Chair of Connected Mobility
School of Computation, Information and Technology
Technical University of Munich

SCALE: Service-Centric Adaptive Load Balancing in Edge Time-Varying Networks

Marcin Bosk[1], Paulo Mendes[2], and Jörg Ott[1]

Motivation ①

Low-Earth Orbit (LEO) constellations promise low latency communication with global coverage

Variable LEO network topology is challenging for provision of internet services with strict QoS requirements:

- ⚠ Service Placement and Discovery ⚠
- ⚠ Service Execution ⚠

🚀 **SCALE** – A service-centric ROSA-based load-balancing mechanism for time-varying networks, aiming to enhance user experience

- Service Edges at the borders of the LEO network
- In-path service name resolution integrated with a load-balancing mechanism
- Consider both service instance load and network path conditions

What is ROSA? ②

ROSA = Routing on Service Addresses [1]

- ★ Novel service-centric networking framework
- ★ Services instead of communication hosts determine the addressing semantics

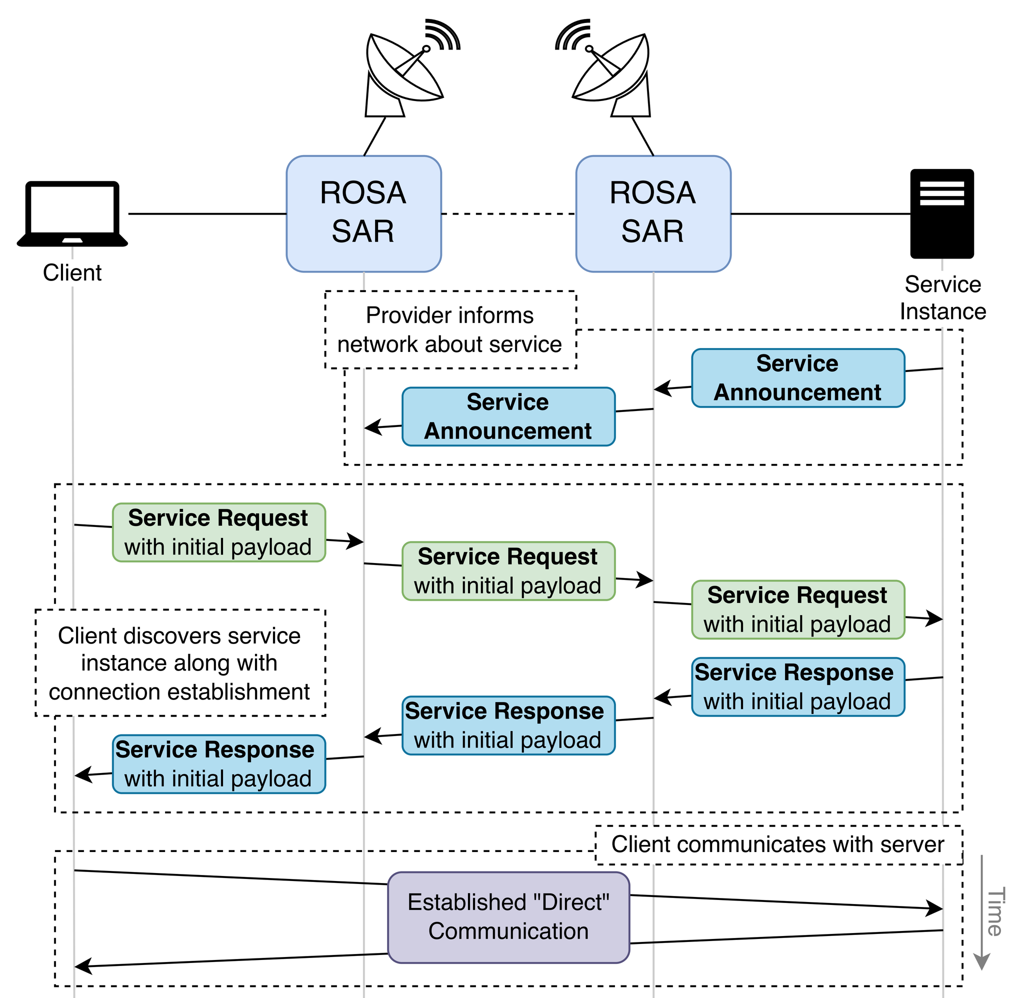

Figure 1 General ROSA message exchange

System Requirements ③

- Need to efficiently distribute the load across globally distributed Service Instances (SIs)
- Need to support a time-varying network

