DEDICATED INFRASTRUCTURE IN A MULTITENANT WORLD

Adobe

Carlos Sanchez / csanchez.org / @csanchez

Cloud Engineer

Adobe Experience Manager Cloud Service

Author of Jenkins Kubernetes plugin

Long time OSS contributor at Jenkins, Apache Maven, Puppet,…

ADOBE EXPERIENCE MANAGER

Content Management System

Digital Asset Management

Digital Enrollment and Forms

Used by many Fortune 100 companies

An existing distributed Java OSGi application

Using OSS components from Apache Software Foundation

A huge market of extension developers

AEM ON KUBERNETES

Running on Azure

18+ clusters and growing

Multiple regions: US, Europe, Australia, Singapore, Japan, more coming

Adobe has a dedicated team managing clusters for multiple products

Customers can run their own code

Cluster permissions are limited for security

Traffic leaving the clusters must be encrypted

Using namespaces to provide a scope

- network isolation
- quotas
- permissions

More details on our Kubernetes setup in my KubeCon 2020 talk

https://tinyurl.com/csanchez-kcna20

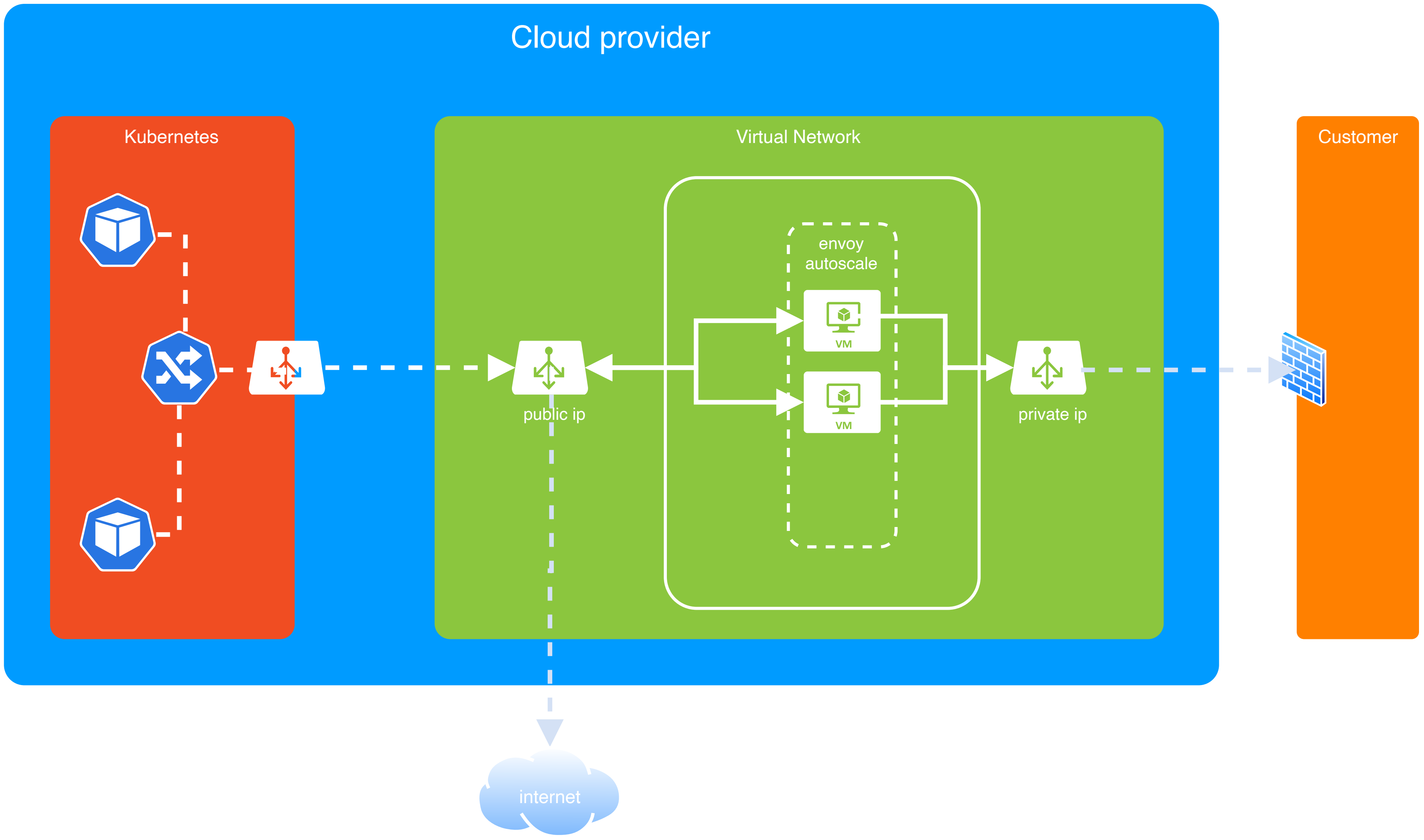
DEDICATED INFRASTRUCTURE

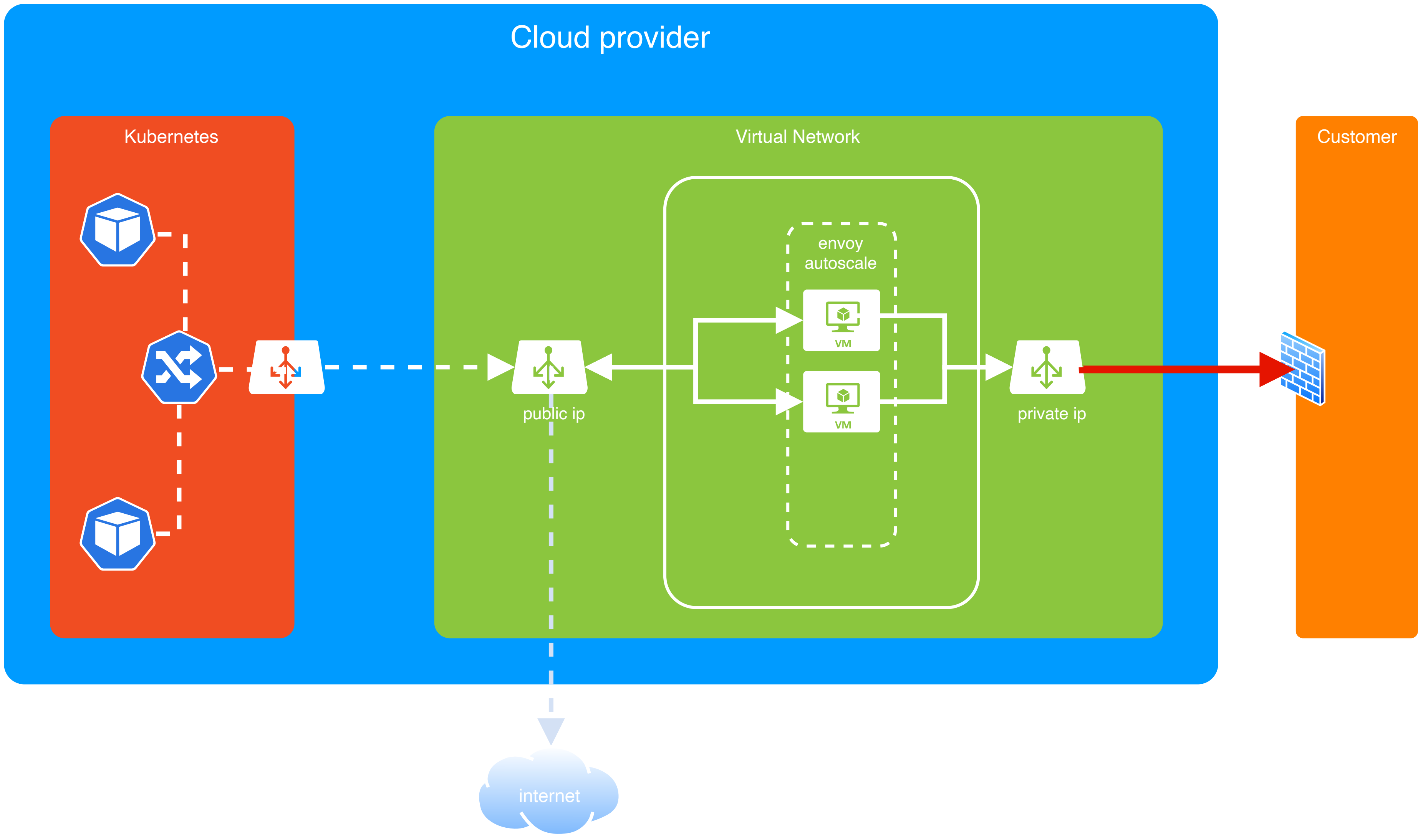
Customers want to have dedicated infrastructure

- Egress IPs
- Private connections (VNET peering, Private Link, ExpressRoute, Direct Connect,...)
- VPN

