



The Computing-Aware Networking Problem



Adrian Farrel : Old Dog Consulting

<adrian@olddog.co.uk>

India Internet Engineering Society (IIESoc)

RFCs We Love : February 24th, 2023

Agenda

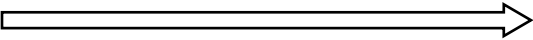
- “Compute” is a misleading term
- Computing isn’t really in the network – where is the edge?
- What problem are we trying to solve?
- A strawman architecture
- Centralised or distributed control
- Doesn’t this look a bit like Service Function Chaining?
- Some RFCs we love
- Potential protocol solutions
- What is the IETF doing about it?

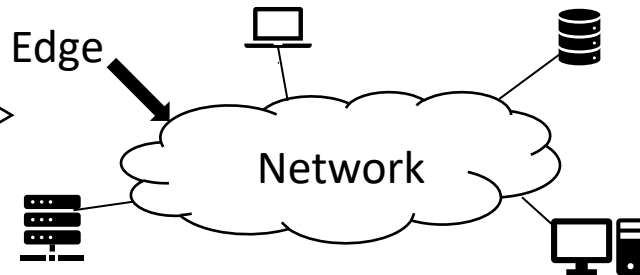
What do we mean by “Compute”

- I spent a lot of time being confused
- We are not talking about performing a specific algorithm on a set of data
- The requirement is to be able to run a program on a server
 - Clearly requires:
 - The program to be present
 - CPU resources to be available
 - May require other features:
 - Storage
 - Access to content or other data
 - Specialist processors
- This is just data centre processing or “edge computing”
 - The processing may be done on super-computers, servers in data centres, virtual machines, or any resource that has the capabilities



Compute In the Network?

- Where is the network edge?
- Traditional thought... 
- Reality...
 - It's not about physical location
 - It's not about ownership of resources
 - The network provides connectivity from client to server
 - The server may be:
 - At an edge site
 - In a data centre
 - Somewhere in the “middle” of the network
- What we care about is how the client uses the network to access the service



So, What is this All About?

- Surely we have been doing data centres, edge compute, and cloud computing for years?
- What's changed?
 - Services are hosted in multiple places in the network
 - Multiple instances of a service may be on the same server
 - Servers have different capabilities
 - Server resources may be overloaded
 - Network connections to different servers have different properties
 - Network connections may be overloaded
- New applications are performance sensitive
 - Need rapid responses
(without service queuing or network latency)
 - May need sophisticated services



What Applications?

- Anything you like to dream of
- Applications that have been mentioned...
 - Real-time image capture and processing
 - Multi-player game servers
 - Networked AR/VR
 - Holographic presence conferencing
 - Interconnected and event-aware “smart cars”
 - Digital twin
- All have different network and service requirements



What was the problem, again?

- Select a remote location for execution of a service
 - Must be able to run the service
 - Must have available resources to run the service
 - Must not be overloaded (able to meet service delivery requirements)
 - Must be reachable over the network
 - Network connectivity must be good enough (able to meet service delivery requirements)
- Not just “first, closest instance”



What Information Do We Need?

- We need to know:
 - Service locations
 - Service capabilities
 - Service location loading (metrics)
 - Network topology (reachability and capabilities)
 - Network state (metrics)
- We also need to know:
 - Service demand requirements



Can't we just use DNS or ALTO?

- DNS is good for service discovery
 - Usually relatively static
 - Doesn't offer a client much choice
- ALTO provides hosts/applications with information to choose service locations
- Neither addresses the problem at hand:
 - We want to allow the client to request a service and have the network choose the best service location and network path
- Other suggestions include:
 - Load balancing: distributes load according to an algorithm, but not aware of individual service requests or requirements
 - Message broking: provides a centralised dispatcher and can use all the metrics, but has problems of centralisation and particularly heavy traffic

