AWS에서 Multi-TxQs로 Catalyst 8000V의 처리량 개선

목차

소개

이 문서에서는 처리량 성능을 개선하기 위해 AWS 환경에 구축된 Catalyst 8000V에서 Multi-TXQ를 활성화하고 활용하는 방법에 대해 설명합니다.

배경 정보

여러 큐가 있으면 수신 및 발신 패킷을 특정 vCPU에 매핑하는 프로세스가 간소화되고 가속화됩니다. Catalyst 8000V에서 Multi-TXQ를 활용하면 할당된 가용 데이터 플레인 코어에서 효율적인 코어 활용을 통해 처리량 성능을 높일 수 있습니다. 이 문서에서는 Multi-TXQ의 작동 방식, 구성 방식, Autonomous 및 SD-WAN Catalyst 8000V 구축에 대한 샘플 CLI 컨피그레이션, 문제 해결 명령을 검토하여 성능 병목을 찾는 방법에 대해 간략하게 설명합니다.

Catalyst 8000V Multi-TxQs를 사용하지 않을 때의 동작

17.18 소프트웨어가 릴리스될 때까지 Catalyst 8000V에 입력되는 패킷은 플로우에 관계없이 모든 vCPU(패킷 처리 코어)에 배포됩니다. PP가 패킷 처리를 완료하면 플로우 순서가 복원되어 인터페이스에서 전송됩니다.

패킷을 전송 큐(TxQ)에 배치하기 전에 Catalyst 8000V는 인터페이스당 하나의 TxQ를 생성합니다. 따라서 사용 가능한 이그레스 인터페이스가 하나뿐인 경우 여러 플로우가 하나의 TxQ로 이동합니다.

Catalyst 8000V는 사용 가능한 인터페이스가 하나뿐인 경우 이 Multi-TxQ 프로세스를 활용할 수 없습니다. 이로 인해 처리량 성능 병목 현상이 발생하고 가용 데이터 플레인 코어에 로드 분배가 불균형해집니다. C8000V 인스턴스에서 데이터를 전송하는 데 이그레스 인터페이스가 하나만 사용되는 경우, 네트워크 트래픽을 전송하는 데 사용할 수 있는 TxQ가 하나뿐이며, 단일 큐가 더 빨리 가득 차서 패킷이 삭제될 수 있습니다.

참고로, 그림 1에서 AWS에 구축된 Catalyst 8000V용 단일 TxQ 아키텍처 모델을 찾을 수 있습니다.

