OVS と Linux ネットワークで実現する HV の透過的プロキシ

近藤智文 / サイバーエージェント

CODT2026

自己紹介

近藤 智文

- サイバーエージェント24新卒
- プライベートクラウド IaaS 基盤の開発・運用をやってます
- OpenStack/OVS/Kubernetes /Go/Python

https://github.com/TOMOFUMI-KONDO/
https://x.com/tomokon_0314

本発表で学べること

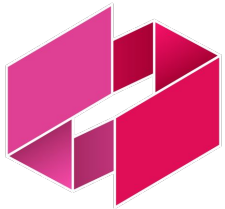
- 高いテナント分離性と undercloud の疎通性を両立する仮想ネットワーク設計
- 上記を構成する OVS や Linux ネットワーク機能の技術的詳細

プライベートクラウド
新リージョンの VPC 要件

プライベートクラウドの概要

社内ユーザに以下の機能を提供

- IaaS (OpenStackベース)
- PaaS/SaaS
  - Kubernetes as a Service
  - Serverless Computing
  - ML Platform
  - GitHub Actions Self-Hosted Runner
  - DB/Storage, etc.



Cycloud

新リージョンの立ち上げ

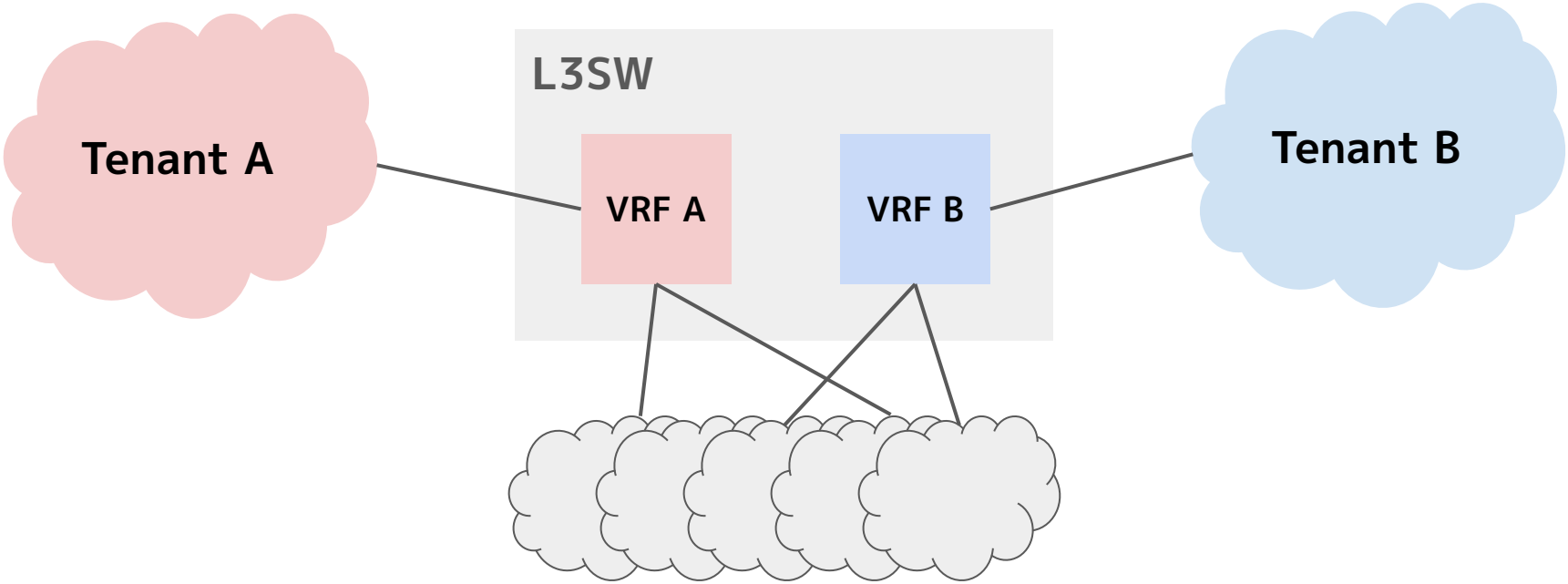
ハードもソフトもゼロベースで構築

- 最新世代のハードウェアによりリソース効率やパフォーマンスの大幅向上
- 統合されたUI/UX・メンテナンスタイムの削減による利用者の負担削減
- 高い**テナント分離性**によるセキュリティ向上←←←

テナント分離とは？

テナント分離

各テナントの VPC をL3レベルで分離

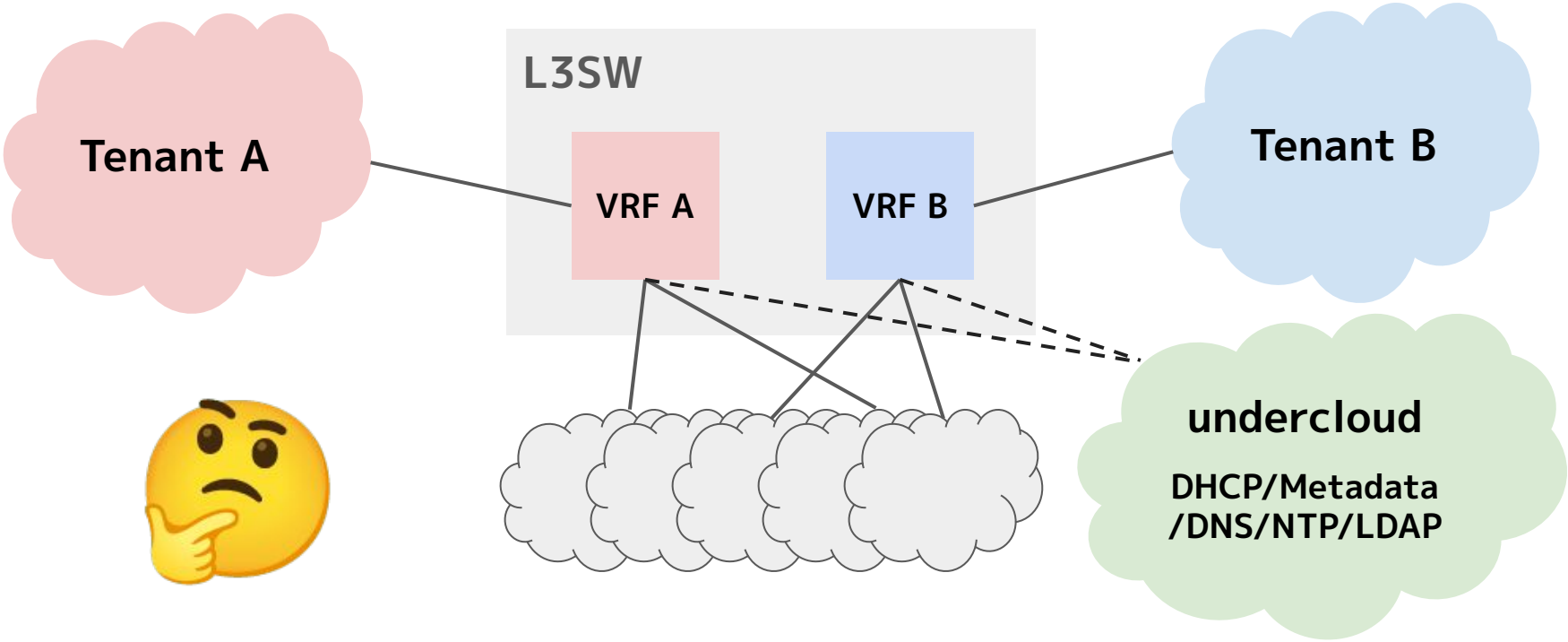


セキュアでいいじゃん！

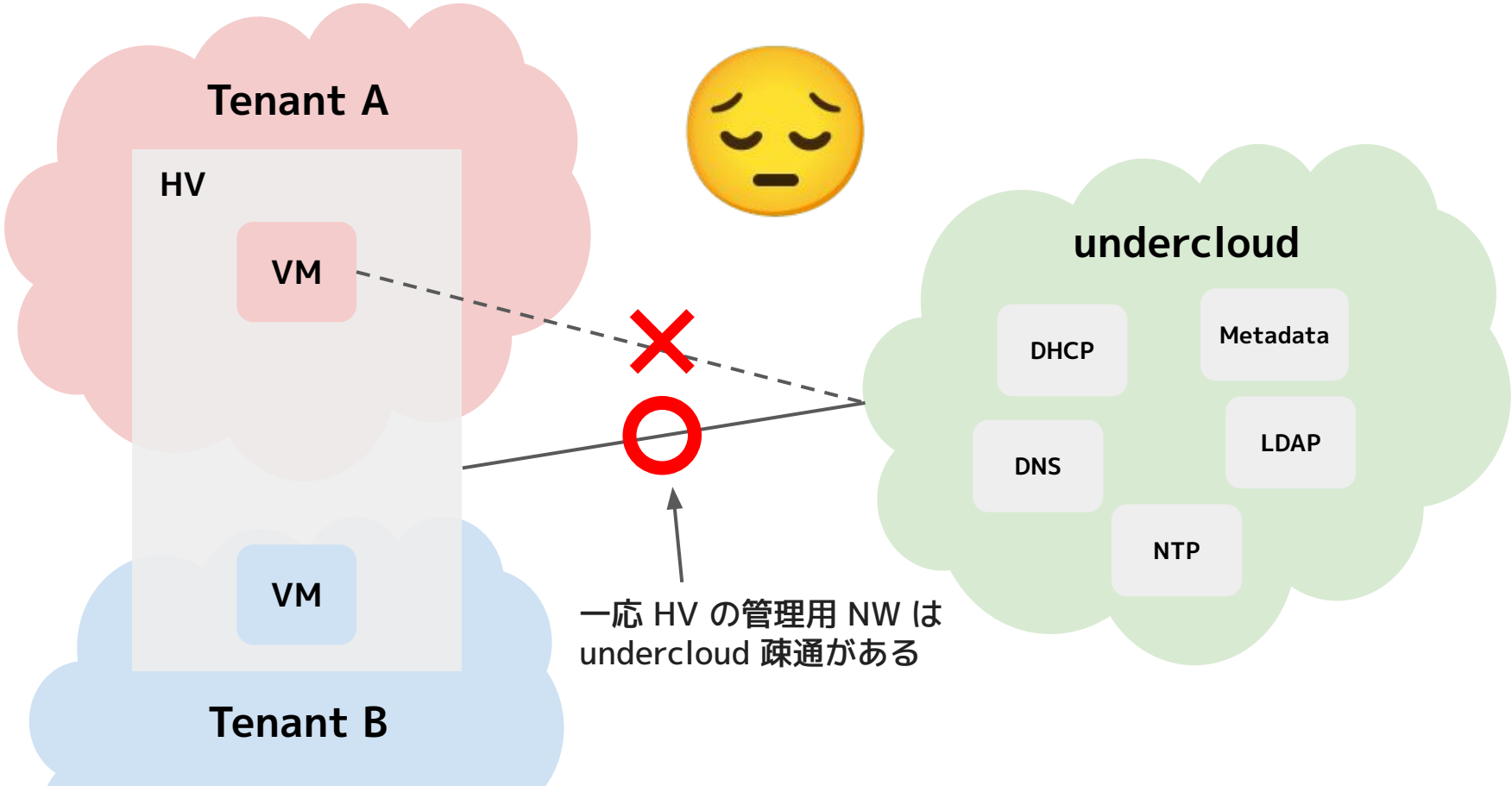
しかし…

undercloud への疎通もない

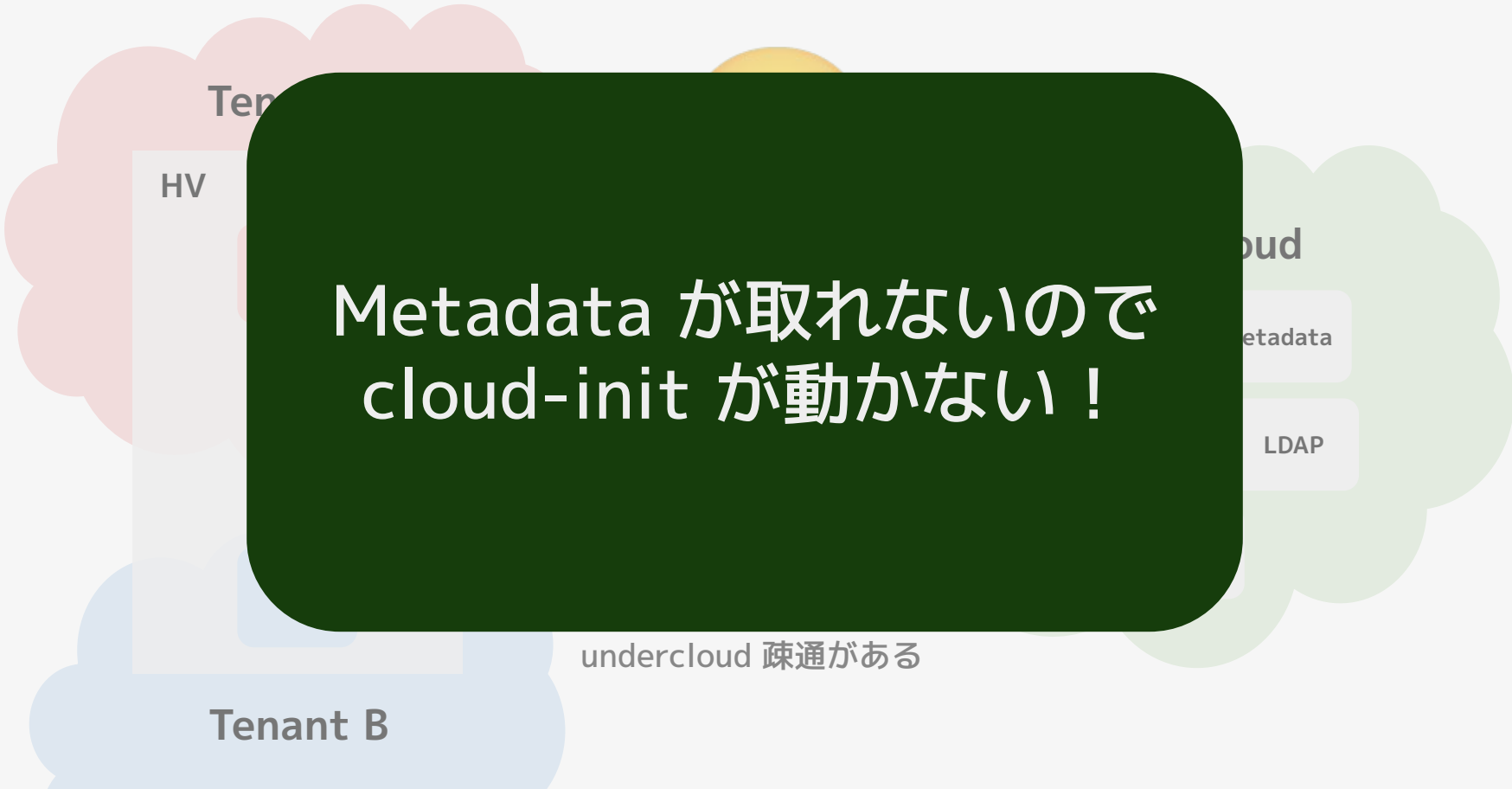
各テナントの VPC をL3レベルで分離



VM が undercloud のサービスにアクセスできない

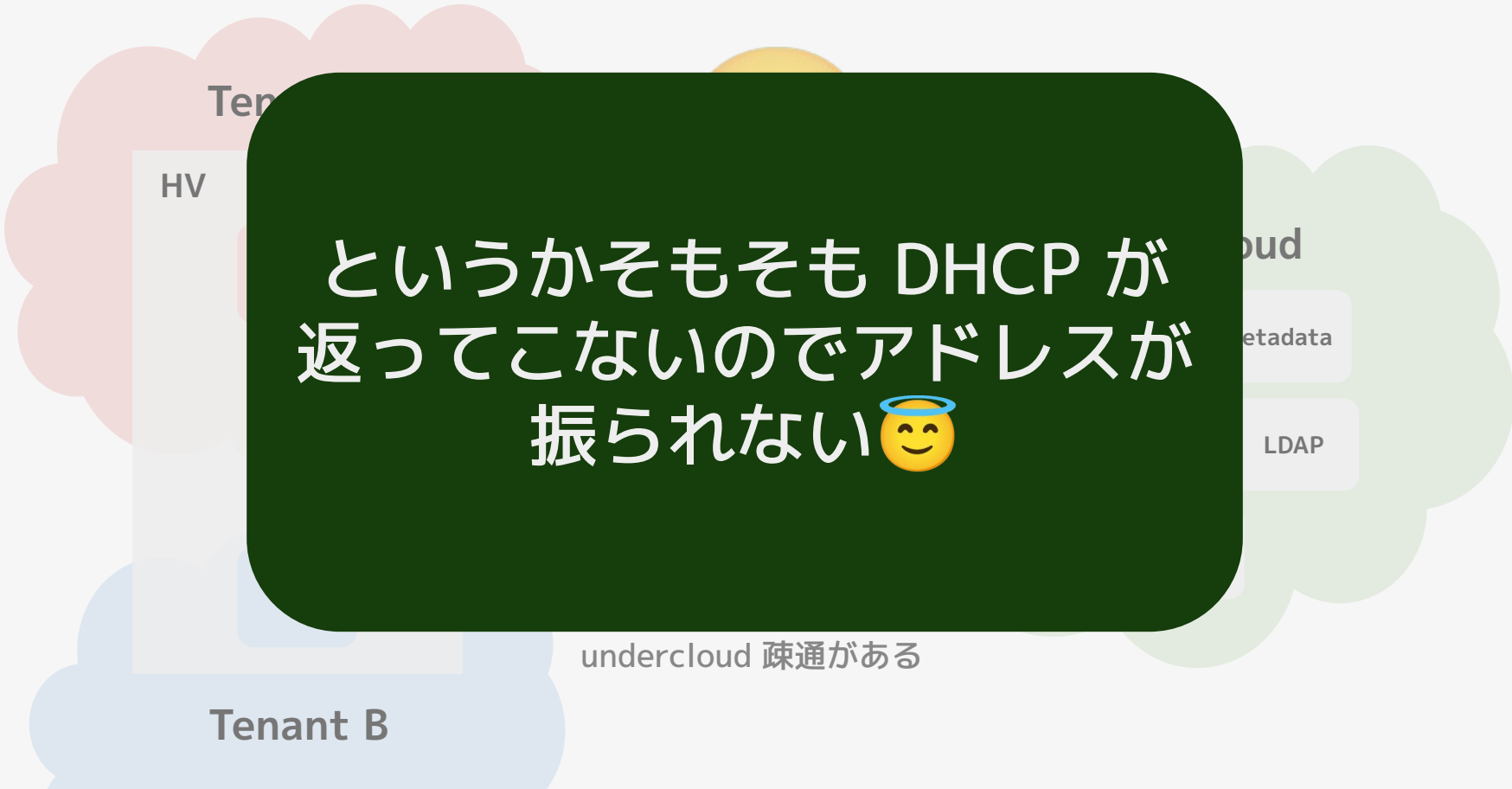


VM が undercloud のサービスにアクセスできない



Metadata が取れないので
cloud-init が動かない！

VM が undercloud のサービスにアクセスできない



undercloud を全テナントに疎通させればいいのでは？

一見良さそう。だが...

