Design/Architecture Proposal Refactoring AWS VPC connectivity to eliminate the AWS VPC limits

Working group: David Nolan

Executive Summary

This is the design proposal for a new generation of connectivity between YBA and customer private VPCs in AWS. By using Transit Gateways instead of VPC Peering we eliminate the current limits which AWS imposes on us.

Problem Statement

A single AWS VPC can only have VPC Peerings to 125 other AWS VPCs. In our current hub-and-spoke model of connectivity from the YBA VPC to the VPCs for databases, this means a single YBA VPC (with multiple YBAs), collectively a "platform network", can only manage clusters in 125 VPCs total.

That includes shared VPCs, of which there are 23 (one per region). That means only about 100 customer dedicated VPCs can be managed by a single platform network. But each customer may have multiple VPCs, so the total number of customers a platform network can manage might be as low as 20 (if each customer had 5 VPCs).

Today we work around this limitation by adding additional platform networks. But to eliminate this restriction we need to eliminate VPC peerings from the platform network to the database VPCs.

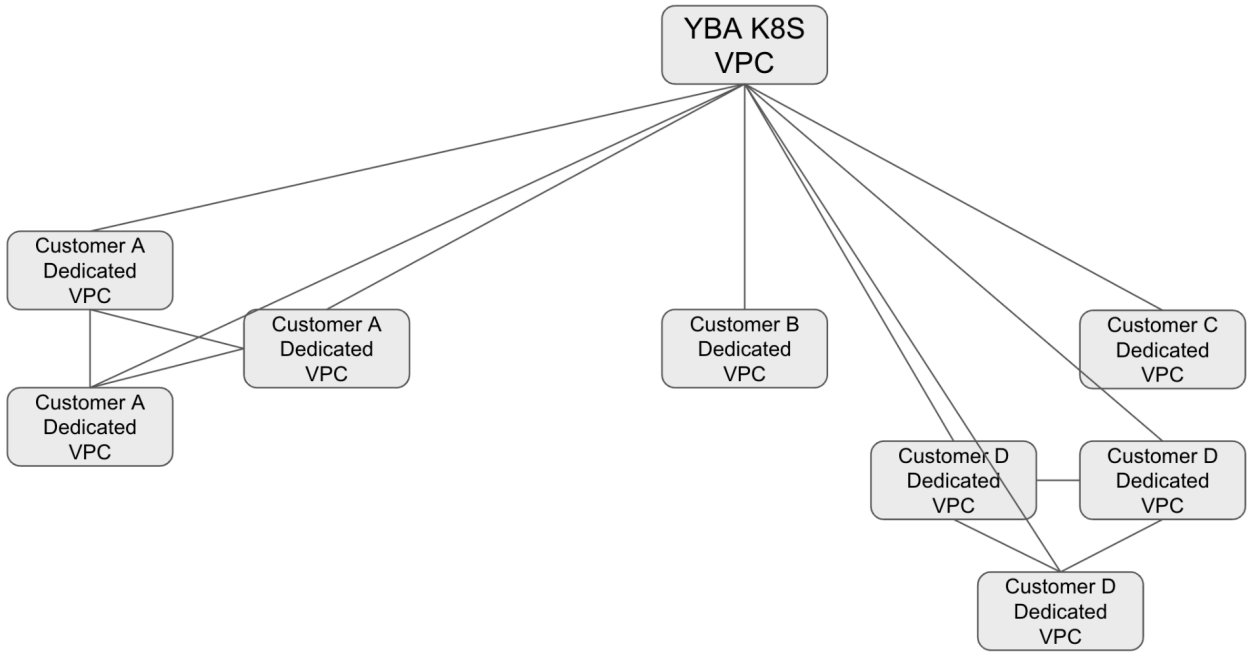
Functional Requirements

Use Cases(options)