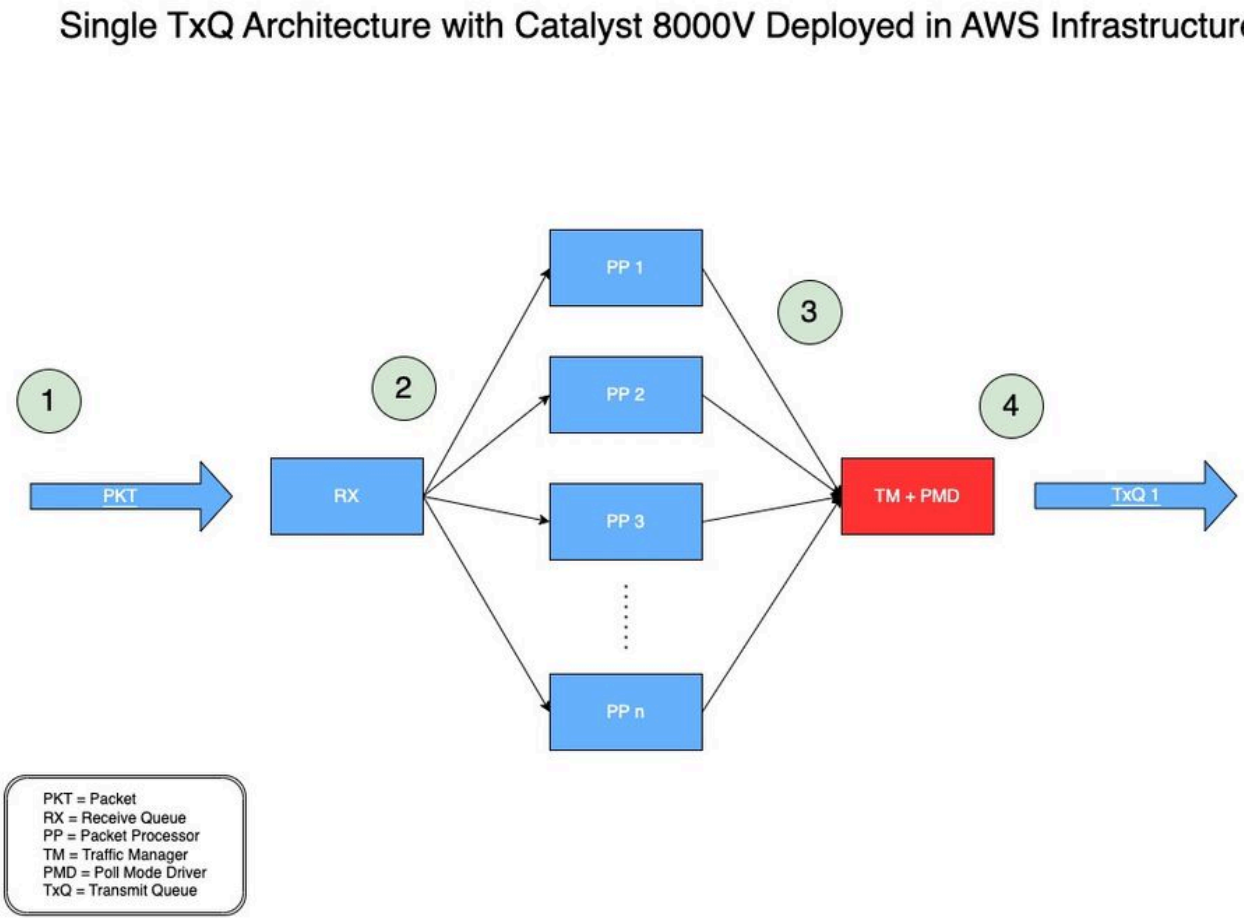


그림 1: AWS에 구축된 Catalyst 8000V용 단일 TxQ 아키텍처 모델.

1. PKT(네트워크 패킷)가 VPC를 통해 이동하고 C8000V의 인그레스 인터페이스에서 수신됩니다.
2. PKT는 수신 큐(RX)에 배치되고 알고리즘에 의해 결정된 PP(Packet Processor) 엔진으

로 전달된다.
3. 패킷 프로세서(PP)가 패킷을 처리한 후 Traffic Manager(TM)로 패킷을 전송합니다.
4. TM 처리가 끝나면 하나의 코어에서 사용 가능한 하나의 TxQ에 패킷을 배치한 다음 Catalyst 8000V의 이그레스(egress) 인터페이스로 전달합니다.

AWS 인프라의 Multi-TXQ란 무엇입니까?

AWS ENA는 내부 오버헤드를 줄이고 확장성을 높이기 위해 다중 전송 큐(Multi-TxQ)를 제공합니다. 여러 큐가 있으면 수신 및 발신 패킷을 특정 vCPU에 매핑하는 프로세스가 간소화되고 가속화됩니다. AWS 및 DPDK 네트워크 참조 모델은 플로우 기반입니다. 각 vCPU는 플로우를 처리하고 해당 플로우의 패킷을 할당된 전송 큐(TxQ)로 전송합니다. 각 vCPU의 RX/TX 큐 쌍은 플로우 기반 모델을 기반으로 유효합니다.

Catalyst 8000V는 플로우 기반이 아니므로 "각 vCPU의 RX/TX 큐 쌍"이라는 명령문은 Catalyst 8000V에 적용되지 않습니다.

이 경우 RX/TX 큐는 vCPU가 아니라 인터페이스별로 이루어집니다. RX/TX 큐는 애플리케이션(Catalyst 8000V)과 AWS 인프라/하드웨어 간의 인터페이스 역할을 하여 데이터/네트워크 트래픽을 전송합니다. AWS는 인스턴스별로 인터페이스당 사용할 수 있는 RX/TX 큐의 속도와 개수를 제어합니다.

Catalyst 8000V에 여러 개의 TxQ를 생성하려면 여러 개의 인터페이스가 있어야 합니다. 한 인터페이스에서 나가는 여러 흐름의 흐름 순서를 유지하기 위해(이 프로세스에 따라 Catalyst 8000V에서 여러 TxQ를 활성화하면) Catalyst 8000V는 5튜플을 기준으로 흐름을 해시하여 적절한 TxQ를 선택합니다. 사용자는 루프백 인터페이스 또는 보조 IP 주소를 사용하여 인스턴스에 연결된 동일한 물리적 NIC를 사용하여 Catalyst 8000V에서 여러 인터페이스를 생성할 수 있습니다.

그림 2에서 AWS에서 Catalyst 8000V를 사용하는 Multi-TxQ 아키텍처를 사용하여 패킷을 처리하는 방법을 확인할 수 있습니다.