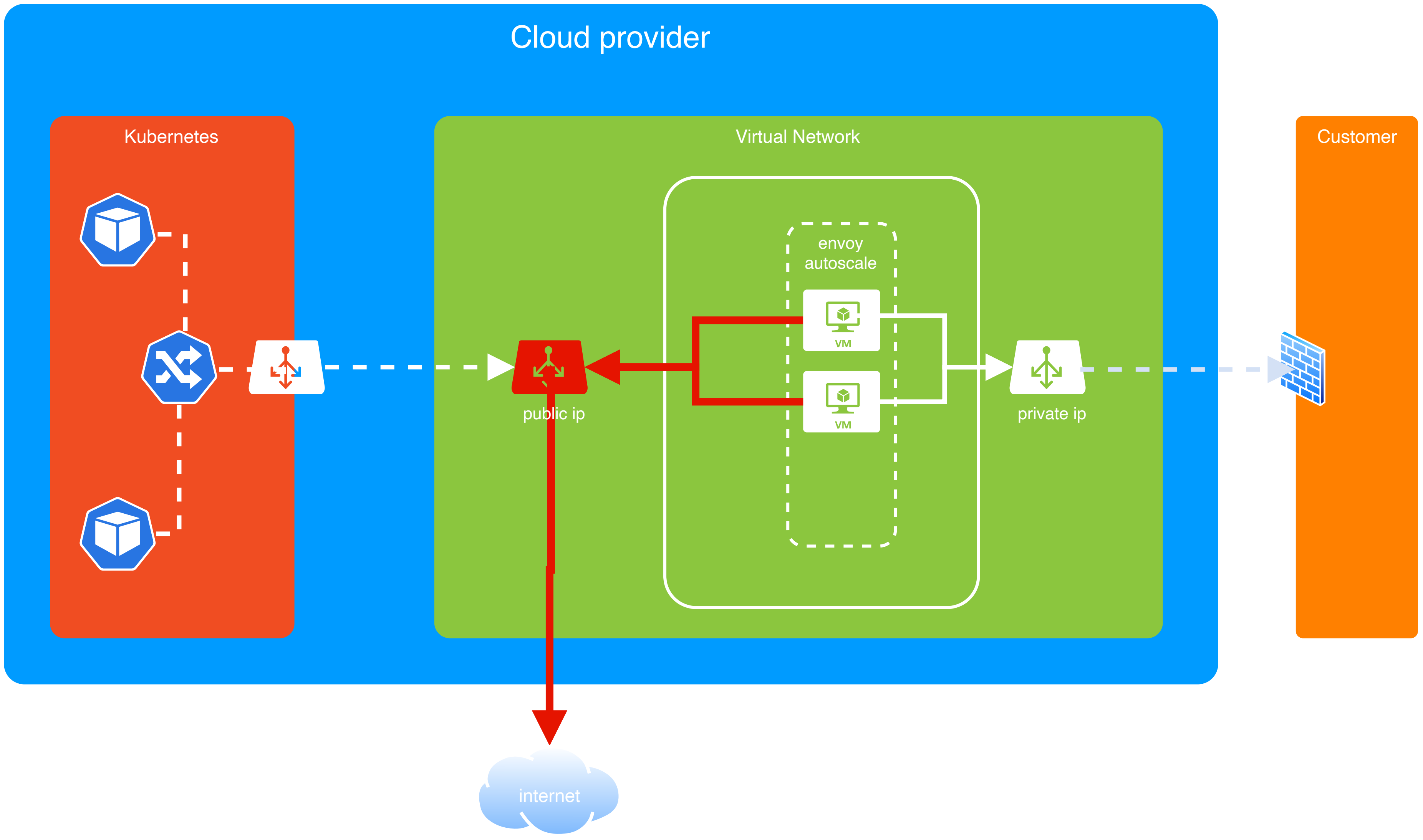
ENVOY

Running Envoy on VMs and pod sidecars in Kubernetes

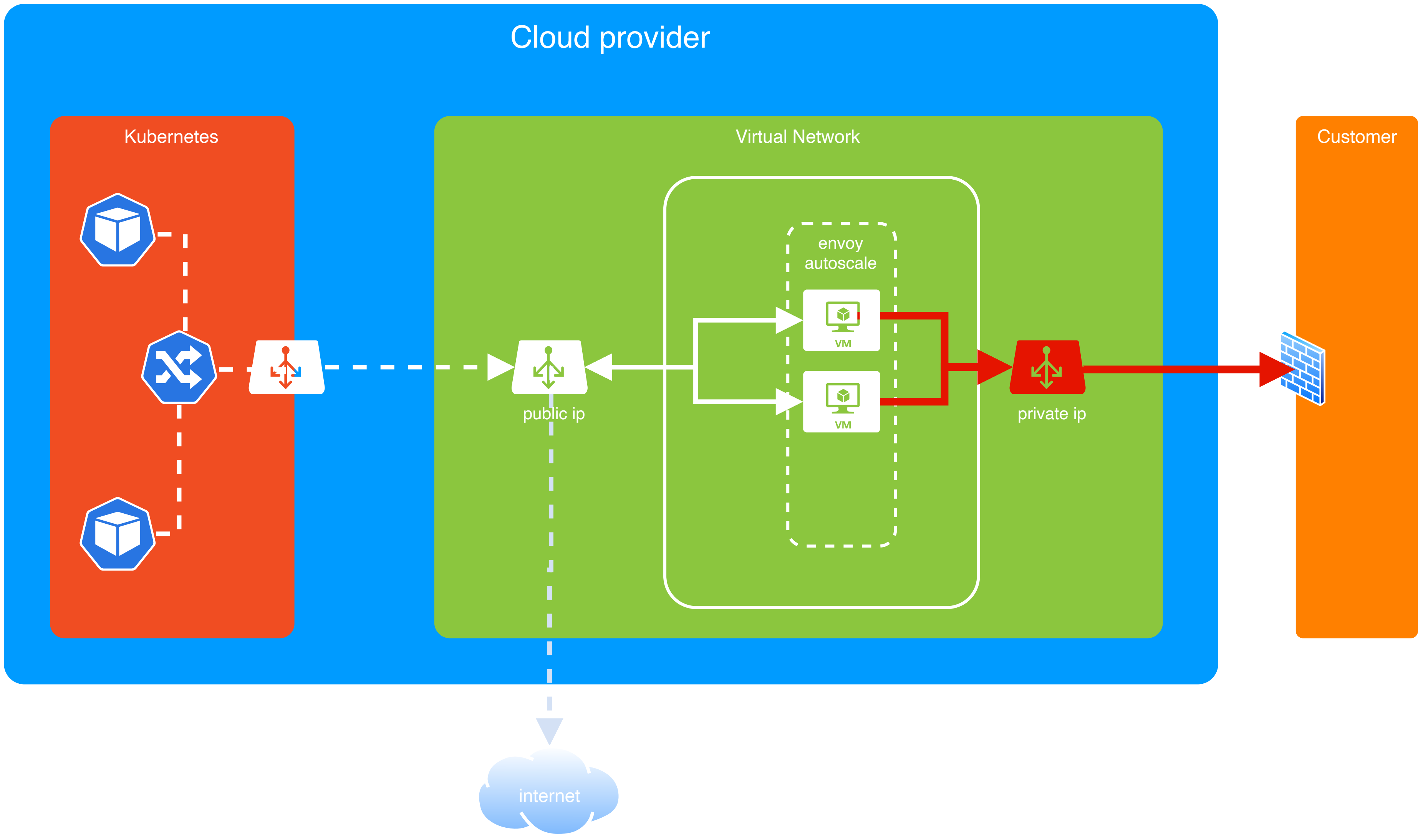

Cloud provider
Kubernetes
Virtual Network
Customer
envoy
autoscale
VM
VM
public ip
private ip
internet



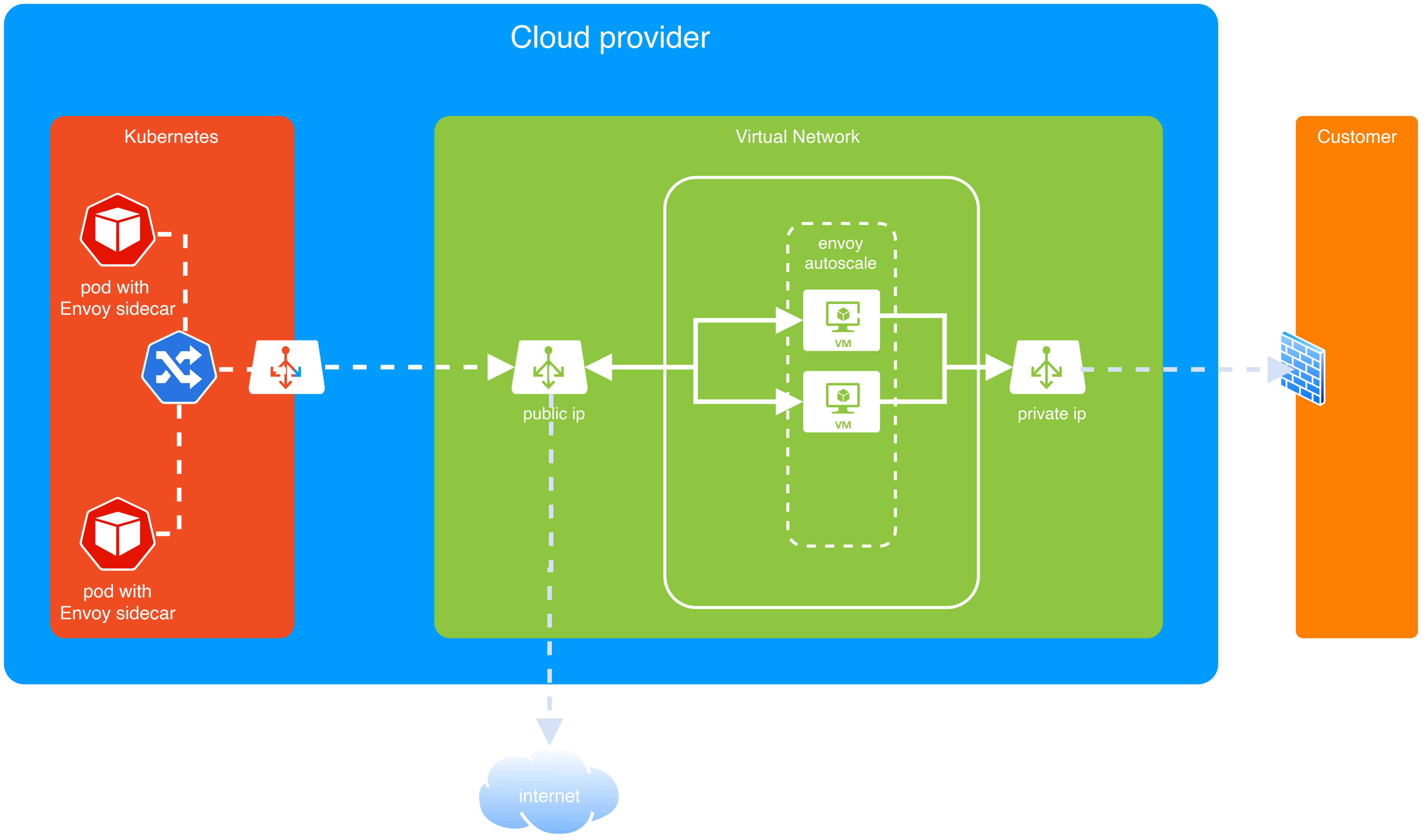
VNET can be privately connected to customer network



