Service mesh. Знакомство с Istio. Envoy

Не забудь включить запись!

План

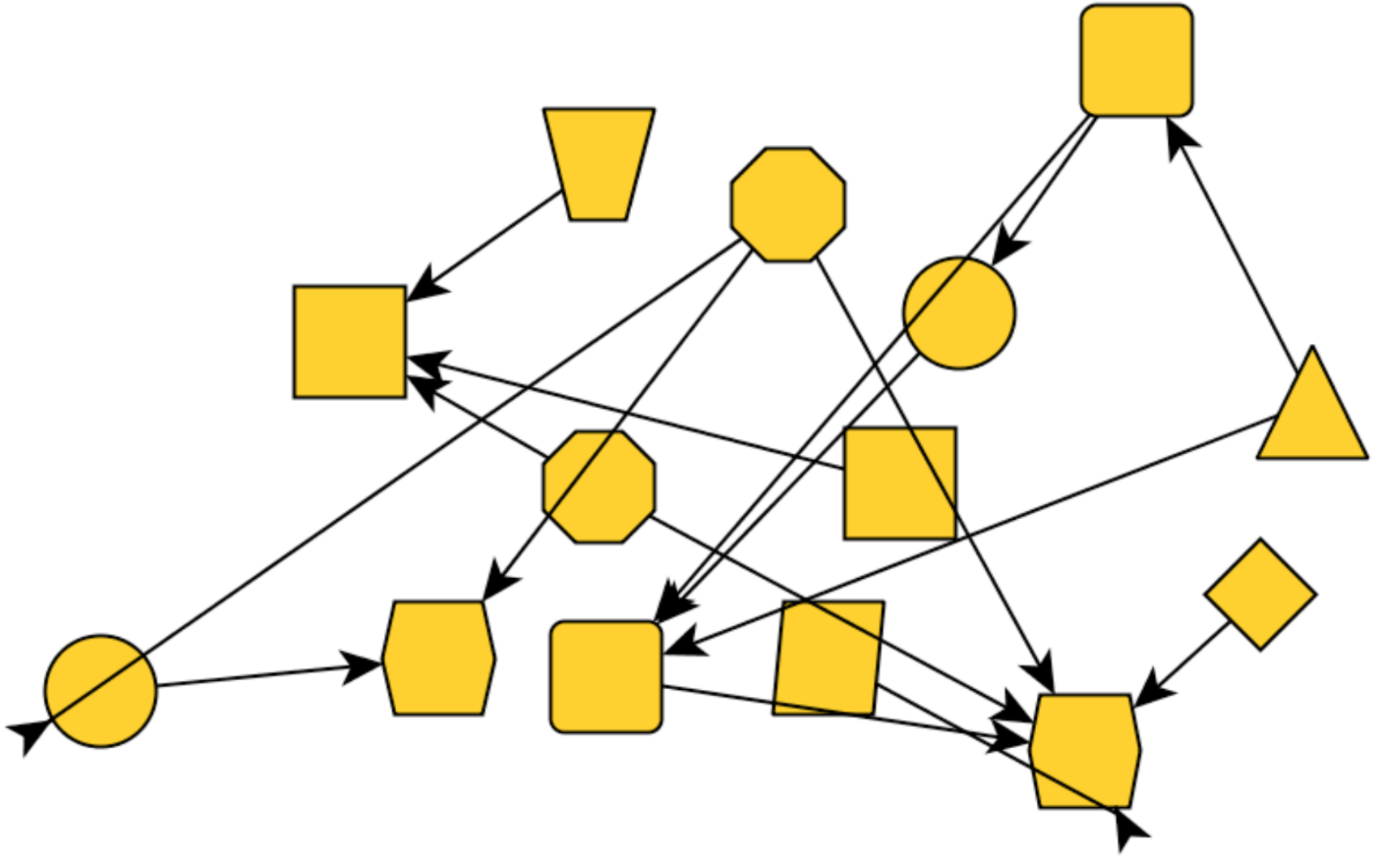
- Проблемы сети между микросервисами
- Service mesh. Control and Data plane
- Компоненты Istio
- Основные возможности Istio
  - Traffic Control
  - Service Resiliency
  - Observability

Немного о микросервисах...

Преимущества микросервисов

- Независимость (Independence)
- Устойчивость (Resilience)
- Масштабируемость (Scalability)
- Автоматизация жизненного цикла (Lifecycle automation)

Особенности икросервисов



Проблема ненадежной сети

HYSTRIX

DEFEND YOUR APP

spring Eureka

[Ribbon]

[Hystrix]

[Eureka]