➔ 逆に言えば、undercloud に侵入されると**全てのテナントに疎通できてしまう**

➔ せっかくテナント分離をしているのにそのような大きな侵入口を作ってしまうのは避けたい

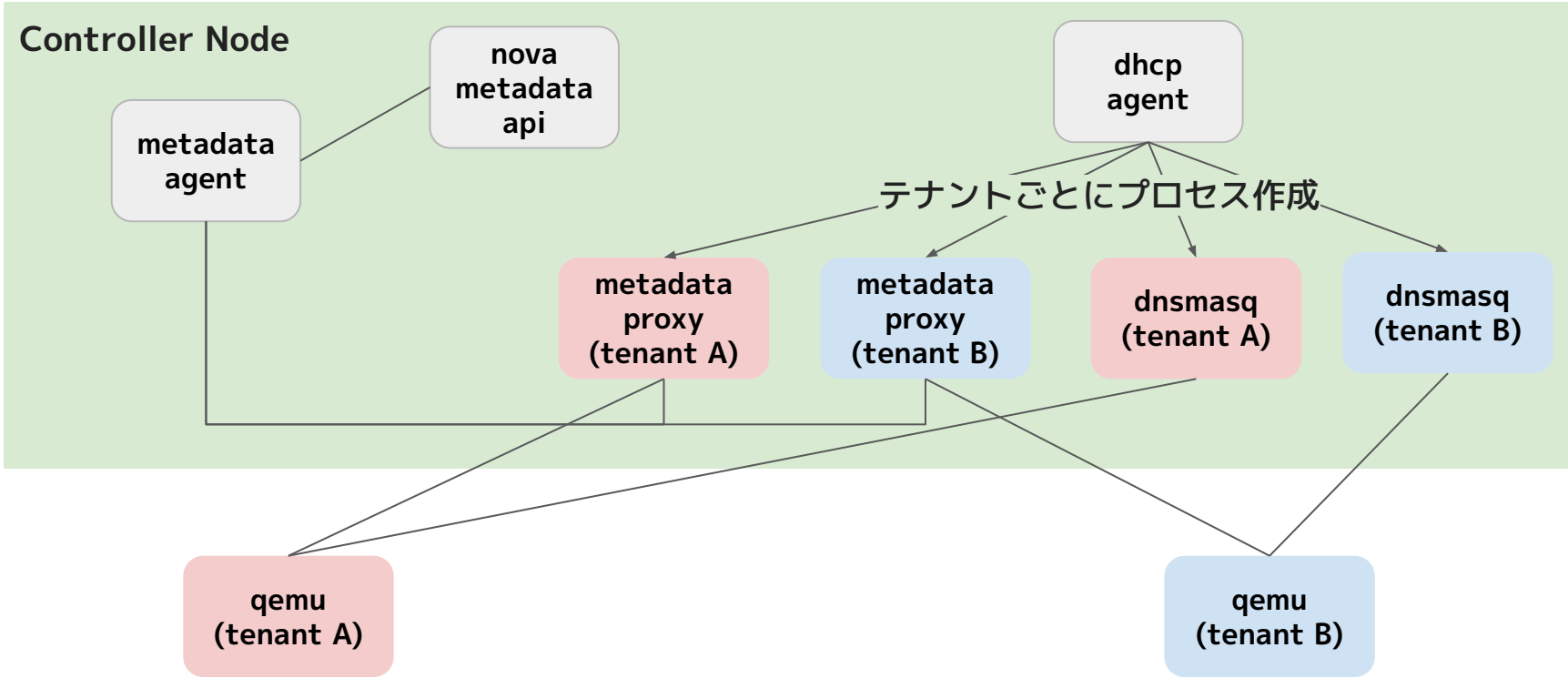
テナント分離 vs undercloud 疎通性

矛盾する二つの要件を同時に満たす方法はあるのか...

- セキュリティの観点から各テナントは NW 的に完全に分離したい
- VM から undercloud のサービスに疎通させたい

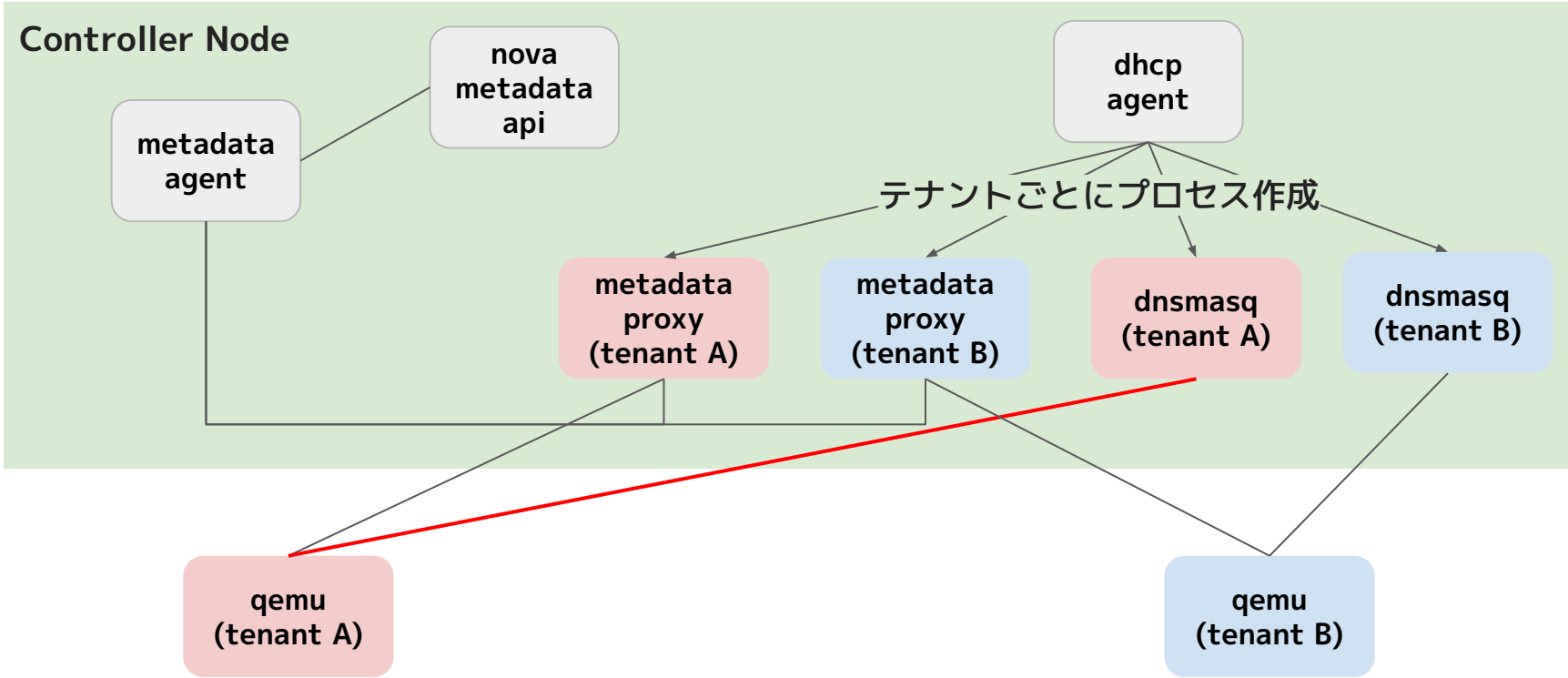
Neutron の DHCP/Metadata おさらい

Neutron の DHCP/Metadata 構成

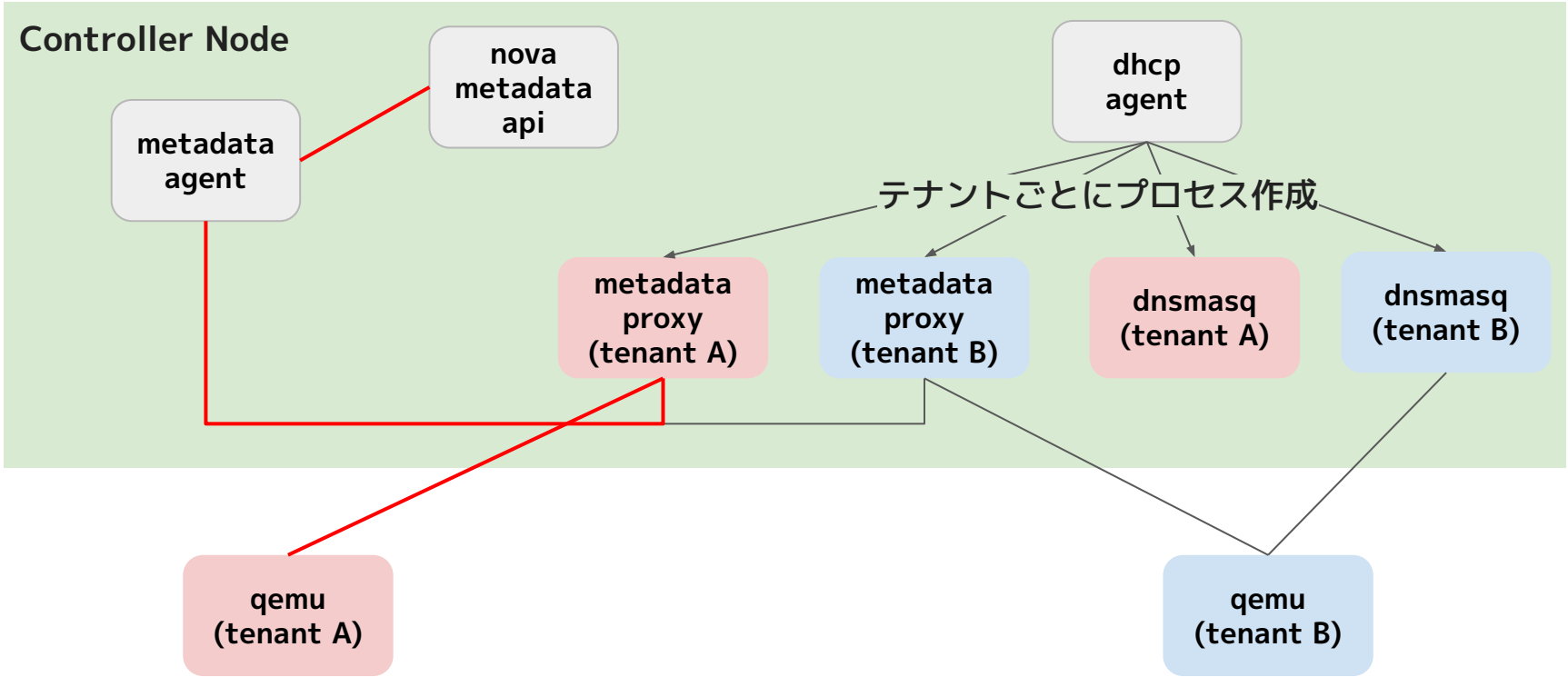


（Neutron: OpenStack のネットワークサービス）

Neutron の DHCP/Metadata 構成



Neutron の DHCP/Metadata 構成



Neutron ではどうしているのか (Metadata)

Neutron 標準構成の課題

- agent を集約すると Control Plane が**全テナントに L2 接続する必要**がある ❌
  - 別途 Network Node を用意したりするのは運用コスト的に厳しい
- DHCP/Metadata 以外の通信に対応できない ❌
  - DNS, NTP, LDAP, etc.

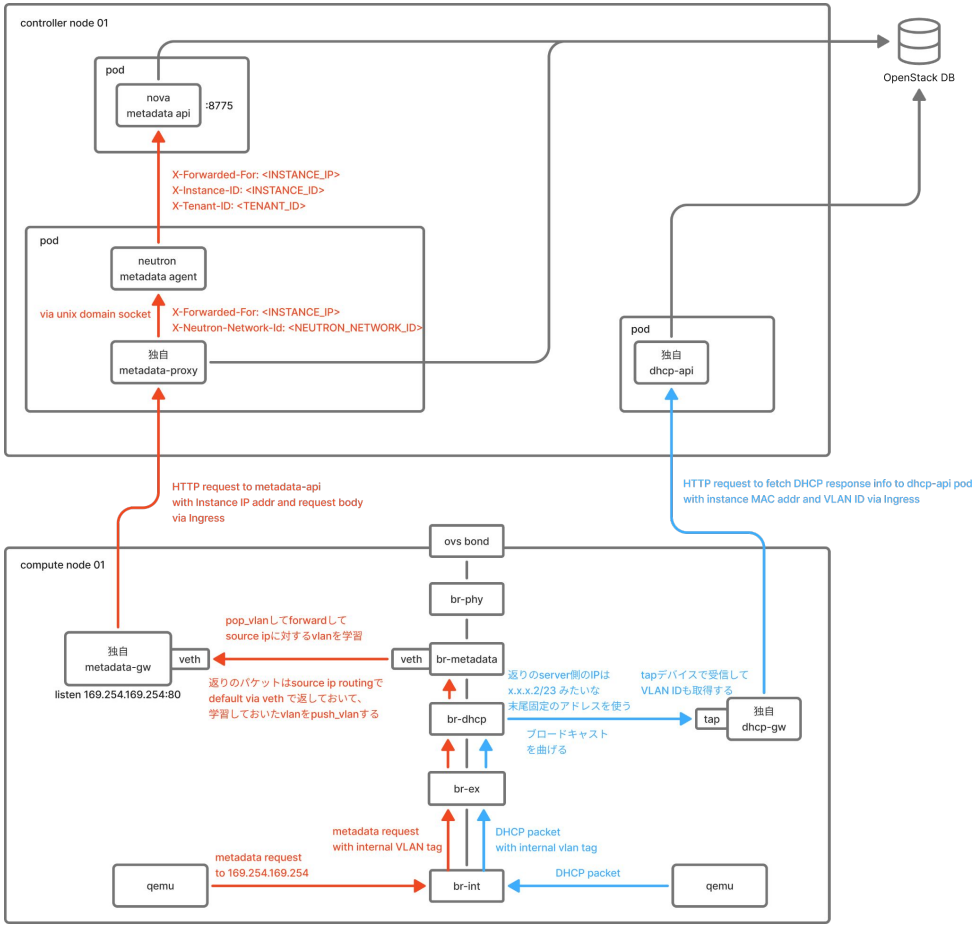
でも大丈夫

我々には OVS がある

Open vSwitch

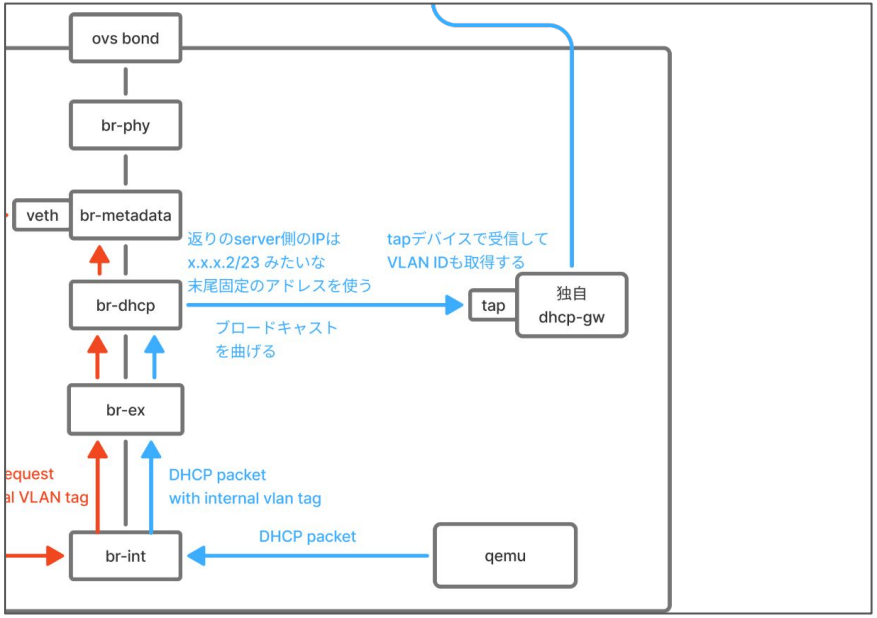
- Linux で動作する仮想スイッチ OSS
- Neutron のデータプレーンとして既に導入済みだった
- 通過するパケットに対して任意のフォワーディングや中身の書き換えが可能