Some Terminology

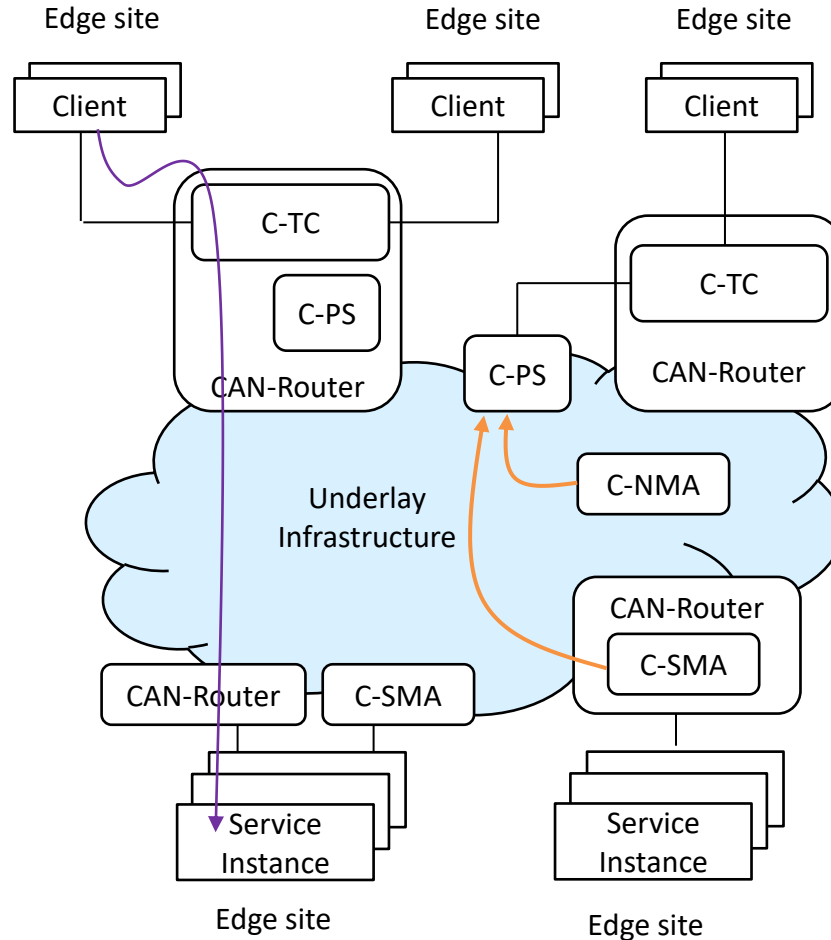
- **CAN:** Computing-Aware Networking takes into account the dynamic nature of computing resource metrics and network state metrics to steer service traffic to a service instance.
- **Service:** A monolithic function that is provided by a network operator according to a specification. A composite service can be built by orchestrating monolithic services.
- **Service instance:** A run-time environment (e.g., a server or a process on a server) that makes the functionality of a service available. One service can have multiple instances running at the same or different network locations.
- **CS-ID:** The CAN Service ID is an identifier representing a service, which the clients use to access said service. Such an identifier identifies all of the instances of the same service, no matter on where they are actually running. The CS-ID is independent of which service instance serves the service demand. Usually multiple instances provide a (logically) single service, and service demands are dispatched to the different instance by choosing one instance among all available instances.
- **CB-ID:** The CAN Binding ID is an identifier of a single service instance of a given CS-ID. Different service instances provide the same service identified through a single CS-ID, but with different CAN Binding IDs.
- **Service demand:** The demand for a specific service identified by a specific CS-ID.
- **Service request:** The request for a specific service instance.

Functional Components

- **CAN-Router:** A network device (usually at the edge of the network) that makes forwarding decisions based on CAN information to steer traffic belonging to the same service demand to the same chosen service instance.
- **Ingress CAN-Router:** A network edge router that serves as a service access point for CAN clients. It steers the service packets onto an overlay path to an Egress CAN-Router linked to the most suitable edge site to access a service instance.
- **Egress CAN-Router:** An Egress CAN-Router is the egress endpoint of an overlay path.
- **C-SMA:** The CAN Service Metric Agent responsible for collecting service capabilities and status, and for reporting them to the C-PS.
- **C-NMA:** The CAN Network Metric Agent responsible for collecting network capabilities and status, and for reporting them to the C-PS.
- **C-PS:** The CAN Path Selector determines the path toward the appropriate service location and service instances to meet a service demand given the service status and network status information.
- **C-TC:** The CAN Traffic Classifier is responsible for determining which packets belong to a traffic flow for a particular service demand, and for steering them on the path to the service instance as determined by the C-PS.

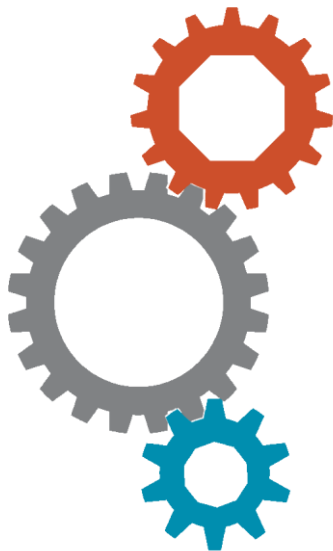
A Functional Architecture

Personal Opinion



Centralised or Distributed Control?

