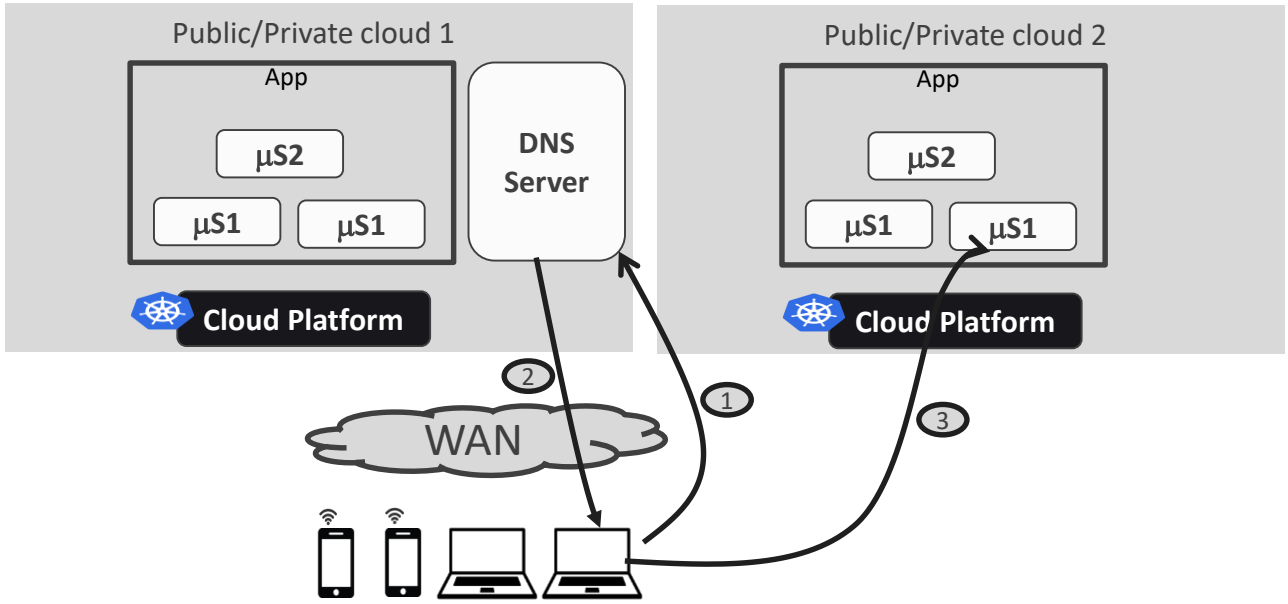
Client (UE) to Edge Service coupling Proposal

(Role of OpenNESS)

Contact: Srinivasa Addepalli (Srinivasa.r.addepalli@intel.com)