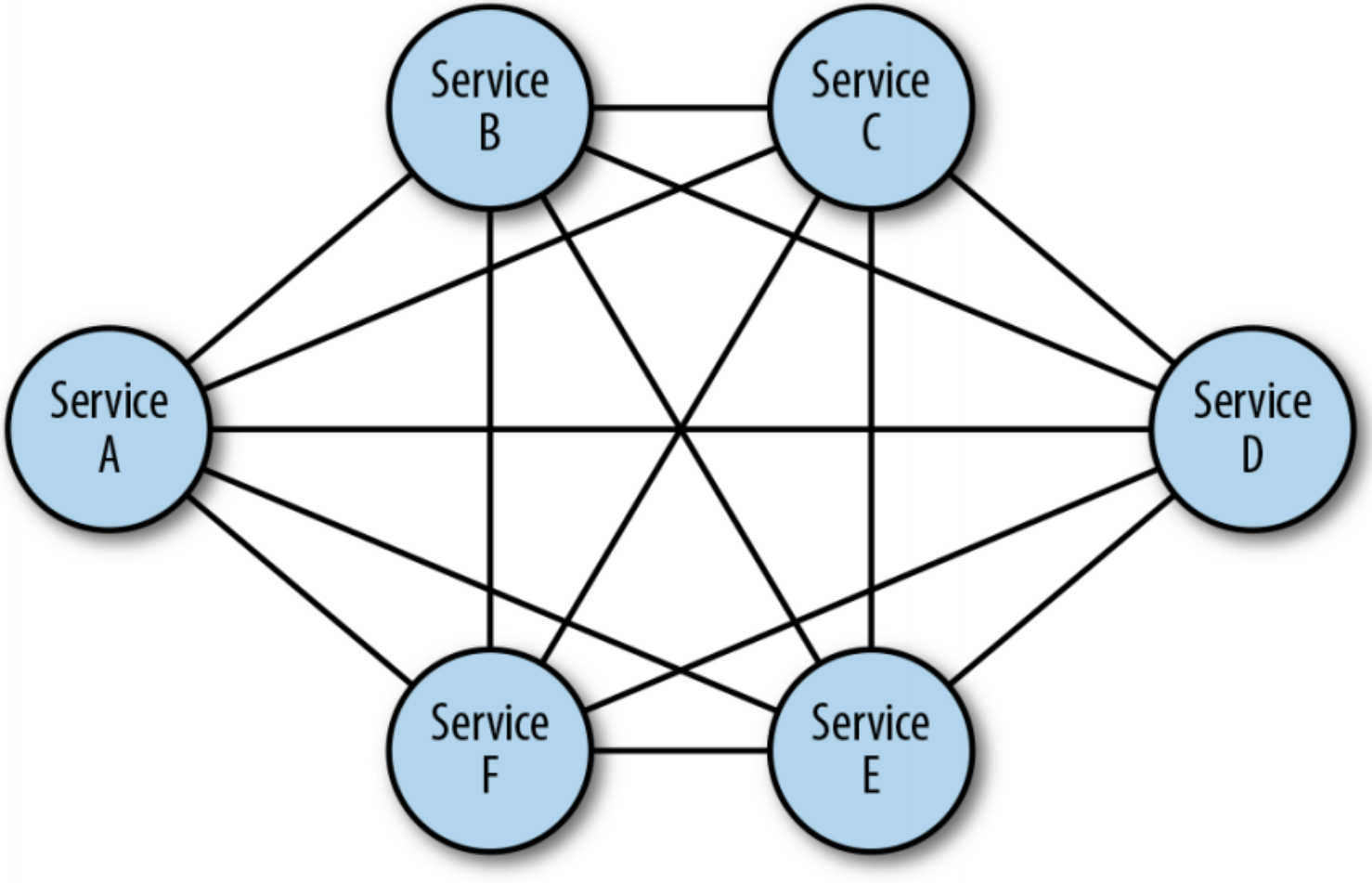
Service mesh

Service mesh

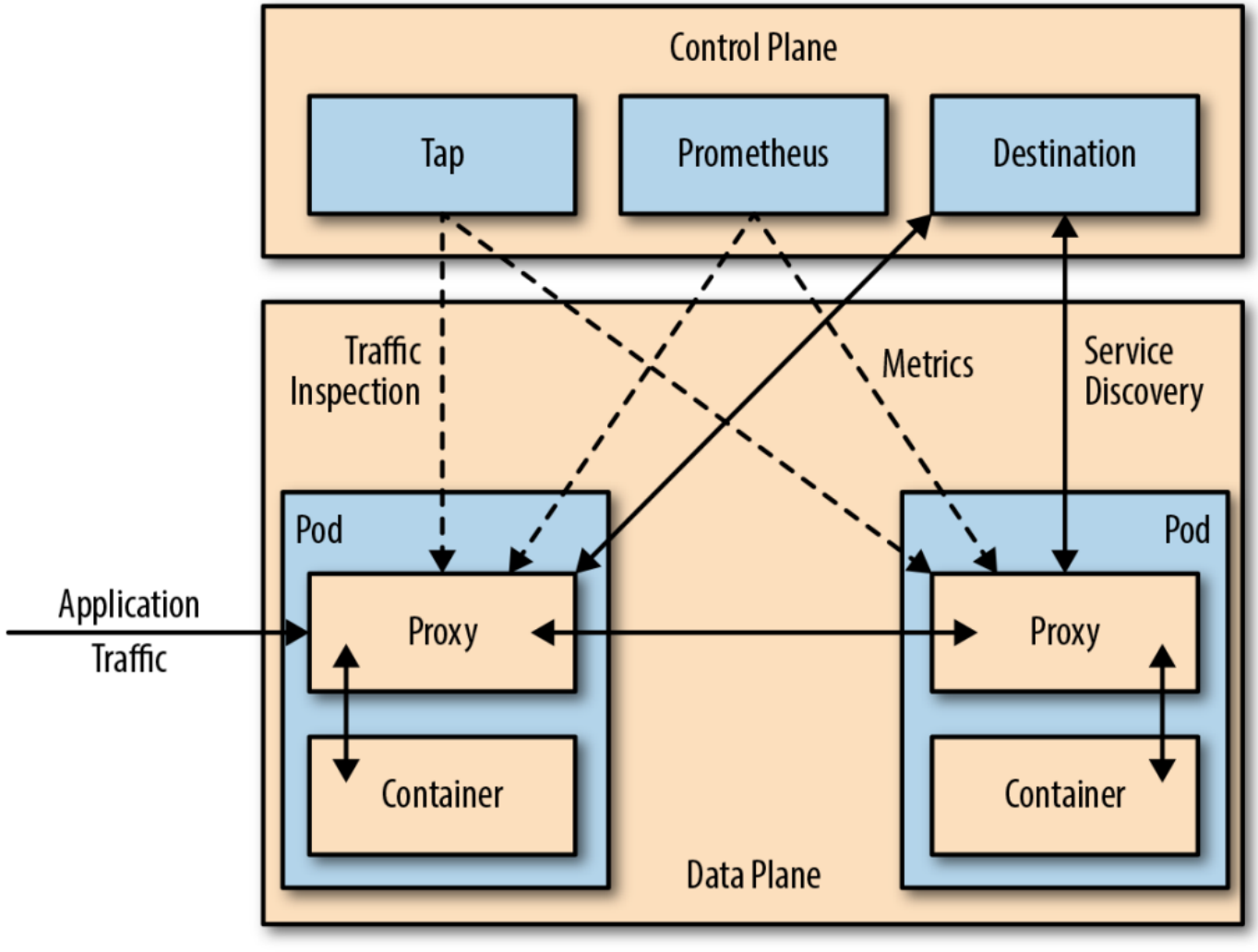
Service mesh решают проблемы взаимодействия микро-сервисов и предлагают функции:

- Service discovery
- Маршрутизация и настройка трафика
- Шифрование, аутентификация, авторизация
- Метрики и мониторинг

Service mesh



Control and Data plane



Istio

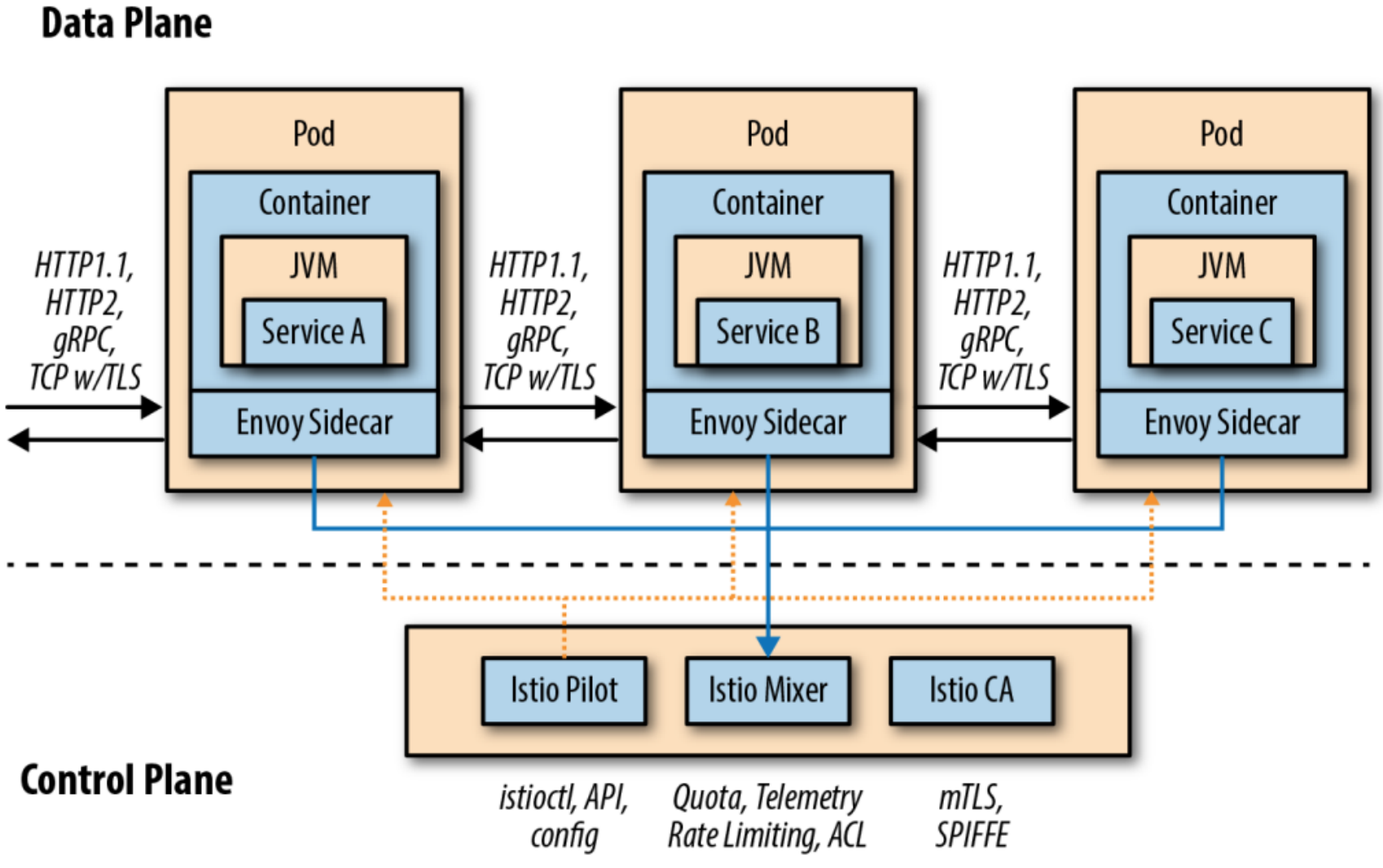
Istio

Что дает Istio командам?

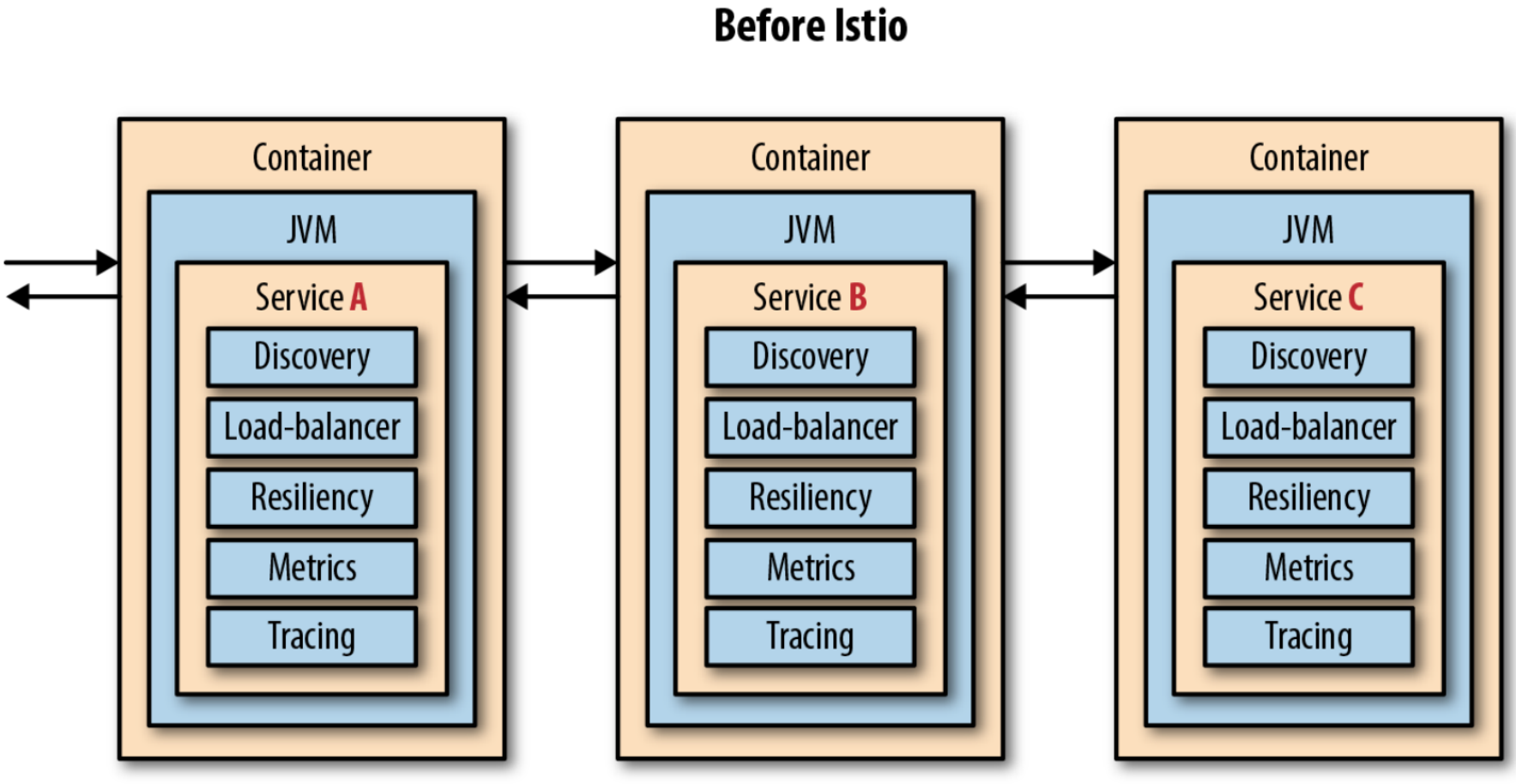
Developers: разработка логики микросервисов