設計 全体像

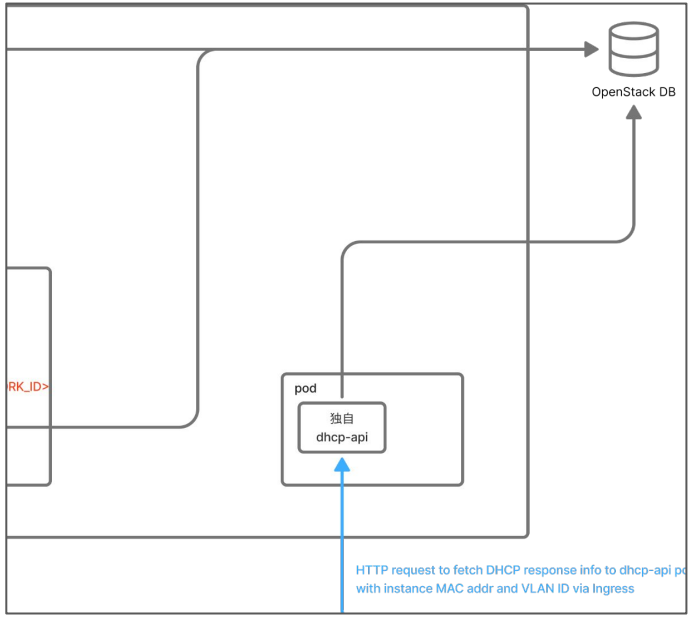

controller node 01
pod
nova
metadata api
:8775
OpenStack DB
X-Forwarded-For: <INSTANCE_IP>
X-Instance-ID: <INSTANCE_ID>
X-Tenant-ID: <TENANT_ID>
pod
neutron
metadata agent
via unix domain socket
X-Forwarded-For: <INSTANCE_IP>
X-Neutron-Network-Id: <NEUTRON_NETWORK_ID>
独自
metadata-proxy
pod
独自
dhcp-api
HTTP request to metadata-api
with Instance IP addr and request body
via Ingress
HTTP request to fetch DHCP response info to dhcp-api pod
with instance MAC addr and VLAN ID via Ingress
compute node 01
ovs bond
br-phy
pop_vlanしてforwardして
source ipに対するvlanを学習
独自
metadata-gw
veth
veth
br-metadata
listen 169.254.169.254:80
返りのパケットはsource ip routingで
default via veth で返しておいて、
学習しておいたvlanをpush_vlanする
返りのserver側のIPは
x.x.x.2/23 みたいな
末尾固定のアドレスを使う
tapデバイスで受信して
VLAN IDも取得する
br-dhcp
tap
独自
dhcp-gw
ブロードキャスト
を曲げる
br-ex
metadata request
with internal VLAN tag
DHCP packet
with internal vlan tag
metadata request
to 169.254.169.254
DHCP packet
qemu
br-int
qemu