Client to Service Discovery (Via DNS) in Cloud world

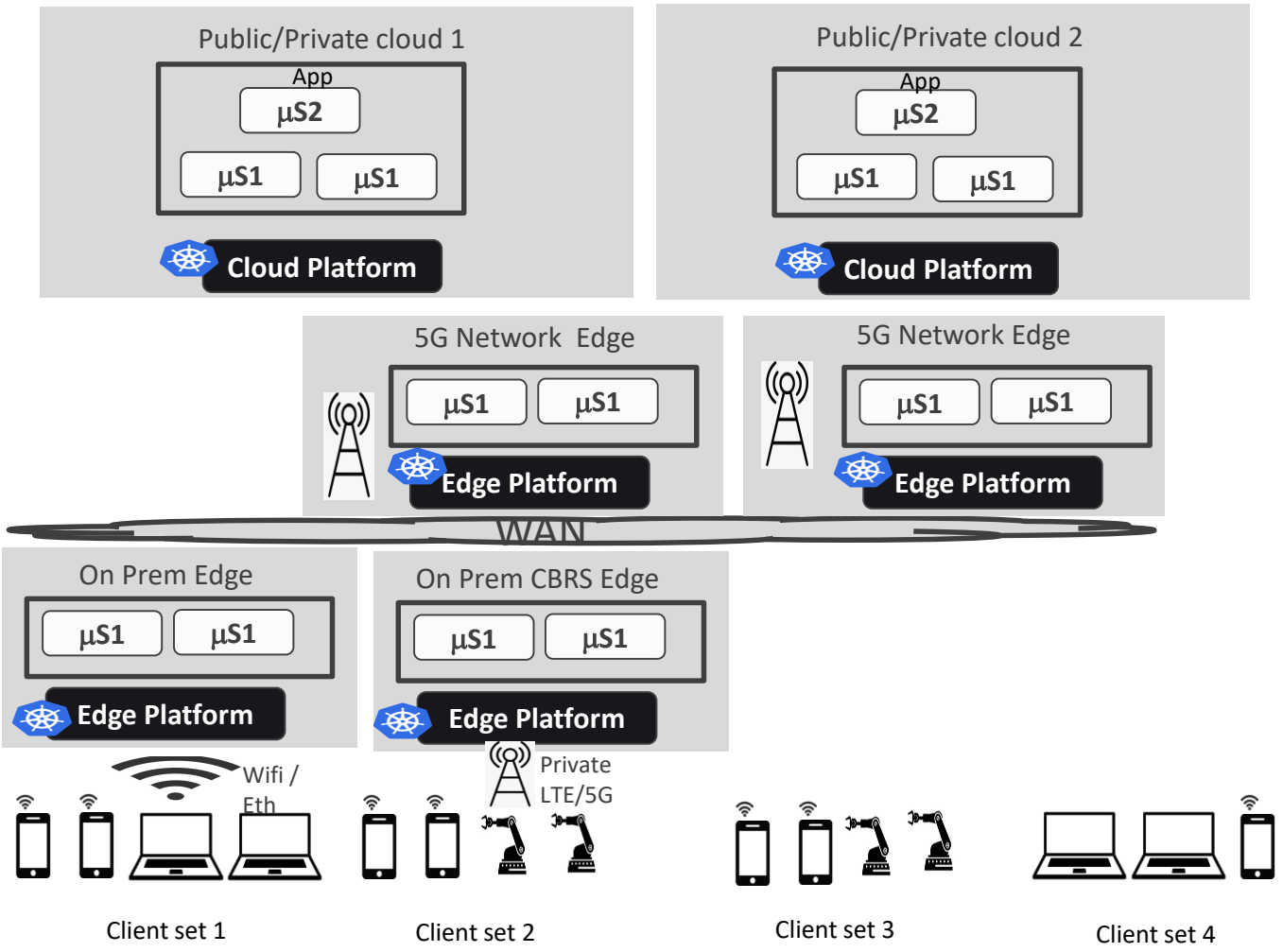


Scenario:
App consisting of two micro-services (MS1 and MS2)
MS1 is user facing service via FQDN s1.example.com
Authoritative DNS Server (e.g. AWS Route 53) is updated with two virtual IP addresses of MS1 service (one at Cloud 1 and another at Cloud2).

Client to Service Matching

1. Client sends DNS request (via local DNS Servers) to authoritative DNS server for s1.example.com
2. DNS Server responds with one of virtual IP addresses of Micro-service 1. DNS Server chooses IP address based on various criteria - Nearest service instance based on IP address of the client, Round Robin etc..
3. Client connects to the IP address returned by DNS Server. Cloud platform (Server Load Balancer) would in turn load balance the incoming connections to one of the instances of the service in that cloud location.

Client to Service Discovery (Via DNS) With Edges - Need



Scenario:

MS1 (user facing service) is deployed at many Edges in addition to cloud locations. Some on On-Prem Edges (such as Enterprise Edges), Some on 5G network edges.

MS1 instances in Edges may be ephemeral and brought up on dynamically.

Need (for performance and latency reasons)

- Client set1 is expected to connect to MS1 of the "On Prem Edge" if MS1 is deployed there. Else connect to MS1 of the public/private cloud.
- Client set2 is expected to connect to MS1 of the "On Prem CBRS Edge" if MS1 is deployed there.
- Client set 3 is going through 5G Edges. Since MS1 is deployed there, Client set3 is expected to give preference to MS1 of those 5G edges.
- Client set 4 is going through 5G where there is no edge computing and hence would be connecting to MS1 of the public/private clouds.

Challenge:

How do one associate clients to the right micro-service instances?