Operations: централизованное управление, конфигурация и мониторинг сети

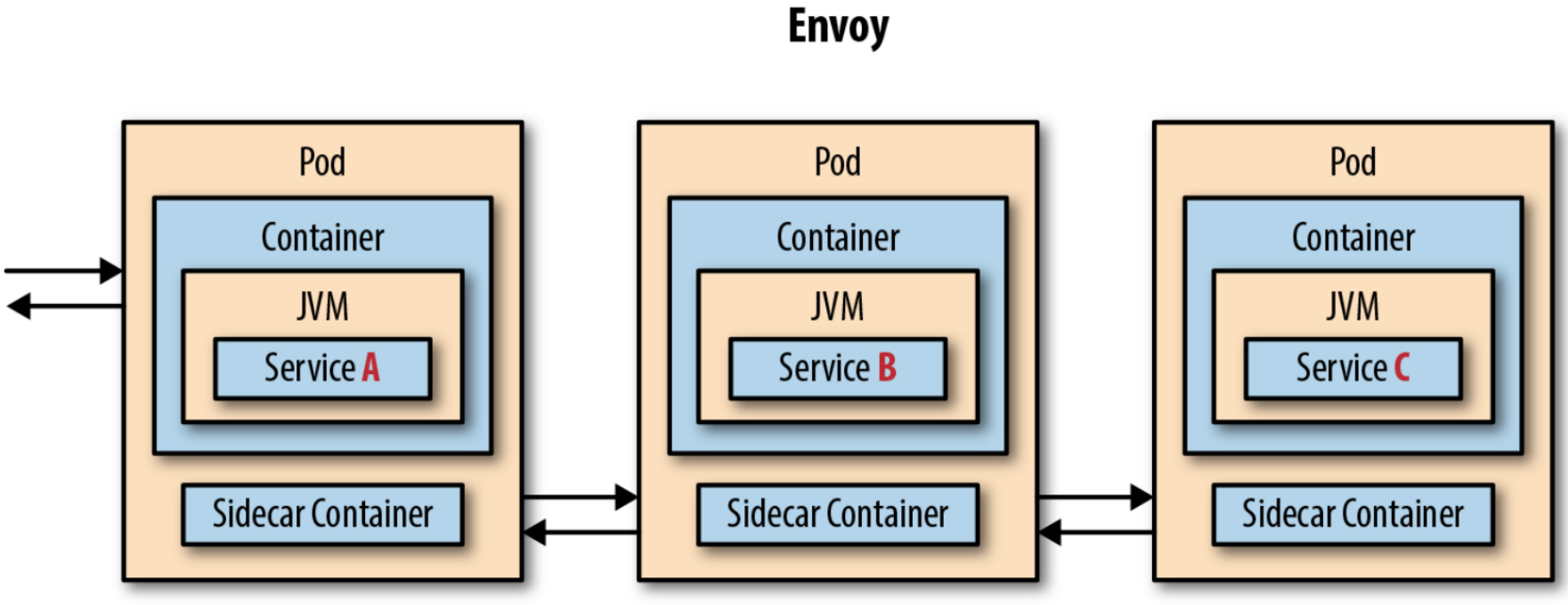
Компоненты Istio



Компоненты Istio



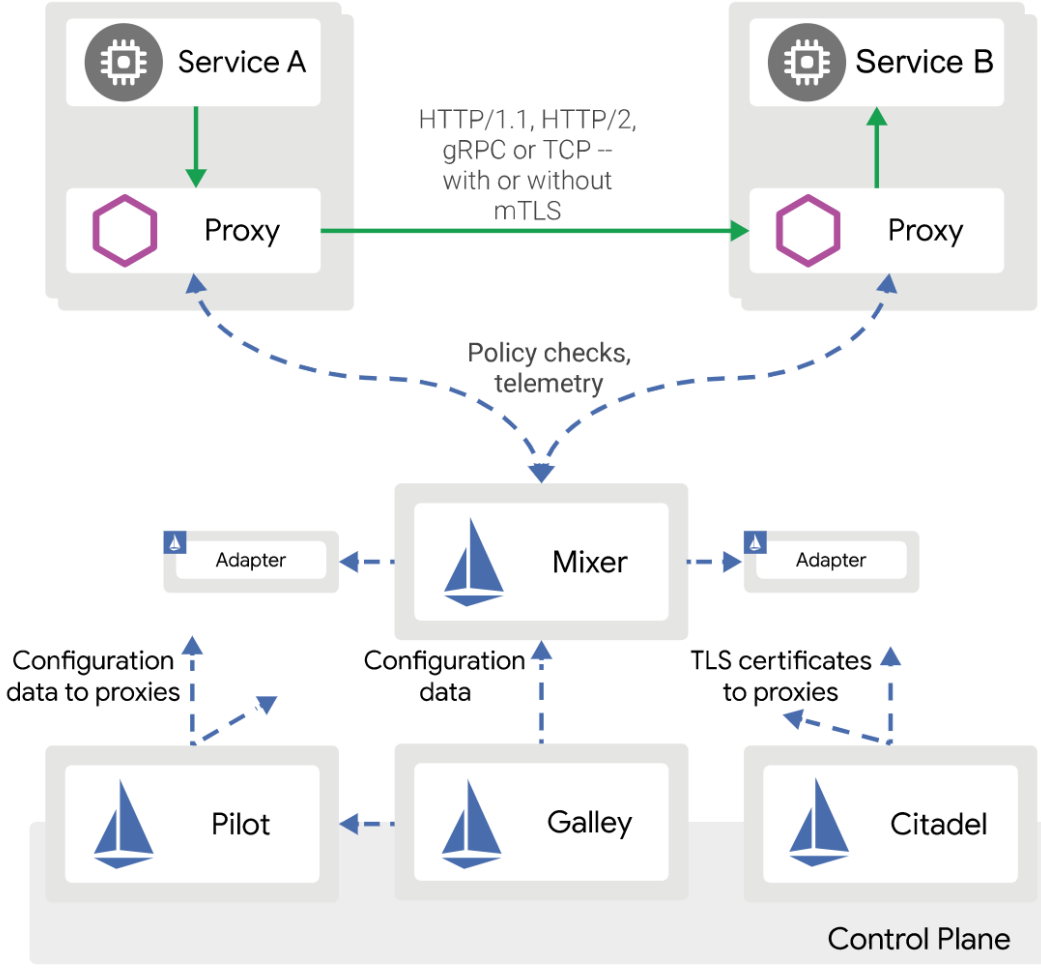
Компоненты Istio



Data plane. Envoy

- Dynamic service discovery
- Load balancing
- TLS termination
- HTTP/2 and gRPC proxies
- Circuit breakers
- Health checks
- Staged rollouts with %-based traffic split
- Fault injection
- Rich metrics

Control plane. Mixer



Istio Architecture

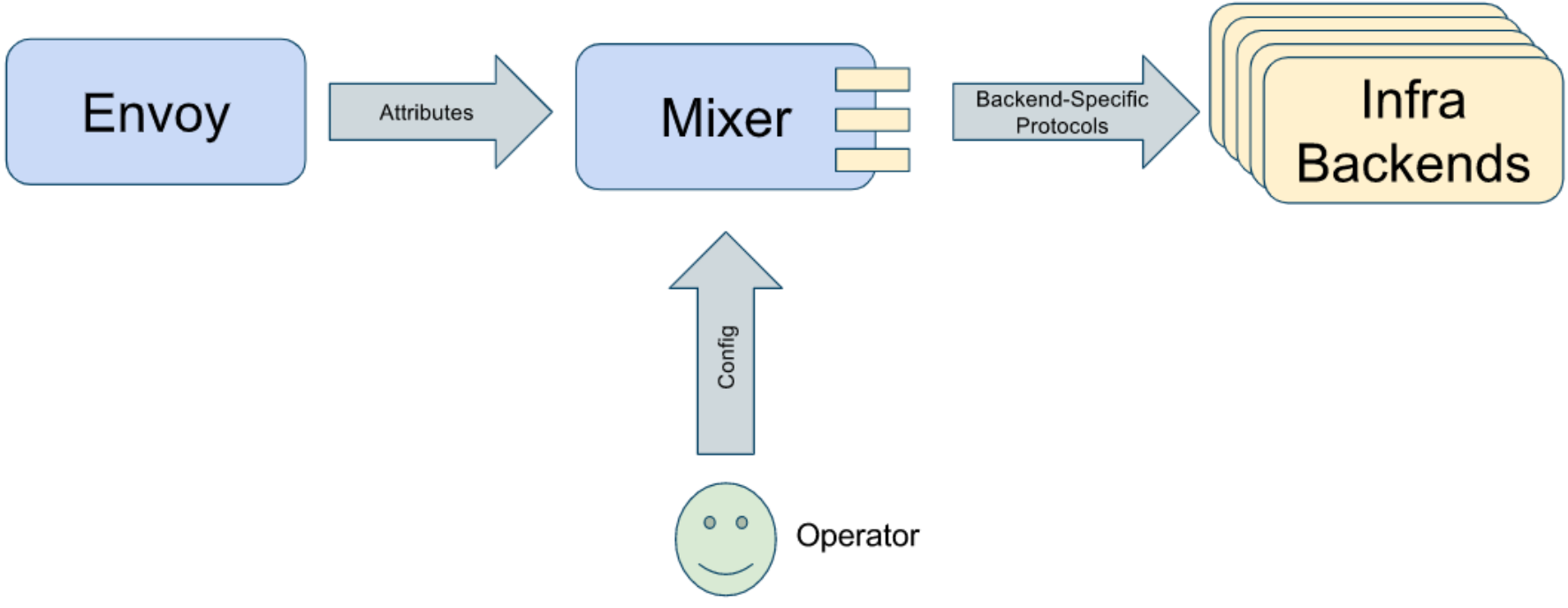
Mixer. Attributes example

```
request.path: xyz/abc
request.size: 234
request.time: 12:34:56.789 04/17/2017
source.ip: [192 168 0 1]
destination.service.name: example
```

Типы атрибутов:

