NFVRG Interim Meeting notes

Interim Meeting Theme: ***Re-thinking VNF Architectures towards a Cloud-native Deployment***

Interim Meeting Time: ***10:00am to 12:00pm Sept. 18th Monday, 2017***

Location: ***VMWare Campus, Bay Area and Remote participation***

Attendees:

1. Aurojit Panda
2. Eric Keller
3. Ramki Krishnan
4. Sarah Banks
5. Ulas Kozat
6. Aaron Smith
7. Akbar Rahman
8. Alberto Rodriguez-Natal
9. Andrea
10. Anna Brunstrom
11. Athanasios Kyparlis
12. Andrew Veitch
13. Ban
14. Bhumip K
15. Cinzia Satori
16. Daniel Nernier
17. Diego Lopez
18. Edel Curley
19. Gil Hellmann
20. Helen Chen
21. Henry Fowler
22. Hesham ElBakoury
23. Herb Patten
24. Ibirisol
25. Isaku Yamahata
26. Jack Kozik
27. Jason Hunt
28. John Kaippallimalil
29. Jorge Zarapinha
30. Jin Fujimiay
31. Julien Maisonneuve
32. Linda Dunbar
33. Madhu Nunna
34. Michal Roszkiewicz

35. Michelle Han
36. Mike Richer
37. Misbah Mahmoodi
38. Nikhil
39. Padma Sudarsan
40. Pasi Vaananen
41. Peter VInsel
42. Rao
43. Raja Krishnasamy
44. Reinhard Schrage
45. Roger Lou
46. Roger Maitland
47. Ryan Hallahan
48. Salagame
49. Sanjay Aiyargi
50. Sastry Isukapallii
51. Satish
52. Scott Blanford
53. Scott Seabolt
54. Shardon Chisolm
55. Steven Wright
56. Tibi
57. TimV
58. Tom Tofigh
59. Vanessa Little
60. Varun
61. Vishakha Gupta
62. Wolfgang Hahn
63. Yapeng Wu

Meeting Kick off
IPR Policy
Administrivia
Agenda

- Meeting is being recorded
- Minutes and slides will be uploaded
- Theme: Rethinking VNF architectures towards a cloud-native deployment
- 2 presentations: Aurojit Panda, Eric Keller are the presenters
- Brainstorming discussion for the last 40 minutes

A New approach to network functions
Presenter: Aurojit Panda

Abstract

Modern networks do far more than just deliver packets, and provide network functions -- including firewalls, caches, and WAN optimizers — that are crucial for scaling networks, ensuring security and enabling new applications. Network functions were traditionally implemented using dedicated hardware middleboxes, but in recent years they are increasingly being deployed as VMs on commodity servers. While many herald this move towards network function virtualization (NFV) as a great step forward, I argue that accepted virtualization techniques are ill-suited to network functions. In this talk I describe NetBricks — a new approach to building and running virtualized network functions that speeds development and increases performance.

Q from Linda: Is NetBricks a Hypervisor?
A: No, I'll get to that answer in a few slides

Q from Linda Dunbar: What is the "isolation" in your context?
A: I'll get to that in the next slide

Q from Ulus: What packet size is used for comparison?
A: 64 bytes

Q from Constantine: You're really talking about memory isolation, you're using some sort of memory mapping technique
A: The NFs in my case don't run in a VM, there is no memory mapping technique

Q from Ramki: There are no containers?
A: No Containers

Q from Mustafa: Is there any relationship between isolation that framework provide and intel SGX. and do you have slide how exactly did get that number

A: No. There is ongoing work to port this in to provide security, but intel SGX is trying to solve a different problem.

Q from Ramki: How did you get the numbers?
A: We used a 40G NIC

Q from Roger: How does your performance compare with Open vSwitch with DPDK: https://software.intel.com/en-us/articles/open-vswitch-with-dpdk-overview ?

A: We always run OVS DPDK

Q from Mike: What are the security implications if memory isolation is mostly at compile time and potentially exploitable

A: You need to check for compiler safety, and you need to check that the compiler was correct. How do you know that the process is the process? You rely on the certificate that's attached to it. We would need roughly the same thing.

Q from Ramki: If you were to add a firewall policy, would you have to recompile the code?
A: Policy? No. It shows up in tables. Policy changes, no. Changing the pipeline at the moment? You have to recompile. You can get rid of the recompile, but we haven't done the work yet.

Q from Ramki: How do you get to that specific area of memory for programming tables such as firewall?
A: There's a daemon that listens on the control port, listens, and goes in and makes the changes appropriately

Q from Mustafa: Does it work in a multi thread environment?
A: yes!

Q from Constantine: You compiled and run on the same interface?
A: Dynamic linking is possible

Q from Constantine: Who does that?
A: you assume the kernel is trusted. We are changing the isolation model. This is equivalent to the exact level of isolation that's provided by processors. Maybe you have to place trust on compilers rather than VMs.

Q from Sarah: So how are you not just kicking the can down the road?
A: I am kicking the can down the road. Everyone is.

Q from Jambi: The results you showed us earlier, that did not use a full core?
A: The one at 25m pps used a full core

Q from Jambi: Is this a proven point that you can share a core?
A: Yes, I'll show you some results to prove that.