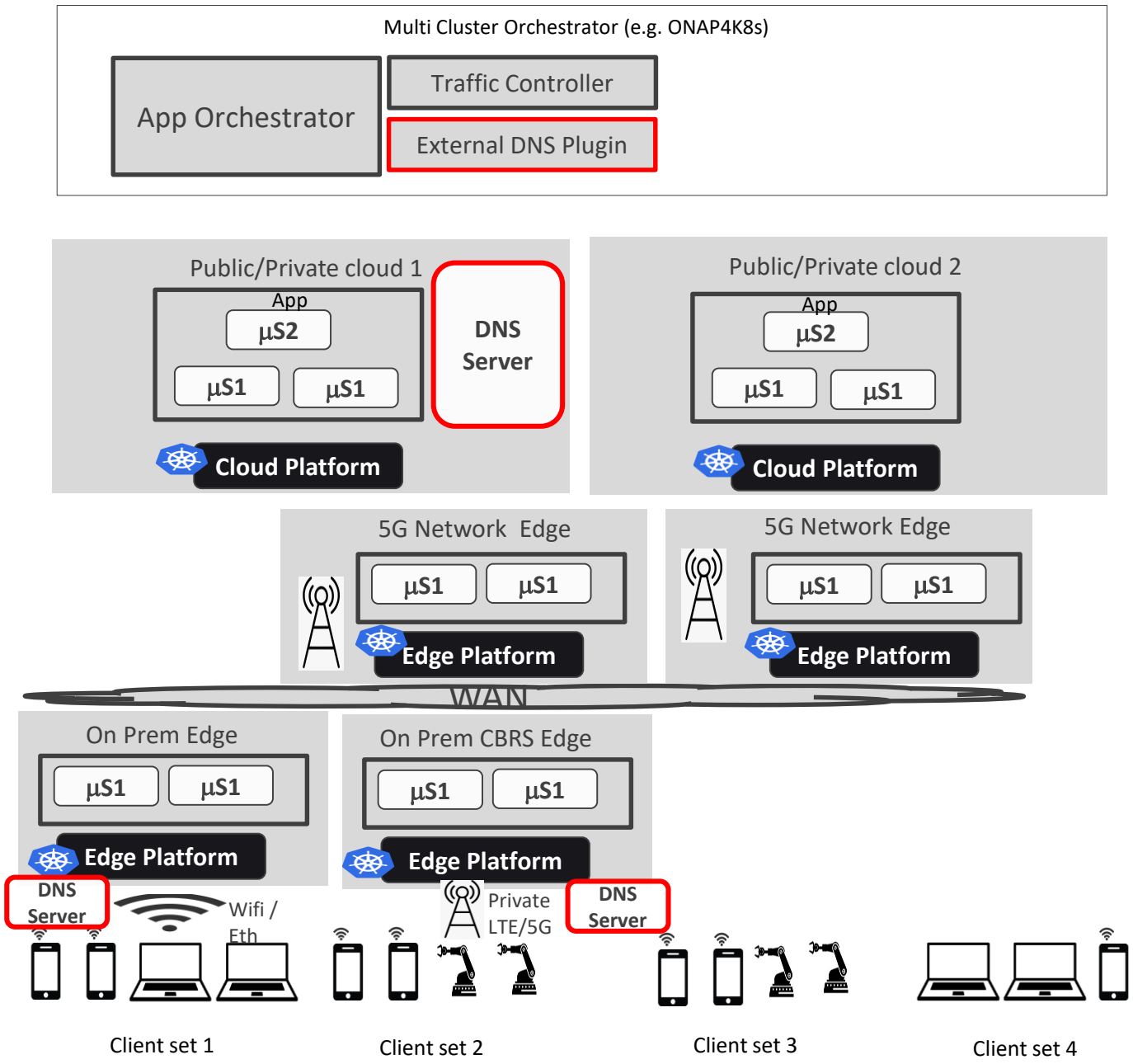
Client to Service Discovery with Edge-Computing

Methods

1. DNS based discovery (Extending traditional model to include services that get replicated at the edge locations)
2. Traffic interception (Intercept traffic that is going to cloud service and redirect to local service at the edge)
3. Client selection (Let the client choose the service instance it connects to)

Akraino ICN Proposal of Client to Service coupling: Support Method 1 (with the help of ONAP4K8s) and Method 2 (With the help of OpenNESS).
BTW, Method 3 is supported by few. I know of one - MobileEdgeX

Client to Service Discovery – DNS based Discovery (Method1)



Solution via Multi-Cluster orchestrators (ONAP4K8s is used by ICN)