Therefore, the load-balancing mechanism must:

R1 Adapt to rapid load and availability changes

R2 Resolve semantic service names (e.g., domain names) to IP addresses in-band

R3 Minimize exchanged messages

R4 Handle inconsistent load-balancing information

R5 Minimize routing loops

R6 Give service providers some control over policy and load-balancing decisions

R7 Allow client to influence load-balancing decisions

Enhanced ROSA Service Address Router (SAR) ④

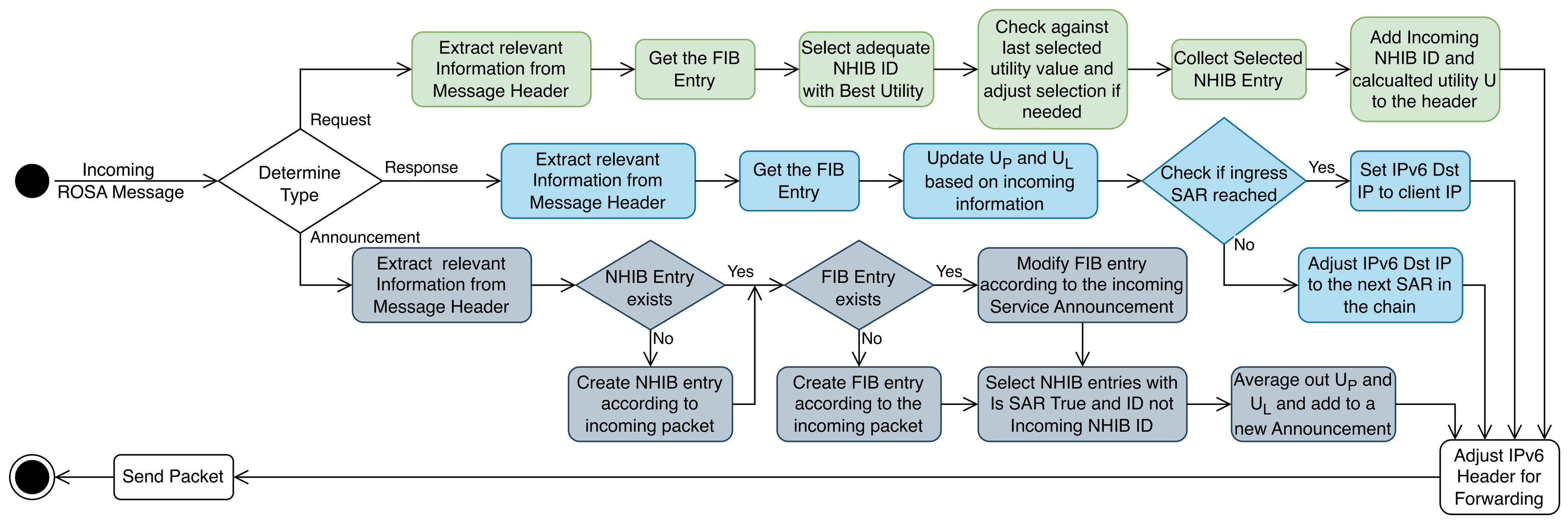

Figure 2 Enhanced ROSA SAR packet handling

- ⚠ ROSA does not support load balancing decisions according to SI load and network path conditions by default
- Extend decisions with forwarding choice utility ⑤
- Utility information saved in extended FIB and NHIB tables ⑥