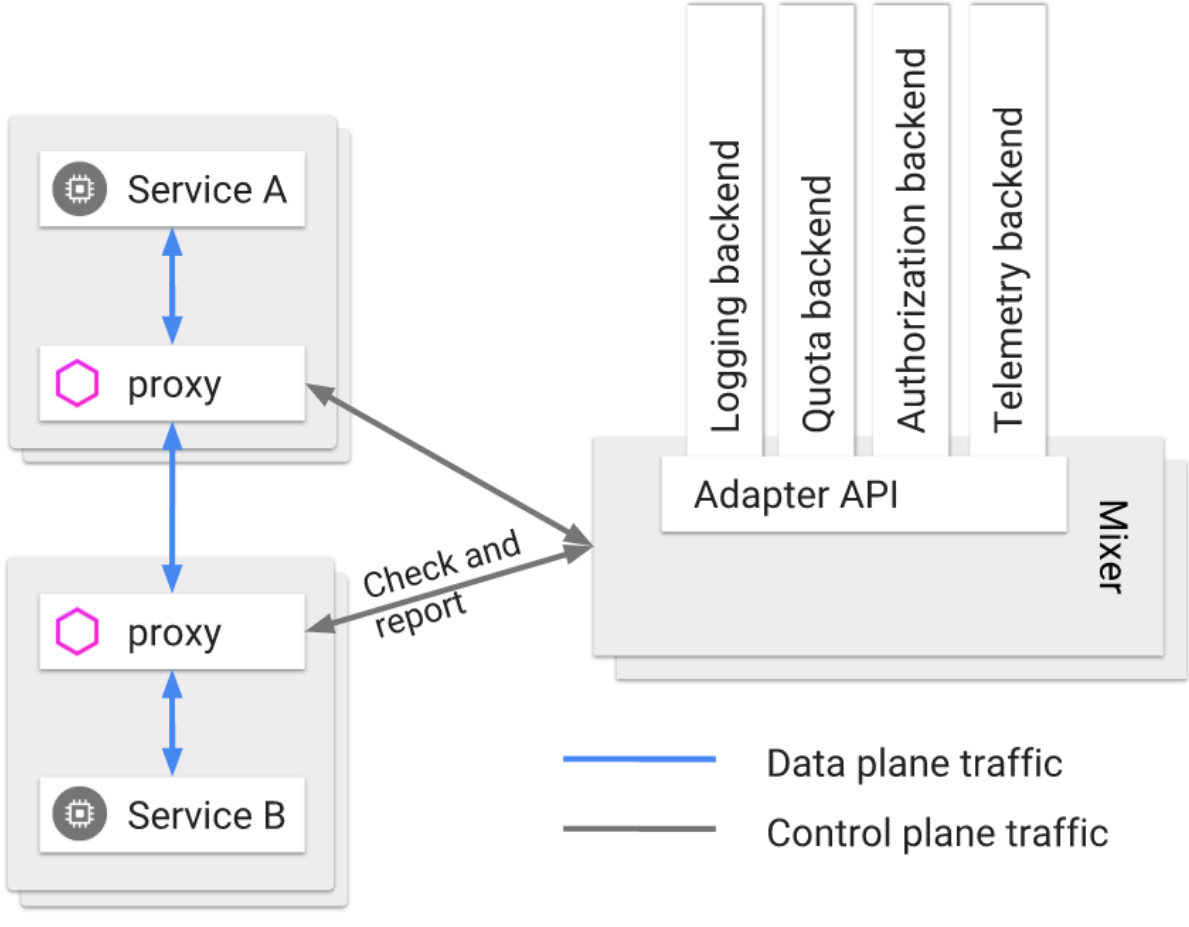
- string
- int64
- ip_address
- map[string, string]
- timestamp
- duration
- boolean

