX ...
```



```
19: 15.4 C  |07...
12: 15.4 ?  |04...
16: 15.4 -  |04...
12: 15.4 ?  |03...
12: 15.4 ?  |05...
12: 15.4 ?  |02...
22: 15.4 ?  |02...
22: 15.4 ?  |02...
```

```
Time       Mote  Message
01:05.201  ID:1  C-TX ...
01:07.189  ID:1  C-RX ...
01:07.189  ID:1  C-TX ...
01:09.175  ID:1  C-RX ...
01:09.175  ID:1  C-TX ...
01:09.306  ID:1  C-TX ...
01:09.175  ID:1  C-TX ...
01:09.306  ID:1  C-TX ...
```

Throughput can be extracted from all the above three different packets and it can easily be concluded that by using Cache Server, the throughput is better than the case with no Cache Server. It is shown in the following snapshot as well:


Activities
>_Terminal ▾
Mon 00:19
en▾
root@instant-contiki: ~/sdn-wise-contiki/contiki/tools/cooja
File Edit View Search Terminal Tabs Help
root@instant-contiki: ~/sdn-wise-contiki/contiki/tools/cooja
user@instant-contiki: ~/01-GetStarted/target

```
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  WARN [Thread-19] (ApplicationRadio.java:219) - Already transmitting, aborting new transmission
     [java]  INFO [Thread-19] (Simulation.java:314) - Simulation main loop stopped, system time: 1489949127721  Duration: 119409 ms     Simulated time
 243841
     [java] 2.2920353982300887
     [java]  INFO [Thread-19] (Simulation.java:338) - Throughput: 2.2920353982300887
```

If we compare the upper snapshot with the one we received for the topology without cache, it can be easily noticed that throughput for Cache server Topology is **“2.2920”** which is more than the pervious throughput received in the last topology which is **“1.0989”.**

vii) Pros and Cons:

Pros:

- Since most requests are made to the same websites, eg. Facebook, Google: Gmail, Youtube. etc , Yahoo, Wikipedia, etc and thus to the same IP addresses, most rules can be set for long times. Only the websites that are not often accessed need new rules. The frequency of new rules setup is rather small when the rules for the most accessed websites have been set up.
- Relocating content implies only a single operation per switch at the base stations.

- The system does not need big computation power. Since it is not fine-grained, the rules are set per website, including thousands of contents. So the number of needed operations is divided by the average number of contents per website.
- A reduced number of configuration operations is required, one per IP address on the switch of the eNodeB and some static rules set up beforehand.
- A single SDN switch is needed per eNodeB .

Cons:

- The contents of a website are not separated. Due to the redirection mechanism, the whole website contents have the same caching location. It is not possible to apply fine-grained policies to favour some Youtube videos in their location among others, for example
- Relocation can only be performed for the whole website, moving all the contents stored from the same website at once. It is not possible to relocate only the most accessed content. Thus the relocation operations take a very long time and are not efficient.
- Contents of the cache are not controlled on a fine-grained level.
- Duplication is not controlled. If duplication is required, it is performed on a website basis, all the contents are duplicated.

viii) Summary:

In order to validate the concept of Caching, we implemented two cases: one without Cache and second with Cache Server. The proposed system has an advantage of simplicity which allows the consumption of resources to a minimal level while improving the performances.

This testbed shows that this concept can optimize caching but that it needs high control level for maximizing the performance increases. Thus by future perspective, the focus should be on improvements to the system for a better control over the contents.

8) Conclusion:

Mobile administrators are presently intrigued into implementing cache solutions for conveying the substance nearer to the clients. The present innovation utilized in LTE systems permits reserving either in base stations or after the P-GW, that are both endpoints of the GTP tunnels. Caching in base stations is not performing so well as a result of the modest number of clients per base stations and their portability. In this manner the store hit rate is lower in the base station than in some other place of the system. Meanwhile, Caching after the P-GW is performing great regarding cache-hits, yet it is further far from the customers and in this manner prompts to latency for the clients and system load for the administrators. Utilizing Software-Defined Networking grants to expel the tunnelings and perform caching in backhaul systems. Consequently it offers a trade off between both circumstances.

SDN can likewise be utilized to enhance the caching systems themselves. A few master postals have been intended to achieve this objective. The present pattern in caching is to decentralize the caches and start a collaboration between the cache neighbours. SDN can partake in that objective, by giving a layer playing out the routing or sending of the solicitations to the correct caches. This can be performed in various ways, utilizing another protocol, developing the IP protocol or building an overlay on IP systems.

Future Works:

This topic opens a wide range of possible future work. The first point for future work is to extend the system, with the help of new location selections. This can also be achieved by relocating triggering algorithms.

Wider future work could include an integration with a work on efficiency in mobile networks. It can be achieved by efficiently bundling the contents, compressing the headers, or compressing the contents very selectively. Thus, it can be expanded to further improve the overall efficiency and the users' Quality of Experience (QoS).

References:

- Software Defined Mobile Networks (SDMN): Beyond LTE Network Architecture: Madhusanka liyanage, Andrei Gurtov, Mika ylianttila
- SDN optimized caching in LTE mobile networks: Jose Costa-Requena, Maël Kimmerlin, Jukka Manner, Raimo Kantola
- Rule Caching in SDN-Enabled Mobile Access Networks: Mianxiong Dong, He Li, Kaoru Ota, and Jiang Xiao
- A Software-Defined Architecture for Next-Generation Cellular Networks: Vassilios G. Vassilakis, Ioannis D. Moscholios, Bander A. Alzahrani, Michael D. Logothetis
- SDN.IEEE.org/education
- A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks: Bruno Astuto A. Nunes, Marc Mendonca, Xuan-Nam Nguyen
- Software-Defined Networking: A Comprehensive Survey: Diego Kreutz, Fernando M. V. Ramos
- LTE architecture (http://www.3gpp.org/LTE)