EPI Framework: Approach for traffic redirection through containerised network functions

Jamila Alsayed Kassem[1], Onno Valkering[2], Adam Belloum[3], Paola Grosso[4]

Abstract

On the road towards personalised medicine, secure data-sharing is an essential prerequisite to enable healthcare use-cases (e.g.training and sharing machine learning models, wearables data-streaming, etc.). On the other hand, working in silos is still dominating today's health data usage. A significant challenge to address, here, is to set up a collaborative data-sharing environment that will support the requested application while also ensuring uncompromised security across communicating nodes.

EPI Framework is a novel data- sharing framework to support healthcare applications via virtualising network Services and automating security function setup.

The EPI Framework

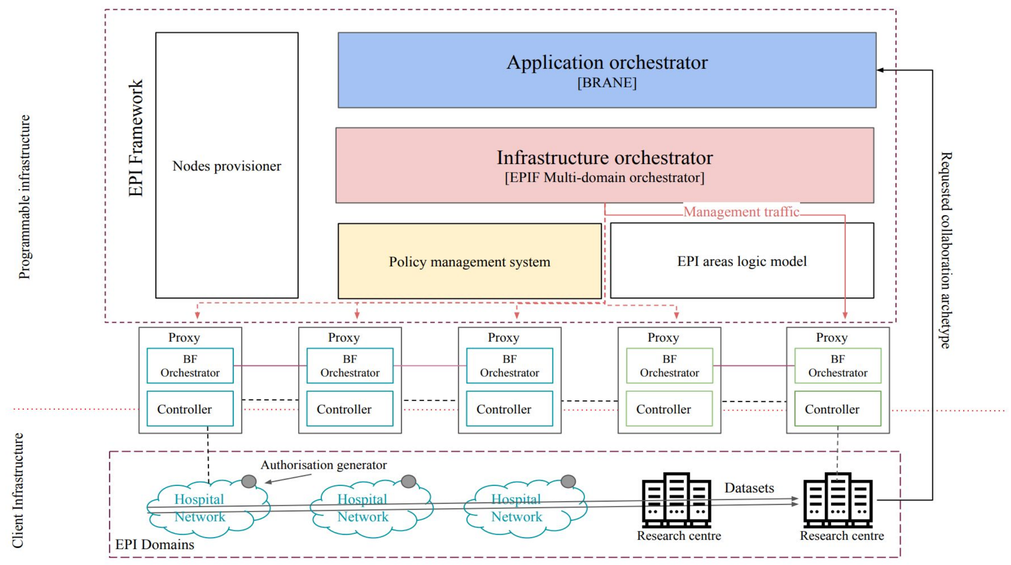

Fig2: The EPI framework architecture with running the different components (including proxy node).

The need for a dynamic infrastructure in healthcare

- ➔ The framework should adapt the underlying infrastructure per use case
- ➔ The adaption is done according to norms and policy agreements, requested application workflow, and network and security policies.
- ➔ Avoid the "one fits all" security standards

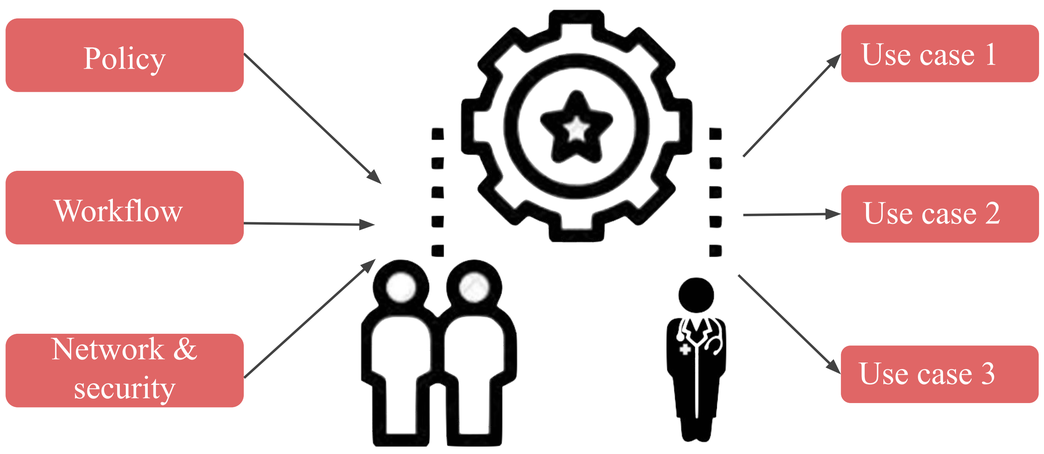

Fig1: The high level view of the infrastructure's considered inputs and outputs

The proxy mid-traffic

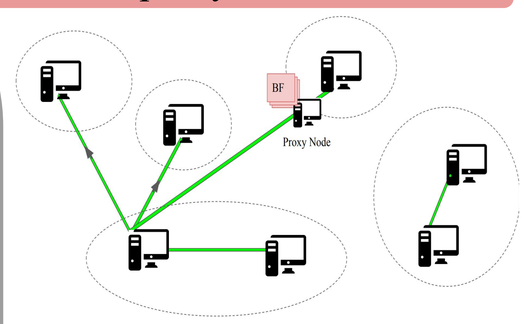

Fig3: An example setup of different nodes within domains belonging to different security areas.