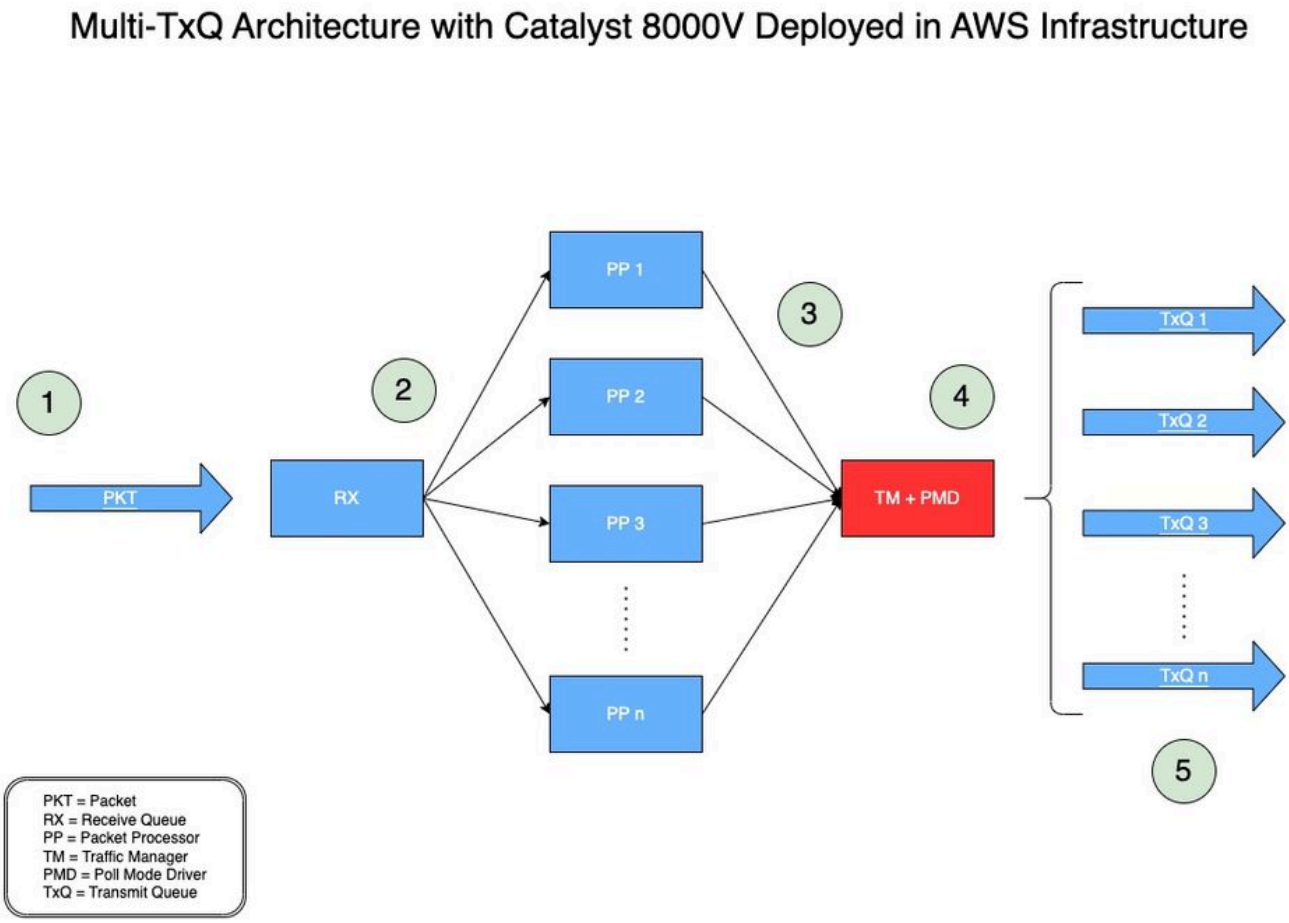


그림 2: AWS에 구축된 Catalyst 8000V용 다중 TxQ 아키텍처 모델.

1. PKT(네트워크 패킷)가 VPC를 통해 이동하고 C8000V의 인그레스 인터페이스에서 수신됩니다.
2. PKT는 수신 큐(RX)에 배치되고 알고리즘에 의해 결정된 PP(Packet Processor) 엔진으로 전달된다.
3. 패킷 프로세서(PP)가 패킷을 처리한 후 Traffic Manager(TM)로 패킷을 전송합니다.
4. TM 처리가 끝나면 패킷을 전송 큐(TxQ)에 배치하기 전에 TM은 패킷 헤더를 확인하고 패킷을 해시합니다(다음 섹션에서 설명). 또 다른 구성 요소인 PMD(Poll Mode Driver)는 인스턴스에서 지원하는 TXQ 수를 구성하는 데 사용됩니다. 하나의 코어는 할당된 TxQ에 패킷을 해싱하고 전송하는 TM + PMD 기능 전용입니다.
5. TxQ는 인스턴스에서 지원하는 TxQ 수와 함께 해시된 5개의 튜플 및 모듈로(modulo)를 기반으로 선택됩니다. 패킷은 선택한 TxQ에 배치되고 Catalyst 8000V의 이그레스 인터페이스로 전달됩니다.

트래픽이 다중 TxQs로 해시되는 방법

그림 2의 4단계와 같이 TM 처리가 끝나면 TM은 패킷을 TxQ에 배치하기 전에 패킷 헤더를 보고 5개의 튜플(목적지 주소, 소스 주소, 프로토콜, 목적지 포트, 소스 포트)을 추출하여 패킷을 TxQ에 해시합니다.

TxQ는 인스턴스에서 지원하는 TxQ 수와 함께 해시된 5개의 튜플 및 모듈로(modulo)를 기반으로 선택됩니다.

다중 TxQs를 지원하는 Catalyst 8000V Software 버전

동일한 인스턴스 패밀리 유형의 AWS EC2 인스턴스는 모두 인스턴스 크기에 따라 서로 다른 수의 TXQ를 지원합니다. C8000V는 IOS® XE 17.7부터 여러 TxQ를 지원하기 시작했습니다.

IOS® XE 17.7부터 C8000V는 C5n.9xlarge에서 최대 8개의 TXQ를 포함할 수 있는 여러 개의 TxQ를 지원합니다.

IOS® XE 17.9부터 C8000V는 C5n.18xlarge 인스턴스 크기를 지원하며, 최대 12개의 TXQ를 가질 수 있습니다(C5n.9xlarge보다 50% 더 많음).

IOS® XE 17.7에서는 Multi-TxQ가 지원되지만, 12개의 TxQ를 지원하므로 소프트웨어 라이프사이클 및 더 높은 처리량 성능 기능을 모두 지원하는 데 IOS® XE 17.9를 사용하는 것이 좋습니다.