Q from Constantine: Are you restricting yourself by serializing packet processing?
A: It's something we haven't explored yet; the challenge is how do you prove that "does simultaneous packet processing not impact performance". We looked at parallel processing but the transactional costs of accounting for changes didn't overcome the performance achieved by serialization

Q from Ibirisol: If you remove NF from virtualization environment, how do you are dealing with distributed environments? Examples, HA function in different instances of NetBricks?

A: Some of this shows up later in the slides; we don't think virtualization is tied to HA, HA was done well before Virtualization was possible

Q from Constantine: Can you comment on the numbers for VMs and containers – you assume every NF runs on a single VM or container, with 1 core.

A: There's no startup time, they're constantly running and polling, in the VM case you're going core to core, so there's the cost of serialization, and there are some queue drops when transitioning from VM to VM, because not all VMs keep up with the others.

Q from Constantine: that doesn't explain fully the difference
A: VM entry and VM exit; vhost adds a cost (vhost for containers just came out with this release of DPDK, isn't fully functional yet)

Q from Mustafa: Do I understand rust has built in memory allocator so what I don't get does NetBricks translate memory allocation request by NF to rust and rust allocate memory from heap and maintain separation?

A: Rust does not have a memory allocator. Most of the allocations here are on stack, or in the cases where you need stable state

Q from Constantine: you implemented all the LB techniques that Maglev supports?
A: yes

Q from Constantine: They have other techniques that aren't published
A: I only implemented what they published

Q from Constantine: How did you obtain the EPC code?
A: We are funded by Intel, Intel ran our code in their lab

Q from Ramki: Is the EPC above using other optimizations, like SR-IOV?
A: the ones not done by Intel are using SR-IOV

Q from Jambi: This commercial EPC gets 5x performance if compiled on NetBricks?
A: No, we implemented the functionality, we ran it through NetBricks, we received the performance noted here

Q from Jambi: We're comparing bare metal to bare metal here?
A: yes

Q from Ramki: Have you tried this on any specific smart Nics?
A: I have no access to Smart NICs. I've only had access to Intel NICs, which are decidedly not smart – they're not meant for programmability

---

Building a better network through disaggregation
Presenter: Eric Keller

Abstract
To improve performance, security, and reliability, network practitioners have, over time, moved away from the principle of a stateless network and added stateful processing to the network with devices such as firewalls, load

balancers, and intrusion detection systems. In doing so, networks have become increasingly complex and brittle, because the state held in these devices (such as the connection tracking information in a firewall) is needed to process the traffic. The conventional approach forces practitioners to configure or architect the network to get the right traffic to the right (physical or virtual) appliance (i.e., where the relevant state is), and introduce costly, and sometimes ineffective, mechanisms to back up state (e.g., to recover from failures). In a world where agility is increasingly important, a new approach is needed.

In this talk, we present our a network architecture based on disaggregated network functions. Our foundational work breaks the underlying assumption that state needs to be tightly coupled to a specific device, the state is maintained separately and the network functions can access that state from anywhere and at any time through a well-defined interface – creating a highly flexible network. After years of research, we proved this architecture viable (publishing the results at NSDI), and now we are commercializing at Stateless, Inc. In this talk we will present the background and technical details of this disaggregated architecture, discuss the challenges we are currently working on, and the use cases driving the commercial adoption.

Q from Ulas: Why did you switch from Infiniband?
A: Various reasons: When moving to Redis, we didn't want to be too dependent on Infiniband and wanted to leverage DPDK and standard Ethernet.

Q from Ramki: packet processing needs to scale out, are you using a stateless approach to direct to those entities?
A: (back to Optimized Data Store Client Interface slide) – processing is happening in Thread 2.

Q from Ramki: If 1 server runs out of capacity and you need more capacity, how do you handle this?
A: We use the SDN switch, and we count on the switch to handle this; stateless hash based

Q from Jambi: What is baseline FW?
A: With all the info onboard, rather than remotely read from database

Q from Sarah: Did you consider using Imix?
A: We replaced a trace with different packet sizes, don't know what the breakouts were

Q from Raja: Are we assuming in memory data stores and what kind of latency?
A: Yes, in memory, and the latencies are really low. RAM Cloud has on the order of 5 micro second read latencies across an infiniband network

Q from Ramki: DB access; is it batch accessed or per packet?
A: per packet for reads and writes, but our Datastore interface will batch them up to optimize the performance

Q from Ramki: You provide that layer?
A: yes

Q from Ramki: how do you fit into the orchestration system?

A: I'll come back to the platform. They don't necessarily need it integrated into an orchestration platform that they already have. We get resources in a DC, we put up a rack, our controller can handle our set of resources. As long they can send traffic to us, it works

Q from Sarah: And your customers are OK with no integration?
A: They don't really have the controllers today; the SPs are interested in OpenStack

Q from Constantine: Would you view OpenStack as an SDN controller?
A: No. Ideally what we need is bare metal, but we can work in a virtualized environment as well

Q from Raja: Have we separated out data store for the control plane, and for the data plane?
A: All of the focus of this talk is on the dataplane; our control plane design is separate