Control plane. Mixer



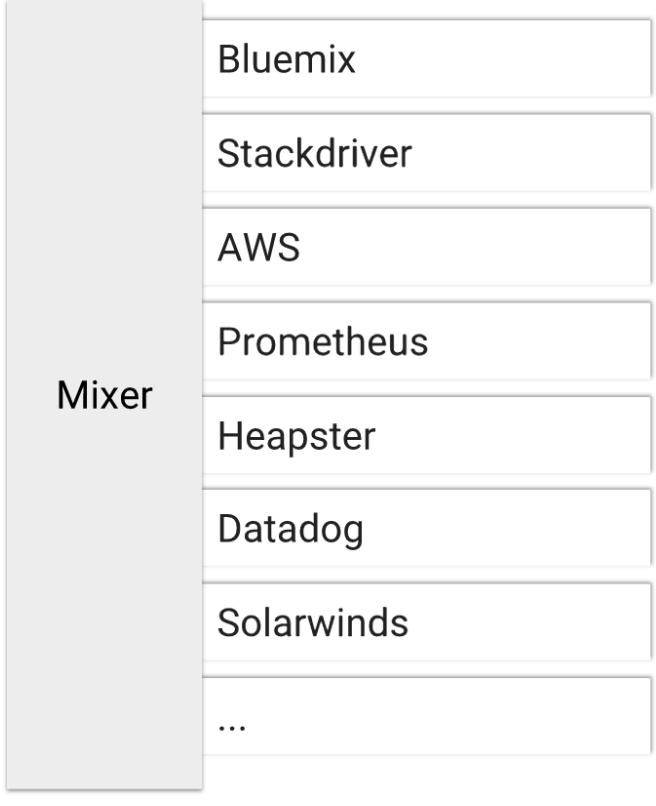
Attribute Machine

Mixer. Adapters



Mixer Topology

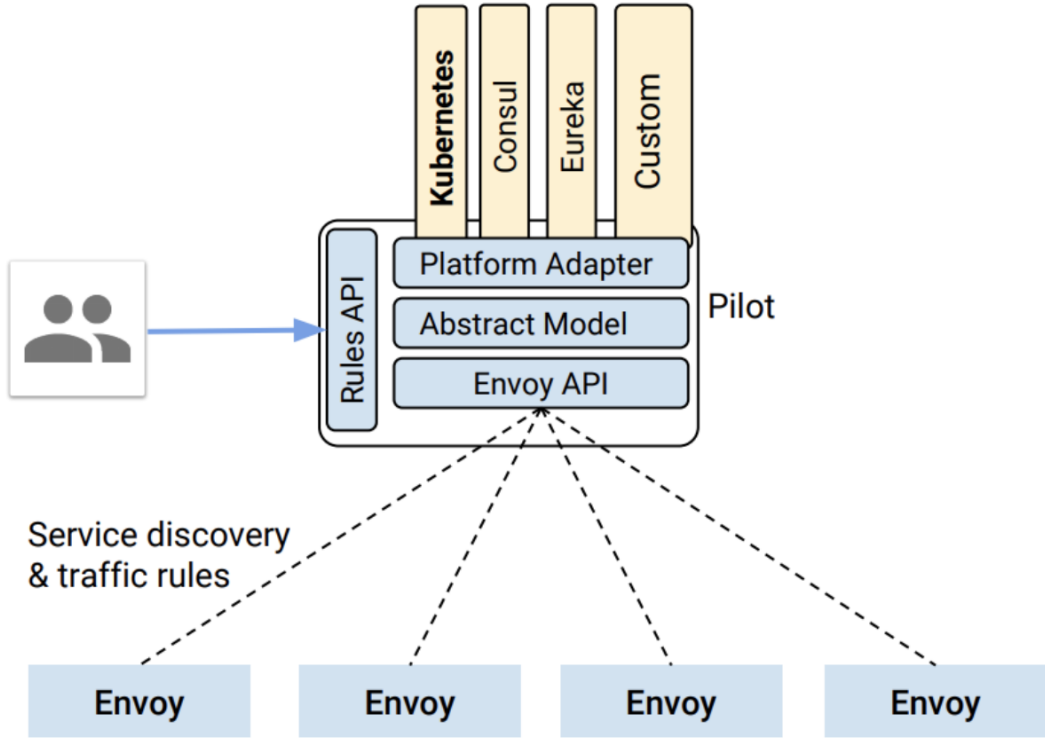
Mixer. Adapters



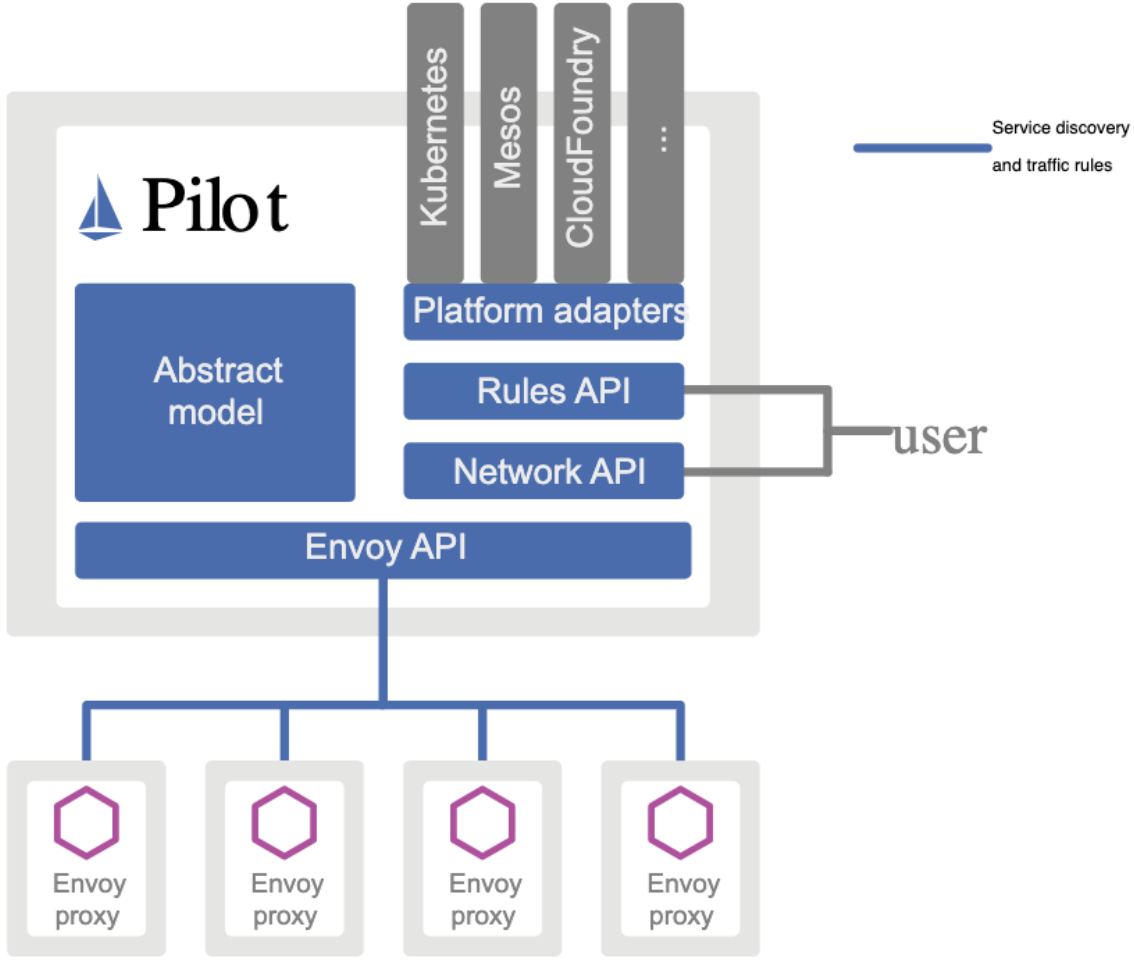
Поддерживаемые адаптеры

Control plane. Pilot

- Service discovery
- Load balancing
- Traffic routing and control
- Resiliency

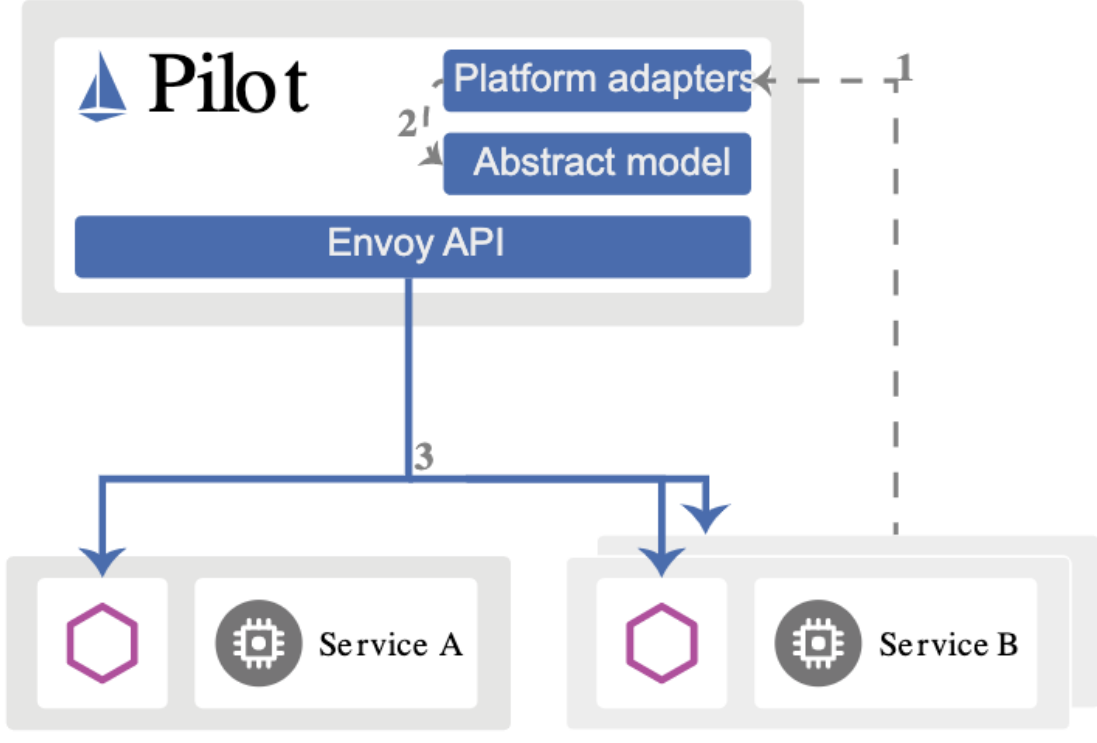


Pilot



Pilot architecture

Pilot



Service discovery

Virtual services

```yaml
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: reviews
spec:
  hosts:
  - reviews
  http:
  - match:
    - headers:
        end-user:
          exact: jason
    route:
    - destination:
        host: reviews
        subset: v2
  - route:
    - destination:
        host: reviews
        subset: v3
```

```yaml
spec:
  hosts:
  - reviews
  http:
  - route:
    - destination:
        host: reviews
        subset: v1
      weight: 75
    - destination:
        host: reviews
        subset: v2
      weight: 25
```

Timeouts

```
 1 apiVersion: networking.istio.io/v1alpha3
 2 kind: VirtualService
 3 metadata:
 4   name: ratings
 5 spec:
 6   hosts:
 7   - ratings
 8   http:
 9   - route:
10     - destination:
11         host: ratings
12         subset: v1
13     timeout: 10s
```

Retries

```
 1 apiVersion: networking.istio.io/v1alpha3
 2 kind: VirtualService
 3 metadata:
 4   name: ratings
 5 spec:
 6   hosts:
 7   - ratings
 8   http:
 9   - route:
10     - destination:
11         host: ratings
12         subset: v1
13     retries:
14       attempts: 3
15       perTryTimeout: 2s
```

Circuit breakers

```yaml
apiVersion: networking.istio.io/v1alpha3
kind: DestinationRule
metadata:
  name: reviews
spec:
  host: reviews
  subsets:
  - name: v1
    labels:
      version: v1
    trafficPolicy:
      connectionPool:
        tcp:
          maxConnections: 100
```

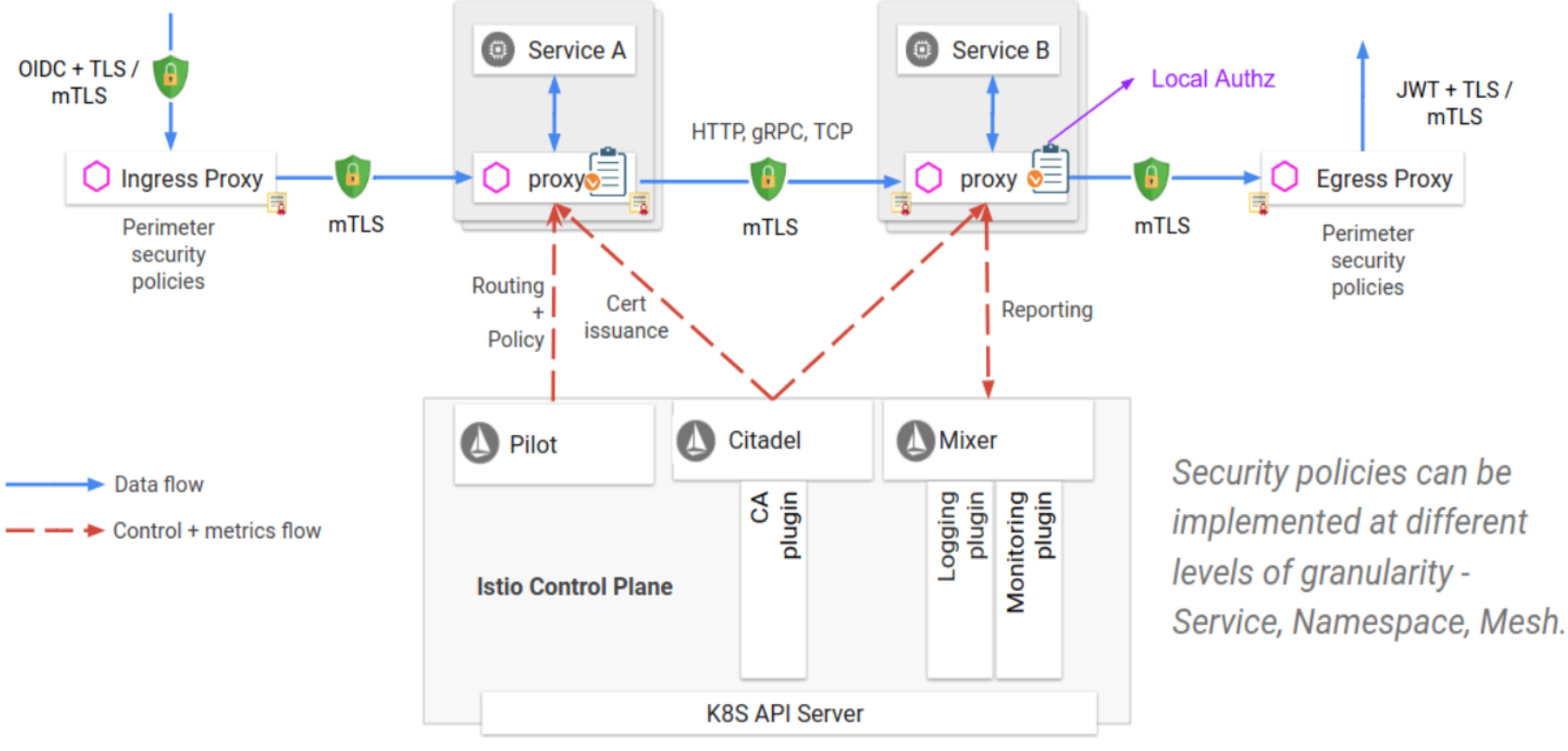
Fault injection

- **Delays:** Задержки - это ошибки синхронизации. Они имитируют увеличенную задержку в сети или перегруженный восходящий сервис.
- **Aborts:** Они имитируют сбои в вышестоящих сервисах. Прерывания обычно проявляются в виде кодов ошибок HTTP или сбоев соединения TCP.