Results

Topology	Name	CP (ms)	PS (ms)	CS (ms)
Proxy-in-between	DOCKER 1	5	5	10
	DOCKER 2	5	10	15
	DOCKER 3	10	5	15
Triangular	DOCKER 4	1	1	1
	DOCKER 5	5	5	5
	DOCKER 6	10	10	10

Table 1: The six network configurations used in our experiments and the respective latencies; three topologies (1-3) are related to proxy-in-between setup and three topologies (4-6) are related to the triangular setup.

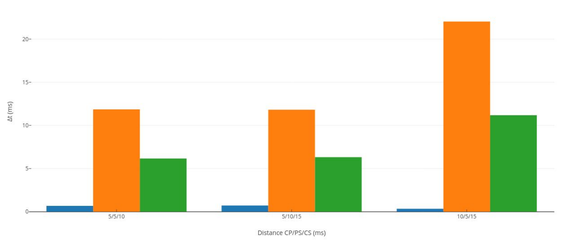

Fig4: The overhead of Δt of different proxy implementations compared to no proxy with changing configured distances.

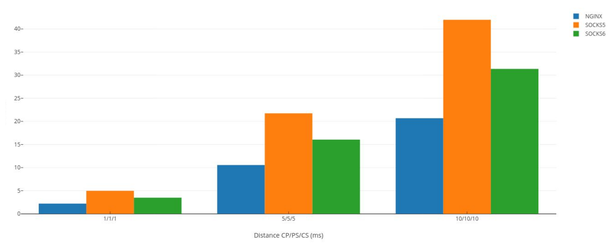

Fig5: The overhead of Δt (ms) of different proxy implementations compared to no proxy with changing configured distances.

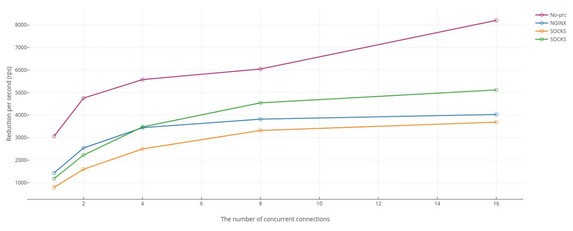

Fig6: The rate of processed transactions resulting via wrk of different proxy implementations with increasing concurrent connections.

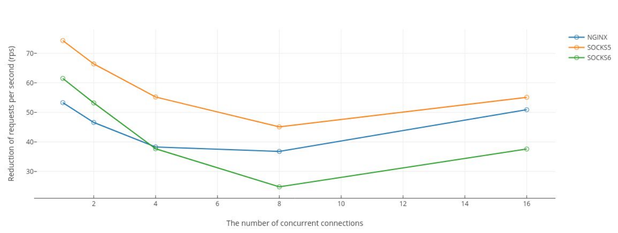

Fig7: The reduction of processed requests per second of different proxy implementations compared to no-proxy.

Conclusion

Parameters	NGINX	SOCKS5	SOCKS6
Δt	✓		
Processing rate			✓
Port scalability		✓	✓
Reconfiguration		✓	✓
Dynamicity		✓	✓
Security		✓	✓

Table 2: The comparison between different proxy implementations according to six performance parameters; where the ✓ represents an advantage over other proxies.

Manipulating traffic is a core feature within the EPI framework to enforce network services route:

- We evaluated and benchmarked two different approaches
- Δt depends on positioning of the proxy

  What proxy to deploy? The choice depends on:

  ⇒The application requirements

  ⇒Specific relevance of performance parameters

  →Time-critical application, NGINX

  →Data streaming application, SOCKS6

Ongoing work:

- Implementing more EPIF functionalities
- Bridging Function Chaining
- Uniform interfaces of bridging functions
- Extra plug-ins needed in the redirection tools

GitHub's code: PoC and Benchmark

The EPI project is funded by the Dutch Science Foundation in the Commit2Data program

[1] j.alsayedkassem@uva.nl
[2] o.a.b.valkering@uva.nl
[3] a.s.z.belloum@uva.nl
[4] p.grosso@uva.nl