Manoj Kumar
Mob: +91-8970533456, 8310562535
Bangalore, India
Email: access2koolmanu@gmail.com

Objective:

To work in an organization where professionalism and enthusiasm is recognized, and work continuously towards increasing my technical skills and professional expertise.

Experience Summary: A Tech Lead Enginner with 12 years of industrial experience in core network products Development (IMS/VoLTE/EPC) domain. Worked as a Developer in High Availability Software Product, DPDK/VPP based application used in user data plane, IP based Route Management, Network security protocols, IP based traffic dispatcher and L4/L2 layer Load balancer in IMS Core Network.

- 2+ Years experience DPDK based Application development ,scaling and performance.
- Worked on Cavium architecture based application development in user plane.
- Experience and understanding of Data Path protocol Ipv4, Ipv6, GRE, GTP, VLAN,
- 6+ Years of Sound Programming knowledge on C.
- 6+ Years of telecom software development experience with Strong knowledge on C++ (c++11, c++14, STL) Development with related tools in Linux environment.
-
- 2+ Years of programming knowledge of Go Lang, and have good understanding of virtualization technology like Docker swarm, Kubernetes(K8s) and Containers.
- Good understanding of TCP/IP protocol suite, TCP/UDP/TLS/IPSec and application layer (HTTP/HTTPS).
- Working knowledge in Transport layer security (SSL/TLS) and openssl library.
- Ip layer (IPSec) security protocol.
- Experience in Multi-Threading, Secure Sockets, Memory Management Programming, Public Key Infrastructure.
- Excellent understanding of distributed computing, scalability, High availability, high performance architectures.
- Good understanding of Linux internal(Process Mgt, File Mgt, Memory Mgt, IPC)
- Possess Good debugging skills equipped with analytical and problem-solving approaches.
- Prefer to be hands-on, taking responsibility and actively participate in design, development, and review activities.
- Good understanding of virtualization technology

Professional Summary:

- **1+** years of experience in **DPDK/VPP** based network application development in EPC at **Samsung R&D (SRIB), Bangalore (March 2021 to Present) in payroll of Aricent Technology.**
- **5+** years of experience of development and maintenance in CSCF IP Dispatcher, IP Routing Management, Network security protocols, and load balancer in IMS Core Network. – at **Nokia Networks**, **Bangalore**. (Jan 2015 to March 2021**)**
- **1** year of experience in development of **Packet Cable Multimedia**– at **Sandvine Technologies, Bangalore**. (Jan 2014 – Dec 2014)

- **3** years of experience in development and maintenance in Middleware (Solaris/Linux based File system and network interface based High Availability Software) Product – **at Radisys Corporation**, **Bangalore.** (Aug 2010 to Dec 2013)

- **6** Months of experience as software trainee in C/C++/Data Structure/Networking/Linux development at **Global Edge Solution, Bangalore. (**March 2010 to Aug 2010)

Technical Skills:

Programming languages	C, C++, Go
Protocol	TCP,UDP,IPv4, IPv6, GTP,GRE,SIP, HTTPS, SSL/TLS, IPsec
Telecom Domain	3GPP IMS and L2/L3 Protocols, DPDK,/Fastpath
Scripting languages	Shell, python
Operating Systems	Linux, Solaris
Tools & Softwares	GDB, TcpDump, Valgrind, Wireshark and Perf tool
Versioning tools	GIT, CVS, IBM Rational Clear case, Source insight
Engineering Practices	Waterfall and Agile

Project #1:
Feature : Data plane micro engine.
Organization: Samsung R&D, Bangalore
Client: Jio, Airtel
Role: Design, Development, Testing, Maintenance task handling.
Team Size: 4
Description: DPDK based network packet processing application runs in S/P Gw of user plane in EPC, which process packet received from network or MME.
My Role:

- Design and develop features for application to handle PDR notification to control plane and Data Usage report handling.
- Handle customer issue/ Module test addition, Maintenance issue handling
- Customer interaction for new requirements understanding, design documents preparation.
- Technology: Network fastpath, DPDK, VPP, Cavium, Virtulization, docker, kubernet.
- Language: C

#Project #2

Feature: ***ANSSI feature to secure communication over Mr interface.***
Organization: ***Nokia Network, Bangalore.***
Client: ***Verizon***
Project Duration: ***3 months***
Role: ***Design, Development and Maintenance***
Team Size: **3**

Description:

Support TLS security mechanism on Mr' interface for traffic between FEE and MRF CSCF node.