Fault injection

```yaml
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: ratings
spec:
  hosts:
  - ratings
  http:
  - fault:
      delay:
        percentage:
          value: 0.1
        fixedDelay: 5s
    route:
    - destination:
        host: ratings
        subset: v1
```

Control plane. Citadel



Security policies can be implemented at different levels of granularity - Service, Namespace, Mesh.

Istio Security Architecture

Control plane. Citadel

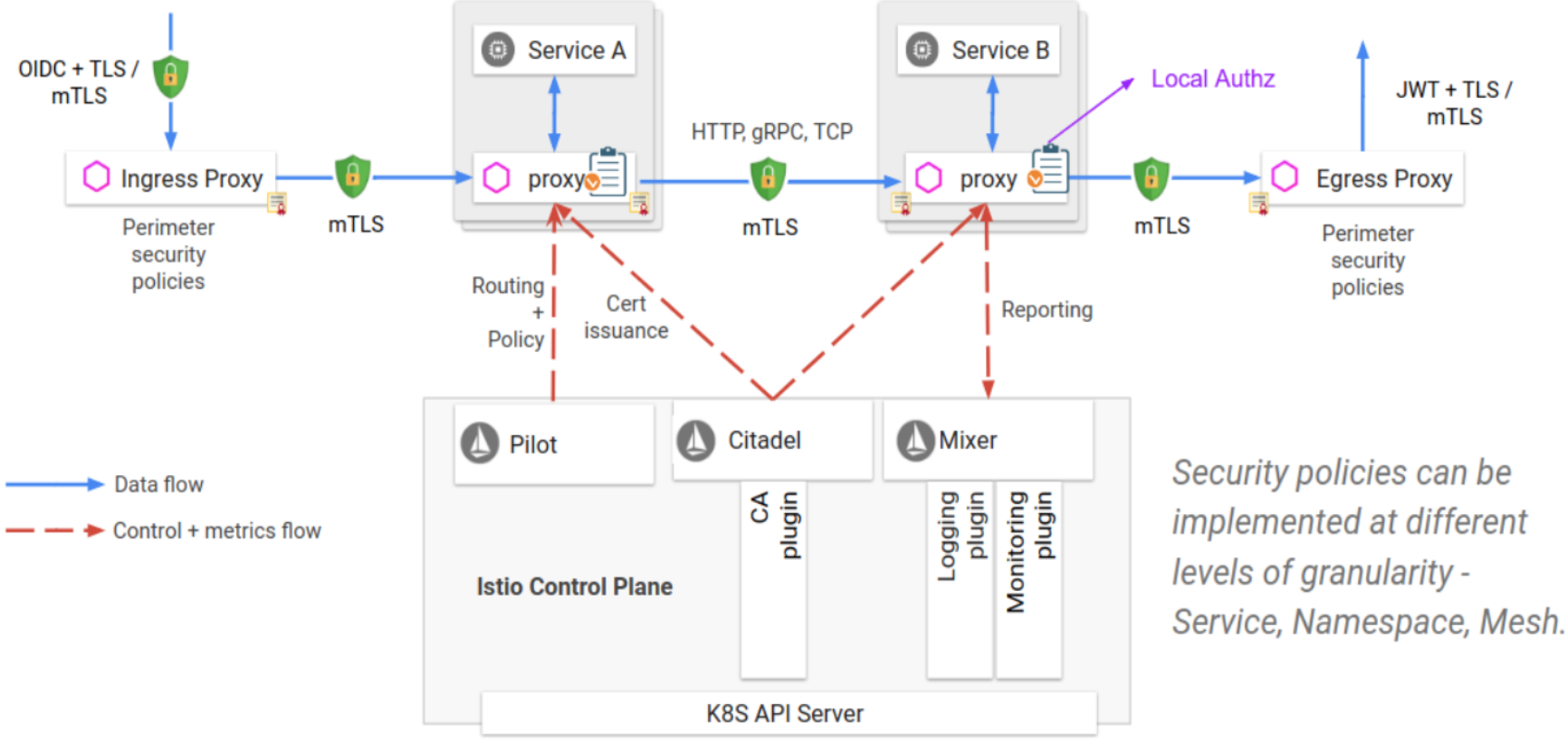
- ***Security by default***: не требуется никаких изменений для кода приложения и инфраструктуры
- ***Defense in depth***: интеграция с существующими системами безопасности для обеспечения нескольких уровней защиты
- ***Zero-trust network***: создание решений для безопасности в ненадежных сетях

Citadel. Policies

- Ограничение трафика к сервису
- Запреты, белые и черные списки, для ограничения доступов к сервисам
- Header rewrites and redirects

```
1 apiVersion: "authentication.istio.io/v1alpha1"
2 kind: "Policy"
3 metadata:
4   name: "reviews"
5 spec:
6   targets:
7   - name: reviews
8   peers:
9   - mtls: {}
```

Control plane. Citadel



Security policies can be implemented at different levels of granularity - Service, Namespace, Mesh.

Istio Security Architecture

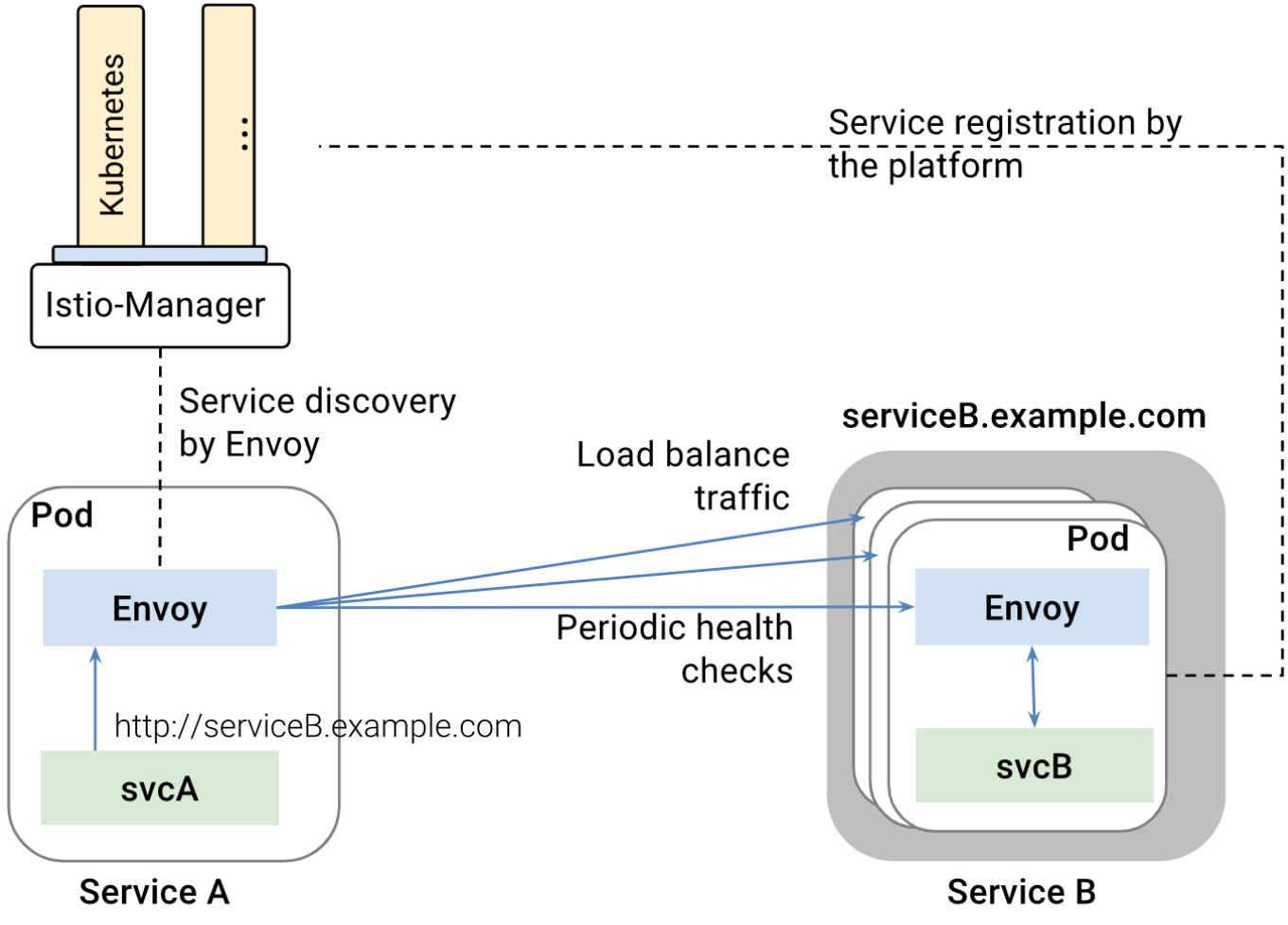
Istio system

```
$ kubectl get pods -n istio-system
NAME                                      READY   STATUS      RESTARTS   AGE
grafana-77b49c55db-bwrdd                  1/1     Running     0          79m
istio-citadel-7f7df4c54c-nvqhw            1/1     Running     0          79m
istio-cleanup-secrets-1.1.16-69txj        0/1     Completed   0          79m
istio-egressgateway-859c9669f6-j2sh6      1/1     Running     0          79m
istio-galley-75c8db7664-9tlhx             1/1     Running     0          79m
istio-grafana-post-install-1.1.16-wn7zm   0/1     Completed   0          79m
istio-ingressgateway-6f4f495fb9-rs8rh     1/1     Running     0          79m
istio-pilot-59b88875c7-b7nf4              2/2     Running     0          79m
istio-policy-5776cffdd5-vv2gj             2/2     Running     2          79m
istio-security-post-install-1.1.16-w26mk  0/1     Completed   0          79m
istio-sidecar-injector-769fff8b78-scxmh   1/1     Running     0          79m
istio-telemetry-7b7fbf56cc-jqpzk          2/2     Running     2          79m
istio-tracing-595796cf54-ctggt            1/1     Running     0          79m
kiali-5c584d45f6-5jxbn                    1/1     Running     0          79m
prometheus-5fffdf8848-ntrkx               1/1     Running     0          79m
```

Основные возможности Istio

Traffic control

Load balancing



Discovery and Load Balancing

Load balancing

- Random: Requests are forwarded at random to instances in the pool.
- Weighted: Requests are forwarded to instances in the pool according to a specific percentage.
- Least requests: Requests are forwarded to instances with the least number of requests.