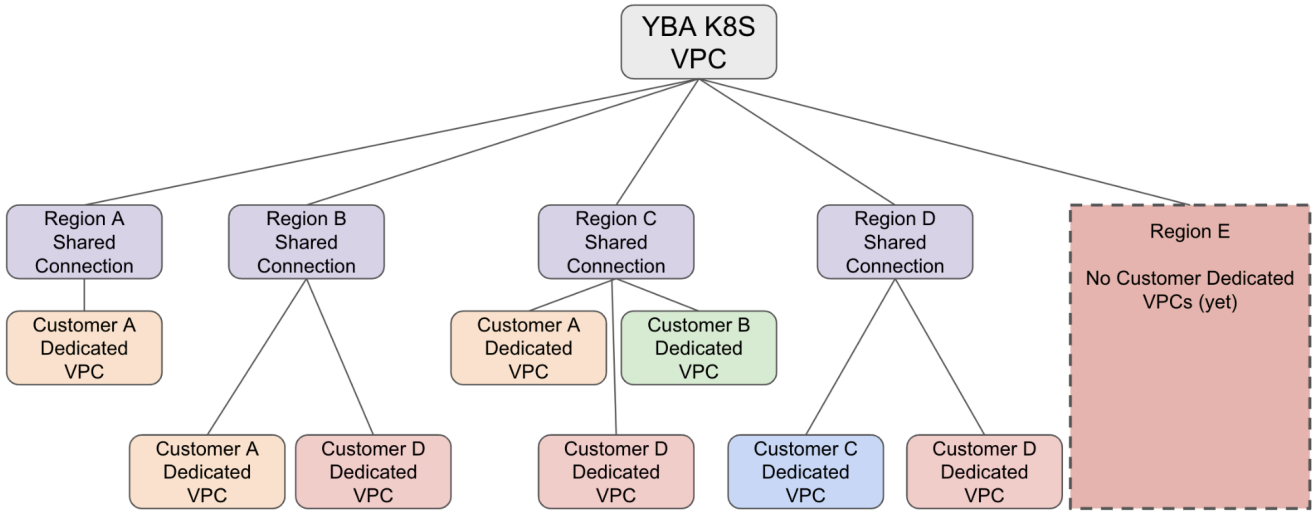
We need to be able to scale our customers without a linear scaling of our infrastructure. As more customers use VPCs, our solution for managing them must scale to handle many customers.

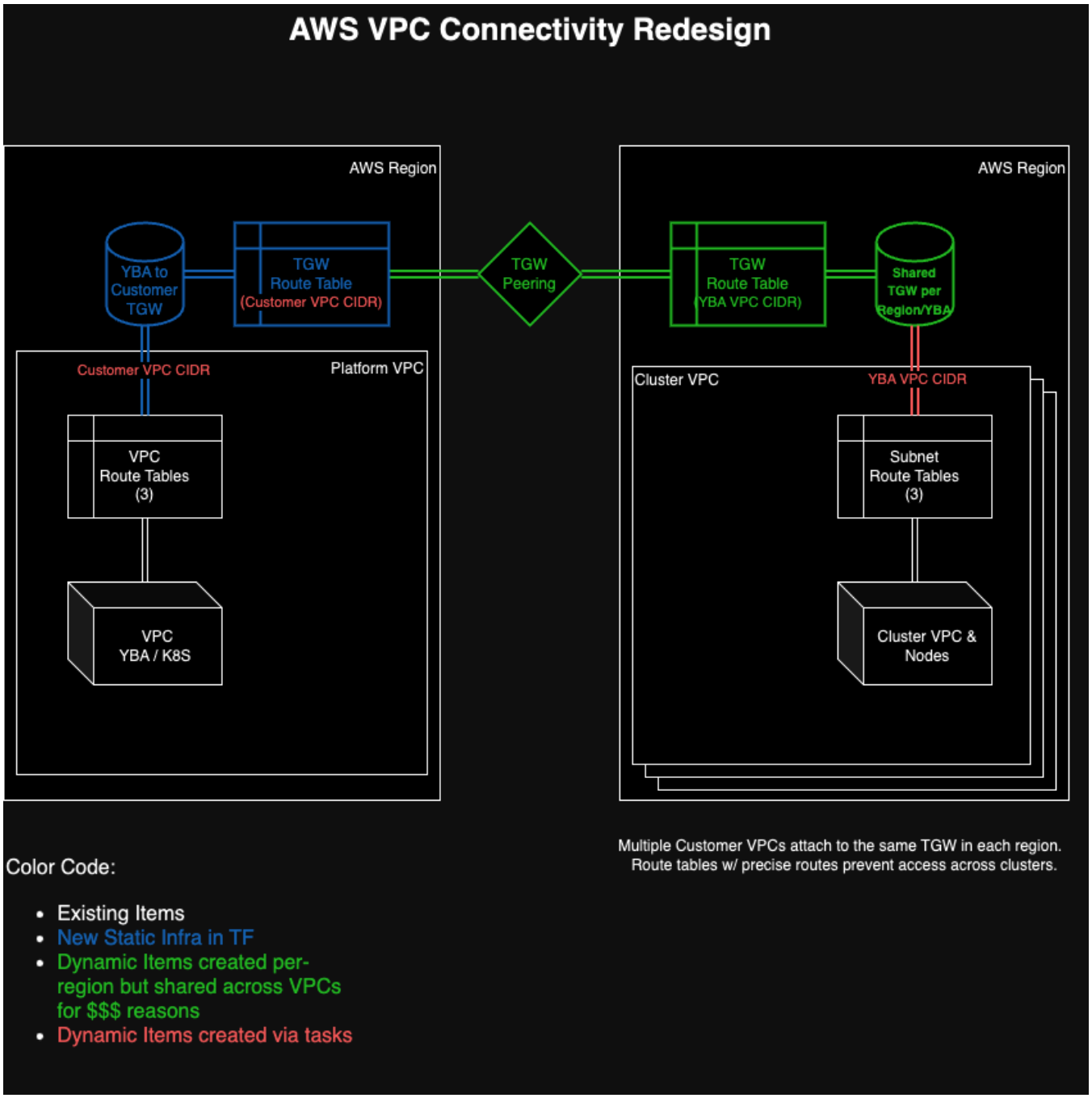
Transit Gateways are expensive to use, so the only TGWs that exist should be for Regions with active clusters in dedicated VPCs. Any region w/o active clusters in dedicated VPCs will not have a transit gateway (yet), and the shared VPCs (non-dedicated) will still be accessed via VPC peering as built today.