My Role:

- Developed TLS client framework to established TLS connection over Tcp to MRF(Media resource function). Added SSL certificate and key storage framework that will store the secret key and certificate based on client security configuration. Added mutual authentication support for TLS connection. Added support for ip blacklisting.

My Role also includes:

- New requirement collections from Product Manager
- Feasibility study of new requirements
- Design for new features. Write the LDD capturing the implementation details, call flows.
- Coding and Module testing for the new features implemented, Code Review
- Identified bug during tls performance run, where Ip dispatcher queue of outgoing packet buffer reached max buffer limits because of tls renegotiation case in mid.

Technology and language: C++, openssl, libcrypto, IMS 3GPP, TLS, TCP

Project #3:

Feature:	***Scalable IP-PBX traffic handling at P-CSCF/S-CSCF***
Organization:	***Nokia Network, Bangalore.***
Client:	***Verizon, Orange France, AT&T***
Project Duration:	***7 months***
Role:	***Design, Development and Maintenance***
Team Size:	**6**

Description:

As per existing load distribution algorithm of CSCF in bare metal or VI architecture, all registration/calls from an IP-PBX get handled by same S-CSCF process. In case the IP-PBX hosts large number of user behind it, the single S-CSCF process is overloaded resulting in call failures in spite of other S-CSCF process sitting an idle and ample CPU available in the system. Similar case happens with P-CSCF too.

It is required that the S-CSCF/P-CSCF should able to successfully handle SIP traffic emanating from a large-scale IP-PBX connected to it. Multiple P-CSCF/S-CSCF processes should be optimally utilized and no single P-CSCF/S-CSCF process should report overload. End user behind IP-PBX should successfully able to place and receive call without any occasional failure.

My Role:

I have added new load distribution algorithm at Gm interface which detects high capacity IP-PBX traffic when IP Dispatcher received on specific scalable port. It supports TCP/UDP/TLS traffic from the UEs. Another load distribution algorithm is added at Mw interface. Bring up new port to support IP-PBX traffic. Handling of different SIP messages at Gm and Mw interface.

My Role also includes:

- New requirement collections from Product Manager
- Feasibility study of new requirements
- Design for new features. Write the LDD capturing the implementation details, call flows.
- Coding and Module testing for the new features implemented, Code Review

Technology and language: C++, SIP, IMS 3GPP, P/S/I CSCF, TCP.

Project #4:

Product: **High Performance Lean Traffic Dispatcher on IMS CFX based on DPDK/Fastpath.**

Requirement:

The Load balancer VM in IMS is very CPU heavy, as per current estimates it requires a minimum of 20 VCPU's to support 2 million subscribers. A heavy VM creates maintenance overheads due to the high resources that it needs and renders virtualization benefits ineffective. Customers expect the VM's to be lean (not needing more than 8 VCPU's).

Current Implementation:

In the current Load balancer architecture an interrupt signals the arrival of a packet at the network card and the Linux OS processes the interrupt, signals the Load Balancer application to read the packet. Furthermore, the LB applications like IPDP and ICM even though lean have overheads because of locking and packet copy across multiple threads.

Solution:

6Wind provides fast path networking stack which works with poll mode drivers (Intel DPDK technology) which significantly increases the speed of packet processing while decreasing the need of CPU. This technology is used to create lean and high-performance traffic dispatchers.

The application library will be loaded as part of the Dispatcher infrastructure and it will do the load distribution at L2 layer. The GM dispatcher will hash the UE IP and identify a CSCF Server to which the packet has to be forwarded.

My Role: Owner of the Following UST

For packets coming from UE: Do the hashing of packet based on source IP. Do ICM table look up to get destination MAC address. Change destination to the one returned by ICM table lookup and source MAC address to LB VMs "Internal LAN2" MAC address. Send the packet out through the "Internal LAN2" interface.

For packets coming from CSCF server VMs: Do a route lookup based on the destination IP address (which will be UE IP address). Based on the lookup, get the next-hop destination IP and its corresponding MAC address. Change the destination MAC address of the packet to that of the gateway MAC address. Change the source MAC address to the MAC address of the Gm interface. Send the packet out through the Gm interface.