- No different to any other networking problem
 - A distributed control plane
 - Reacts well to small changes
 - Is strong in the face of network failures
 - Is a complex system requiring management
 - Doesn't coordinate different client requests
 - A centralised control plane
 - Provides better coordination between service requests
 - Simplifies the network control system
 - Has a single point of failure
 - Is less reactive to changes
- A hybrid solution may be the answer



Isn't this Service Function Chaining?

- Yes and no
- Yes
 - Discovering service instances
 - Selecting them
 - Steering traffic to them using overlay tunnels
- No
 - (For now) Only one service function is invoked
 - Destination for traffic is the service instance
 - That is, traffic does not take in multiple service functions on the path to a final destination
- But functional requirements are very notably similar



Some RFCs We Love

- Technologies not for this purpose
 - RFC 1035 : DNS
 - RFC 7285 : ALTO Protocol
- Some related / useful work
 - RFC 4655 : PCE
 - Maybe the C-PS is similar and uses objective functions
 - RFC 7752 : BGP-LS
 - Maybe how the C-NMA communicates
 - RFC 5305 : TE extensions for IS-IS
 - RFC 7810 : IS-IS extensions for network performance metrics
- A potential basis for a control plane solution
 - RFC 9015 : A BGP control plane for SFC



BGP for SFC?



- Wait! What?
 - Didn't you say CAN is not the same as SFC?
- Yes, but look at the similarities
 - CAN requires the distribution of information about edge sites
 - Looks a lot like L3VPN
 - CAN requires the identification of service function instances and their locations
 - Looks a lot like SFC
 - CAN requires the selection of overlay paths
 - Looks a lot like SFC
 - CAN requires integration between network state and edge point for routing
 - So the use of a routing system is not crazy
- Future-proofing
 - Today, CAN is a single end-point function
 - Tomorrow there may be additional on-path functions needed (such as authentication)
 - Today, CAN is looking at single-domain deployments
 - It seems inevitable that multi-domain will be in scope quite soon

Reminder : How does RFC 9015 work?

- Data plane agnostic (NSH, MPLS, SR, ...)
- Service Function Discovery
 - Service Function Forwarder (SFF) announces locally attached Service Function Instances (SFIs)
 - Equivalent to a C-SMA announcing local service functions
 - Route Target
 - Identifies the overlay network. Other nodes only import when the RT matched
 - Route Distinguisher (SFIR-RD)
 - Identifies this SFI advertisement
 - SF Type (SFT)
 - We will still need to know the service function types
 - Controllers use advertisements to build Service Function Chains (SFCs)
 - Spec already allows controllers to know about available resources and locations
- Service Function Path (SFP) Advertisement
 - Allows SFFs to know the next Service Function (SF) on a chain
 - SFF uses local routing processes to reach next SFF
 - Life is easy when there is only one SF : this part isn't needed
- Traffic classification
 - The Classifier is a standard part of SFC
 - Determines which packets are sent to which SFC (i.e., service function)
 - Programmed by Controller when a new SFC is installed
 - I.e., when (or before) a new service demand is made



What's The IETF Doing?



- BoF at IETF-113 online and in Vienna (March 2022)
- Presented at RTGWG and TSVWG at IETF-114 in Philadelphia (July 2022)
- Mailing list for CAN
 - <https://www.ietf.org/mailman/listinfo/can>
 - Created October 2022
- BoF at IETF-115 in London (November 2022)
 - Clear interest in the work, but very unfocussed meeting
- Draft charter for a CAN working group
 - <https://datatracker.ietf.org/doc/charter-ietf-can/>
 - Currently out for review by the IETF community
- Planning a meeting at IETF-116 in Yokohama (March 2023)
 - Probably a WG (maybe a BoF)
 - Kick-start discussions on architecture and metrics
- Various Internet-Drafts exist, but lacking tight focus
 - <https://datatracker.ietf.org/doc/search?&name=computing-aware&sort=&activedrafts=on&olddrafts=on>



What Would a CAN Working Group Do?

From the draft charter...

- Analyse the problem
- Produce an architecture aiming to use existing tools
- Do “groundwork”
 - Problem statement
 - Use cases
 - **Metrics**
- Applicability of existing tools
 - Control plane
 - Management plane (FCAPS)
 - Data plane and potential encapsulations



Questions and Follow-up
adrian@olddog.co.uk