Forwarding Choice Utility ⑤

$$U = \alpha \cdot U_P + (100 - \alpha) \cdot U_L$$

Defined as a combination of:

SI load – U_L ⟷ U_P – Cost of reaching the SI $

Client influence via α attached to *Service Requests*

Utility components saved in the FIB and NHIB tables

Updates using the *Service Response* and *Service Announcement* packets

FIB and NHIB Tables ⑥

Table 1 Example FIB table with two entries correlating service names with potential next hops and their associated utility

Service Name	Next Hop NHIB IDs	Known SI Tokens	Path Utility Map NHIB ID	Path Utility Map U_P	Load Utility Map NHIB ID	Load Utility Map U_L
EG1.COM	1,2	abdefgh, ijklmnop	1	30	1	90
			2	25	2	50
EG2.COM	3	qrstuvwx	3	30	3	110

Table 2 Example NHIB table with two entries correlating entry IDs with IP addresses of associated next hop

ID	Target IPv6 Address (corresponding device)	Next Hop Network Cost	Is SAR
1	fc00::1 (e.g. SAR 1)	20	true
2	fc00::2 (e.g. SI 1)	15	false
3	fc00::3 (e.g. SI 2)	10	false

Service Edge Design ⑦

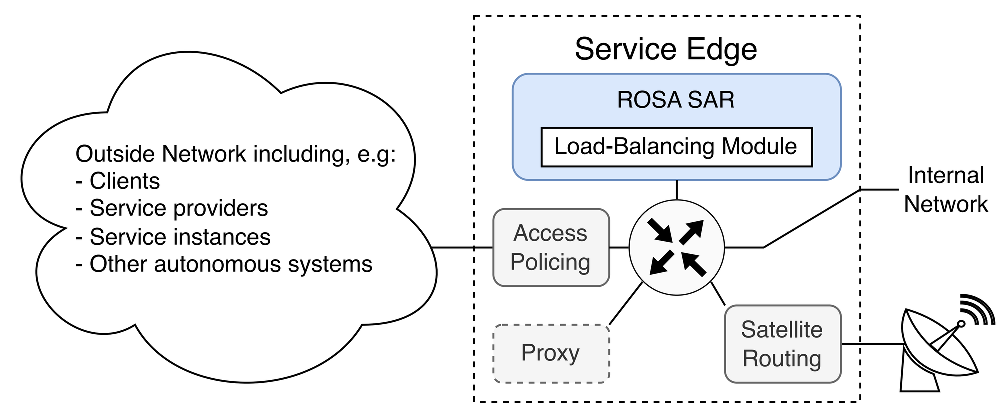

Figure 3 Simplified diagram of a *Service Edge*

Initial Deployment ⑧

- *Service Edge* implemented in Go
- Client- and Server-side gateways enabling use of generic TCP applications