設計 DHCP

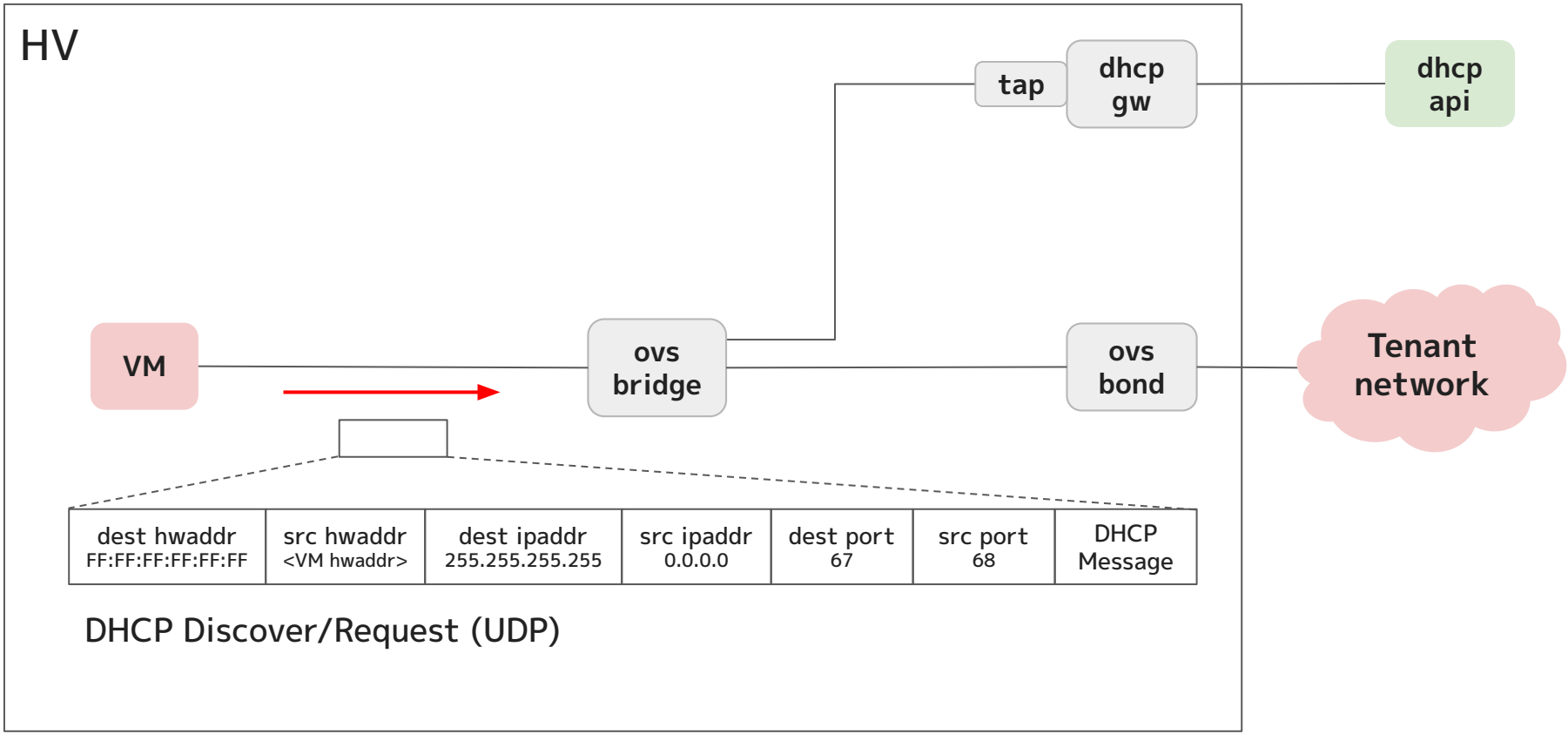
HV



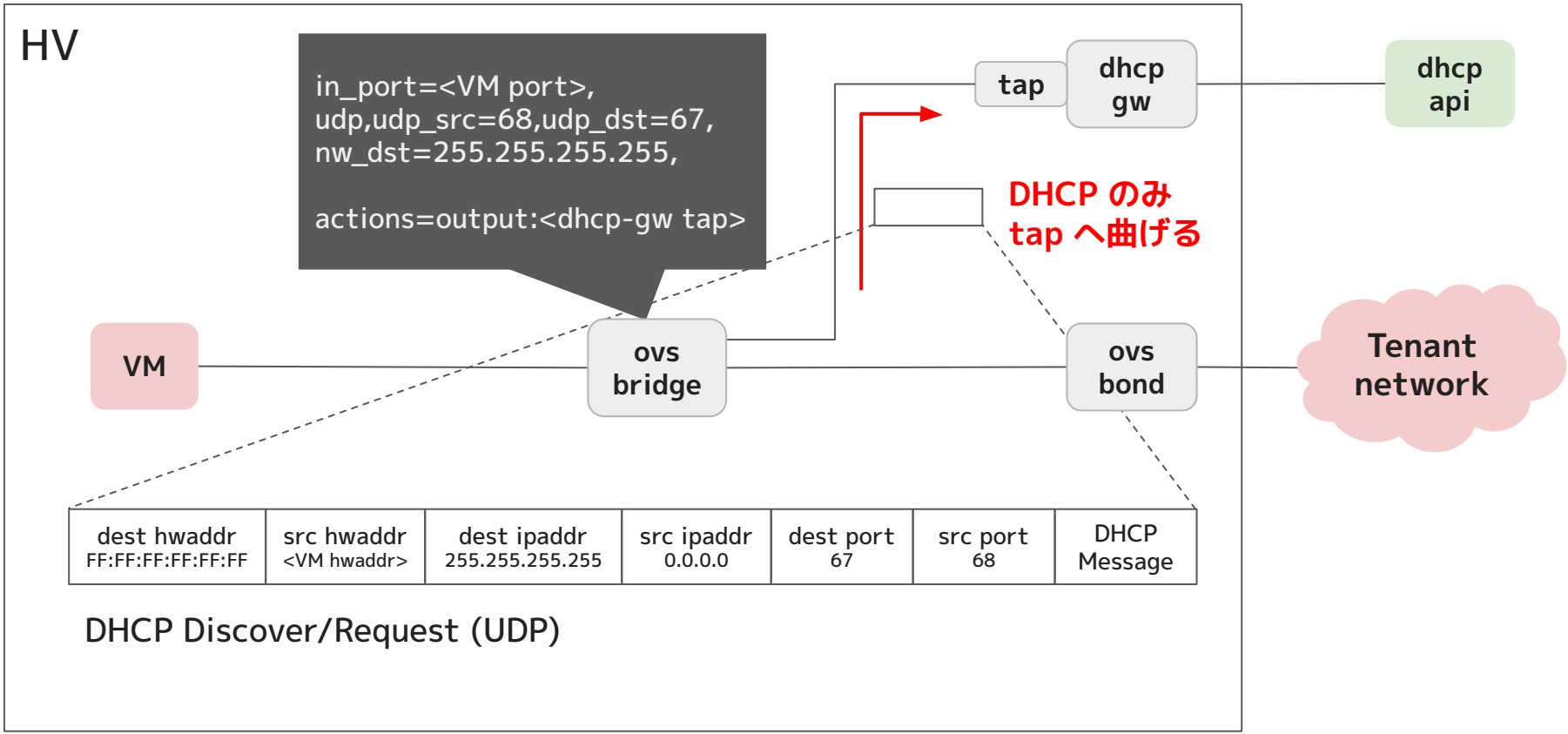
Controller



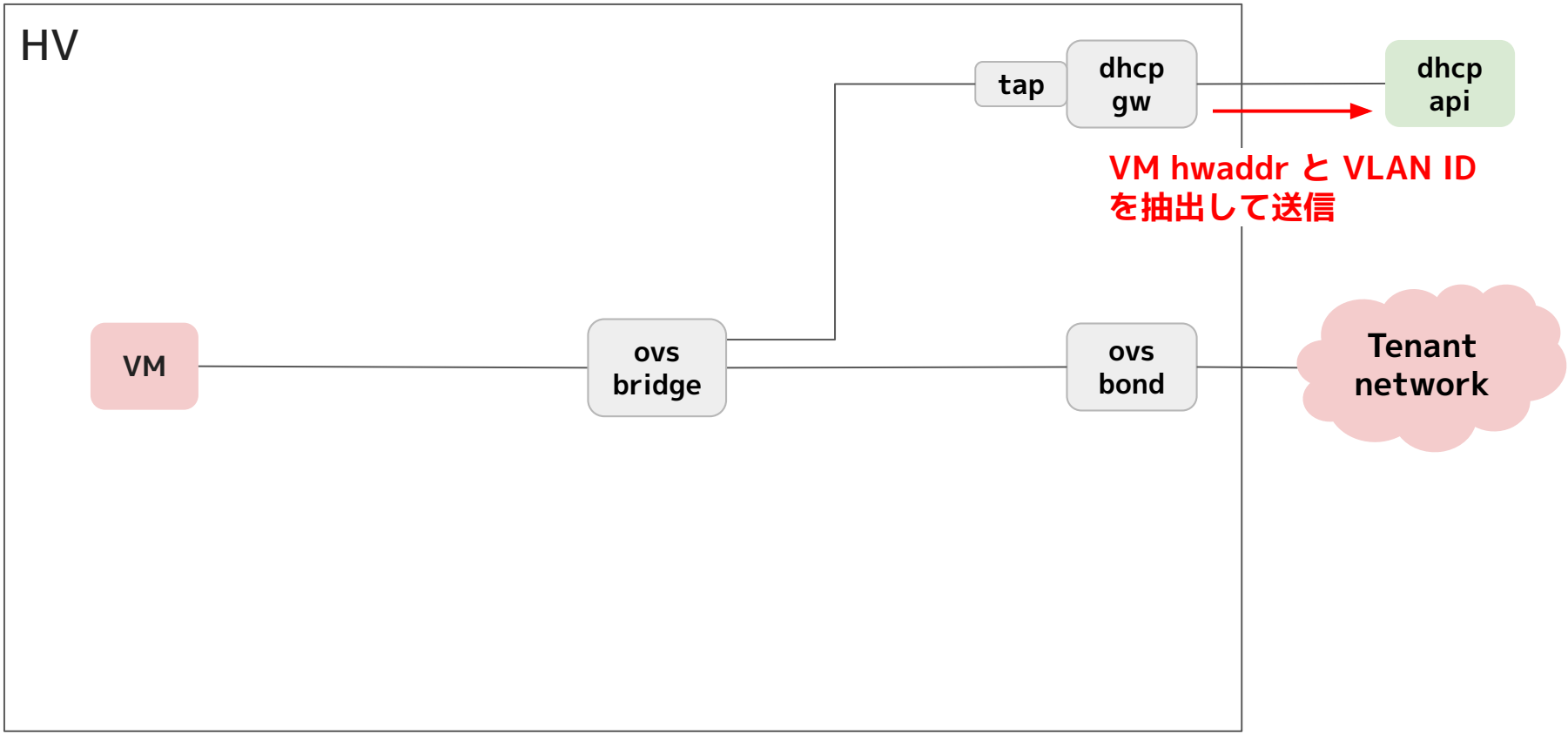
DHCP OVS 詳細（行き）



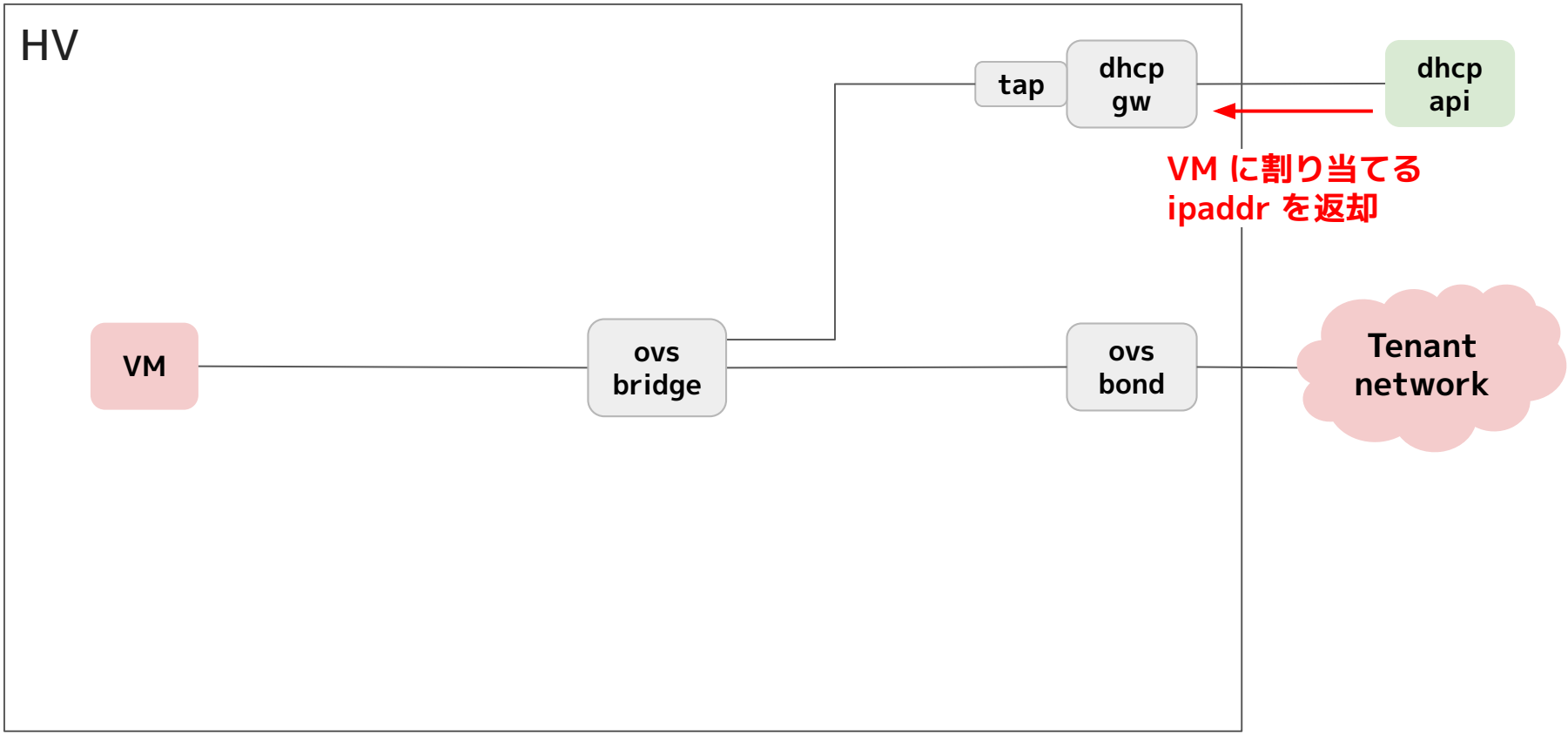
DHCP OVS 詳細（行き）



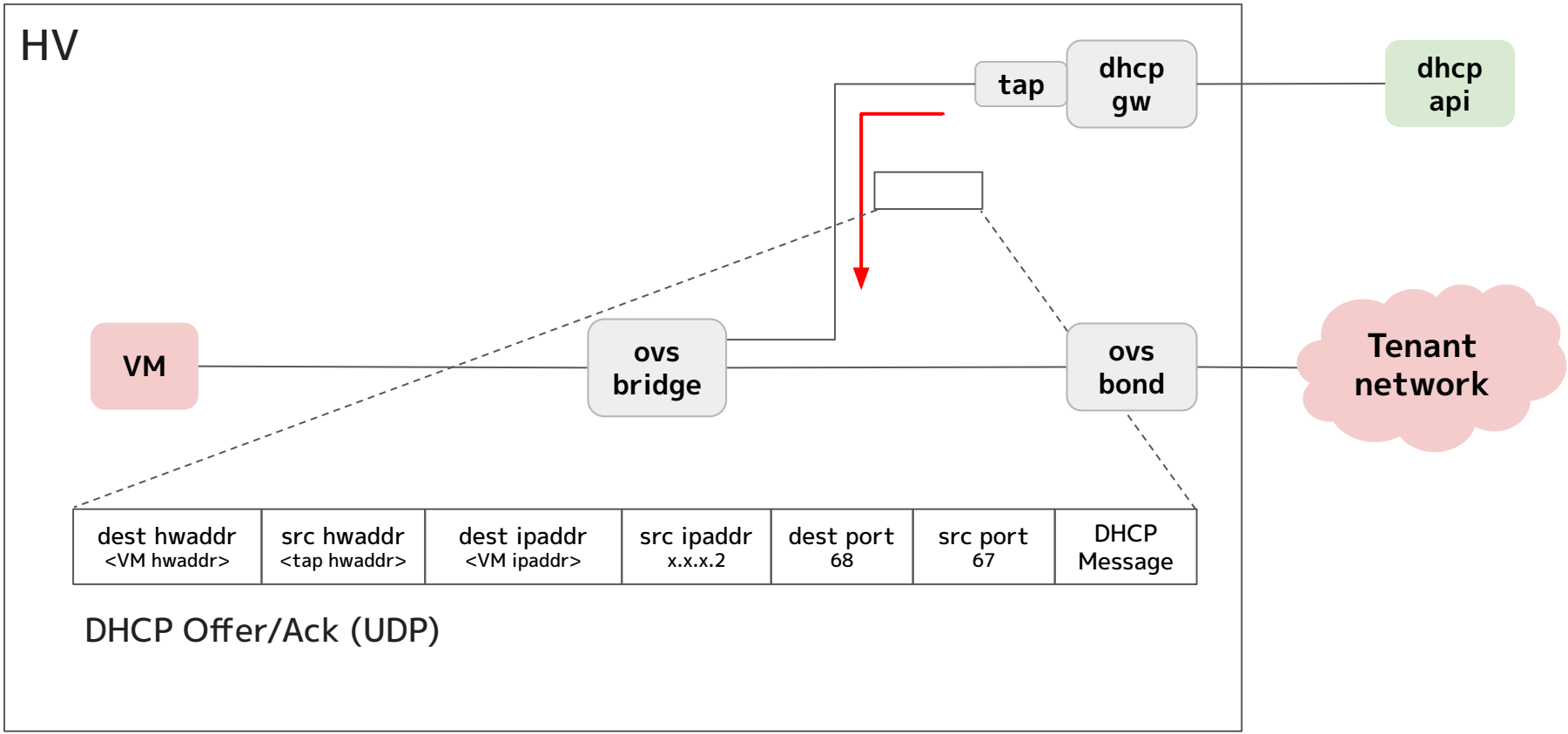
DHCP OVS 詳細（行き）



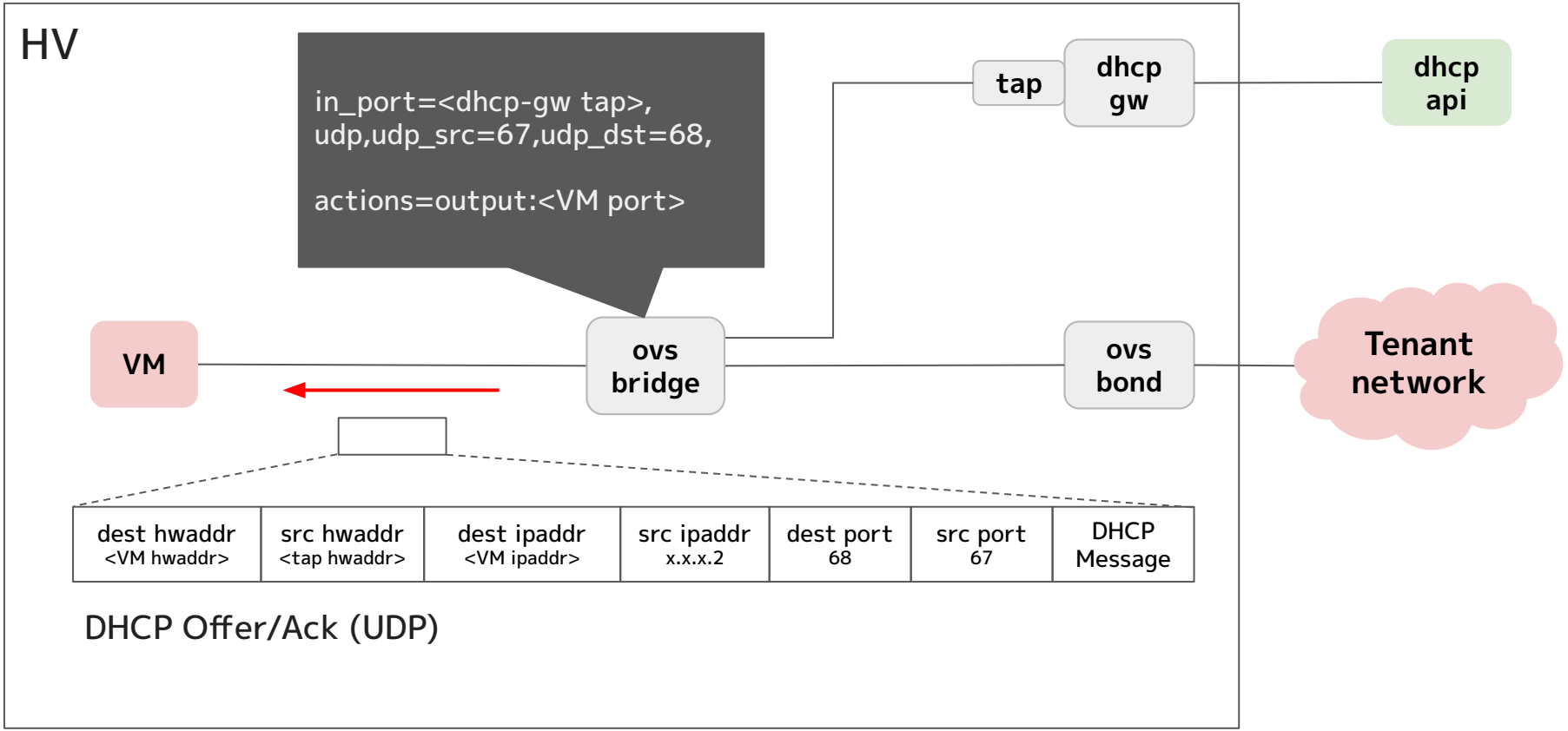
DHCP OVS 詳細（帰り）



DHCP OVS 詳細（帰り）

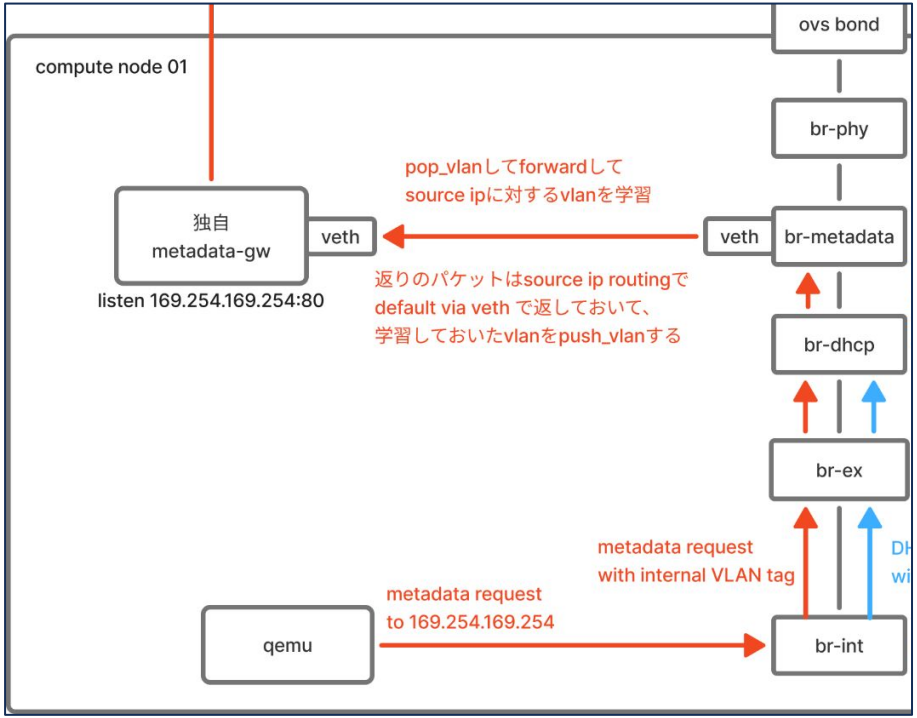


DHCP OVS 詳細（帰り）

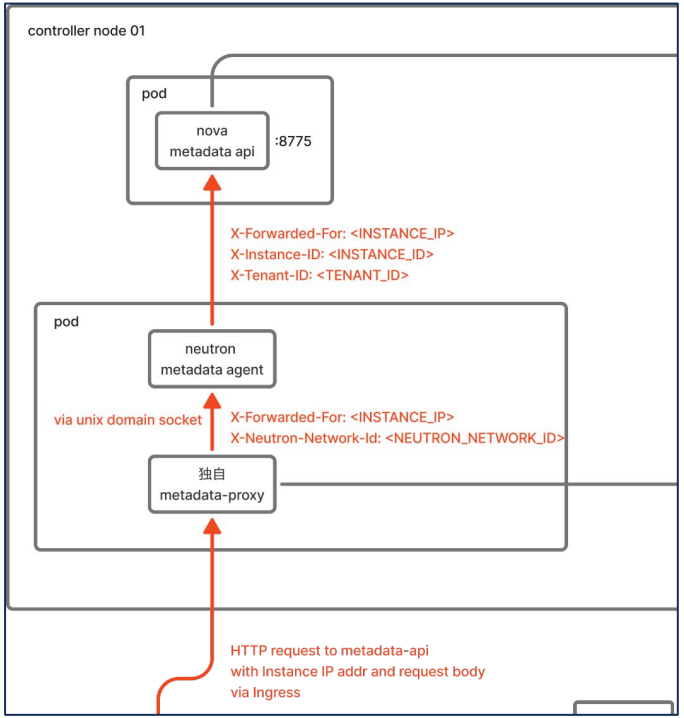


設計 Metadata（DNS / NTP / LDAP もほぼ同様）

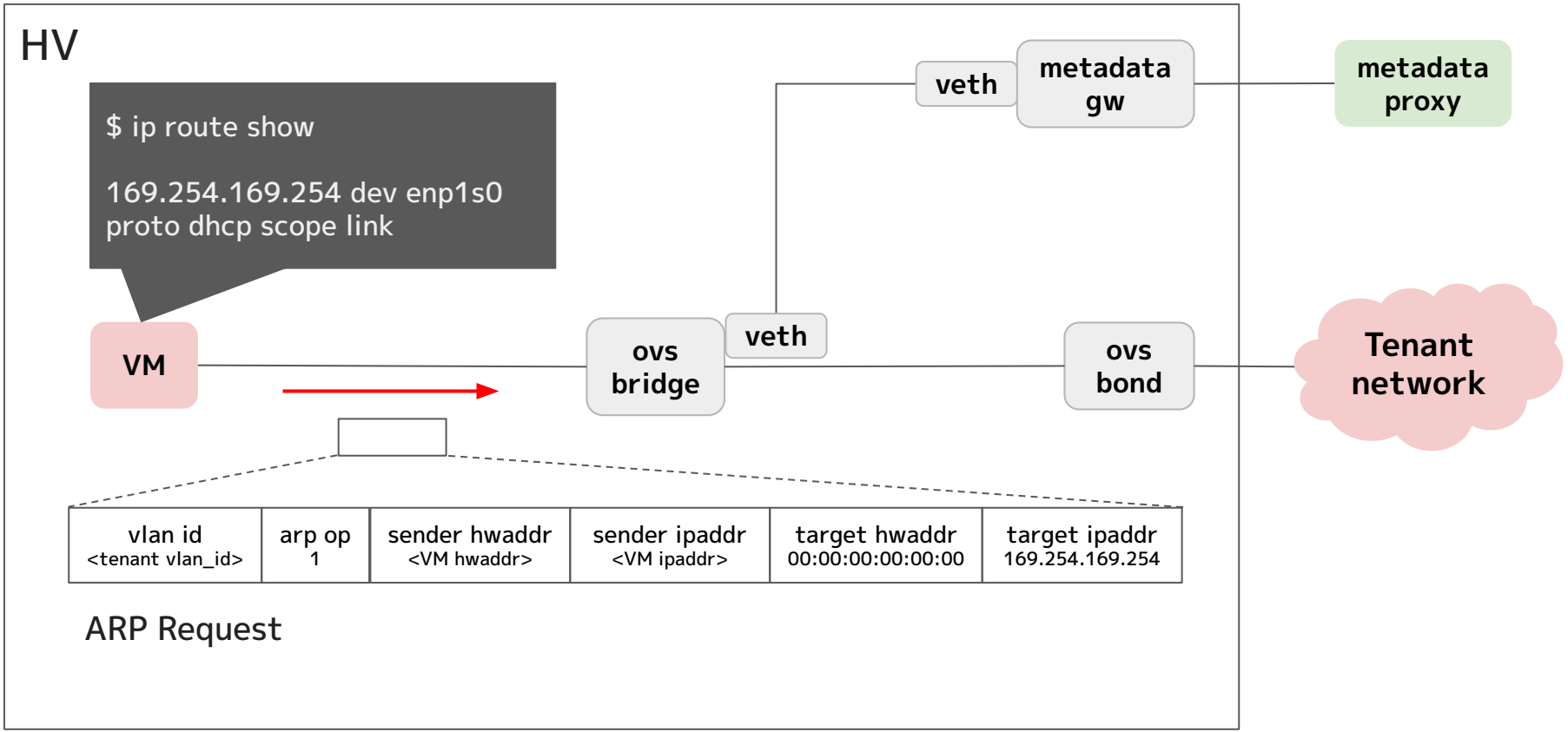