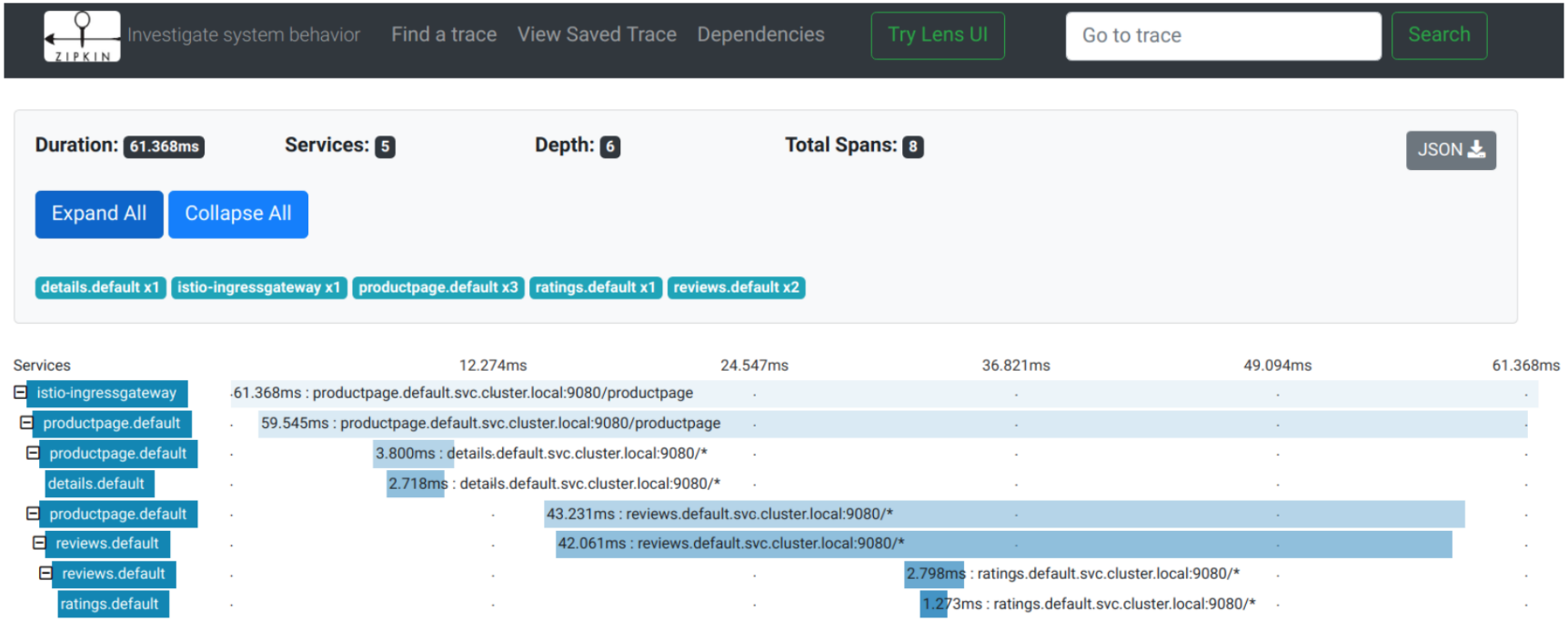
Envoy load balancing documentation

Load balancing. Example

```yaml
apiVersion: networking.istio.io/v1alpha3
kind: DestinationRule
metadata:
  name: my-destination-rule
spec:
  host: my-svc
  trafficPolicy:
    loadBalancer:
      simple: RANDOM
  subsets:
  - name: v1
    labels:
      version: v1
  - name: v2
    labels:
      version: v2
    trafficPolicy:
      loadBalancer:
        simple: ROUND_ROBIN
  - name: v3
    labels:
      version: v3
```

Observability

Tracing



Distributed Trace for a single request

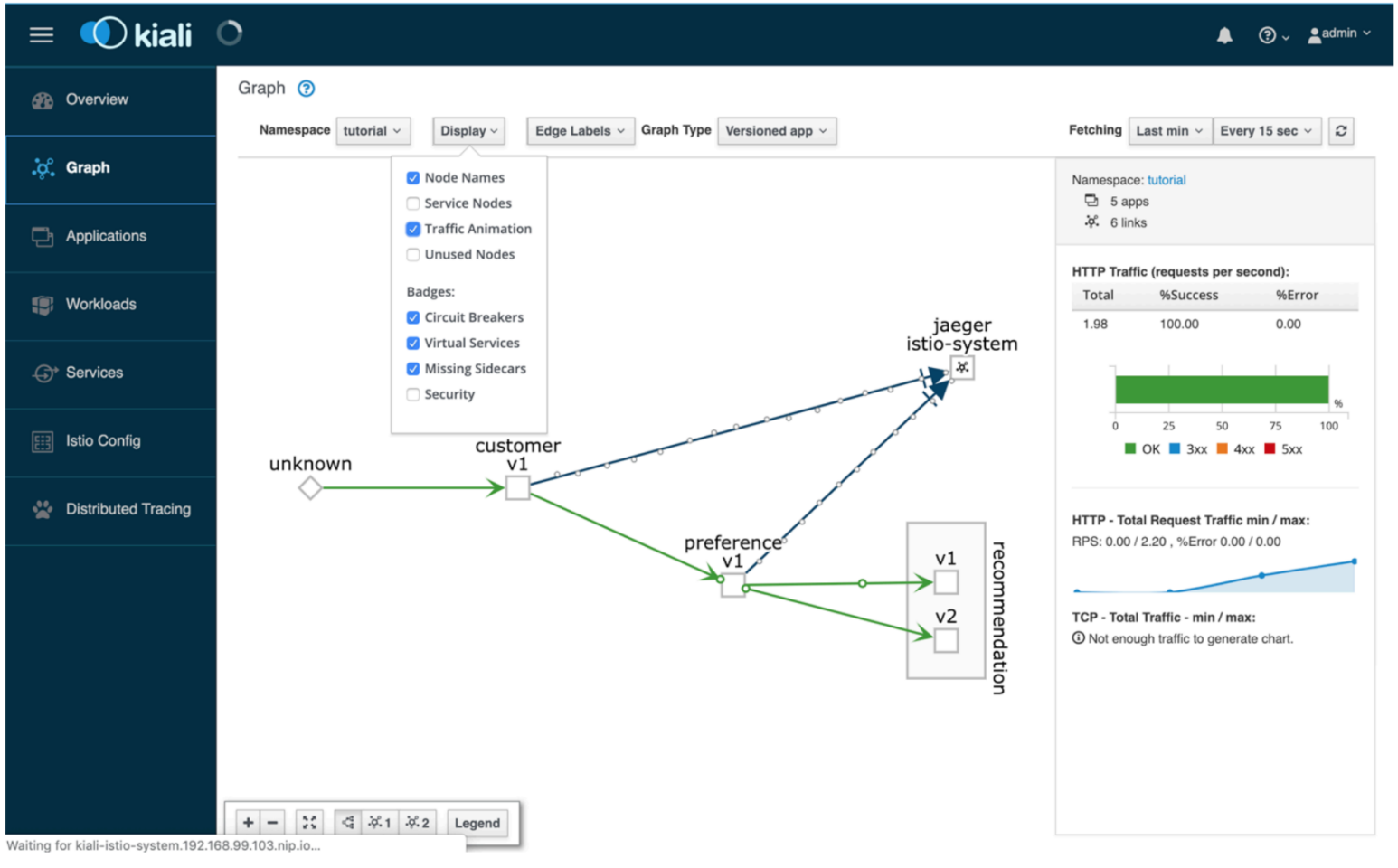
Proxy-level Metrics

```
envoy_cluster_internal_upstream_rq{response_code_class="2xx",cluster_name="xds-grpc"} 7163

envoy_cluster_upstream_rq_completed{cluster_name="xds-grpc"} 7164

envoy_cluster_ssl_connection_error{cluster_name="xds-grpc"} 0

envoy_cluster_lb_subsets_removed{cluster_name="xds-grpc"} 0

envoy_cluster_internal_upstream_rq{response_code="503",cluster_name="xds-grpc"} 1
```

Service-level Metrics

```
istio_requests_total{
  connection_security_policy="mutual_tls",
  destination_app="details",
  destination_principal="cluster.local/ns/default/sa/default",
  destination_service="details.default.svc.cluster.local",
  destination_service_name="details",
  destination_service_namespace="default",
  destination_version="v1",
  destination_workload="details-v1",
  destination_workload_namespace="default",
  reporter="destination",
  request_protocol="http",
  response_code="200",
  response_flags="-",
  source_app="productpage",
  source_principal="cluster.local/ns/default/sa/default",
  source_version="v1",
  source_workload="productpage-v1",
  source_workload_namespace="default"
} 214
```

Servicegraph



Load Testing

Сеть Istio 1.3.2 состоит из 1000 сервисов, 2000 sidecars. После запуска тестов были получены следующие результаты при 70,000 запросов в секунду:

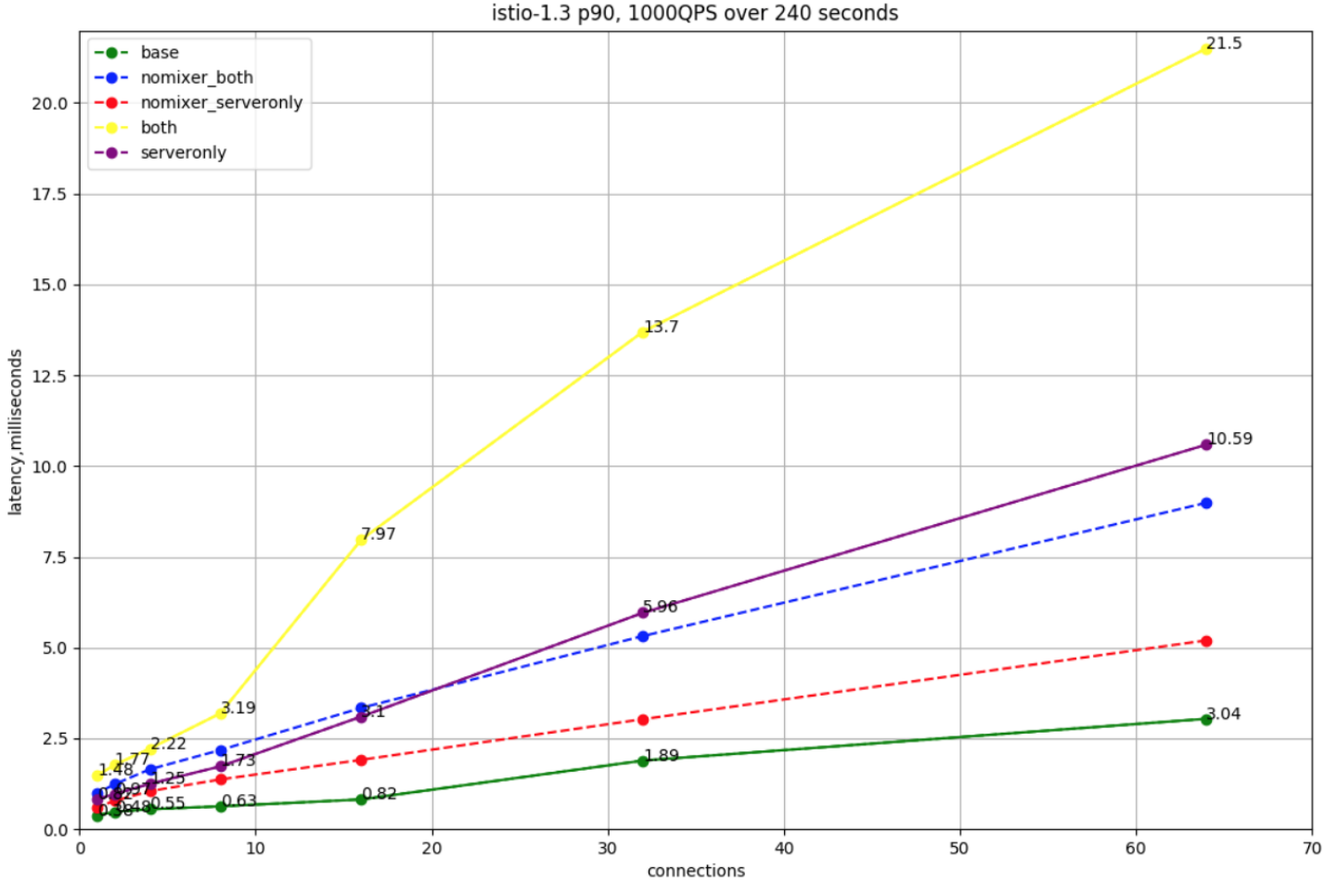
- Envoy proxy: 0.6 vCPU и 50 MB memory per 1000 requests per second going through the proxy.
- Istio-telemetry service: 0.6 vCPU per 1000 mesh-wide requests per second.
- Pilot: 1 vCPU and 1.5 GB of memory.
- Envoy proxy adds 8ms to the 90th percentile latency.

Load Testing

Производительность Data Plane зависит от многих факторов, например:

- Число клиентских подключений
- Частота запросов
- Request size and Response size
- Number of proxy worker threads
- Protocol
- CPU cores
- Number and types of proxy filters, specifically Mixer filter.
- The latency, throughput, and the proxies' CPU and memory consumption are measured as a function of said factors.

Latency

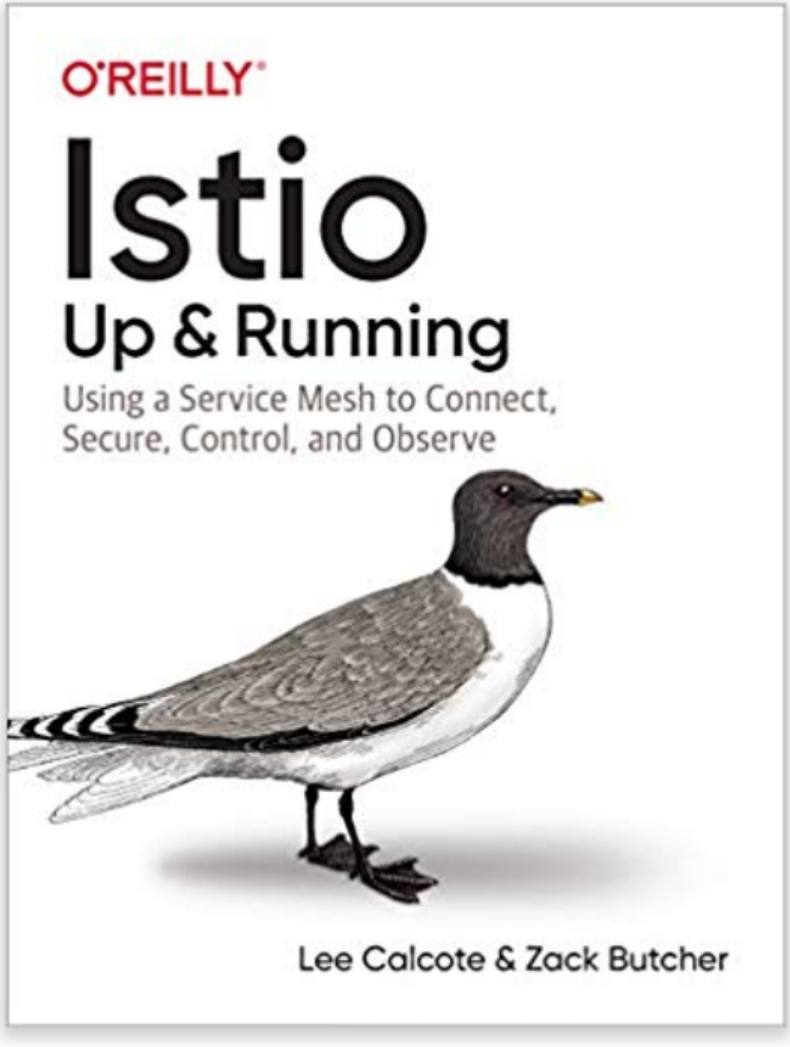


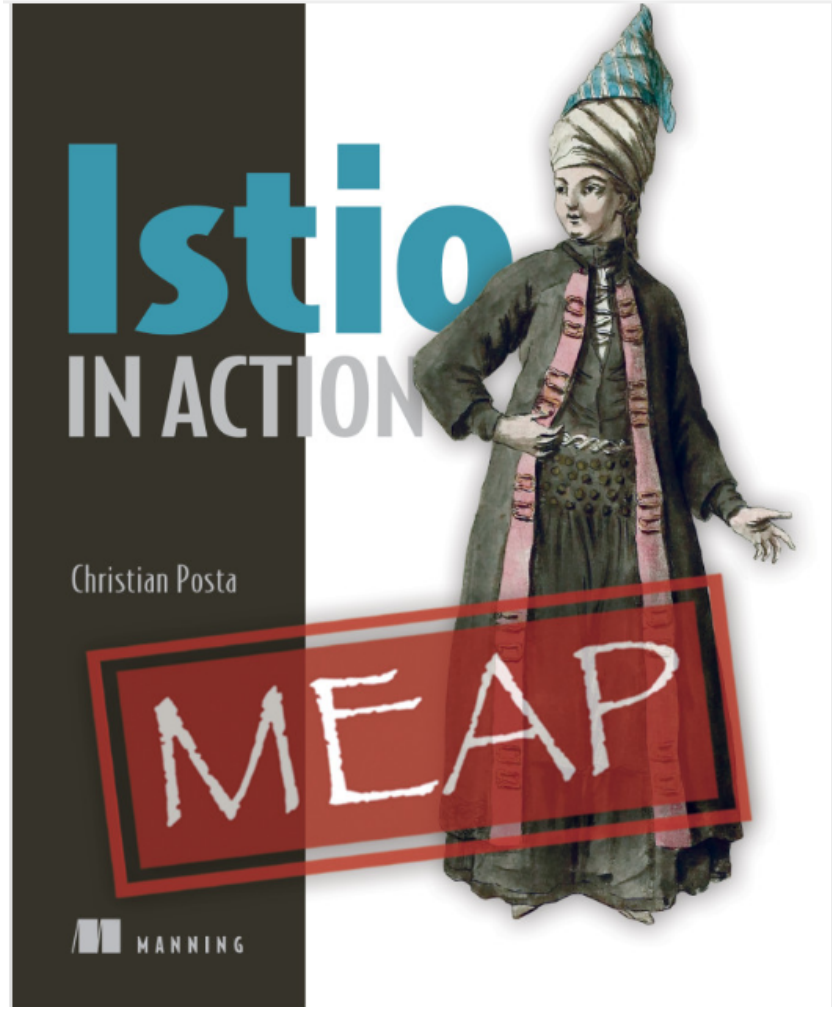
P90 latency vs client connections

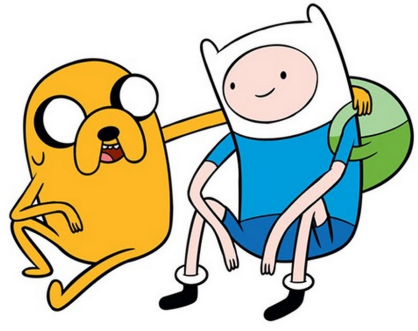
Ссылки

- Service mesh для микросервисов
- Блог из 12 небольших статей про Istio
- Ликбез по запуску Istio
- Benchmarking Istio & Linkerd CPU at Scale

Ссылки


O'REILLY
Istio
Up & Running
Using a Service Mesh to Connect, Secure, Control, and Observe
Lee Calcote & Zack Butcher


Istio
IN ACTION
Christian Posta
MEAP
MANNING



Спасибо за внимание!

Время для ваших вопросов!