Project #5:

Product:	***IP-Dispatcher in IMS***
Feature:	***Buffer Chaining Management***
Organization:	***Nokia Networks, Bengaluru.***
Project Duration:	***4 months***
Role:	***Design, Development***
Team Size:	***1***

Description:

In current scenario, If the received message length is bigger than available free size of Ip Dispatcher buffer pool, then it is start dropping the messages which resultant in call failure. And in second scenario, IP Dispatcher drops the messages if it receives in socket binding state but message will be sent out only in socket open state.

Feature Implementation:
As part of first solution provided, if the received message length is bigger than available buffer size, then it copies the part of message to the available free size. Thereafter, it retrieves another buffer from IpDp buffer pool and copy the remaining length of message to it. It allows to chain new buffer along with existing chained buffer list (Max allowed IpDP buffers in the chained list is 3).

As part of second solution, Ip Dispatcher allows to receive maximum one message in binding state and copy this message to chained IpDp buffer list if socket state transits from '**binding'** to '**connecting'**. Subsequent messages will be dropped, if socket state still in **connecting** state. Once the connection request accepted and socket state change to 'open' state then it will begin copying message to the IpDp chained buffer.

Role:
I was sole owner of this feature, involved in requirement gathering, feature analysis, implementation, review, LDD, Module testing, feature release meeting.

Technology and language: C++, STL, Memory Management, TCP, STL.

Project #6:

Product:	***HttpSv and HttpCl application Implementation on X1 interface.***
Organization:	***Nokia Networks, Bangalore.***
Client:	***AT&T, Verizon.***
Project Duration:	***1 year.***
Role:	***Design, Development and Maintenance***
Team Size:	**3**

Description:

HttpSv and HttpCl are Go language-based middleware applications, which are used for subscriber provisioning at X1 interface in IMS. It has two interfaces, one towards CSCF role-based application and other toward the network.

HttpSv server:

It provides a message interface for (P/I/S-CSCF role based) application layer and uses a TCP interface for external HTTP clients. It opens Tcp socket listener for all configured ports and gets data request from the remote external Http client, parse and validate http message and forward to the configured application. It receives response from CSCF application and send it to the external client.

- Addition, deletion and modification of existing service at runtime.
- Create secure Tcp connection to external http client based on SSL/TLS configuration.
- Support Bulk message request from the client. It sends data in chunks.
- Whitelist check for untrusted client.

Http Client:

It creates new Tcp connection (secure/non-secure based on configuration) to the external Lawful interception server on request of CSCF role based application It sends request message to external http server and get response from them and distributes to the mapped role based applications. It has support of persistent/non-persistent Tcp connection.

My Role:

- Learnt Go language and develop both applications HttpCl and HttpSv from scratch in Go.
- Involved in requirement gathering, HLD/LLD, coding and module testing.
- Created message interfaces API for applications, SSL/TLS implementation for secure connection, Chunk data support for bulk message request, message flow control handling, Runtime configurations parameter changes handling.
- Implemented alarm, counter and logger.
- Added IP black list feature for non-registered client.
- Implemented support of permanent and non-permanent connection in http client to the external server.
- Bug fix in feature testing.

Technology and language: **Go, CGO, TLS cryptographic, HTTPS/HTTP Go based APIs**

Project #7:

Feature:	**Permanent configuration of CSCF IPv6 MTU Size**
Organization:	**Nokia Siemens Networks.**
Client:	**Bell Canada.**
Duration:	**6 months**
Role:	**Design, Development and MT and feature bug fixing**.
Team Size:	**3**

Description: This Feature allows customer to have network elements which does not support IPv6 MTU size (1500 bytes) as this provides a provision to configure specific persistent MTU size for every interface/LAN/route.

There is no Path MTU Discovery support for UDP protocol from the RHEL OS to adjust the MTU size to the lower value based on the ICMP error. Although SIP level retry of the packet exists, it does not help as the RHEL OS still sends the packet with MTU size of 1500. This ultimately results into every IPv6 UDP packet with size greater than PMTU size to get dropped in the network with ICMP packet too big error.

IPv6 do not allow fragmentation of packets by routers. Packets are only allowed to be fragmented by host initiating a packet. CSCF always uses a fixed MTU of 1500 for application LANs. There is no support to customize the MTU size of Routes. But PGW does not support IPv6 packets with MTU size 1500.