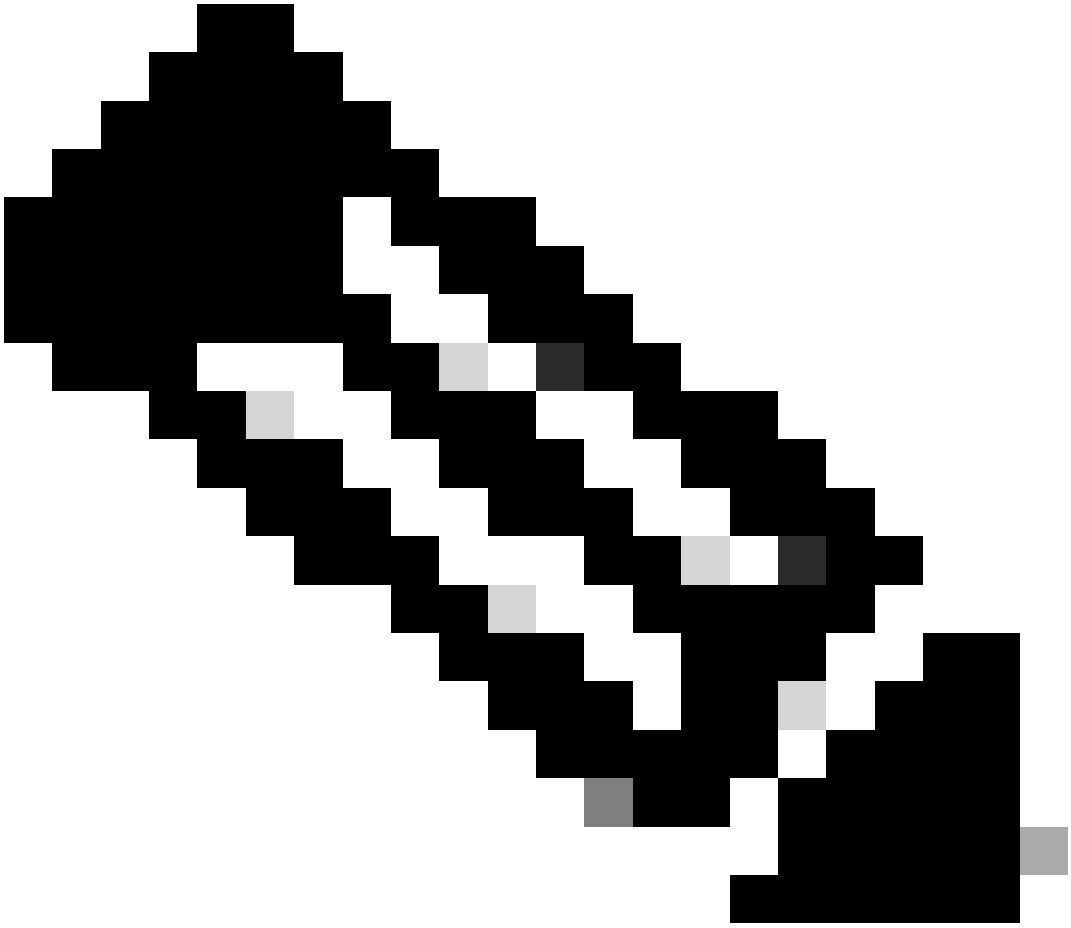
해싱을 계산하기 위해 IP 주소 지정 체계를 설계하는 방법

사용 가능한 모든 TxQ 간에 트래픽을 균등하게 해시하려면 Catalyst 8000V에서 IPsec/GRE 터널을 종료할 때 특수 IP 주소를 사용해야 합니다.

이러한 터널 종료를 담당하는 Catalyst 8000V 인터페이스를 구성하는 데 사용할 특수 IP 주소를 생성하는 데 사용할 수 있는 공개 스크립트가 있습니다. 이 섹션에서는 스크립트를 다운로드하고 사용하여 Multi-TxQ 해싱에 필요한 IP 주소를 엔지니어링하는 방법에 대한 지침을 제공합니다.

Catalyst 8000V에서 TCP/UDP와 같은 일반 텍스트 트래픽을 처리하는 경우 특별한 IP 주소 지정 체계가 필요하지 않습니다.

원래 지침은 여기에서 확인할 수 있습니다. https://github.com/CiscoDevNet/python-c8000v-aws-multitx-queues/



참고: 17.18 이상을 실행하는 Catalyst 8000V의 경우 패킷이 다르게 배포됩니다. 따라서, 다른 해싱 알고리즘이 사용될 필요가 있다.

사전 요구 사항

- Python 스크립트를 실행할 수 있는 Linux/MacOS 또는 Windows 시스템이 있어야 합니다.
- Python 버전이 3.8.9 이상인지 확인합니다. 'python3 —version'을 사용하여 Python 버전 확인
- 아직 설치되지 않은 경우 PIP를 설치합니다. 그렇지 않은 경우 다음을 실행합니다.
  - curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py
  - python3 get-pip.py

'python3 —version' 명령을 사용하여 시스템에서 사용 중인 Python 버전을 확인할 수 있습니다.

```
user@computer ~ % python3 --version

Python 3.9.6
```

Python 버전이 확인되어 실행 중이면 3.8.9 이상의 버전을 설치하고 최신 버전의 PIP를 설치합니다.

```
user@computer ~ % curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py

  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current

                                 Dload  Upload   Total   Spent    Left  Speed

100 2570k  100 2570k    0     0  6082k      0 --:--:-- --:--:-- --:--:-- 6135k
```