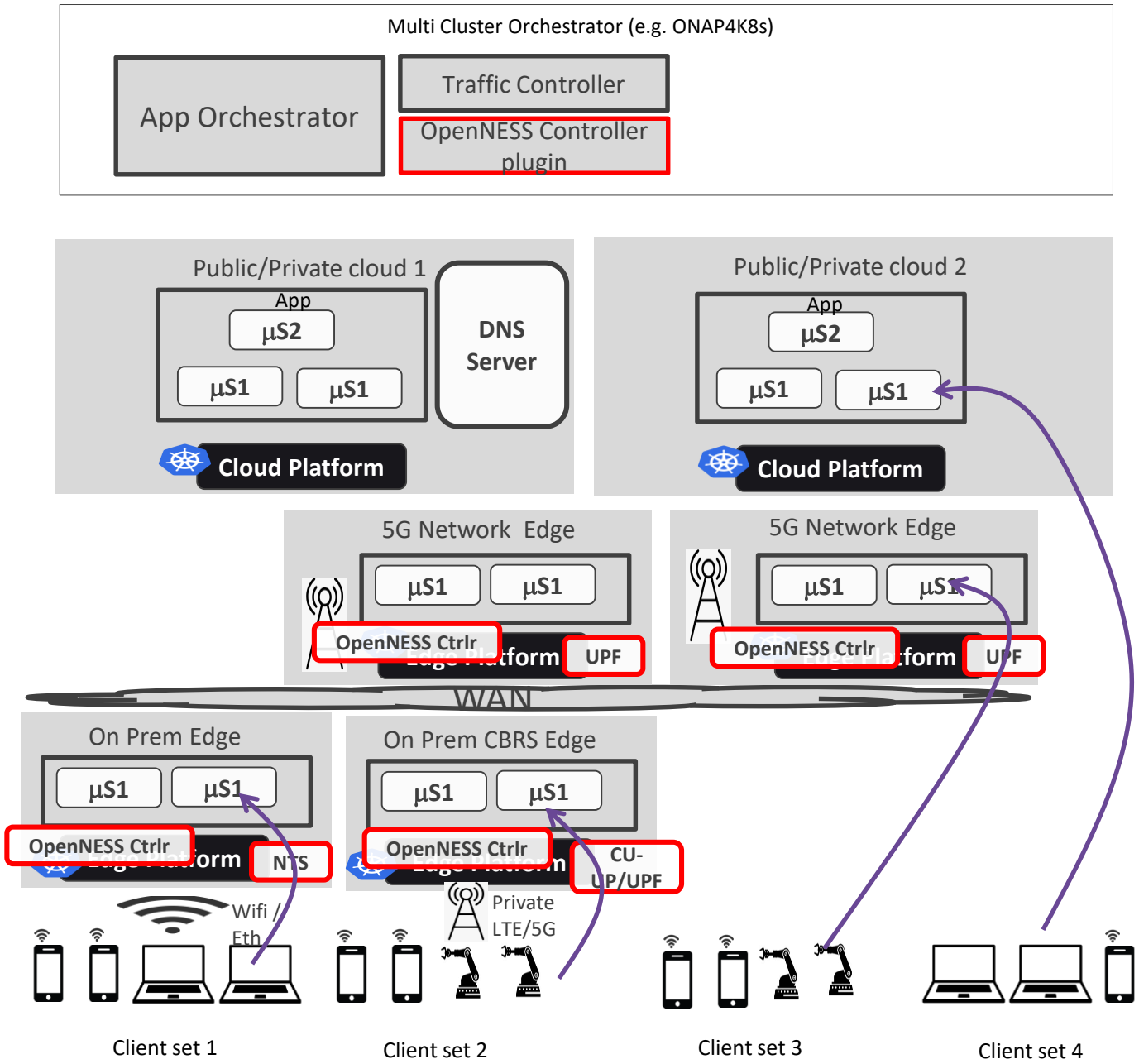
DNS Server Update:

- All publicly accessible MS1 instance IP addresses are updated in global DNS Server in public/private clouds by the traffic orchestrator once app orchestrator bring up the workloads in various places (based on deployment intent)
- On-Prem Micro-service IP addresses (which are privately reachable locally) are updated in local DNS servers in each On-prem location by traffic orchestrator.
- Multi-Cluster orchestrator keeps monitoring new relevant edges coming up and takes care of deploying MS1 as well updating appropriate DNS Servers.
- As and when Edge is decommissioned or service is brought down, Multi-Cluster orchestrator will take care of updating the DNS servers by removing IP addresses.

Pros:
- Traditional and hence proven approach.
- Simple to understand and troubleshoot.

Cons:
- Depends on DNS server to decide on nearest instance. May not be accurate in the world of NAT and VPNs.

Client to Service Discovery – Traffic Interception



Traffic interception method

As and when Multi Cluster orchestrator bring up services, it is expected to do following:

- Update DNS Server with IP addresses of Micro-services that are brought up in public/private clouds.
- When Micro-service is brought up in Edges (that support OpenNESS), configure NTS, UPF, CU-UP traffic redirection (steering) rules via OpenNESS controller plugin, which in turn talks to OpenNESS controller which programs rules in UPF/UP/NTS.

At later time, thought the clients (in set1, set2 and set3) would make DNS request (as clients do) and get micro-service IP address of the clouds, but when they make connection to those resolved IPs, UPF/NTS intercept the traffic (as they are inline) and redirects the packet to local services.

Pros:

- No changes to clients.
- More accurate selection of service instances.

Cons:

- Works for 5G. Good network deployment are needed to ensure that the traffic from Wifi and Ethernet goes through the Edge-computing site for redirection to happen.

Thank you

AKRAINO EDGE STACK