Feature Implementation:

- Development of APIs to interface with cafProxy and data handling. Requirement grooming and feature analysis, workshop.
- MTU_SIZE parameter is configured in LAN table, IP Cluster Routing table, and IP Dynamic Routing table in Technical Product Descriptor (TPD).
- CafProxy process reads MTU size column value from IP_CLUSTER_ROUTING in cmrepo/DB and it writes to /opt/SMAW/INTP/generations/liveUpdate/iprouting6.conf. AdvIpRouting process read MTU column value from iprouting6.conf file and add route to the system.
- Added Interface for reading the configured MTU size from the cafproxy generated file iprouting6.conf
- Generate ipcfg-<interface-name> file with given MTU_SIZE, add API to SET and GET for setting and retrieving MTU size.

- INTPaacfr_ipConfig.sh script invoked by call action script that stored all added routes to the system that later used for persistency.

- During interface/routes/LAN or route creation, Common application framework (CAF) access the MTU size from LAN/IP Cluster Routing/Dynamic Routing table along with the other parameter present in the tables.

- If MTU size is configured in TPD, MTU value is taken from LAN table(ipv6), IP Cluster Routing (IPv6) and IP Dynamic Routing(ipv6) table.

- During Fallback, MTU value is restored to default value of 1500 bytes.

- MTU_SIZE parameter of LAN/Routes table can be add/modified through cmcli as well.

Technology and language: C++, Linux Ip routing cmd, RTLINK, Shell Script.

Project #8:

Product: **UpSuite**
Organization: **Radisys Corporation.**
Client: **T Mobile, Bell Canada.**
Duration: **1 year**
Role: **Development, maintenance, and feature bug fixing**.
Team Size: **5**

Introduction:

Upsuite is a high availability software product for Solaris/RHEL Linux that instantly provides high availability for any application or server.it detects software hardware, network failure and provides sub-second failure to a standby system.

Description:

- Sub-second failure detection and recovery.
- Run on TCP/IP LAN
- Tunable heartbeat control failover.
- Efficient application-transparent failover.
- Efficient repair and replay recovery
- Minimal use of CPU and bandwidth
- Simple to maintain active and standby architecture

Role & Responsibility:

- Provided fix for kernel panic due to corruption **in in-core** hash list.
- Fix for Kernel panic:"**mutex_enter:bad mutex**" – This KP happened during dataset repair operation and same has averted by freeing the inactive inode to the free inode pool.
- **Kernel panic after server reboot** – This was consequence of double free of already freed buffer and is triggered in the boundary case of the ipfs chunk deletion.

Featured developed as mmap() support for IP File System.

Project #9:

Product: **UpLink**
Organization: **Radisys Corporation.**

Client: **T Mobile, Bell Canada.**
Duration: **1 year**
Role: **Development, maintenance, and feature bug fixing**.
Team Size: **5**

Introduction:

Uplink is a middleware that provides high-availability Ethernet services. Uplink creates a single virtual Ethernet interface to represent two physical Ethernet interfaces. Uplink detects heartbeats over the two physical interfaces and provides traffic failover from one interface to another in case of link failure or heartbeat timeout. While Uplink is managing the physical interface uses, application use the virtual interface to achieve connectivity through a single highly available IP address.

Description:

Uplink enable following features:

- Provide a single virtual Ethernet interface with single MAC address that failover between two phy interface
- Implemented a multiplexing STREAM driver work over existing Ethernet interface.
- Supported high level protocol, including TCP/IP,UDP,SCTP or DLPI request to transmit raw Ethernet frame.

Tunable failover in response to physical link

Role & Contribution:

- Bug fixing and feature development
- Kernel panic in uplink during reboot operation
- Uplink entry in arp table with **"d"** status
- Fixed **Duplicate MAC address** after reboot
- Support for **e1000g** interface
- Implement **DAD** (Duplicate address detection)

Educational Qualification:

- B.Tech in Information Technology, School of Engineering(CUSAT), Kochi, Kerala, in June 2009, with an Aggregate of 71.7(1st to 8th Sem) %.

Personal Details:

Name	Manoj Kumar
Sex	Male
Marital Status	Married
Nationality	Indian