Systematic Behavioral Analysis and Adaptive Traffic Routing for Services within Microservice Architecture Service Mesh

RAGHU MEDARAMETLA

Dr. Sharmin Jahan, Dr. Johnson Thomas, Dr. Anirudh Paranjothi

Introduction

Microservice architecture is a popular topic in developing large software applications. In these applications, the service mesh functions as a layer that allows microservices to communicate with each other without modifications to their implementations. It efficiently handles service-to-service communications, improving the application's maintainability. However, the existing service mesh does not have an autonomous adaptability to enhance the security. There is a need for protection mechanisms to assess risks.

Our solution is to add an Adaptation component to the service mesh that changes traffic patterns on demand based on real-time analysis. Specific parameters from service requests are used as environmental conditions, and they are compared to a threat model to see how likely it is that security rules for authorized data transfer will be violated. We improve the application's security by keeping an eye on how services behave, and traffic is routed to stop known threats. This evaluation takes the service mesh's decision-making, letting preventive measures be taken to reduce possible security risks and protect the integrity of the flow of information between services.

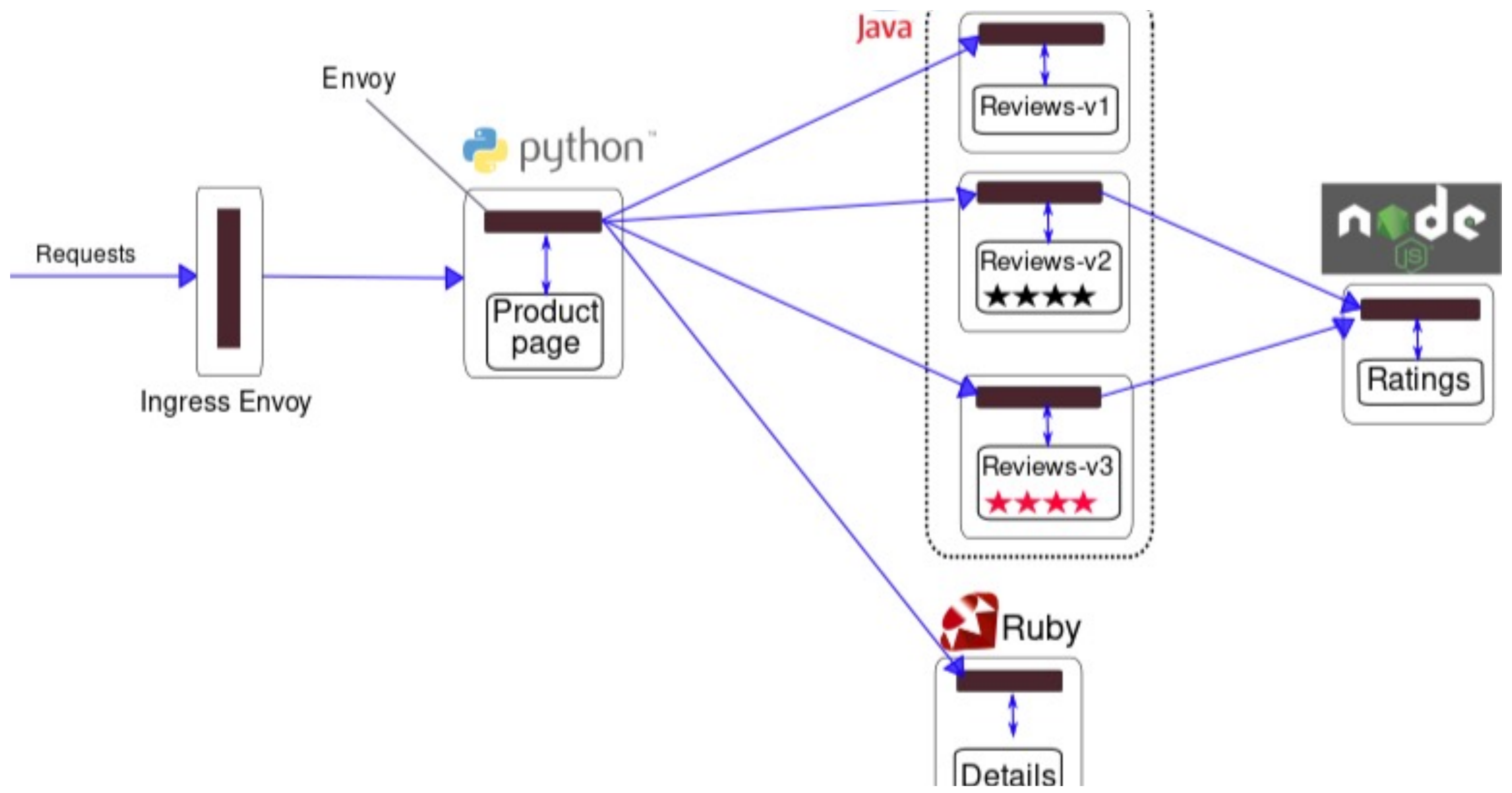
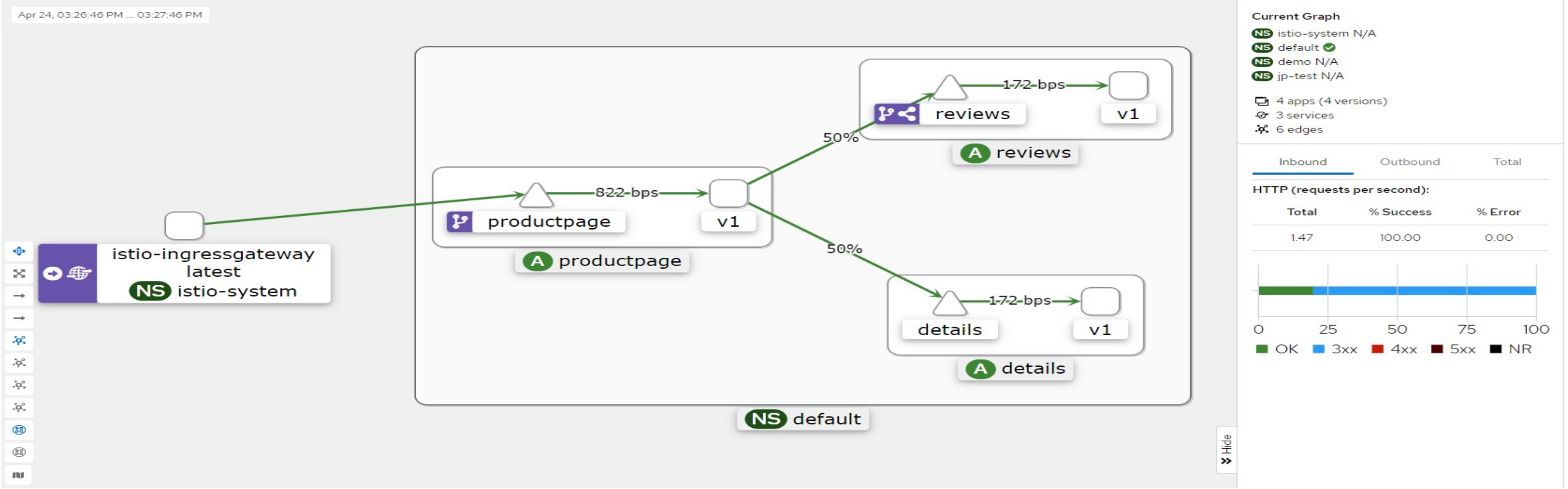


Figure 1. Book Info Application with Istio Service Mesh.

Related Work

Intelligent Service Mesh Framework for API Security and Management: Our method incorporates an adaptation component directly into the service mesh, whereas this work uses machine learning for anomaly detection. This enables real-time analysis of service requests and dynamic traffic routing based on predefined security policies, offering a more active and detailed security advancement within the microservice architecture.