<#root>

```
user@computer ~ % python3 get-pip.py

Defaulting to user installation because normal site-packages is not writeable
Collecting pip
  Downloading pip-23.3.1-py3-none-any.whl.metadata (3.5 kB)
Downloading pip-23.3.1-py3-none-any.whl (2.1 MB)

   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.1/2.1 MB 7.4 MB/s eta 0:00:00

Installing collected packages: pip

  WARNING: The scripts pip, pip3 and pip3.9 are installed in '/Users/name/Library/Python/3.9/bin' which

  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-scri

Successfully installed pip-23.3.1


[

notice

]

 A new release of pip is available: 21.2.4 -> 23.3.1


[

notice

]

 To update, run: /Applications/Xcode.app/Contents/Developer/usr/bin/python3 -m pip install --upgrade pi
```

가상 환경 만들기

사전 요구 사항이 설치되면 가상 환경을 만들고 Multi-TxQ에 대한 고유한 IP 주소 체계를 생성하는 데 사용되는 IP 주소 해싱 스크립트를 다운로드합니다.

명령 요약:

1. python3 -m venv c8kv-hash
2. cd c8kv-hash
3. 원본 저장소/활성화
4. git 복제 https://github.com/CiscoDevNet/python-c8000v-aws-multitx-queues/
5. cd c8kv-aws-pmd-hash
6. python3 -m pip 설치 - 업그레이드 pip
7. pip install -r requirements.txt

Python의 가상 환경은 다른 프로젝트나 종속 항목에 영향을 주지 않는 격리된 작업 공간을 만드는 데 사용됩니다. 다음 명령을 사용하여 가상 환경 'c8kv-hash'를 생성합니다.

```
user@computer Desktop % python3 -m venv c8kv-hash
```

가상 환경 내에서 'c8kv-hash' 폴더(이전에 생성됨)로 이동합니다.

```
user@computer Desktop % cd c8kv-hash
```

가상 환경을 활성화합니다.

```
user@computer c8kv-hash % source bin/activate
```

Multi-TxQ 해싱 파이썬 스크립트가 있는 리포지토리를 복제합니다.

```
(c8kv-hash) user@computer c8kv-hash % git clone https://github.com/CiscoDevNet/python-c8000v-aws-multit

Cloning into 'c8kv-aws-pmd-hash'...
remote: Enumerating objects: 82, done.
remote: Counting objects: 100% (82/82), done.
remote: Compressing objects: 100% (59/59), done.
remote: Total 82 (delta 34), reused 57 (delta 19), pack-reused 0
Receiving objects: 100% (82/82), 13.01 KiB | 2.60 MiB/s, done.
Resolving deltas: 100% (34/34), done.
```

리포지토리가 복제되면 'c8kv-aws-pmd-hash' 폴더로 이동합니다. 생성된 가상 환경에 위치하므로 최신 버전의 PIP를 설치합니다.

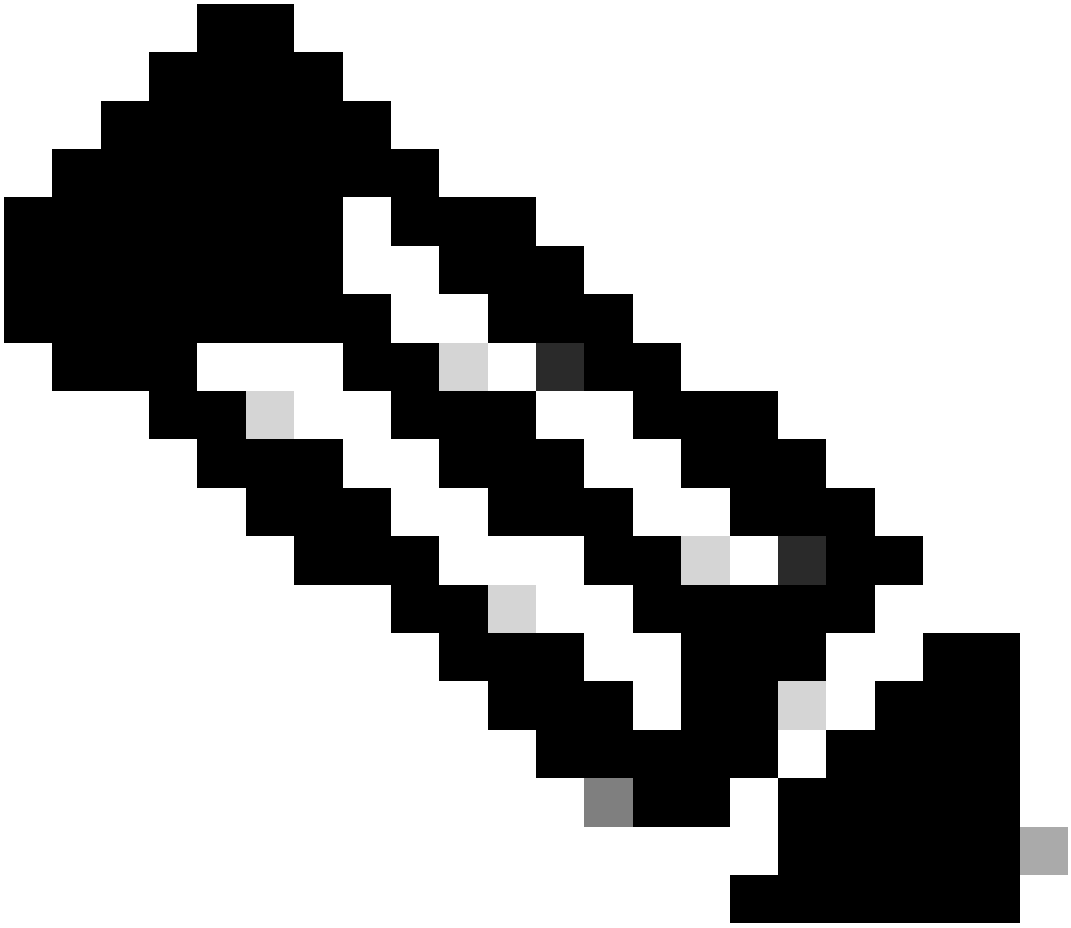
```
(c8kv-hash) user@computer c8kv-hash % cd c8kv-aws-pmd-hash
(c8kv-hash) user@computer c8kv-aws-pmd-hash % python3 -m pip install --upgrade pip

Requirement already satisfied: pip in /Users/name/Desktop/c8kv-hash/lib/python3.9/site-packages (21.2.4)
Collecting pip
  Downloading pip-23.3.1-py3-none-any.whl (2.1 MB)
     |████████████████████████████████| 2.1 MB 2.7 MB/s
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 21.2.4
    Uninstalling pip-21.2.4:
      Successfully uninstalled pip-21.2.4
Successfully installed pip-23.3.1
```