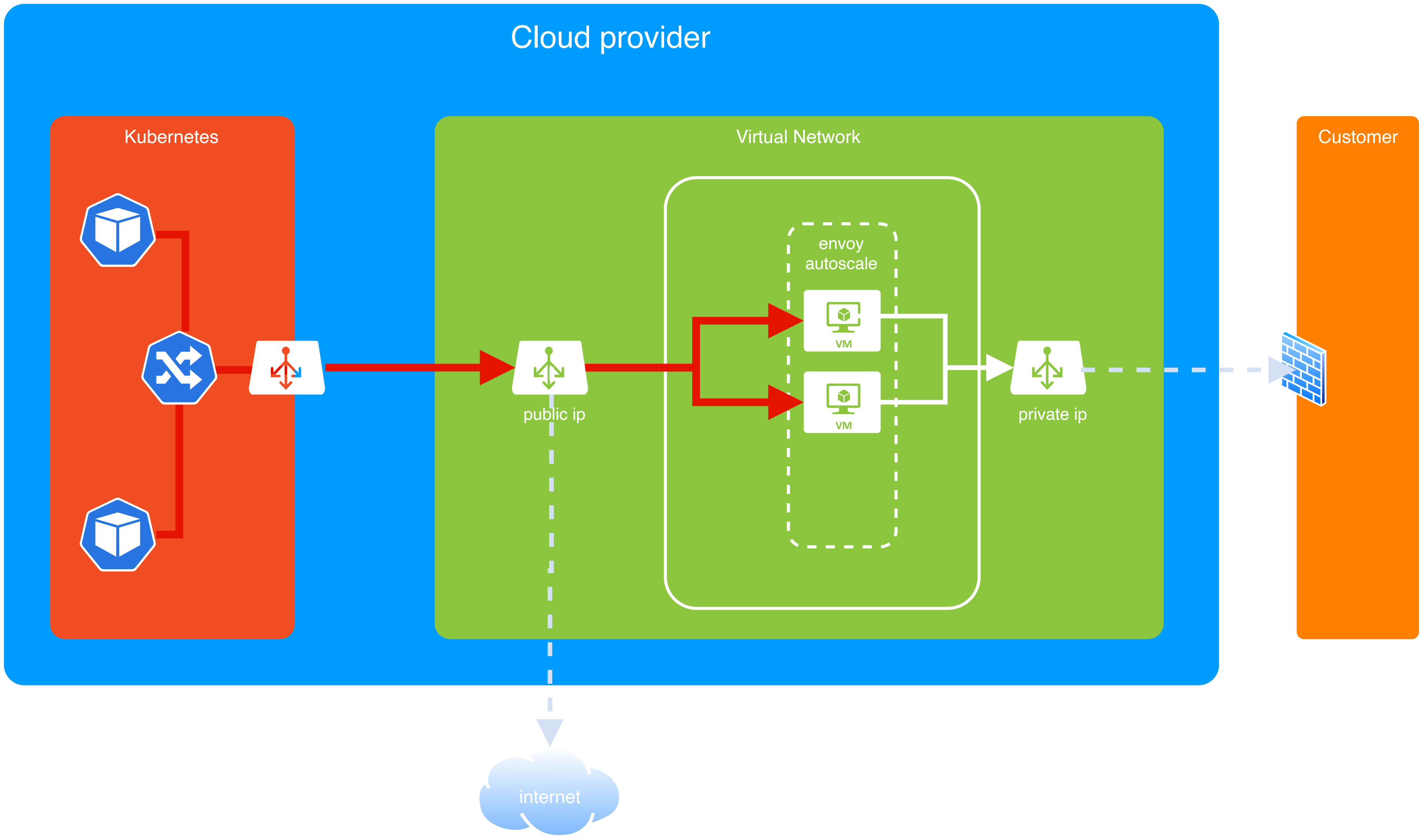
Load Balancer in front of VMs gives a dedicated public egress ip



Private Load Balancer gives a dedicated private egress ip



JVM configured with Envoy sidecar as HTTP proxy



HTTP2 tunnel between sidecar Envoy and VM Envoy with mTLS

ENVOY: PROS

Simple and transparent config in JVM using http proxy system properties

Any protocol supported using different listeners in sidecar

All traffic is encrypted

ENVOY: PROS

VNET allows configuration of VPN, private connections at cloud level as a service

mTLS prevents unauthorized connections and one tenant to connect to another tenant Envoy

