TAG Network Project Review - Sermant

The *Sermant* project had a presentation for TAG Network at our meeting held the date. The project was presented by *Yang Yi*, *name02*, *etc*. The TAG leadership was represented by *Lee Calcote*, *Nic Jackson*.

Recording of the meeting: https://www.youtube.com/watch?v=IhOMBfbezmk&t=846s
Meeting notes: https://docs.google.com/document/d/18hYemFKK_PC_KbT_TDBUgb0rknOuIhikkRxer4_bv4Q/edit#heading=h.fu1n3n4qiw3y

Project Summary

Sermant is a Service Mesh created in 2022, that is built around Java Agent and has seamless integration to non Java services via Istio. It is intended to create a low latency, zero friction way for Java developers to take advantage of service mesh.

Sermant can be used for architectural transformation, continuous delivery, automated service registration and load balancing.

History

Notes on the project's history, what has been achieved so far, and other relevant information.'

The project authors believe that one of the reasons for slow adoption of service mesh in China was due to the increased consumption of resources on the cluster and increased latency caused by service mesh components like Envoy.

While much of this can be offset by using technologies like eBPF, eBPF causes its own issues with architectural complexity. Service meshes that are provided as an SDK which need to be embedded into the application code introduce tight coupling and a software maintenance problem.

Since it is believed that 90% of microservices that are developed in China use Java, Sermat was created to leverage the power of Java Agent to create a service mesh that is lightweight, performant, and can be easily integrated with existing applications with 0 code changes. This approach was inspired by existing Application Performance Management (APM) tools which also leverage the Java Agent to dynamically instrument the application.

- *Ability to tag requests with a context that is passed between service calls, this facilitates routing between different service instances later in the request chain.*

Architecture

Notes on architecture, like what language it is written in, how the service or application should be run, and other interesting information.

The data plane for Sermant uses Java Agent, a special feature of Java that allows the instrumentation API to modify the Java Byte Code of an existing application loaded into a JVM.

https://www.baeldung.com/java-instrumentation#what-is-a-java-agent

To support ease of upgrade each of the components within the agent is a separate plugin, currently it supports Monitoring, Routing, Flow Control, Load Balancing and Fault Injection. Plugins can be dynamically updated rather than having to update the entire agent.

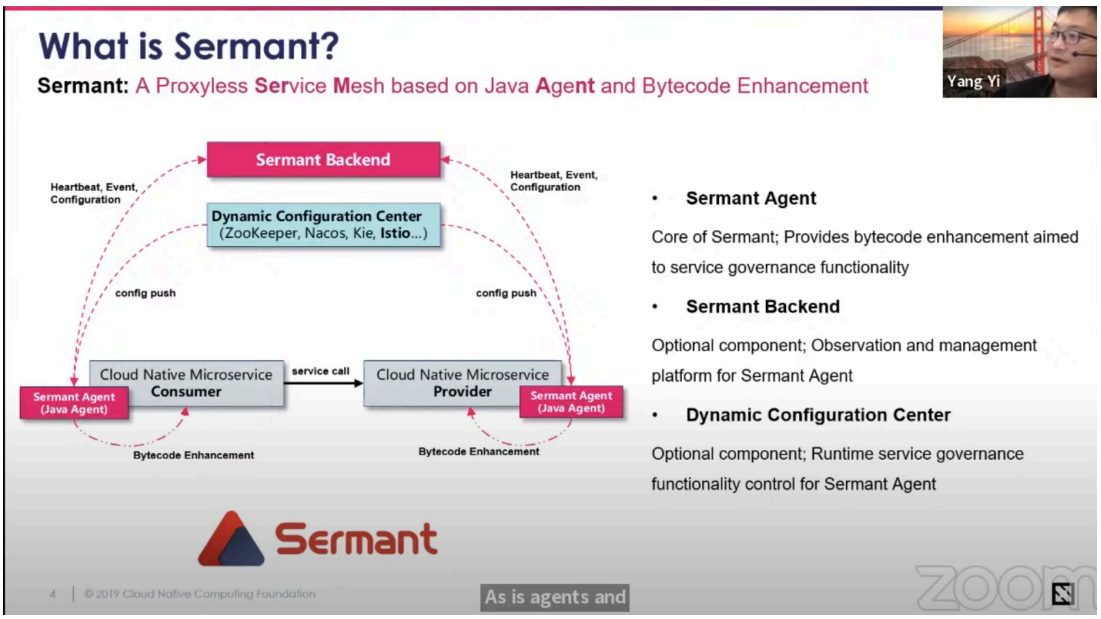
Currently a wide range of range of Web and RPC frameworks are supported by the project, these include:

- SpringBoot 1.5.x
- Servlet 3.0+
- Jetty 8.x+
- Tomcat 7.x+
- gRPC 1.13+
- Dubbo 2.6.x 2.7.x, 3.x
- ServiceComb Java Chassis 2.x
- SofaRpc 5.x

Because Sermant is embedded into the request flow of the application it can automatically tag and append metadata to a request. This enables smart routing where an individual request can be automatically routed through a set of microservices. For example, the end user may wish to send a particular user through v2 of many different microservices.

Configuration for the mesh is pushed from the Dynamic Configuration Center (DCC); this is a replaceable control plane that can utilise ZooKeeper, Nacos, Kie, or Istio. Since the data plane supports the xDS protocol, any control plane that integrates xDS could also be used.

In addition to the DCC, an optional component is the Sermant Backend, this stores heartbeats and events from each of the augmented services. It provides observability and health into the individual agents.



https://www.youtube.com/live/IhOMBfbezmk?si=-vb5zQ2lZR2Q3yOw&t=731

Goals & Roadmap

Notes on the current goals of the project, what features are planned in the near and distant future.

Current Roadmap:

- Expand exposure and usage outside of China.
- Support mTLS and Authentication Policy.
- Increase xDS integration and feature compatibility with the Istio control plane.

Future Roadmap 2025+:

- Integrate with Open Telemetry, not a current priority as application developers have other options available to them.

Key Considerations:

Community and Growth

The number and affiliation of maintainers, number of contributions, activity in discussion forums (GitHub issues and/or discussions, Slack, Discord, etc), representation at conferences and meetups, etc.

Currently Sermant has 26 contributors and has an active code commit history. It has 152 Folks and 1200 GitHub stars. The website https://sermant.io is available in Chinese and English.

The website details their code of conduct which follows the CNCF Code of Conduct and also the contribution policy.

Adopters:

- Huawei
- Jingling
- z-ONE
- Mashang
- Dobest
- Yonyou
- Yunchewang

Maintainers:
Currently Sermant has 4 official maintainers of whom are employed by Huawei, and 2 core maintainers who are employed outside of Huawei.

Sermant/MAINTAINERS.md at develop · sermant-io/Sermant · GitHub

TAG Recommendation to TOC

It is TAG Network's opinion that Sermant has a valuable role to play in both the CNCF ecosystem and with Service Mesh technologies. The ease of adoption for Java developers along with the ability to integrate with existing xDS enabled service meshes like Istio has huge benefits for the community. We feel that the architectural pattern is well thought out and the project has been developed to operate at scale and within large financial institutions.

We think that Sermant could benefit from speaking with the Istio, and other Service Mesh teams to show their current or potential integrations. Sermant has real potential to drive adoption of Service Mesh among Java developers and could be a real benefit to the CNCF ecosystem and community.