HV



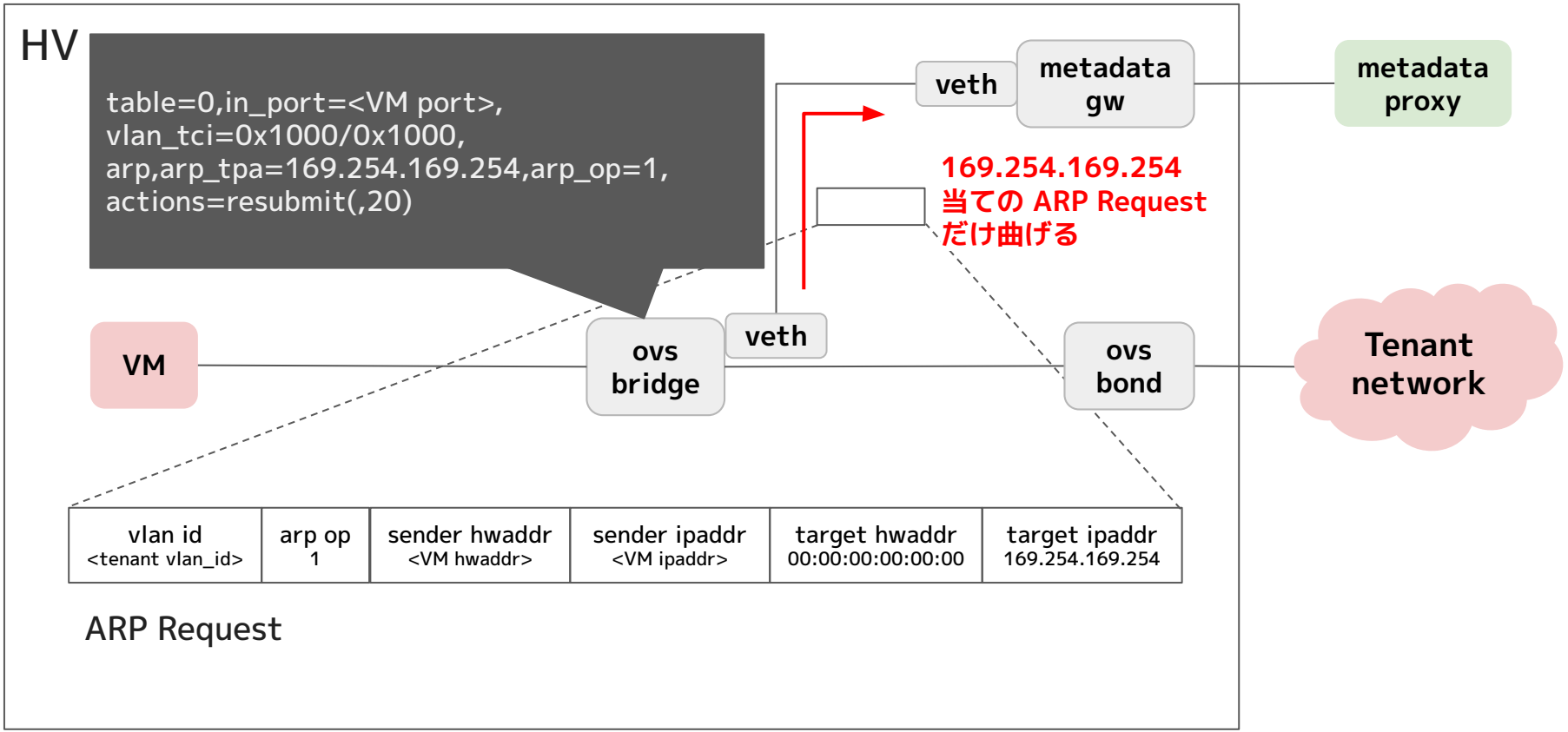
Controller



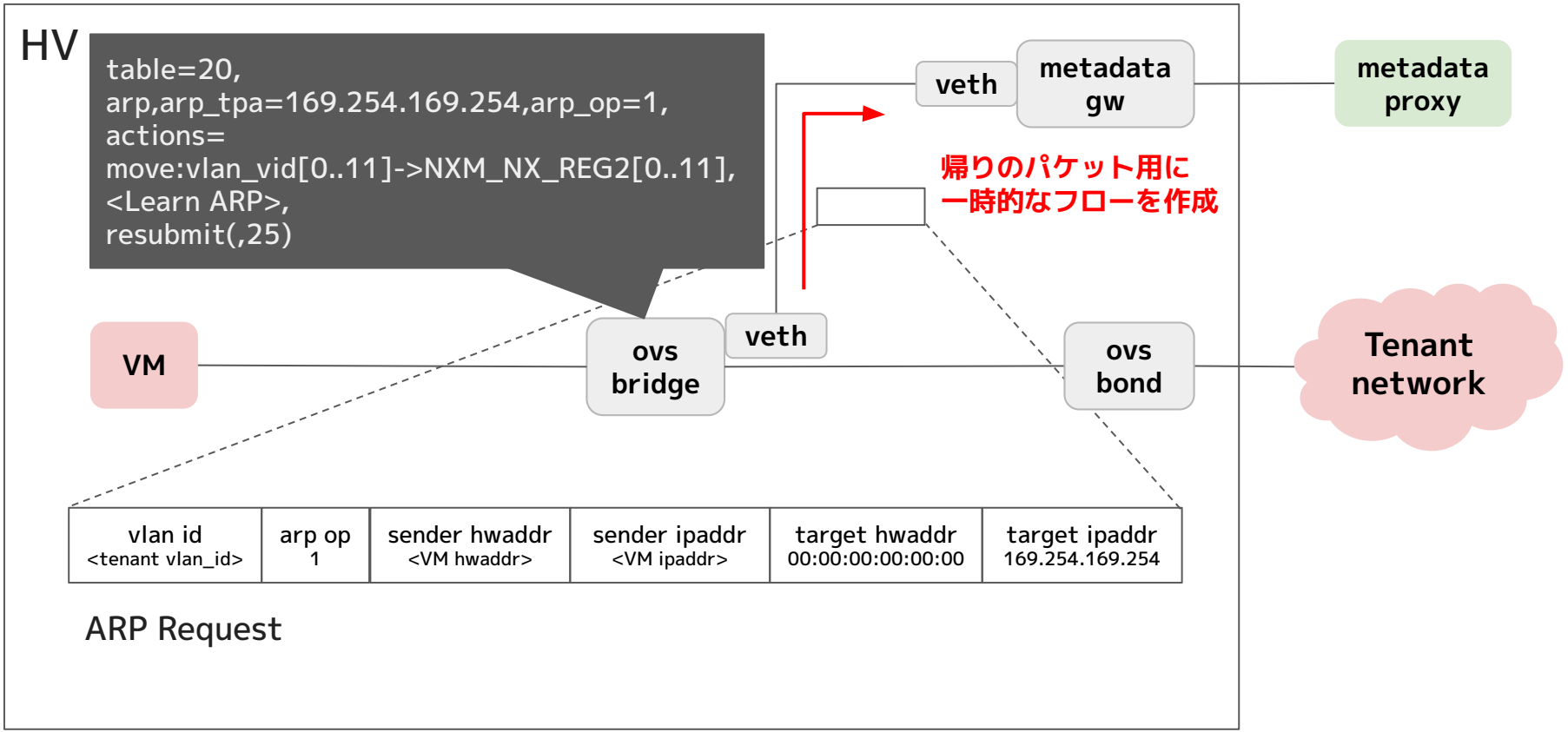
Metadata OVS 詳細（ARP / 行き）



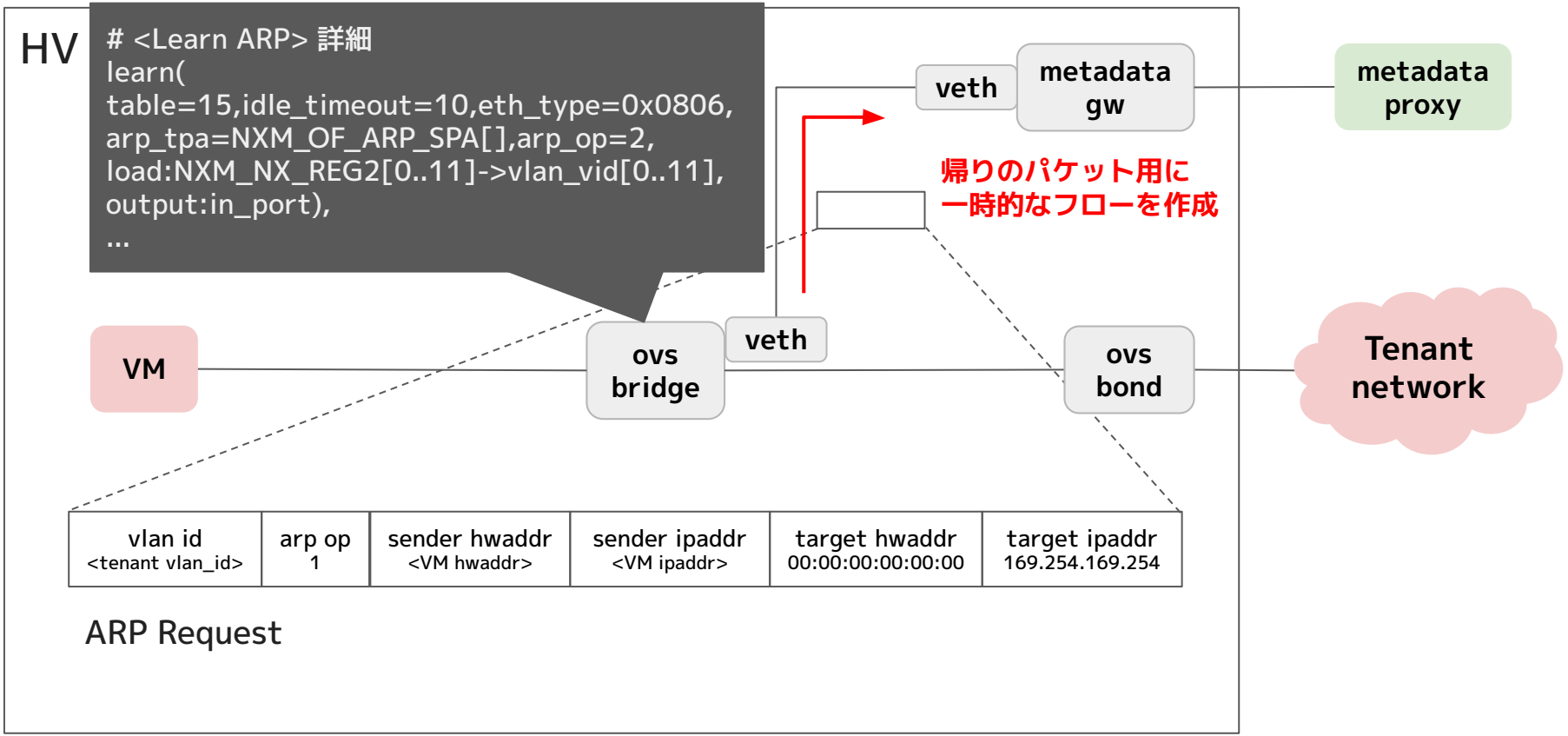
Metadata OVS 詳細（ARP / 行き）



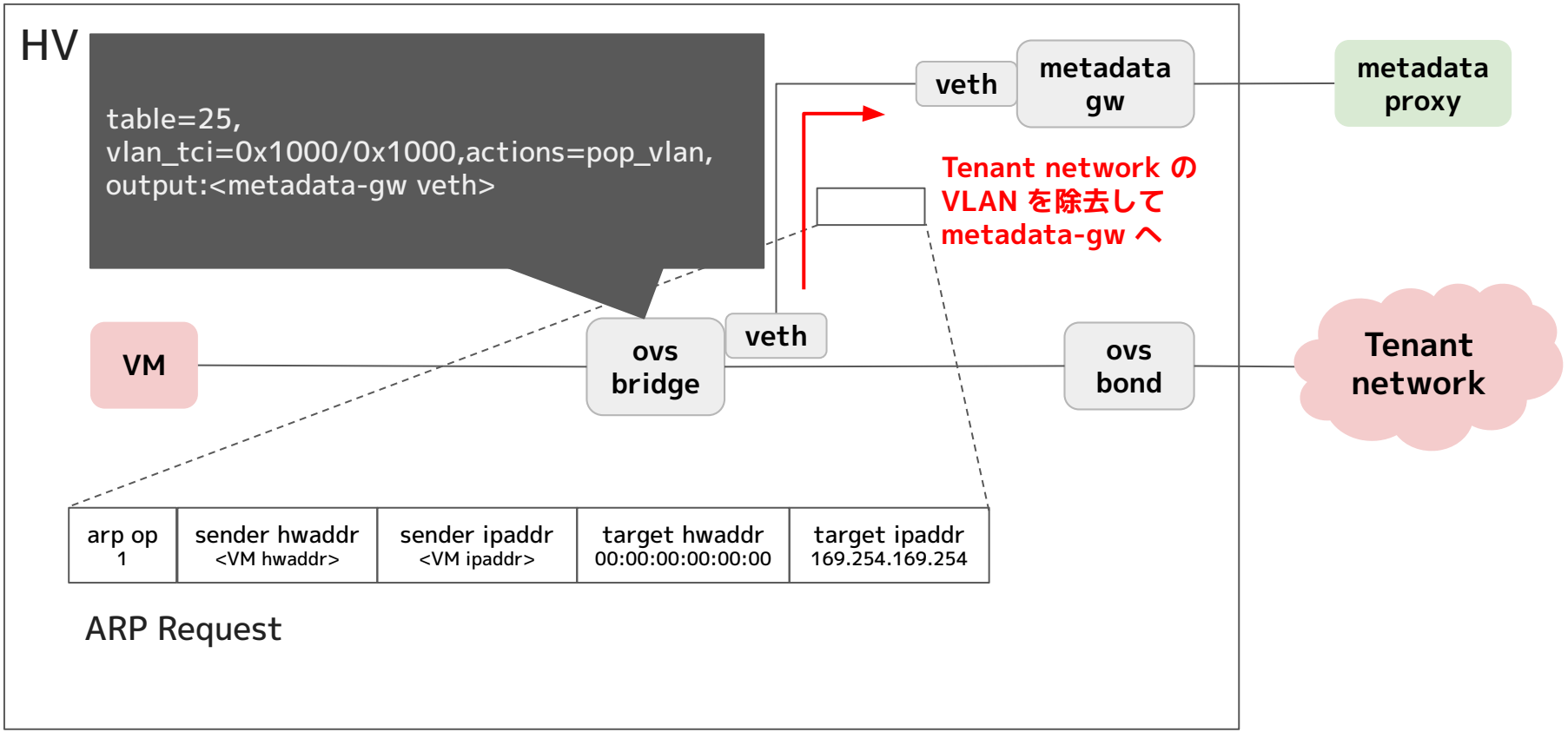
Metadata OVS 詳細（ARP / 行き）



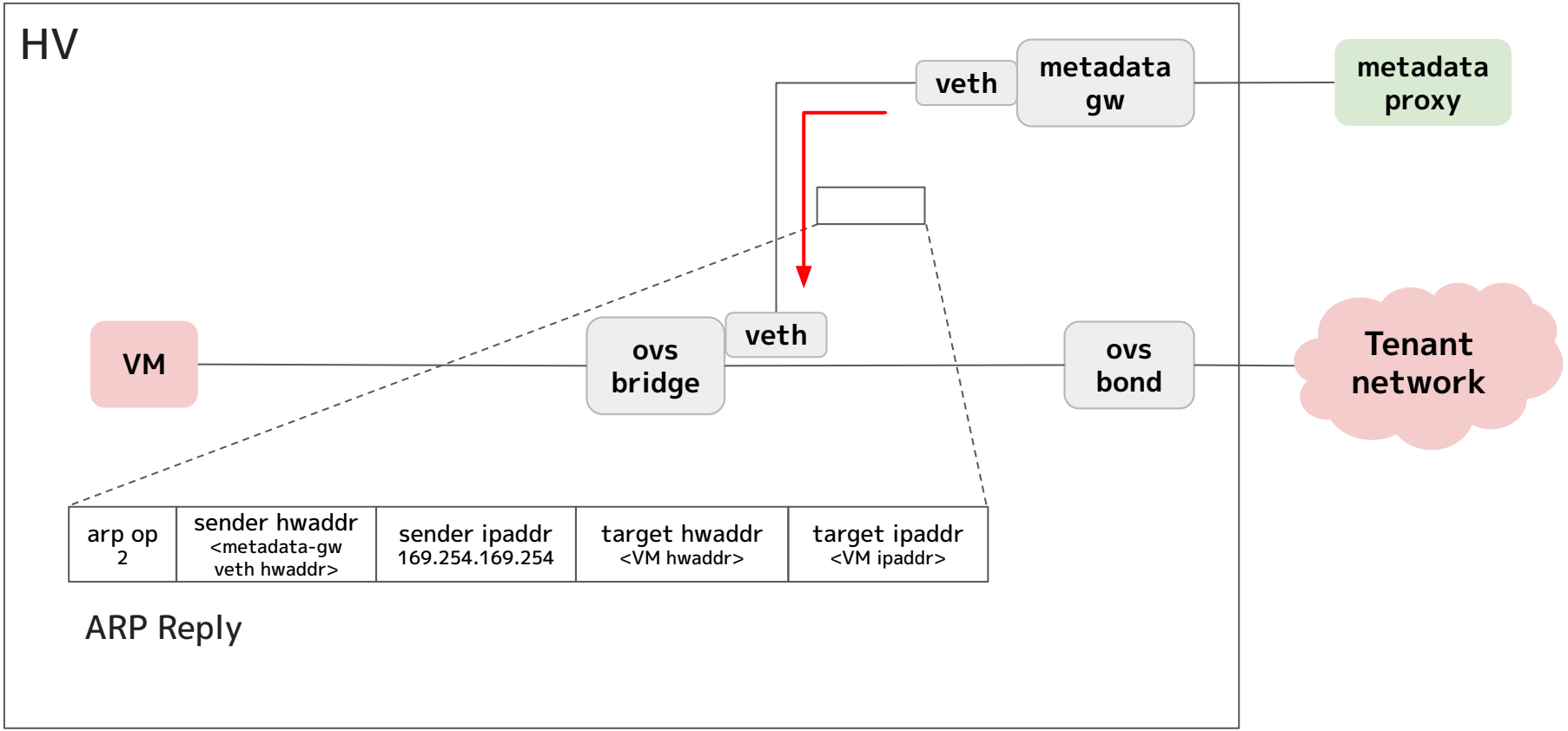
Metadata OVS 詳細（ARP / 行き）



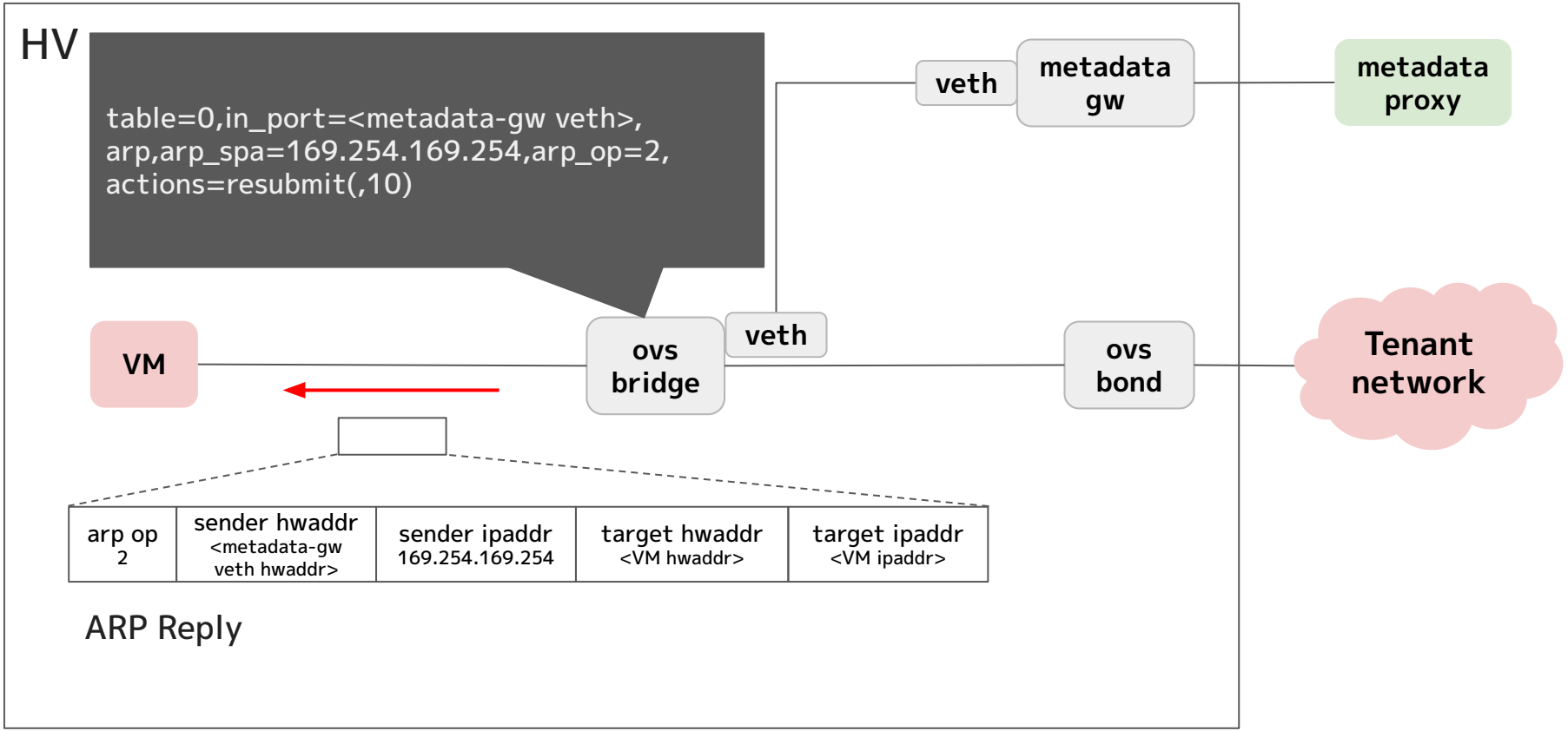
Metadata OVS 詳細（ARP / 行き）



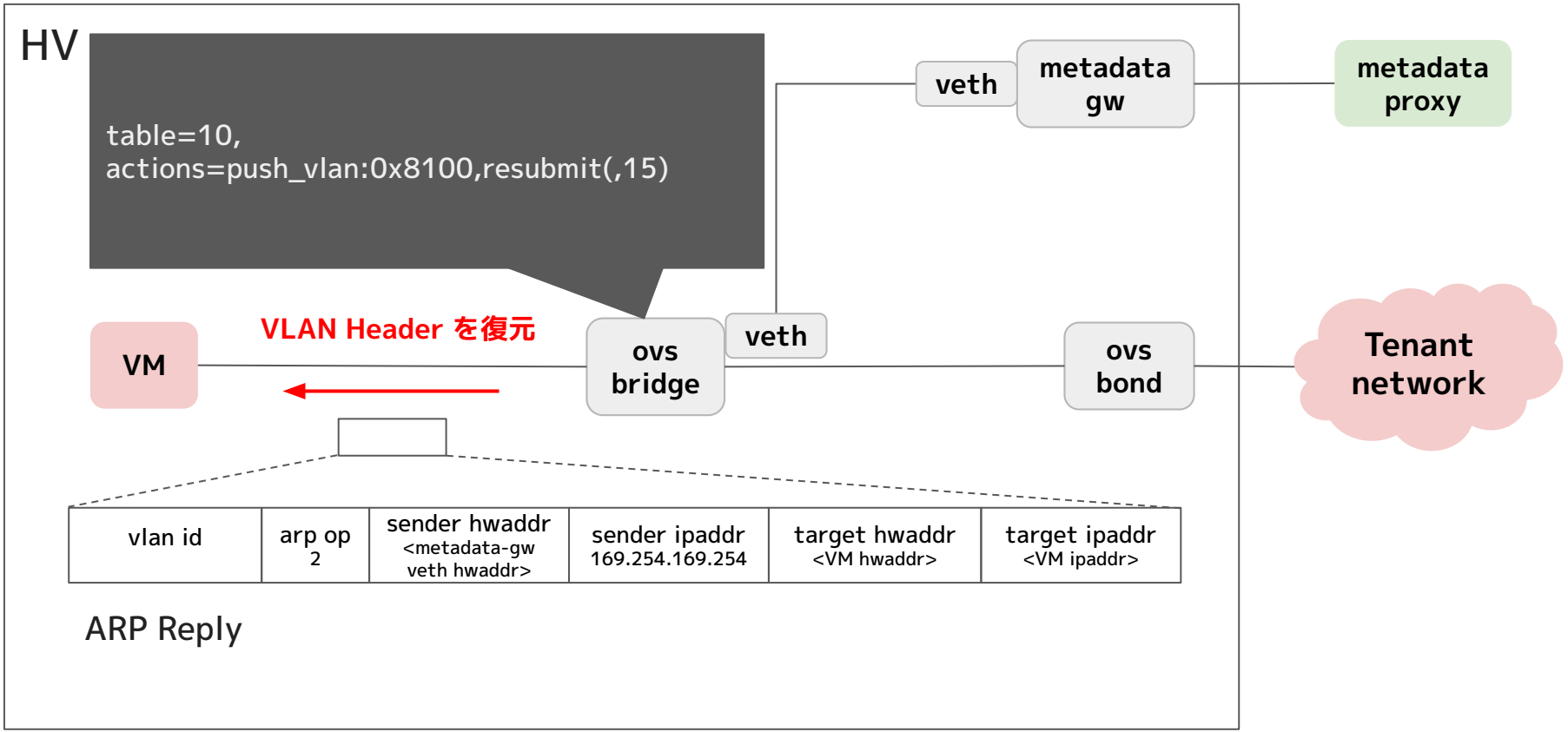
Metadata OVS 詳細（ARP / 帰り）



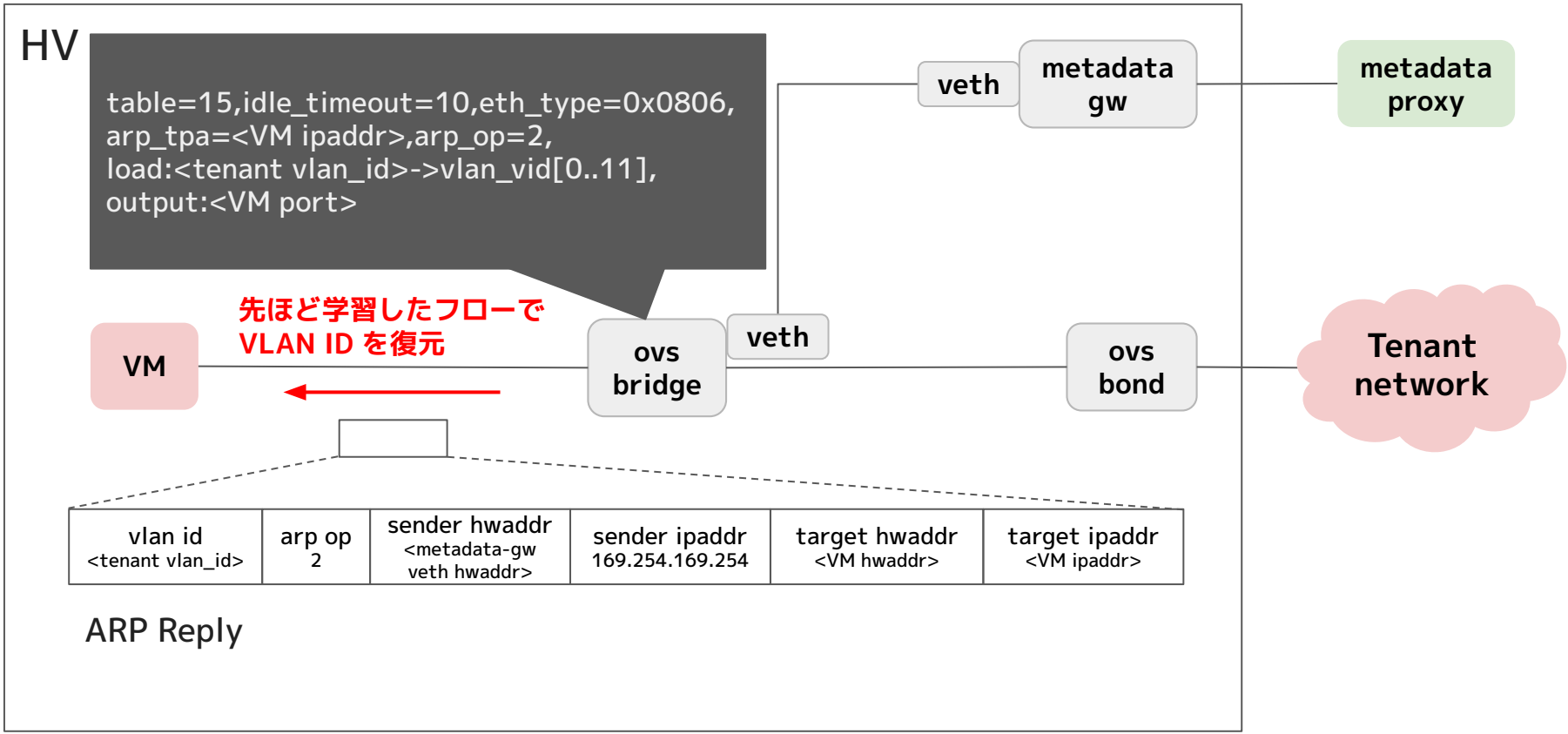
Metadata OVS 詳細（ARP / 帰り）