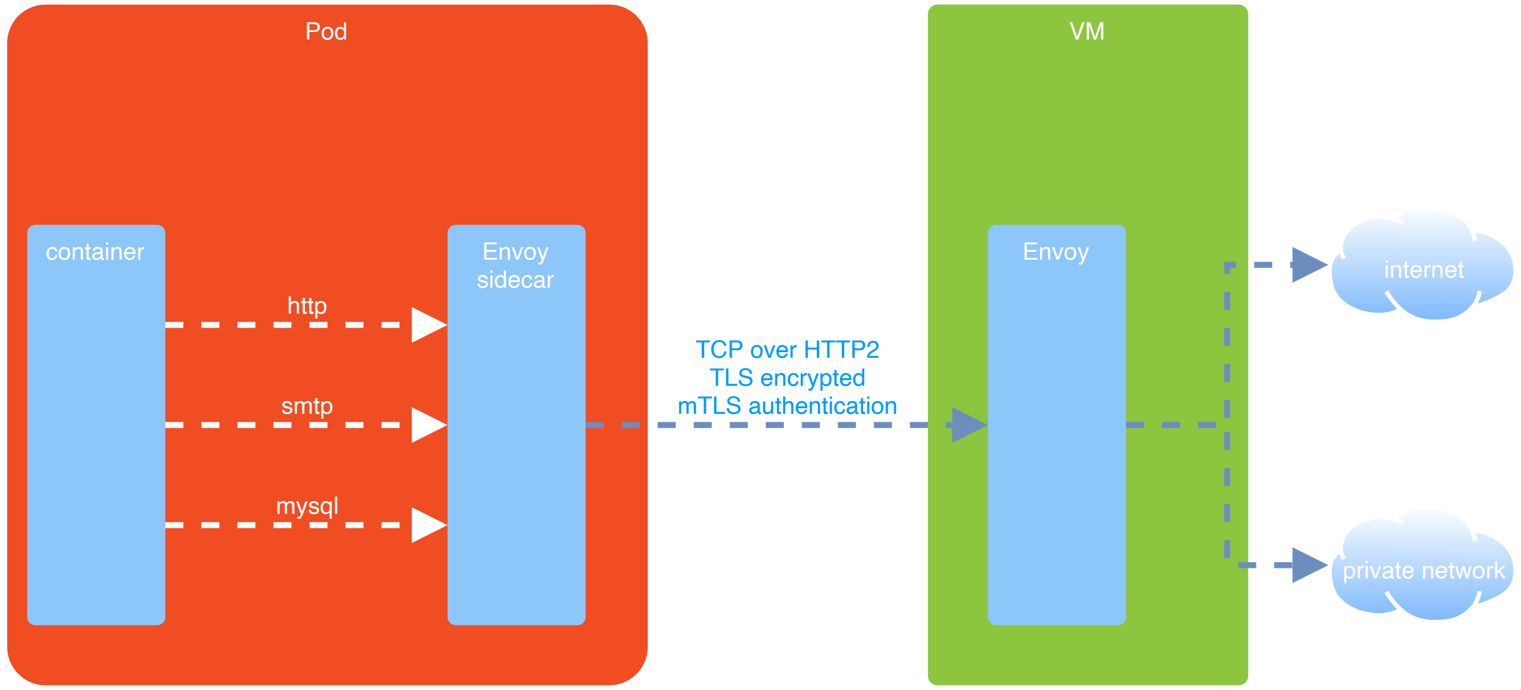
ENVOY: CONS

Needs one set of certificates for each tenant for sidecars and VMs

rotation, expiration,...