PIP가 업그레이드되면 requirements.txt 파일에 있는 종속성을 폴더에 설치합니다.

```
(c8kv-hash) user@computer c8kv-aws-pmd-hash % pip install -r requirements.txt
Collecting crc32c==2.3 (from -r requirements.txt (line 1))
  Downloading crc32c-2.3-cp39-cp39-macosx_11_0_arm64.whl (27 kB)
Installing collected packages: crc32c
Successfully installed crc32c-2.3
```

가상 환경이 최신 상태이며 Multi-TxQ에 대한 IP 주소 체계를 생성하는 데 사용할 수 있습니다.

17.7 및 17.8 릴리스에 대해 Python Hash Index Script를 사용하여 IP 주소 체계 계산(감가상각)



참고: 7.7 및 17.8 해시 스크립트는 곧 사용이 중단됩니다. 17.9 해시 스크립트를 사용하는 것이 좋습니다

명령 요약:

- python3 c8kv_multitxq_hash.py —old_crc 1 —dest_network 192.168.1.0/24 —src_network 192.168.2.0/24 --unique_hash 1

'—old_crc 1'은 지원되는 PMD TXQ와 일치하도록 모듈로 8이 포함된 17.7 및 17.8 릴리스를 기반으로 해시 인덱스를 생성합니다(수정하지 않음).

'—dest_network'는 대상 네트워크 주소 서브넷을 정의합니다(네트워크 IP 주소 체계에 따라 수정).

'—src_network'는 소스 네트워크 주소 서브넷을 정의합니다(네트워크 IP 주소 체계에 따라 수정).

'—unique_hash 1'은 고유하게 해시된 IP 주소 집합 1개(8개의 TXQ에 대해 8쌍)를 생성합니다. 수정할 수 있습니다.

<#root>

```
(c8kv-hash) user@computer c8kv-aws-pmd-hash % python3 c8kv_multitxq_hash.py --old_crc 1 --dest_network 1

Dest:            Src:             Prot  dstport  srcport          Hash: Rev-hash:

192.168.1.0      192.168.2.0      2     5
192.168.1.0      192.168.2.1      2     7
192.168.1.0      192.168.2.2      2     1
192.168.1.0      192.168.2.3      2     3
192.168.1.0      192.168.2.4      2     5
192.168.1.0      192.168.2.5      2     7
192.168.1.0      192.168.2.6      2     1
192.168.1.0      192.168.2.7      2     3
192.168.1.0      192.168.2.8      2     5
192.168.1.0      192.168.2.9      2     7
192.168.1.0      192.168.2.10     2     1
.
. ### trimmed output ###
.
192.168.1.255    192.168.2.247    5     2
192.168.1.255    192.168.2.248    5     4
192.168.1.255    192.168.2.249    5     6
192.168.1.255    192.168.2.250    5     0
192.168.1.255    192.168.2.251    5     2
192.168.1.255    192.168.2.252    5     4
192.168.1.255    192.168.2.253    5     6
192.168.1.255    192.168.2.254    5     0
192.168.1.255    192.168.2.255    5     2

Unique hash:

------ Tunnels set 0 ---------

192.168.1.37<===>192.168.2.37<===>0


192.168.1.129<===>192.168.2.129<===>1


192.168.1.36<===>192.168.2.36<===>2


192.168.1.128<===>192.168.2.128<===>3


192.168.1.39<===>192.168.2.39<===>4


192.168.1.131<===>192.168.2.131<===>5


192.168.1.38<===>192.168.2.38<===>6


192.168.1.130<===>192.168.2.130<===>7
```

17.9 이상 릴리스의 Python Hash Index Script를 사용하여 IP 주소 체계 계산

명령 요약:

- python3 c8kv_multixq_hash.py —dest_network 192.168.1.0/24 —src_network 192.168.2.0/24 —port udp —src_port 12346 —dst_port 12346 —unique_hash 1

IOS® XE 릴리스 17.9 이상에서는 스크립트가 지원되는 PMD TXQ와 일치하는 modulo 12를 —old_crc 옵션 없이 사용합니다.