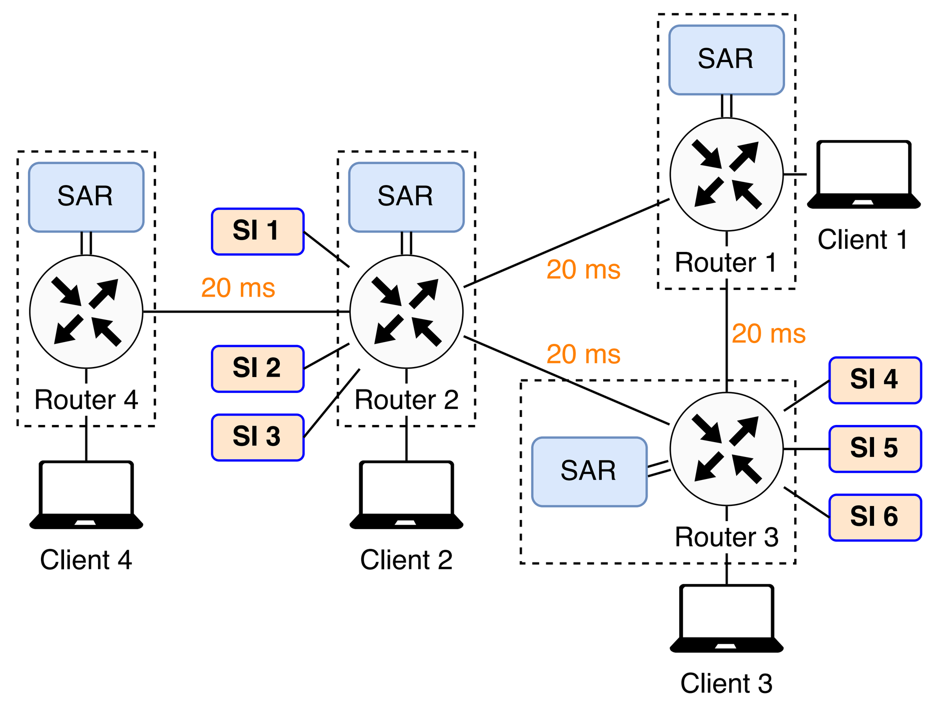

Figure 4 Network showcasing the trial system deployment

Initial Validation ⑨

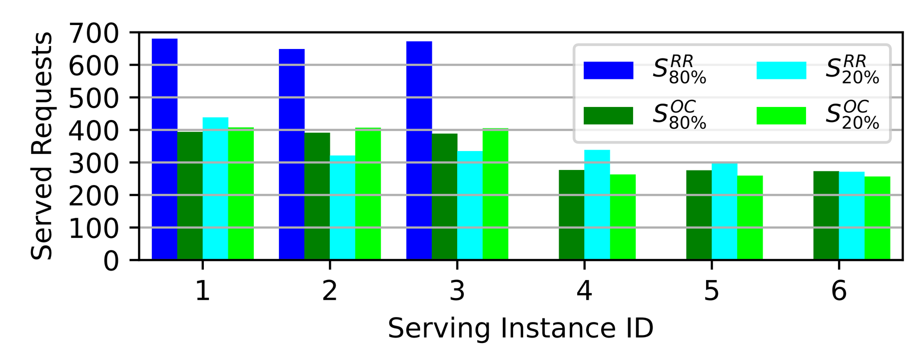

Figure 5 Requests served by each SI in the S experiment

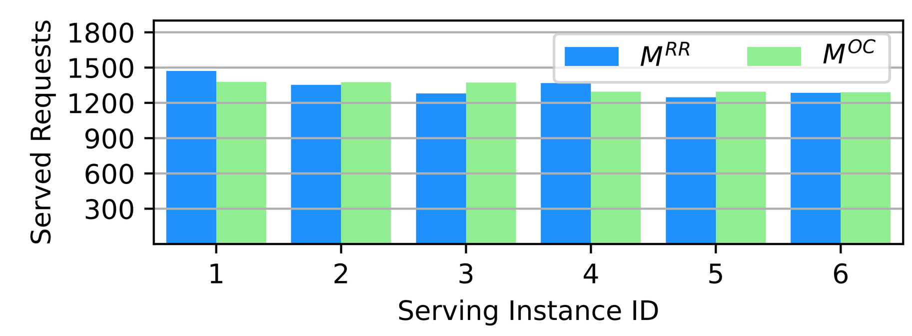

Figure 6 Requests served by each SI in the M experiments

- ☑ Decentralized forwarding and balancing decisions
- ☑ Agile reactions to system changes
- ☑ Even request distribution among SIs
- ☑ Consideration of client needs
- ☑ Control for service provider and network operator

[1]School of Computation, Information and Technology, Technical University of Munich
[2]Central Research and Technology, Airbus

[1] D. Trossen, L. M. Contreras, J. Finkhäuser, and P. Mendes. Architecture for Routing on Service Addresses. Internet-Draft draft-trossen-rtgwg-rosa-arch-01, July 2023.