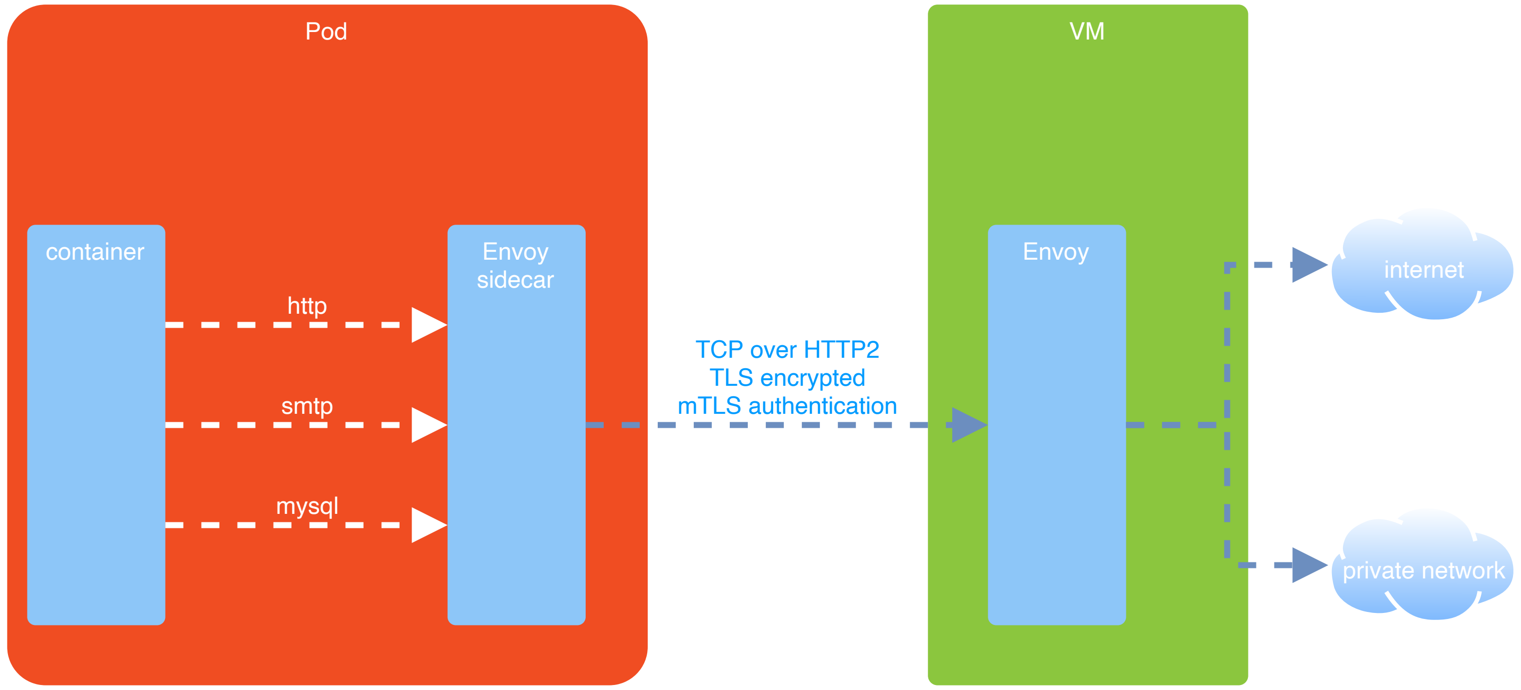
ENVOY SIDECAR

ENVOY SIDECAR



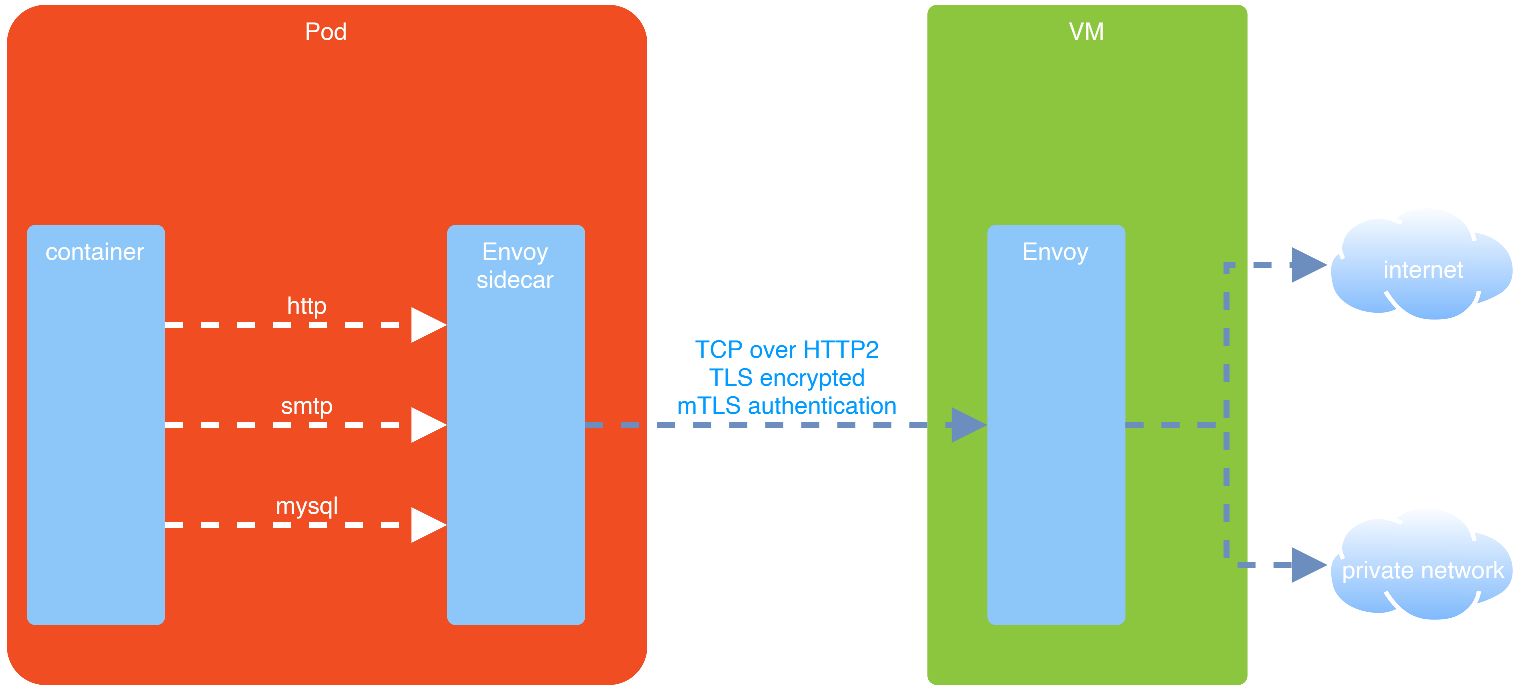
One listener with TcpProxy filter for http/s. `HTTP CONNECT` gives Envoy the destination

ENVOY SIDECAR



One listener for each non http port. Destination hardcoded in `tunneling_config`

ENVOY SIDECAR



One cluster with the VM Envoy LB as endpoint and TLS `transport_socket` config

ENVOY SIDECAR: HTTP LISTENER

```
- name: listener_0
  "@type": type.googleapis.com/envoy.config.
  listener.v3.Listener
  address:
    socket_address:
      protocol: TCP
      address: 0.0.0.0
      port_value: 3128
  filter_chains:
  - filters:
    - name: tcp
      typed_config:
        "@type": type.googleapis.com/
        envoy.extensions
        .filters.network.tcp_proxy.v3.TcpProxy
        stat_prefix: tcp_stats
        cluster: cluster_0
```

ENVOY SIDECAR: NON HTTP LISTENER