'—dest_network'는 대상 네트워크 주소 서브넷을 정의합니다(네트워크 IP 주소 체계에 따라 수정).

'—src_network'는 소스 네트워크 주소 서브넷을 정의합니다(네트워크 IP 주소 체계에 따라 수정).

'—포트 udp'는 사용되는 프로토콜을 정의합니다. 사용자는 프로토콜 매개변수를 "gre", "tcp" 또는 "udp" 또는 임의의 십진수 값으로 지정할 수 있습니다(선택 사항)

'—src_port'는 사용되는 소스 포트를 정의합니다(선택 사항).

'—dst_port'는 사용되는 대상 포트를 정의합니다(선택 사항).

'—unique_hash 1'은 고유하게 해시된 IP 주소의 한 집합(12개의 TXQ에 대해 12쌍)을 생성합니다. 수정할 수 있습니다.

<#root>

```
(c8kv-hash) user@computer c8kv-aws-pmd-hash % python3 c8kv_multitxq_hash.py --dest_network 192.168.1.0/

Dest:           Src:            Prot  dstport  srcport          Hash: Rev-hash:

192.168.1.0     192.168.2.0     17    12346    12346       ==>    4        4       <-- Unique Hash Va
192.168.1.0     192.168.2.1     17    12346    12346       ==>    4        4
192.168.1.0     192.168.2.2     17    12346    12346       ==>    8        8       <-- Unique Hash Va
192.168.1.0     192.168.2.3     17    12346    12346       ==>    0        0       <-- Unique Hash Va
192.168.1.0     192.168.2.4     17    12346    12346       ==>    0        0
192.168.1.0     192.168.2.5     17    12346    12346       ==>    0        0
192.168.1.0     192.168.2.6     17    12346    12346       ==>    4        4
192.168.1.0     192.168.2.7     17    12346    12346       ==>    0        0
192.168.1.0     192.168.2.8     17    12346    12346       ==>    9        9       <-- Unique Hash Va
192.168.1.0     192.168.2.9     17    12346    12346       ==>    9        9
192.168.1.0     192.168.2.10    17    12346    12346       ==>    9        9
192.168.1.0     192.168.2.11    17    12346    12346       ==>    1        1       <-- Unique Hash Va
192.168.1.0     192.168.2.12    17    12346    12346       ==>    1        1
.
. ### trimmed output ###
.
192.168.1.255   192.168.2.250   17    12346    12346       ==>    1        1
192.168.1.255   192.168.2.251   17    12346    12346       ==>    1        1
192.168.1.255   192.168.2.252   17    12346    12346       ==>    9        9
192.168.1.255   192.168.2.253   17    12346    12346       ==>    1        1
192.168.1.255   192.168.2.254   17    12346    12346       ==>    5        5       <-- Unique Hash Va
192.168.1.255   192.168.2.255   17    12346    12346       ==>    9        9
```

Unique hash:

------ Tunnels set 0 ---------

192.168.1.38 <===> 192.168.2.38<===>0

192.168.1.37 <===> 192.168.2.37<===>1

192.168.1.53 <===> 192.168.2.53<===>2

192.168.1.39 <===> 192.168.2.39<===>3

192.168.1.48 <===> 192.168.2.48<===>4

192.168.1.58 <===> 192.168.2.58<===>5

192.168.1.42 <===> 192.168.2.42<===>6

192.168.1.46 <===> 192.168.2.46<===>7

192.168.1.40 <===> 192.168.2.40<===>8

192.168.1.43 <===> 192.168.2.43<===>9

192.168.1.36 <===> 192.168.2.36<===>10

192.168.1.56 <===> 192.168.2.56<===>11

루프백 인터페이스와 함께 8개의 TXQ를 사용하는 샘플 토폴로지 및 CLI 컨피그레이션

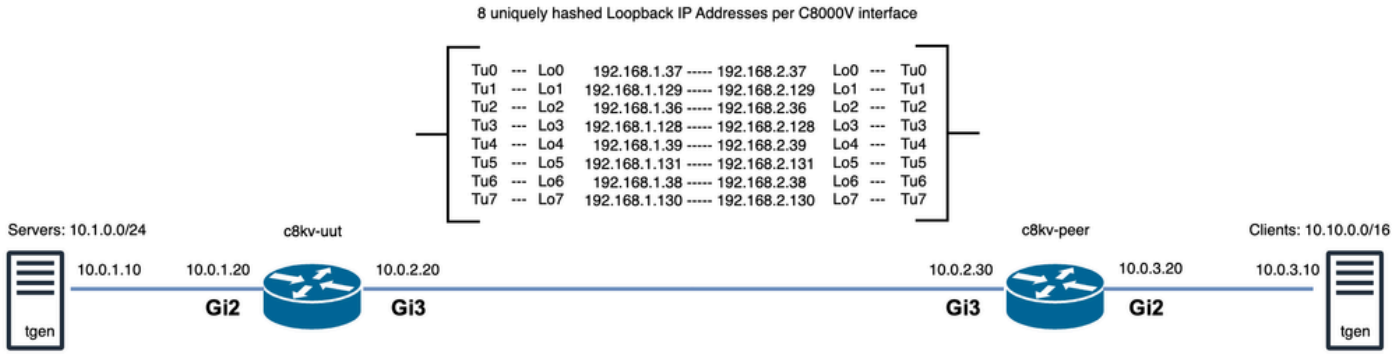


그림 3: 루프백 인터페이스를 사용하여 8개의 TxQ를 사용하는 샘플 토폴로지입니다.

