Dhanraj Vedanth Raghunathan

919-345-4552 | draghun@ncsu.edu | linkedin.com/dhanrajraghunathan | github.com/dhanraj-vedanth | draghun.com

Education

North Carolina State University — Raleigh, NC
Master of Science in Computer Networking (3.77/4) — *August 2019 – May 2021*

Anna University — Chennai, India
Bachelor of Engineering in Electronics and Communication (8.56/10) (Distinction) — *August 2014 – May 2018*

Skills

Languages: Python, C, Shell Scripting, Ruby
Networking: HTTP, DNS, DHCP, TCP, UDP STP, IP, BGP, OSPF, ICMP, ARP, LAN, VLAN, SDN
Tools: Git, Mercurial, Travis CI, Rails, Ansible, Wireshark, IPtables, Docker, KVM, Libvirt, Sensu, Grafana, Thrift

Experience

Facebook Inc. — Fremont, CA, USA
Network Automation Engineer Intern — *May 2020 – August 2020*

- Reduced time taken to handle a network break-fix event by over **1 hr/ticket** by automating **80%** of the network event steps using a workflow
- Developed 3 core modules for an asynchronous service to spin workflows resulting in a loosely coupled service with **30%** reduced DB hits

Tata Communications — Chennai, TN, India
Network Automation Engineer — *June 2018 – June 2019*

- Led the end-end implementation of the Looking Glass project over major PoPs in India resulting in improved polling response time of the servers by **85.3%** by using concurrent SSH, Python, and Ansible for orchestration
- Used Versa's APIs with Python to automate **60%** of the on-boarding steps resulting in **40%** increase in on-boarding speed
- Developed two core Python modules for the Ansible stack : Parse and Compute
- Built a generic traffic analyzer network devices using Python for real-time fault detection. Introduced checks for server and script failure by deploying Sensu
- Optimized memory and CPU usage by **44%** in the Linux cluster by virtualizing servers using KVM and ESXi

Projects

IaaS using Linux server Virtualization | *Networking, Virtualization*

- Built a multi-tenant virtual private cloud solution (on-demand VMs, subnets, and containers) with multiple KVM/QEMU hypervisors, docker engine, L2 and NAT/DHCP Linux, and OVS bridges
- Used network namespaces, IPtables, and GRE/VXLAN tunnels between multiple hypervisors for isolated multi-cloud/hybrid cloud setup

Automation and Orchestration | *Networking, Virtualization, Automation*

- Designed automation of a VPC infrastructure using KVM/QEMU, Libvirt, Docker, and Ansible. Developed the cloud infrastructure and network orchestrator using Python, Shell scripts, and JSON data model
- Followed model-driven architecture with clearly defined northbound, logic layer, and southbound

VPC-CDN and Edge Computing | *Networking, Virtualization, Automation*

- Designed and deployed a per-tenant file-sharing Edge Computing based CDN service. Edge nodes are present in different locations on demand and use a custom file-based load balancing scheme.
- Used InfluxDB and Grafana to visualize and alert. Implemented a custom self-healing feature for containers (Kubernetes inspired)

Point to Multi-point system with ARQ | *Socket Programming*

- Implemented a data transfer system using UDP and developed a reliability scheme on the end-systems

Peer to Peer system with a Distributed Index | *Socket Programming*

- Designed an application-level protocol to accomplish an asynchronous, multi-threaded request-response P2P communication system (TCP) with a centralized registration server

Athena | *Web Development*

- Designed an E-Learning web application using Rails. Admin has control over the application and can view history. Students can purchase teacher specific courses. Teachers can choose to teach certain courses. Students can view purchase history and teachers can view reviews

Spotify and Youtube Automation | *Automation, APIs*

- Built a CLI service to add songs from a selected Youtube playlist to Spotify liked songs list using Youtube Data API and Spotify Web API

Xinu | *Operating Systems*

- Implemented a robust 'exponential-distribution' scheduler and 'Linux-Like' scheduler to overcome starvation. Accomplished memory virtualization through demand paging using swapping algorithms: Second Chance and Aging. Designed a reader/writer lock and overcame the priority inversion problem through priority-inheritance.