```
- filters:
  - name: tcp
    typed_config:
      "@type": type.googleapis.com/envoy.extensions.
      filters.network.tcp_proxy.v3.TcpProxy
      stat_prefix: tcp_stats
      cluster: cluster_0
      tunneling_config:
        hostname: mysql:3306
```

ENVOY SIDECAR: CLUSTER

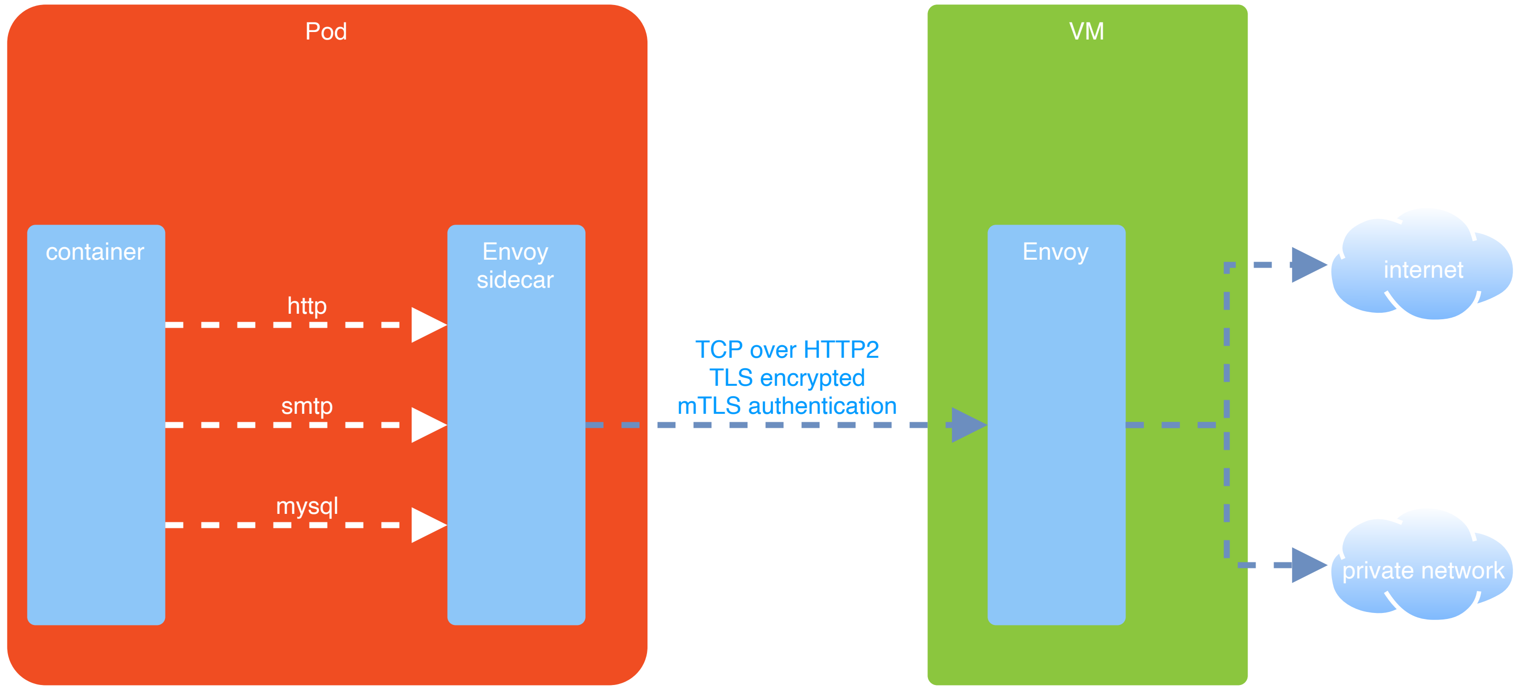
```yaml
- name: cluster_0
  "@type": type.googleapis.com/envoy.
  config.cluster.v3.Cluster
  type: logical_dns
  typed_extension_protocol_options:
    envoy.extensions.upstreams.http.v3.
    HttpProtocolOptions:
      "@type": type.googleapis.com/envoy.extensions.
      upstreams.http.v3.HttpProtocolOptions
  respect_dns_ttl: true
  load_assignment:
    cluster_name: cluster_0
    endpoints:
      - lb_endpoints:
          - endpoint:
              address:
                socket_address:
                  address: envoy_vm
                  port_value: 443
```