Securing Microservices with Service Mesh: This work discusses the role of service mesh in securing microservices architectures. While it highlights the importance of service mesh for security, our work goes a step further by proposing a concrete solution for enhancing security within the service mesh itself.



Current Graph
NS istio-system N/A
NS default
NS demo N/A
NS jp-test N/A

4 apps (4 versions)
3 services
6 edges

Inbound Outbound Total

HTTP (requests per second):

Total	% Success	% Error
1.47	100.00	0.00

Figure 5. Observability for malicious users.

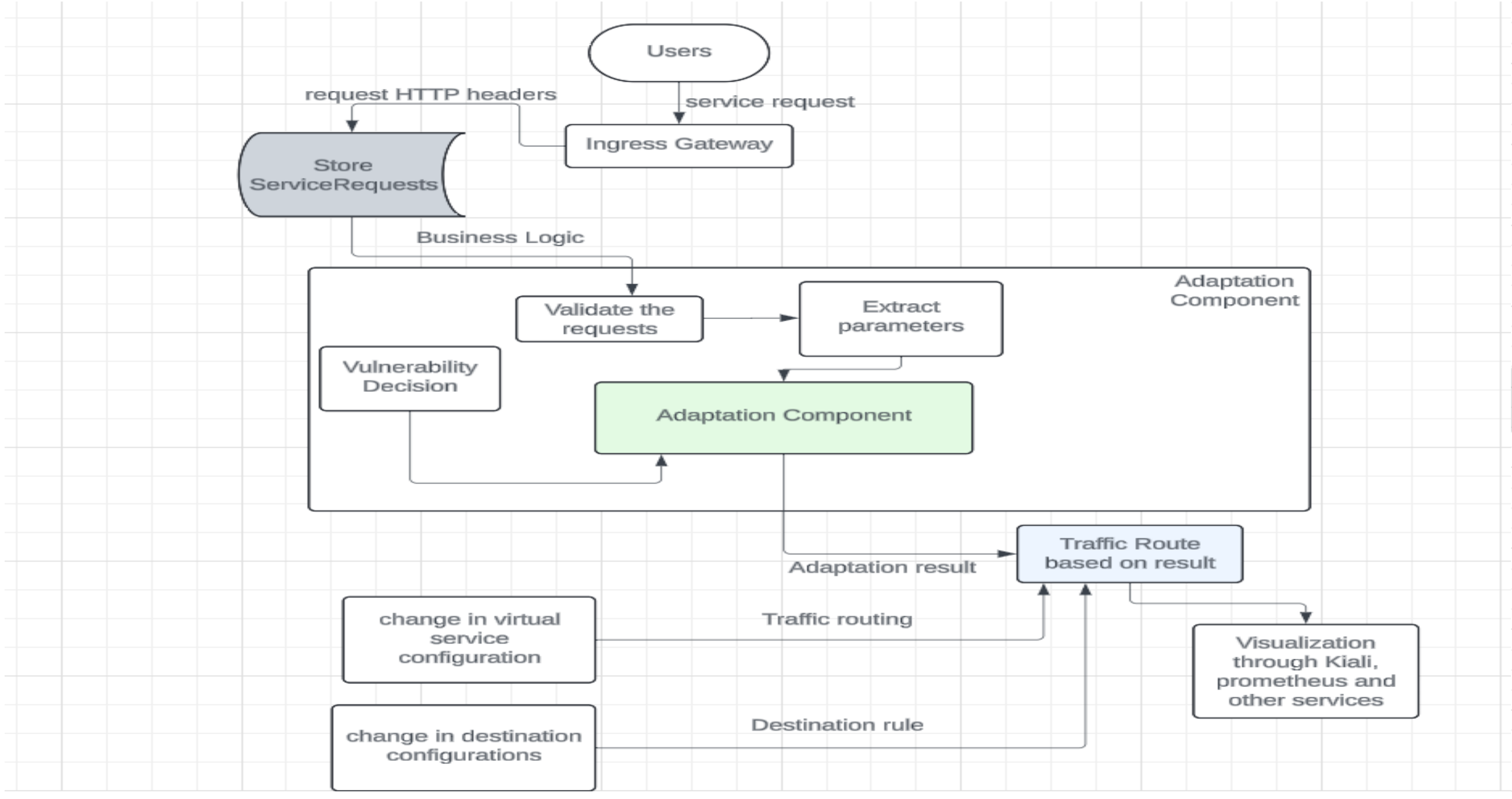


Figure 2. Dataflow diagram of AC as a service in service mesh.