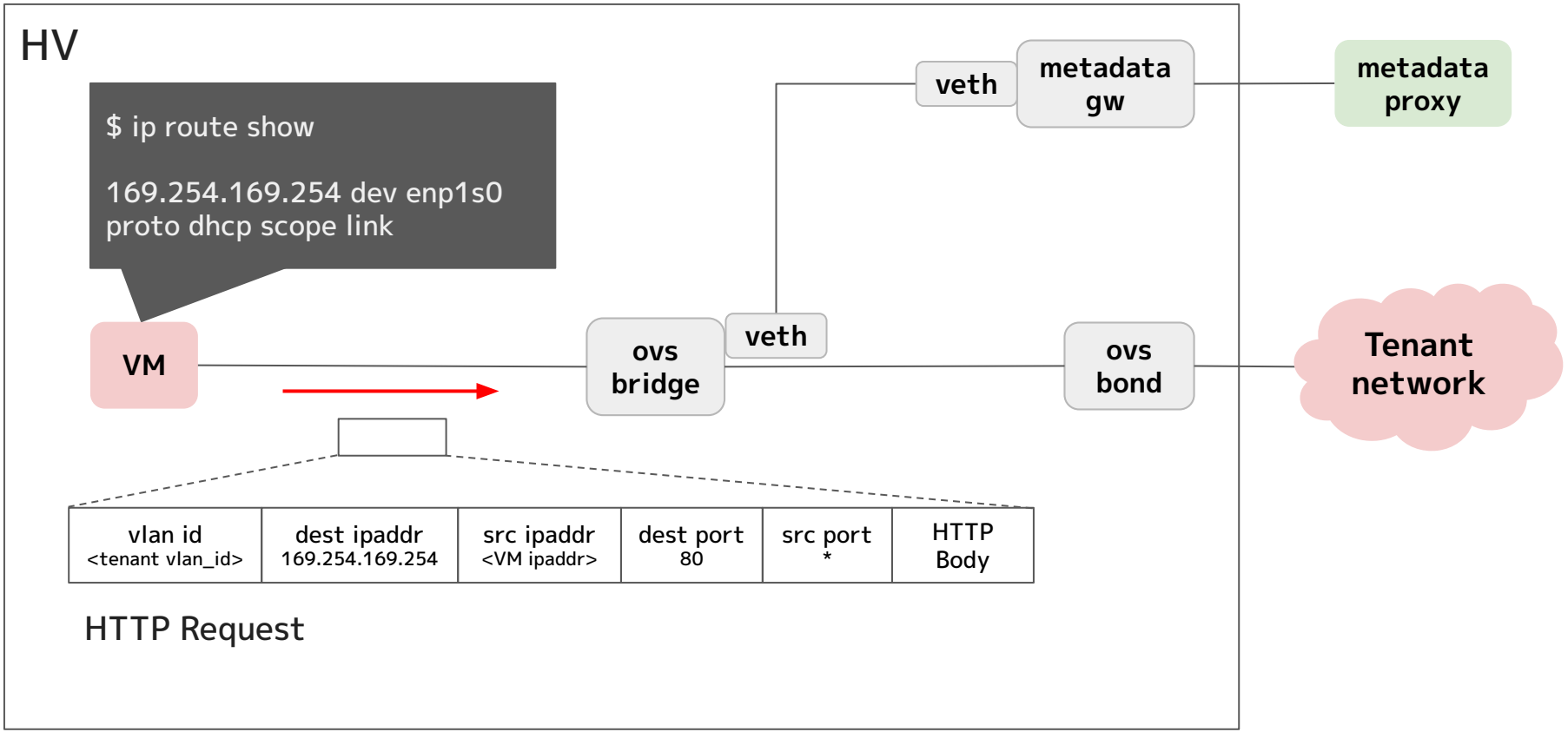
Metadata OVS 詳細（ARP / 帰り）



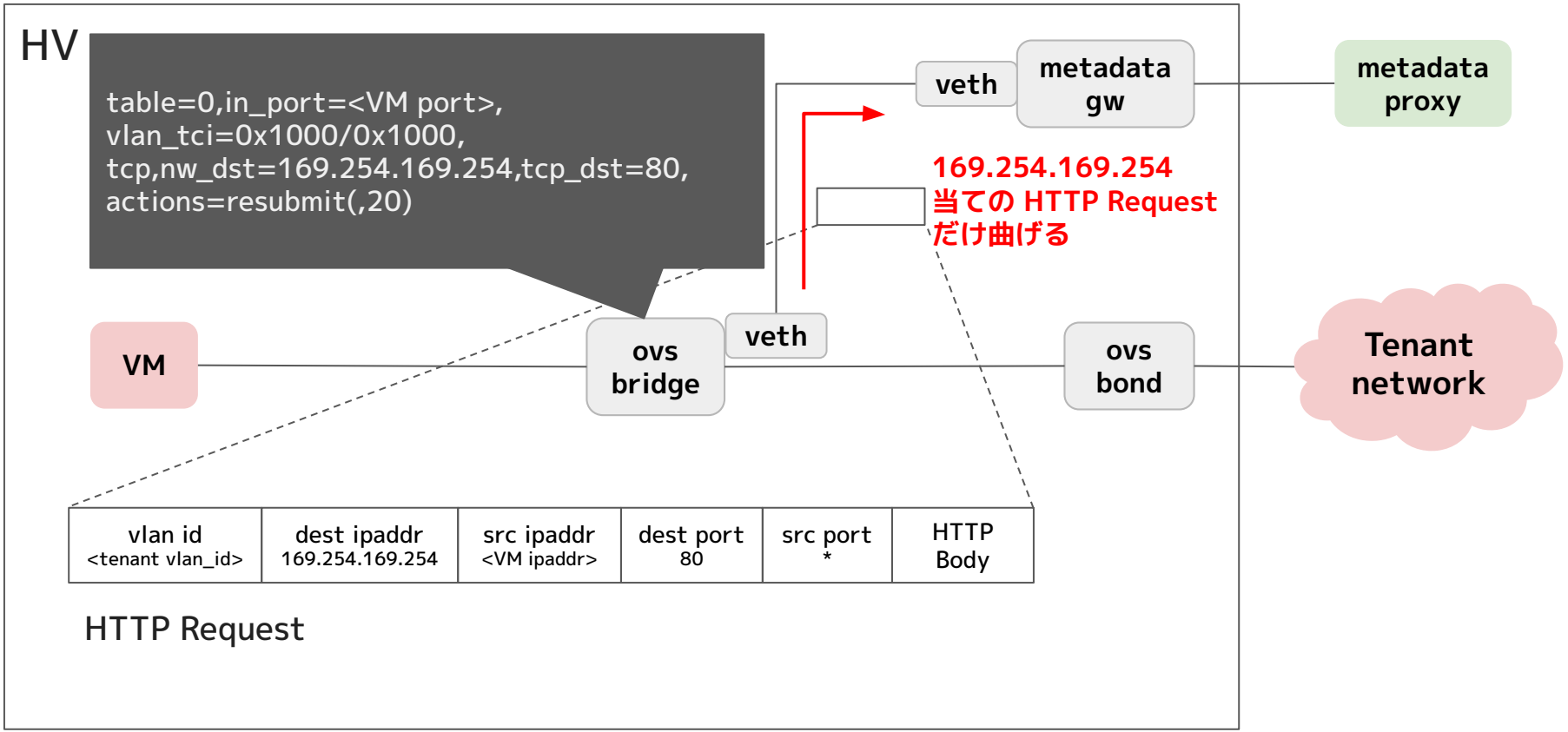
Metadata OVS 詳細（ARP / 帰り）



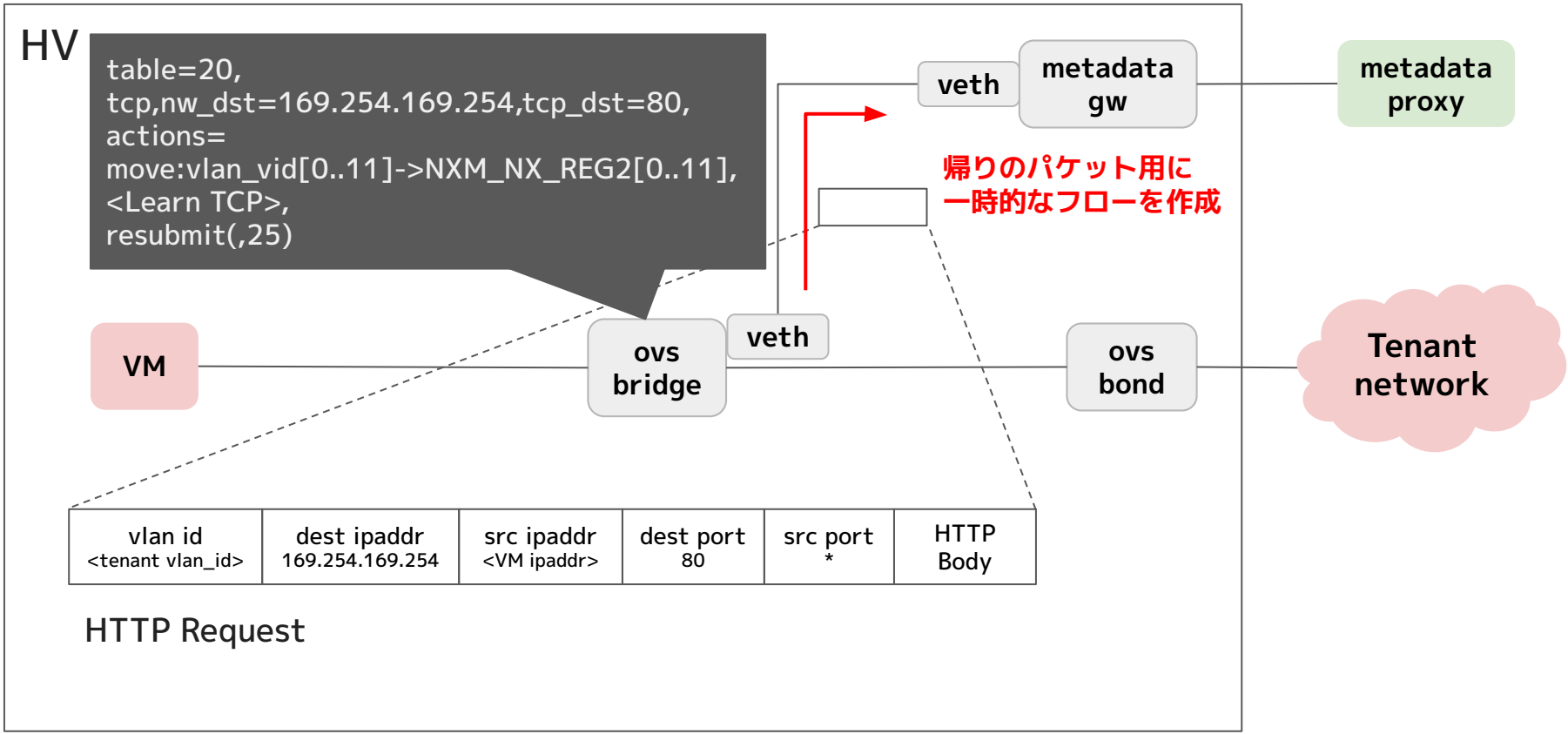
Metadata OVS 詳細（HTTP / 行き）



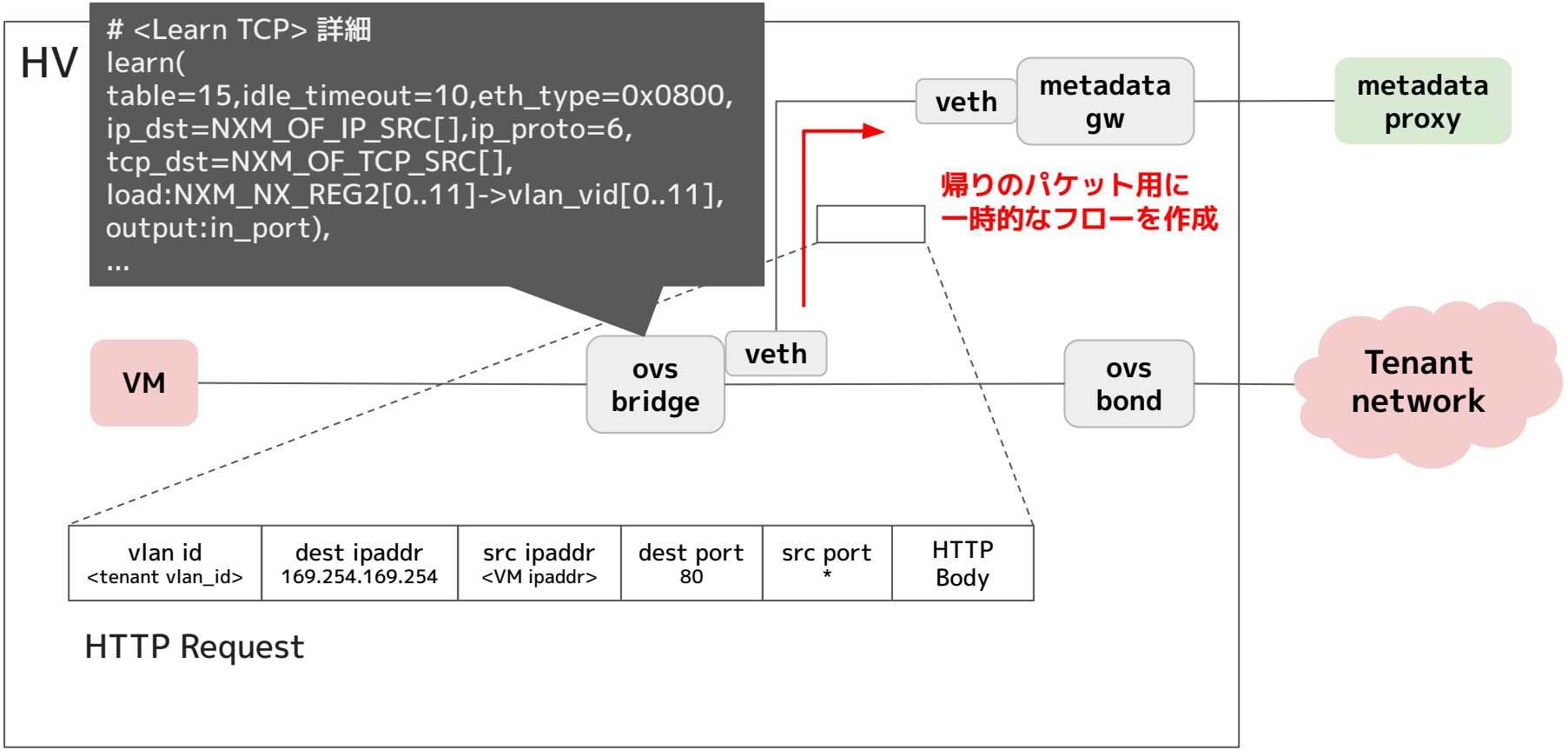
Metadata OVS 詳細（HTTP / 行き）



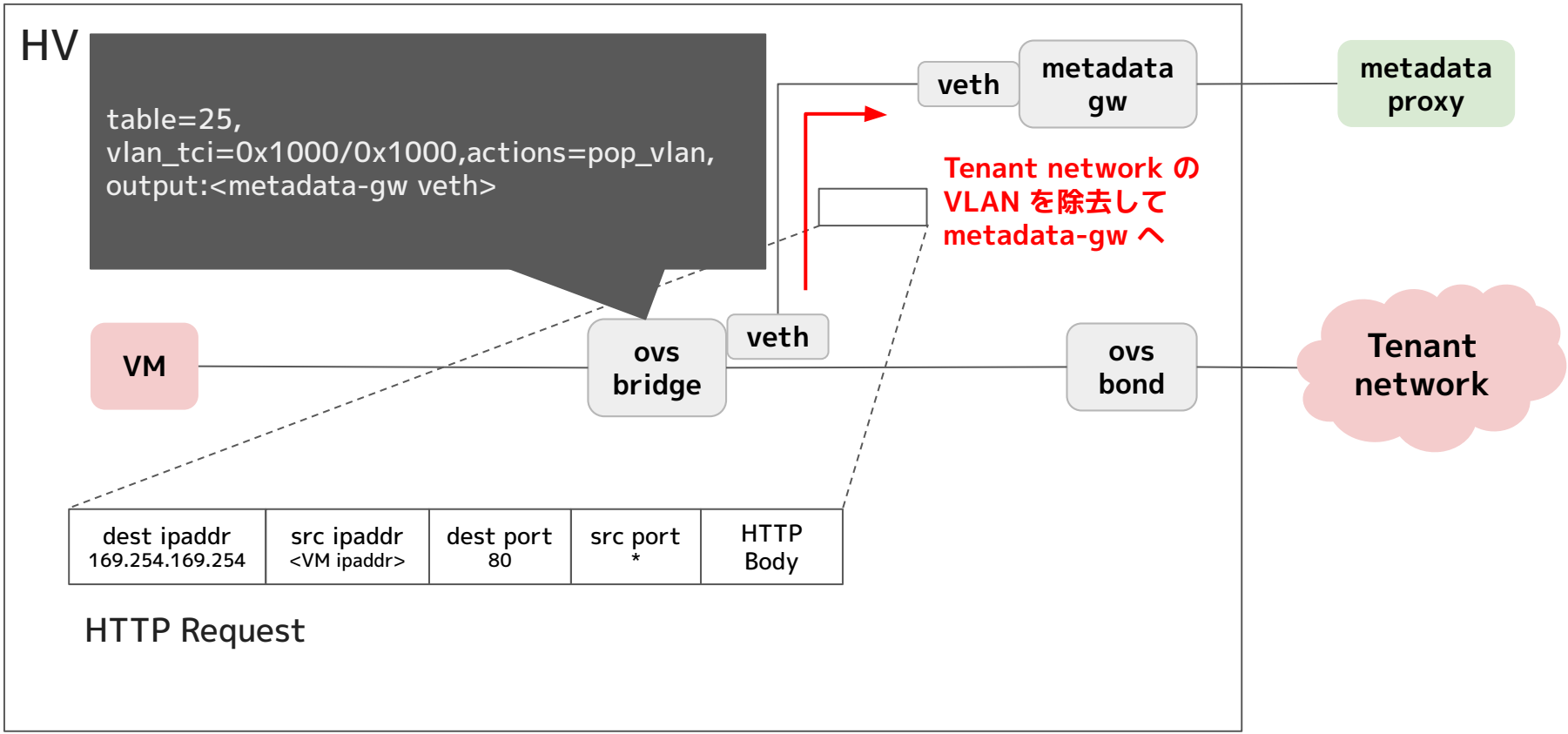
Metadata OVS 詳細（HTTP / 行き）



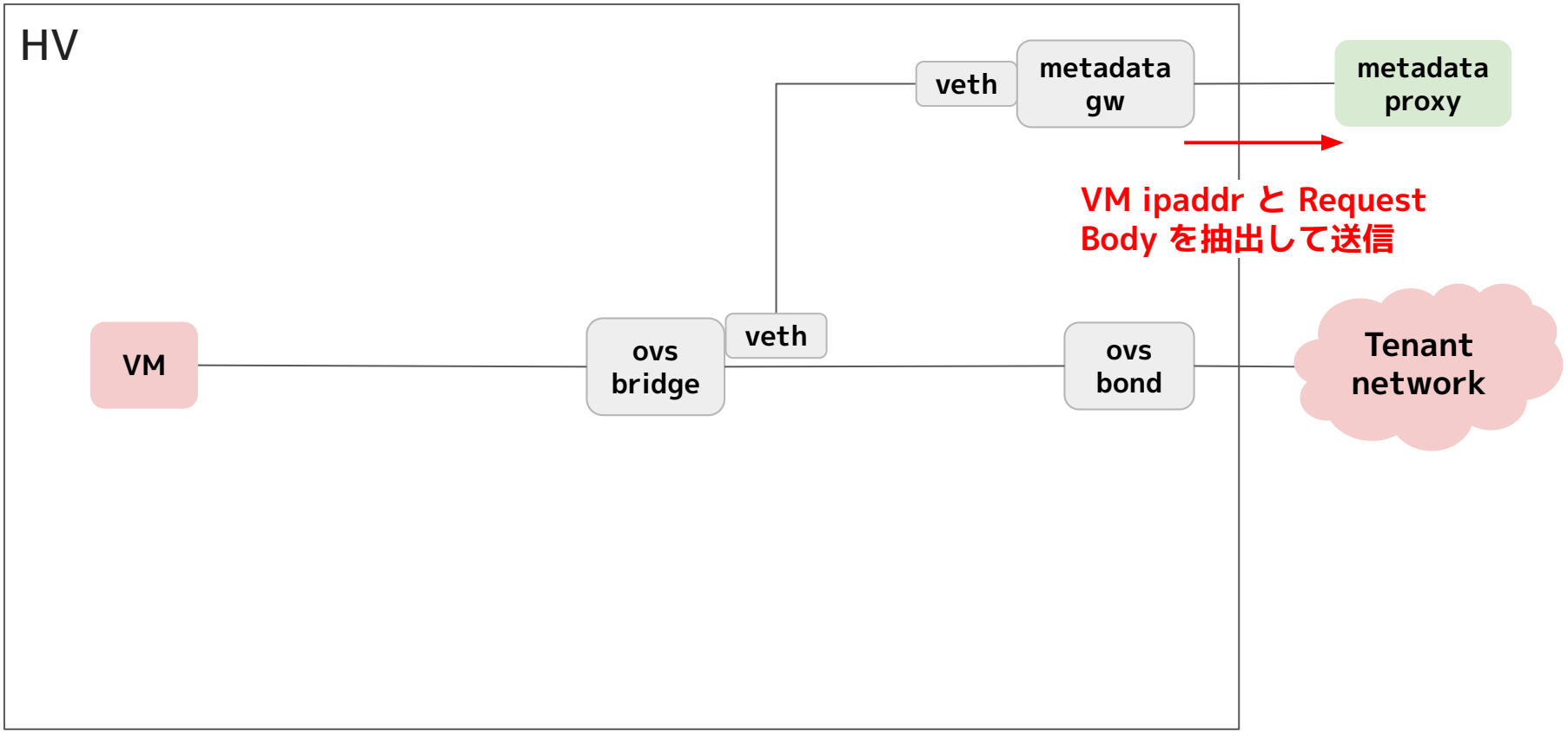
Metadata OVS 詳細（HTTP / 行き）



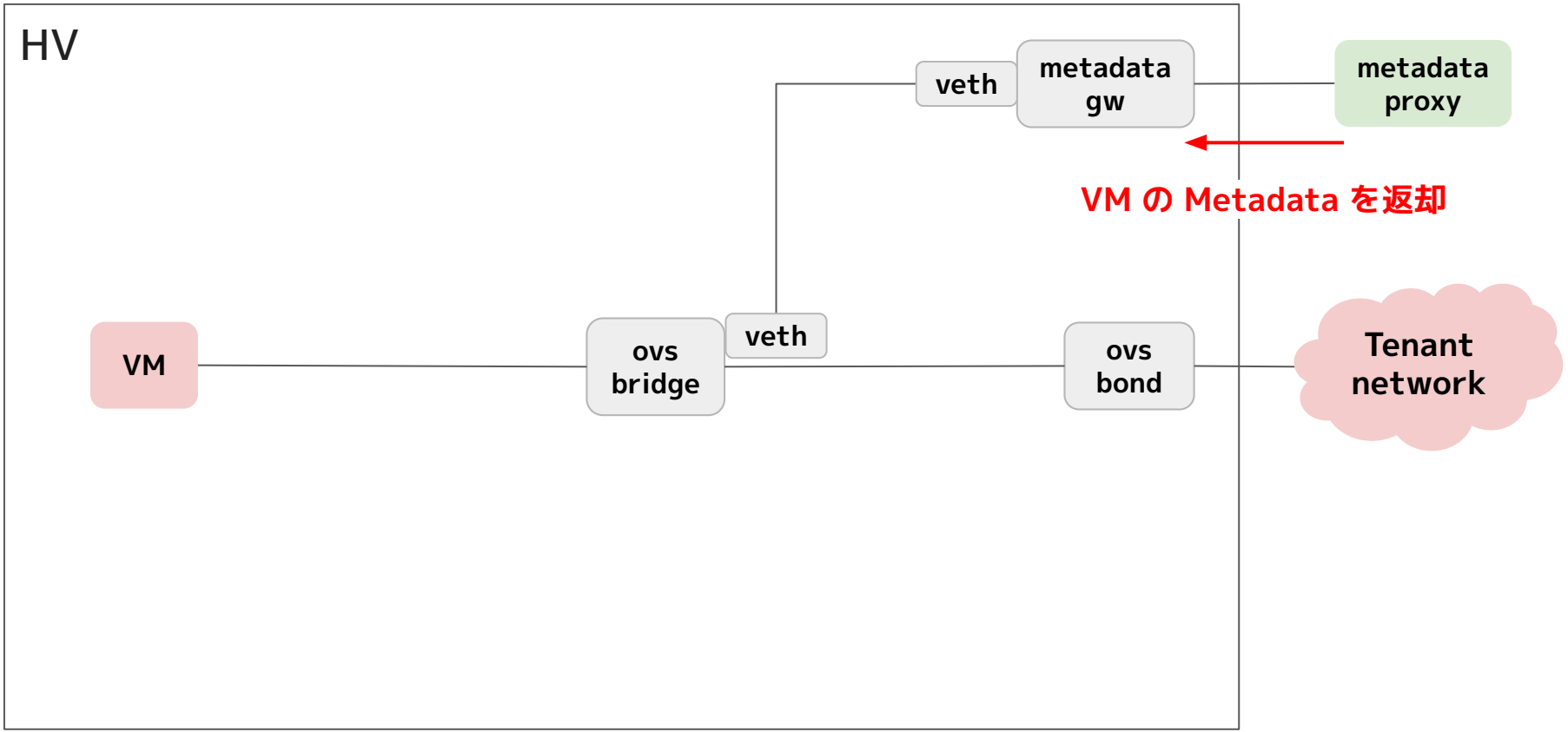
Metadata OVS 詳細（HTTP / 行き）



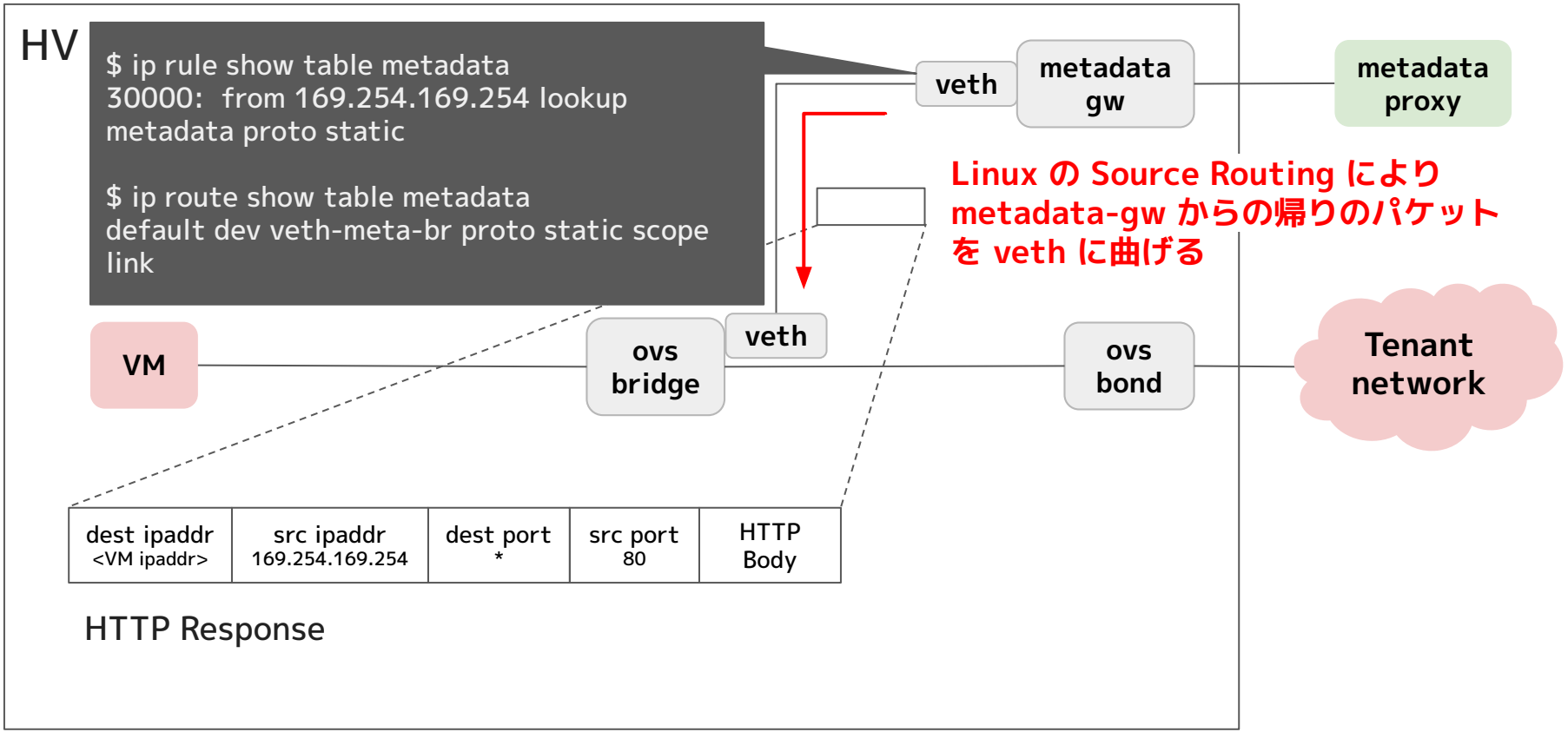
Metadata OVS 詳細（HTTP / 行き）