ENVOY SIDECAR: CLUSTER

```yaml
  transport_socket:
    name: envoy.transport_sockets.tls
    typed_config:
      "@type": type.googleapis.com/envoy.extensions.
      transport_sockets.tls.v3.UpstreamTlsContext
      common_tls_context:
        tls_certificate_sds_secret_configs:
          - name: upstream_cert
            sds_config:
              path: /var/lib/envoy/sds.yaml
        tls_params:
          tls_minimum_protocol_version: TLSv1_2
        validation_context_sds_secret_config:
          name: upstream_cert_ca
          sds_config:
            path: /var/lib/envoy/sds-ca.yaml
```

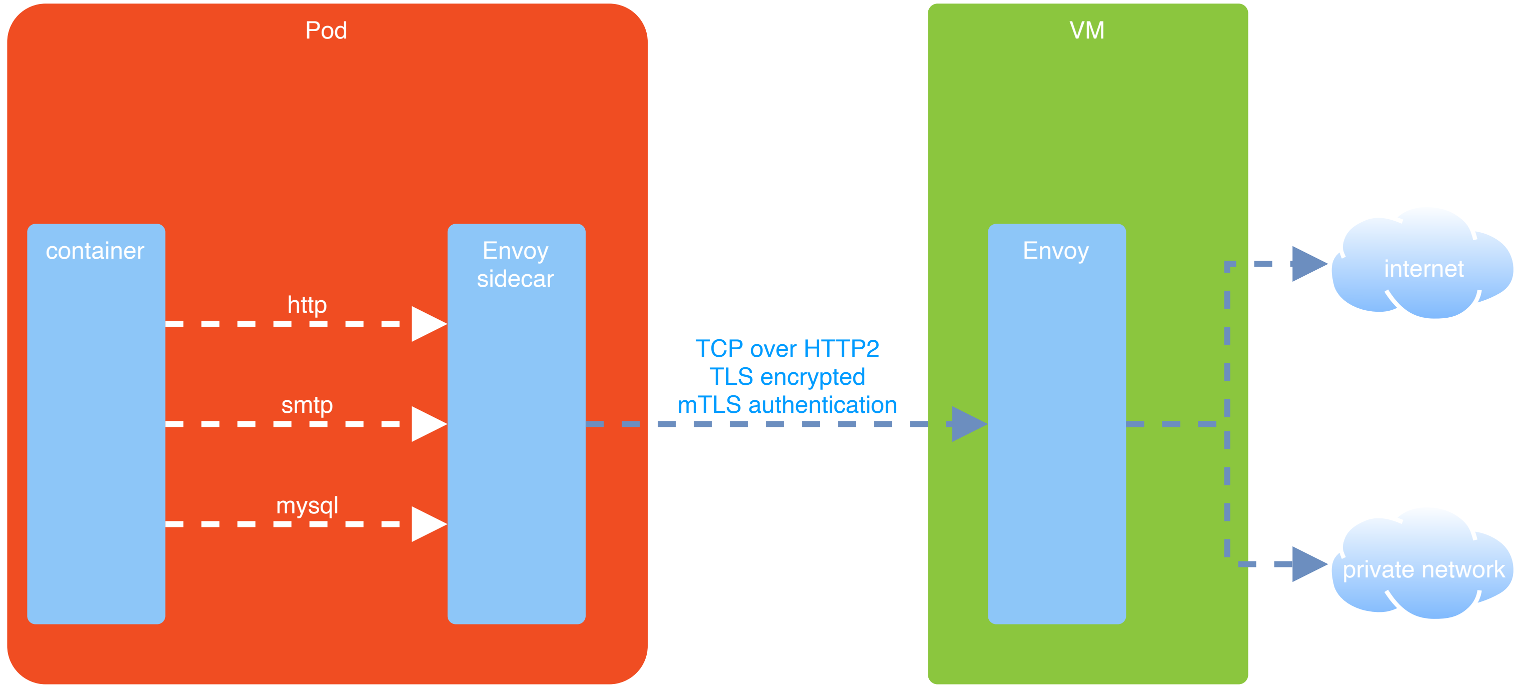
ENVOY VM

ENVOY IN VM



One `HttpConnectionManager` listener with `CONNECT` upgrade

ENVOY IN VM



One `dynamic_forward_proxy` cluster for all destinations

ENVOY IN VM: LISTENER

```yaml
- name: listener_0
  "@type": type.googleapis.com/envoy.config.listener.v3.Listener
  address:
    socket_address:
      protocol: TCP
      address: 0.0.0.0
      port_value: 443
  filter_chains:
  - filters:
    - name: envoy.filters.network.http_connection_manager
      typed_config:
        "@type": type.googleapis.com/envoy.extensions
        .filters.network.http_connection_manager.v3.
        HttpConnectionManager
```

ENVOY IN VM: LISTENER

```
route_config:
  name: local_route
  virtual_hosts:
  - name: local_service
    domains: ["*"]
    routes:
      - match:
          connect_matcher: {}
        route:
          cluster: dynamic_forward_proxy_cluster
          upgrade_configs:
            - upgrade_type: CONNECT
              connect_config: {}
      # needed to be used as a proxy with http (not s)
      - match:
          prefix: "/"
        route:
          cluster: dynamic_forward_proxy_cluster
```