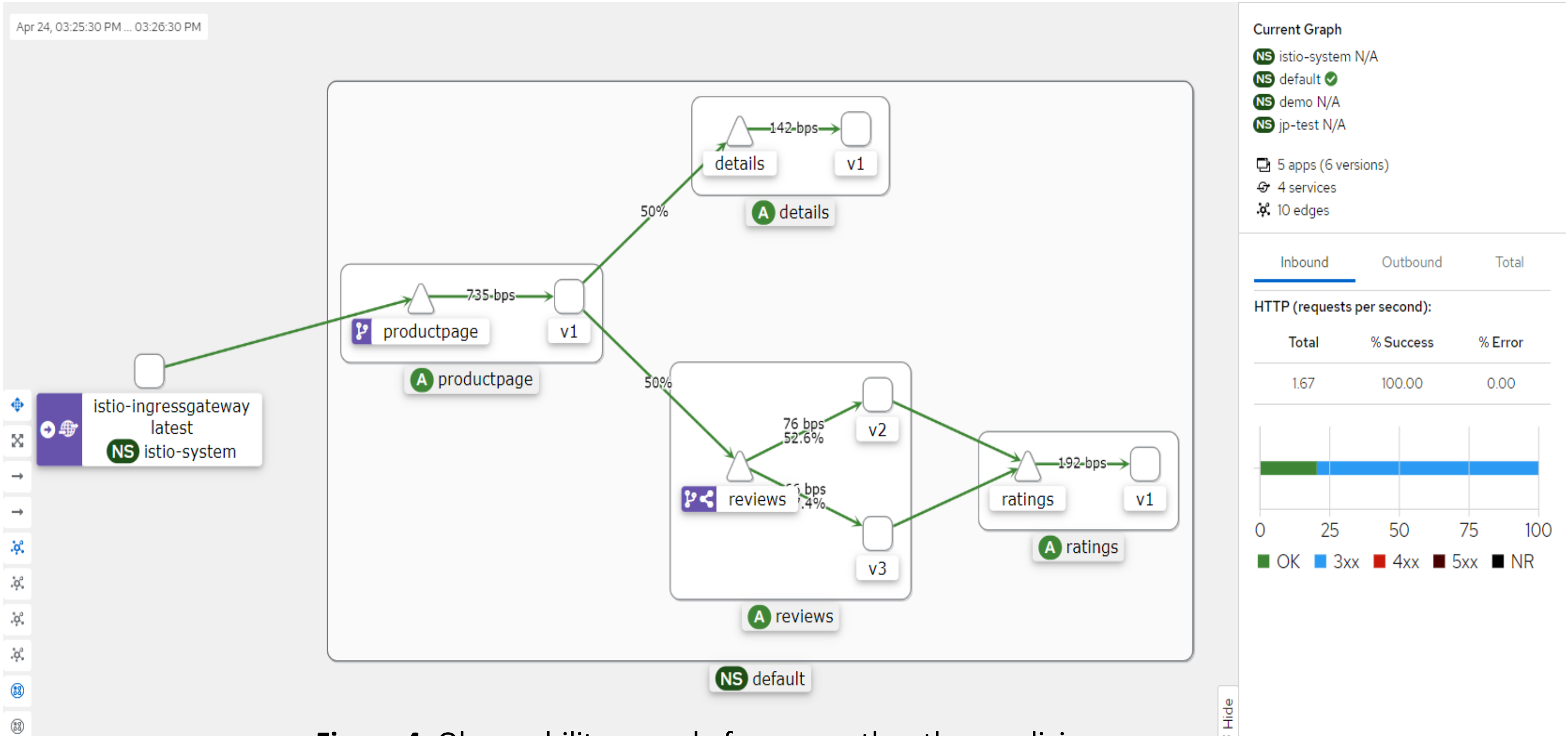
Technical Approach

In this proposed approach, we used Kubernetes as an orchestrator and Istio as the service mesh.

- Istio was chosen over many available meshes because of the capabilities of Envoy proxy and session affinity, which we use to promote the protection decision based on the result using the HTTP headers.
- A microservice application Book info is deployed within the Istio system, book info consists of ratings, details and three versions of reviews. So, when the application is injected with envoy proxies. These are responsible for setting up iptables rules so that in/out bound of traffic go through proxies as in Figure 1.
- Next, The adaptation component (AC) is deployed as a service within a pod in the control plane of the service mesh. The AC service utilizes proxies to impose its actions on other services in the data plane as shown in figure 2.
- The ingress gateway calls AC, which intercepts and stores each envoy proxy HTTP data in JSON format. The AC assess these service requests and get the required parameters. That result is used to adjust the traffic routing for protection and distributed to other services.
- In our case, we consider to get request id of each service request and if a user sends more than 5 requests within a minute there is a high chance it comes from suspicious user. So, for that user we limit the request routing to specific versions to reduce probability of attack as shown in figure 4,5.

Example Conditions	Decided traffic route
Malicious user	Route traffic to V1
Not a malicious user	Route traffic to V2, V3

Table 1. Experiment and Results.



Current Graph
NS istio-system N/A
NS default
NS demo N/A
NS jp-test N/A

5 apps (6 versions)
4 services
10 edges

Inbound Outbound Total

HTTP (requests per second):

Total	% Success	% Error
1.67	100.00	0.00

Figure 4. Observability console for users other than malicious user.