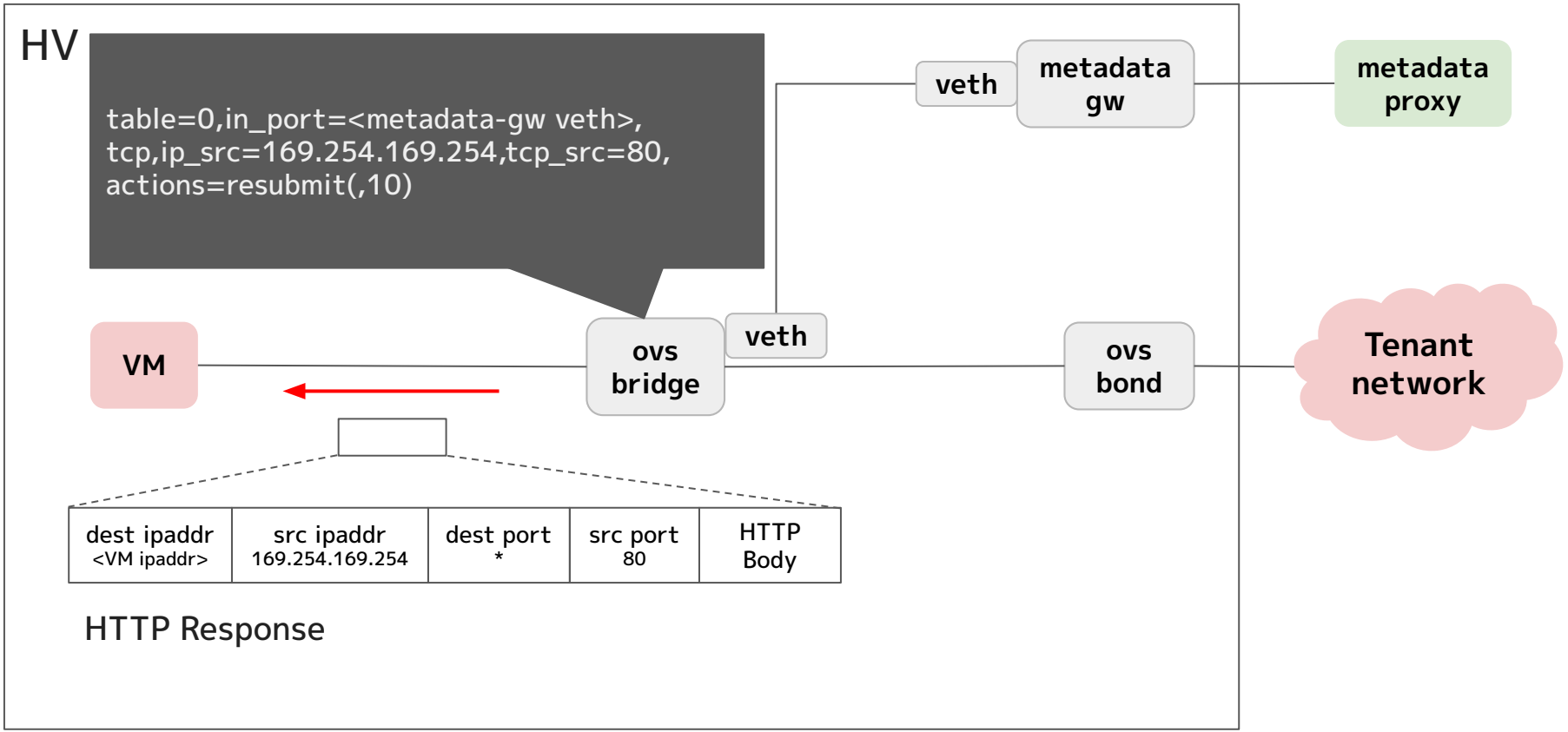
Metadata OVS 詳細（HTTP / 帰り）



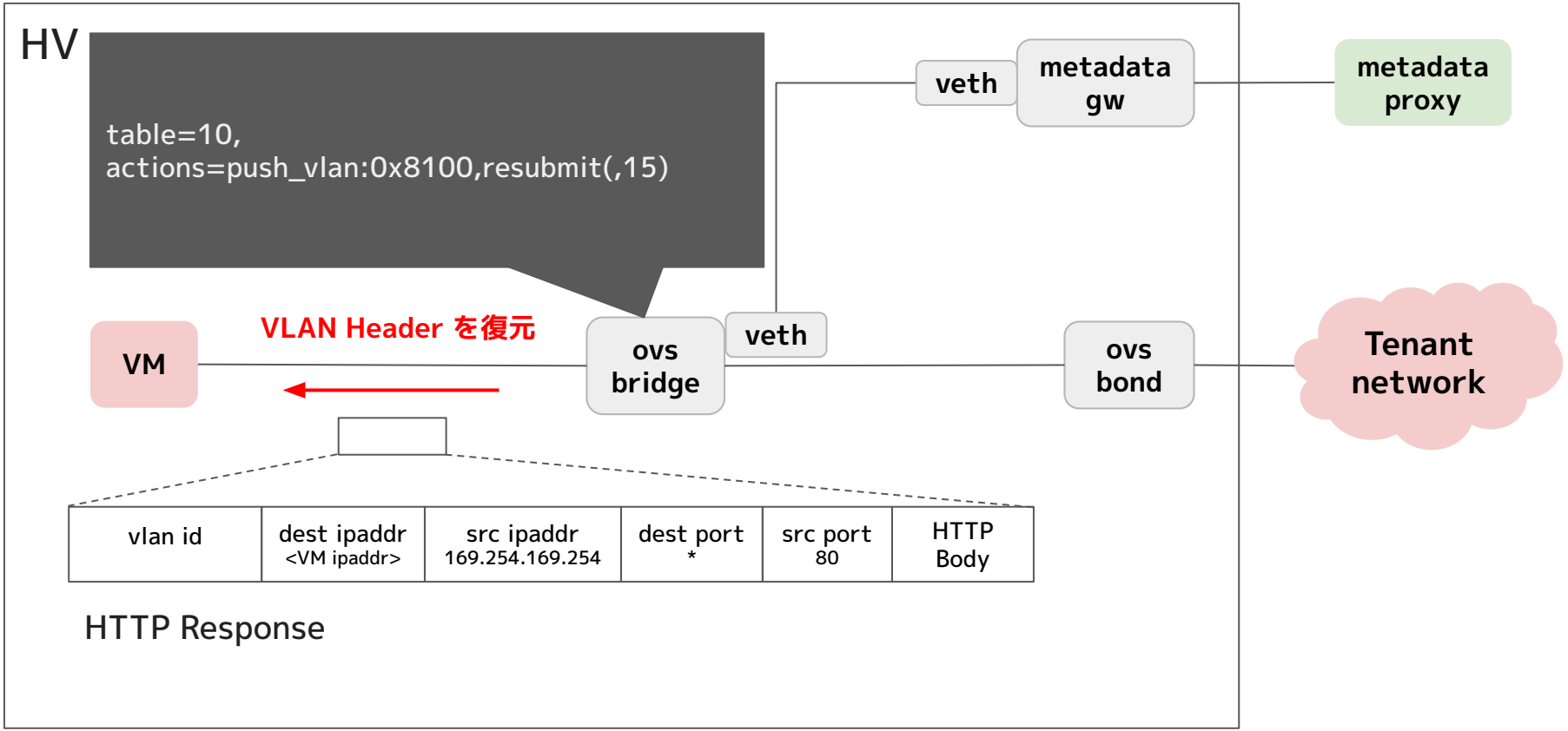
Metadata OVS 詳細（HTTP / 帰り）



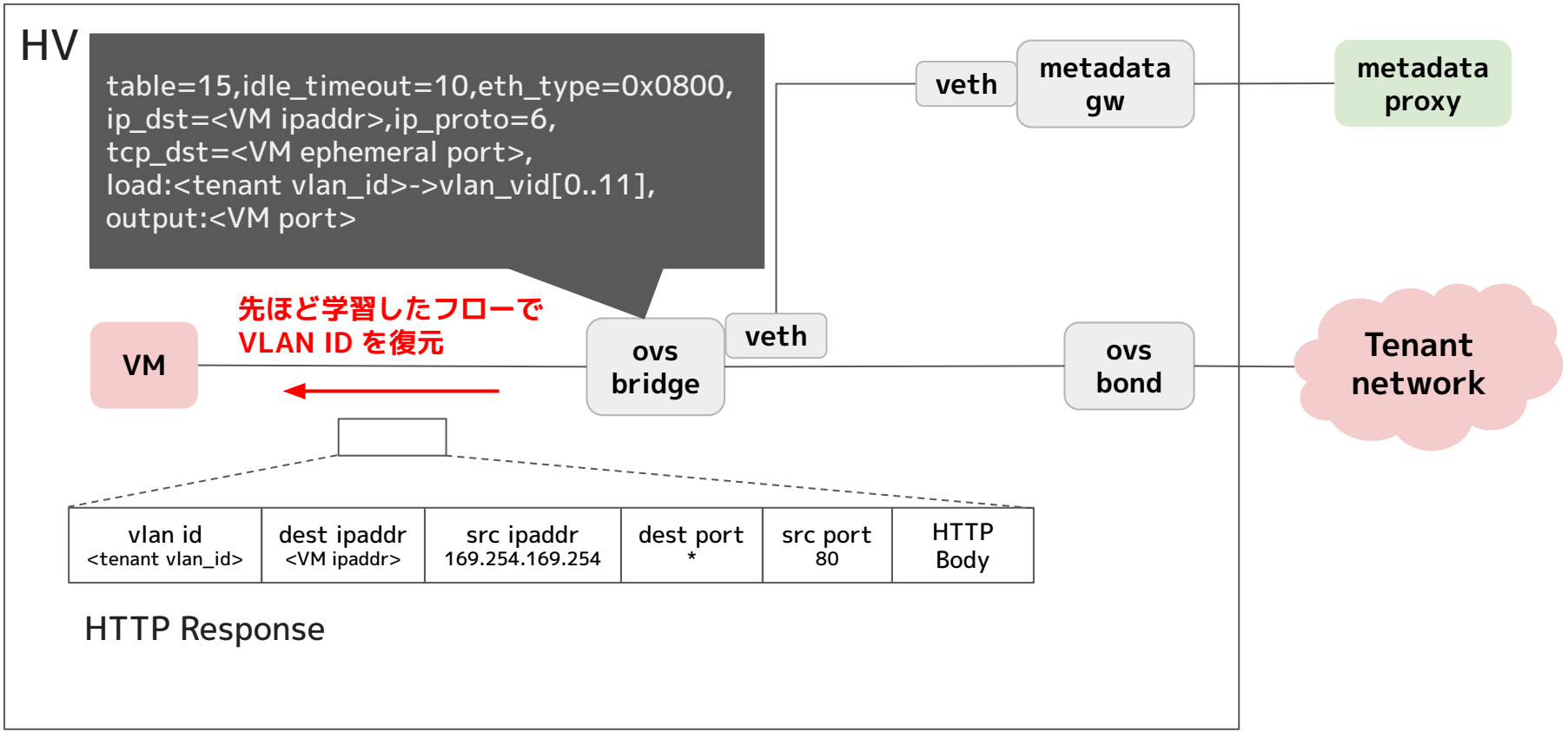
Metadata OVS 詳細（HTTP / 帰り）



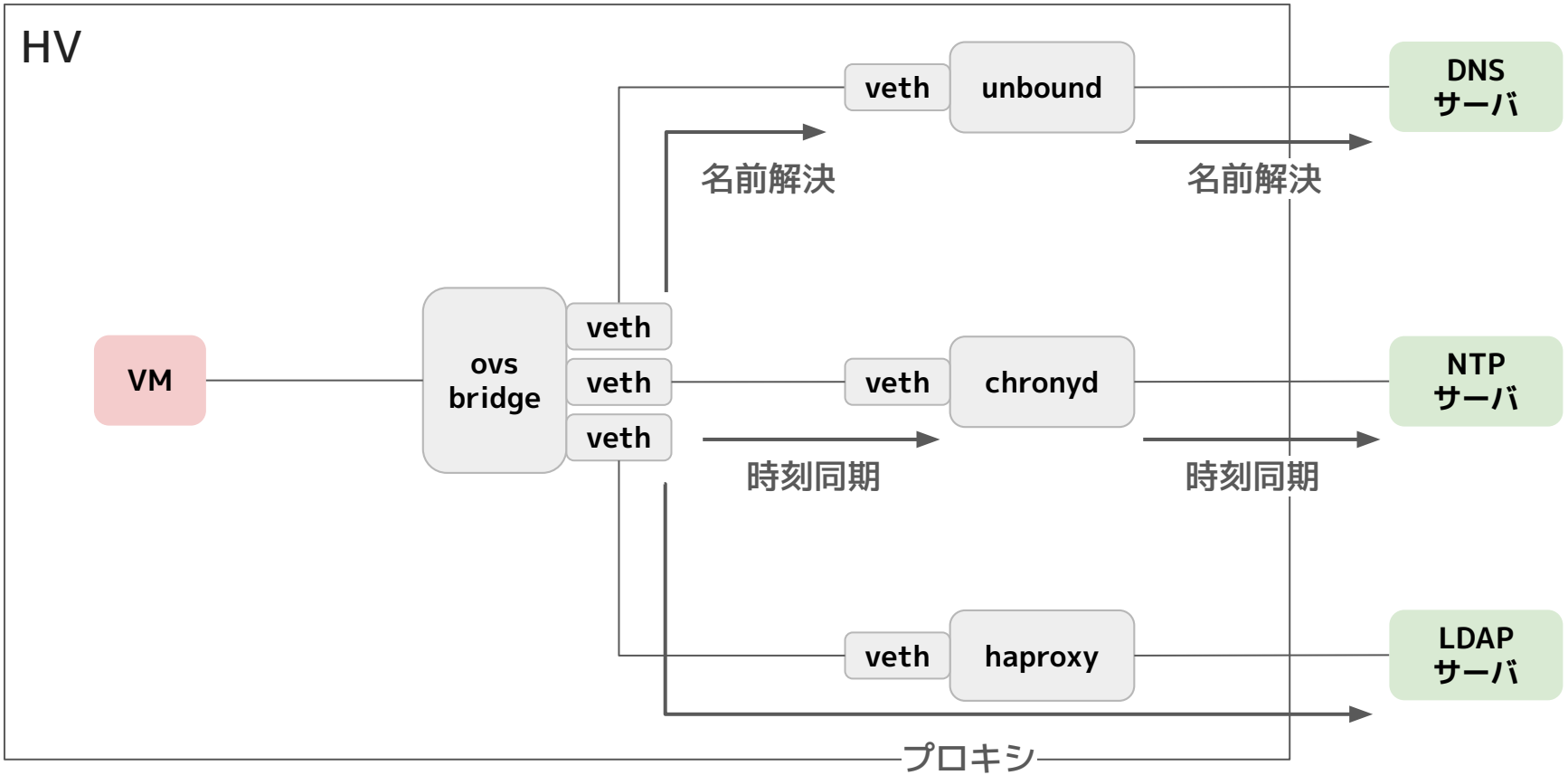
Metadata OVS 詳細（HTTP / 帰り）



sMetadata OVS 詳細（HTTP / 帰り）



DNS / NTP / LDAP への横展開



設計振り返り

- HV 上で通信を折り返すことで、**テナント分離を崩さず VM からの undercloud 疎通を実現**
- DHCP / Metadata / DNS / NTP / LDAP 等の様々なサービスを HV 上のプロキシで適切に処理
- OVS Flow Rule を用いて**対象の通信だけを捕捉**し、VLAN 情報を保持・復元して返りの通信も可能に
- Linux Source Routing で**帰りの通信だけ**を OVS データパスへ誘導

まとめ

本発表での学び

- 高いテナント分離性を満たす必要がある制約の中で、OVS と Linux Source Routing を用いて undercloud との疎通性を確保する手法
  - OVS による柔軟なフォワーディングやパケットの書き換え、データパスの設計
  - Source Routing によって特定の通信を OVS のデータパスに渡す手法

We are hiring!

一緒に働く仲間を大募集しています！

- プライベートクラウドや IaaS 基盤の開発・運用に興味がある方
- 仮想化基盤や仮想ネットワークの設計に携わることに意欲がある方

お気軽にお声がけください 👋

https://it.cyberagent.group/team/ciu/
https://www.cyberagent.co.jp/way/list/detail/id=31427
https://www.cyberagent.co.jp/way/list/detail/id=31367
https://www.cyberagent.co.jp/way/list/detail/id=29497