Current Solution(if any)



Proposed Solution





- Each Platform Network will have a statically built Transit Gateway, and route table, and be attached to the VPC for the platform network.
- Each customer region will have a *statically* built Transit Gateway, shared between all customer VPCs, but all attachments to that TGW, including the peering back to the platform network, will be dynamically built to save costs..
    - Peering from the DB-side TGW and the YBA-side TGW will be built only when the first customer cluster is deployed to a VPC in that region.
    - When no customer clusters exist in the region for a period of time (six hours proposed), the transit gateway peering will be deleted from the region.
    - When the TGW peering is created, appropriate route table entries will also be created to point to the platform network.
- Each customer VPC will be awttached to the TGW, when a cluster needs to be deployed

- A TGW Attachment of type VPC is created, linking the TGW to the VPC in the same region
- Route table entries are created:
  - in the customer VPC pointing to the TGW for the platform network VPC
  - in the platform network VPC pointing to the platform-network TGW, for the customer VPC CIDR
  - in the platform-network TGW route table, pointing to the peered TGW, for the customer VPC CIDR

Detailed Design

API Specification

No changes, all internal?

Object Model/Database Schema

Draft from Dmitry & Artem:
https://drive.google.com/file/d/1pBdUhJ2U2Yl5iHC1GGwiWMt20387sEG0/view?usp=sharing

Version Compatibility/Upgrade

Existing VPCs will remain as-is, using VPC peering. Only new VPCs moving forward will use the transit gateways.

VPC Peering *between* customer VPCs will remain as is. Only the network connectivity between the platform network and the customer dedicated VPCs is being included in this design.

Deployment/Implementation Plan

Terraform will be used to create all static components, and provide a list of those components to the YBM apiserver via configuration.

The APIServer will use the AWS SDK to create and manage all other AWS objects, and track the creation/deletion of those objects in its database.

Quality Attributes

Performance

Network performance of TGWs is expected to be comparable with VPC peering

HA

TGWs provide inherent redundancy as part of how AWS builds them

Security

Technically the TGW in each region will now be a shared component, used by multiple customers. Careful application of route entries to the VPC route table and TGW will prevent traffic from routing between two independent customer VPCs.

Alternatives Considered/Discarded

- Building our own connectivity (too complex and complicated, requires HA design, etc)
- Using a third party network routing vendor (just as expensive, but with an extra vendor involved, adds more complexity)

Links(related documents)

https://docs.aws.amazon.com/vpc/latest/tgw/tgw-getting-started.html

Action Items

- David Nolan review whether AWS TGW are free when nothing is attached to them, and adjust above plan if so
-