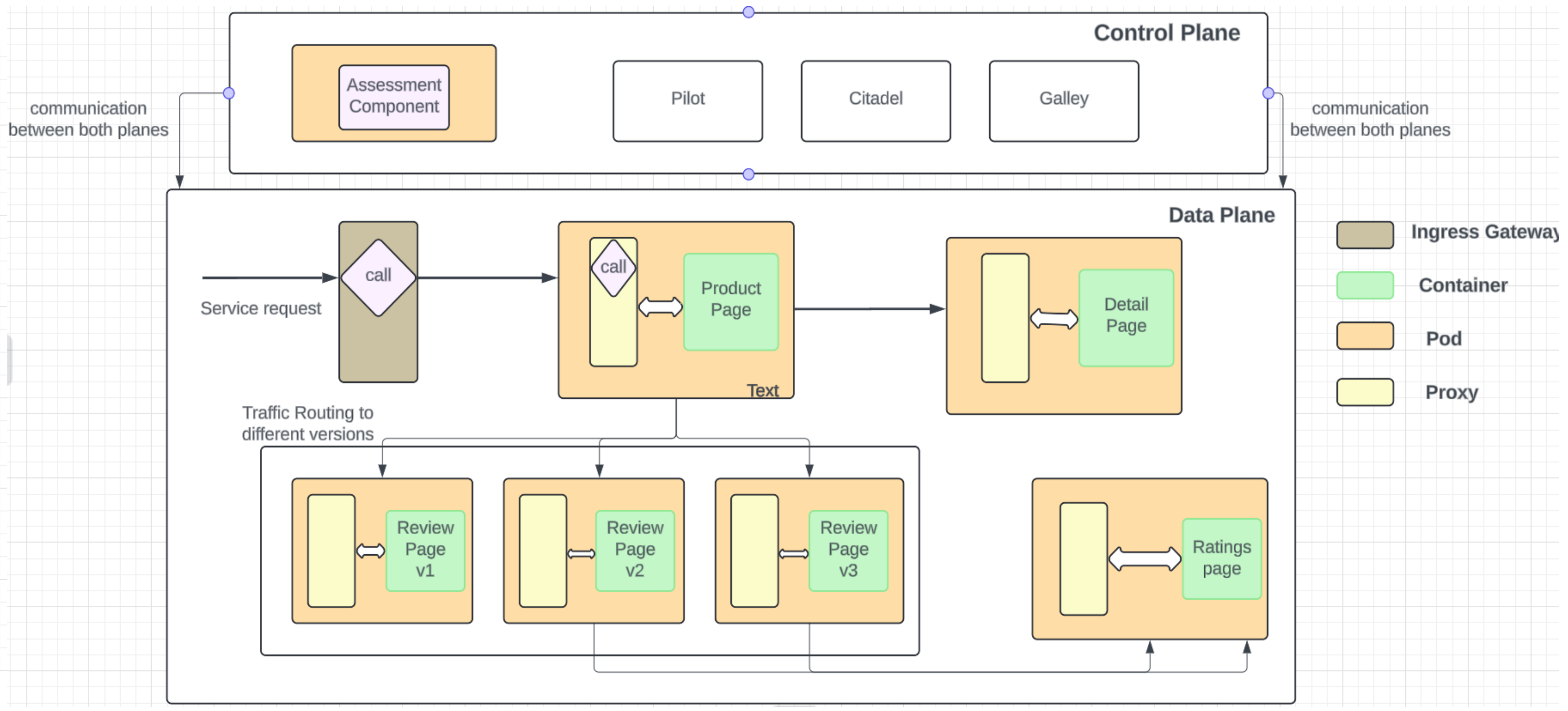


Figure 3. Service mesh architecture for Book info with an adaptation component.

Results

The interesting thing I found when I am working on this approach is how the traffic routing works on different users. I learned that the additional effort required to identify and analyze service requests could affect the microservice architecture's overall speed, especially when there is a lot of traffic. The importance of implementing the adaptation component in the best way possible to minimize performance impacts and maximize security benefits is emphasized by this.

Furthermore, our tests made us think about the solution's ability to grow and be accurate, especially in complex microservice deployments where different services interact with each other. The adaptation part may not be as good at identifying and stopping threats as it used to be as the number of microservices and the complexity of their communication patterns rise. This makes it clear that more research needs to be done on scalable threat modeling techniques and methods for those within the service mesh to make decisions.

Conclusions

Our approach to integrate an adaptation component into the service mesh is a novel step toward improving the security and adaptability of microservice architectures. We add an adaptive security measure that reduces potential threats and keeps the integrity of data flow between services by changing traffic routing on demand based on real-time analysis of service request parameters. This innovative solution addresses a critical gap in existing service mesh frameworks as they lack autonomous adaptability to changing operations and security threats.

Although our approach makes microservice security much better, it also raises questions and suggests areas for further research. For instance, looking into more threats and machine learning models to look at service request parameters could make threat detection in the service mesh more accurate. Additionally, looking into the performance of adding the adaptation part to large-scale microservice deployments would give us useful information about how well our approach works in real life and how much it costs.

References

[1] W. Li et al., “Service mesh: challenges, state of the art, and future research opportunities,” in Proceedings - 13th IEEE International Conference on Service-Oriented System Engineering, SOSE 2019, 2019, pp. 122–127, 2019.

[2] R. Chandarmouli, “Security strategies for microservices-based application systems,” National Institute of Standards and Technology (NIST), Gaithersburg, MD, NIST Special Publication (SP) 800-204, 2019.

[3] S. R. Boyapati and C. Szabo, "Self-adaptation in microservice architectures: a case study," 2022 26th International Conference on Engineering of Complex Computer Systems (ICECCS), pp. 42-51, 2022