ENVOY IN VM: LISTENER

```
http_filters:
- name: envoy.filters.http.dynamic_forward_proxy
  typed_config:
    "@type": type.googleapis.com/envoy.extensions.
    filters.http.dynamic_forward_proxy.v3.FilterConfig
    dns_cache_config:
      name: dynamic_forward_proxy_cache_config
      dns_lookup_family: V4_ONLY
- name: envoy.filters.http.router
  typed_config:
    "@type": type.googleapis.com/envoy.extensions.
    filters.http.router.v3.Router
upgrade_configs:
  - upgrade_type: CONNECT
```

ENVOY IN VM: LISTENER

```
transport_socket:
  name: envoy.transport_sockets.tls
  typed_config:
    "@type": type.googleapis.com/envoy.extensions.
    transport_sockets.tls.v3.DownstreamTlsContext
    common_tls_context:
      tls_certificate_sds_secret_configs:
        - name: upstream_cert
          sds_config:
            path: /var/lib/envoy/sds.yaml
      tls_params:
        tls_minimum_protocol_version: TLSv1_2
      validation_context_sds_secret_config:
        name: upstream_cert_ca
        sds_config:
          path: /var/lib/envoy/sds-ca.yaml
    require_client_certificate: true
```

ENVOY IN VM: CLUSTER

```
- name: dynamic_forward_proxy_cluster
  "@type": type.googleapis.com/envoy.config.cluster.
  v3.Cluster
  connect_timeout: 30s
  lb_policy: CLUSTER_PROVIDED
  cluster_type:
    name: envoy.clusters.dynamic_forward_proxy
    typed_config:
      "@type": type.googleapis.com/envoy.extensions.
      clusters.dynamic_forward_proxy.v3.ClusterConfig
      dns_cache_config:
        name: dynamic_forward_proxy_cache_config
        dns_lookup_family: V4_ONLY
```

ENVOY IN VM: SDS

```
resources:
  - "@type": type.googleapis.com/envoy.extensions.
    transport_sockets.tls.v3.Secret
    name: upstream_cert_ca
    validation_context:
      trusted_ca:
        filename: /etc/envoy/certs/cacert.pem
      # only allow connections with certificates
      # that have this SAN
      match_subject_alt_names:
        - exact: "envoy_sidecar"
```

CERTIFICATE ROTATION

CERTIFICATE ROTATION

Using Vault Agent to generate short lived certificates (sidecar and VM)

CA stored in Vault

Kubernetes and Azure passwordless authentication to Vault

ENVOY SIDECAR: SDS

Configuring certificates as SDS in separate files to ensure certificates are automatically reloaded on change.

```
resources:
  - "@type": type.googleapis.com/envoy.extensions.
    transport_sockets.tls.v3.Secret
    name: upstream_cert
    tls_certificate:
      certificate_chain:
        filename: "/etc/envoy/certs/tls.crt"
      private_key:
        filename: "/etc/envoy/certs/tls.key"
```

ENVOY SIDECAR: SDS-CA

```
resources:
  - "@type": type.googleapis.com/envoy.extensions.
    transport_sockets.tls.v3.Secret
    name: upstream_cert_ca
    validation_context:
      trusted_ca:
        filename: "/etc/envoy/certs/cacert.pem"
```

CERTIFICATE ROTATION: ALTERNATIVES

Can use certmanager to do the rotation

only in K8S

Spiffe is another option

ENVOY DEBUGGING

ENVOY DEBUGGING

TLS connection errors only show up in `connection` component debug logs

Client only sees socket closing messages

`OPENSSL_internal` ERRORS

`KEY_VALUES_MISMATCH` `[warning/config]`

key does not match cert

`SSLV3_ALERT_CERTIFICATE_EXPIRED` `[debug/connection]`

`CERTIFICATE_VERIFY_FAILED`

`SSLV3_ALERT_CERTIFICATE_UNKNOWN` `[debug][connection]`

cert SAN does not match expected SAN

Example: certificate SAN does not match `match_subject_alt_names`

VM side

```
envoy_vm_1     [debug][connection]
[source/extensions/transport_sockets/tls/ssl_socket.cc:224]
[C0] TLS error: 268435581:SSL routines:
OPENSSL_internal:CERTIFICATE_VERIFY_FAILED
```

Sidecar side

```
envoy_sidecar_1 [debug][connection]
[source/extensions/transport_sockets/tls/ssl_socket.cc:224]
[C1] TLS error: 268436502:SSL routines:
OPENSSL_internal:SSLV3_ALERT_CERTIFICATE_UNKNOWN
envoy_sidecar_1 [debug][connection]
[source/common/network/connection_impl.cc:241]
[C1] closing socket: 0
```

HOW IS IT USED

- http/s: as a http proxy
- other ports: open port in localhost

HTTP PROXY: JAVA

System properties

- `http.proxyHost` & `http.proxyPort`
- `https.proxyHost` & `https.proxyPort`
- `http.nonProxyHosts`

HTTP PROXY: JAVA

Apache HTTP client ignores proxy by default

```
HttpClient httpClient = HttpClientBuilder.create().
  useSystemProperties().build();
```

LONG RUNNING CONNECTIONS

Connection pools or long running connections ar typical (ie. db)

Configure `max_stream_duration` and `stream_idle_timeout`

Increase load balancer timeouts

CONNECTION POOLS

Must configure connection pools to validate connections before use

In JDBC drivers it is a configuration option

ENVOY TWEAKS

Some defaults can to be tweaked on the VM side listener

```
typed_config:
  "@type": type.googleapis.com/envoy.extensions
  .filters.network.http_connection_manager.v3.
  HttpConnectionManager
  common_http_protocol_options:
    # do not reset connections if they are not idle, set to 24h
    max_stream_duration: 86400s
  # increase the stream idle timeout, it ignores tcp keepalives
  stream_idle_timeout: 7200s
```

HTTP PROXY: HTTPD

```
SSLProxyEngine on

ProxyRemote "https://example.com"
  "http://${HTTP_PROXY_HOST}:${HTTP_PROXY_PORT}"
ProxyPass "/somepath" "https://example.com"
ProxyPassReverse "/somepath" "https://example.com"
```

httpd sends proxy requests with `CONNECT HTTP/1.0`

we need to set `http_protocol_options.accept_http_10`

ENVOY CONFIG: RESOURCES

envoyproxy.io

arch_overview/http/upgrades

sandboxes/tls

sandboxes/double-proxy

csanchez.org

csanchez

carlossg

Adobe