이는 이전 섹션에서 계산된 해시된 IP 주소(192.168.1.X)를 사용하여 루프백 인터페이스를 사용하여 8개의 IPsec 터널을 생성하는 'c8kv-ut'에 대한 샘플 CLI 컨피그레이션입니다(그림 3).

유사한 컨피그레이션이 나머지 8개의 계산된 해시된 IP 주소(192.168.2.X)와 함께 다른 라우터 엔드포인트(c8kv-peer)에 적용됩니다.

```
ip cef load-sharing algorithm include-ports source destination 00ABC123

crypto keyring tunnel0
  local-address Loopback0
  pre-shared-key address 192.168.2.37 key cisco
crypto keyring tunnel1
  local-address Loopback1
  pre-shared-key address 192.168.2.129 key cisco
crypto keyring tunnel2
  local-address Loopback2
  pre-shared-key address 192.168.2.36 key cisco
crypto keyring tunnel3
  local-address Loopback3
  pre-shared-key address 192.168.2.128 key cisco
crypto keyring tunnel4
  local-address Loopback4
  pre-shared-key address 192.168.2.39 key cisco
crypto keyring tunnel5
  local-address Loopback5
  pre-shared-key address 192.168.2.131 key cisco
crypto keyring tunnel6
  local-address Loopback6
  pre-shared-key address 192.168.2.38 key cisco
crypto keyring tunnel7
  local-address Loopback7
  pre-shared-key address 192.168.2.130 key cisco

crypto isakmp policy 200
 encryption aes
 hash sha
 authentication pre-share
 group 16
 lifetime 28800
```

```
crypto isakmp profile isakmp-tunnel0
   keyring tunnel0
   match identity address 0.0.0.0
   local-address Loopback0
crypto isakmp profile isakmp-tunnel1
   keyring tunnel1
   match identity address 0.0.0.0
   local-address Loopback1
crypto isakmp profile isakmp-tunnel2
   keyring tunnel2
   match identity address 0.0.0.0
   local-address Loopback2
crypto isakmp profile isakmp-tunnel3
   keyring tunnel3
   match identity address 0.0.0.0
   local-address Loopback3
crypto isakmp profile isakmp-tunnel4
   keyring tunnel4
   match identity address 0.0.0.0
   local-address Loopback4
crypto isakmp profile isakmp-tunnel5
   keyring tunnel5
   match identity address 0.0.0.0
   local-address Loopback5
crypto isakmp profile isakmp-tunnel6
   keyring tunnel6
   match identity address 0.0.0.0
   local-address Loopback6
crypto isakmp profile isakmp-tunnel7
   keyring tunnel7
   match identity address 0.0.0.0
   local-address Loopback7

crypto ipsec transform-set ipsec-prop-vpn-tunnel esp-gcm 256
 mode tunnel
crypto ipsec df-bit clear

crypto ipsec profile ipsec-vpn-tunnel
 set transform-set ipsec-prop-vpn-tunnel
 set pfs group16

interface Loopback0
 ip address 192.168.1.37 255.255.255.255
!
interface Loopback1
 ip address 192.168.1.129 255.255.255.255
!
interface Loopback2
 ip address 192.168.1.36 255.255.255.255
!
interface Loopback3
 ip address 192.168.1.128 255.255.255.255
!
interface Loopback4
 ip address 192.168.1.39 255.255.255.255
!
interface Loopback5
 ip address 192.168.1.131 255.255.255.255
!
interface Loopback6
 ip address 192.168.1.38 255.255.255.255
!
```

```
interface Loopback7
 ip address 192.168.1.130 255.255.255.255
!

interface Tunnel0
 ip address 10.101.100.101 255.255.255.0
 load-interval 30
 tunnel source Loopback0
 tunnel mode ipsec ipv4
 tunnel destination 192.168.2.37
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel1
 ip address 10.101.101.101 255.255.255.0
 load-interval 30
 tunnel source Loopback1
 tunnel mode ipsec ipv4
 tunnel destination 192.168.2.129
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel2
 ip address 10.101.102.101 255.255.255.0
 load-interval 30
 tunnel source Loopback2
 tunnel mode ipsec ipv4
 tunnel destination 192.168.2.36
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel3
 ip address 10.101.103.101 255.255.255.0
 load-interval 30
 tunnel source Loopback3
 tunnel mode ipsec ipv4
 tunnel destination 192.168.2.128
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel4
 ip address 10.101.104.101 255.255.255.0
 load-interval 30
 tunnel source Loopback4
 tunnel mode ipsec ipv4
 tunnel destination 192.168.2.39
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel5
 ip address 10.101.105.101 255.255.255.0
 load-interval 30
 tunnel source Loopback5
 tunnel mode ipsec ipv4
 tunnel destination 192.168.2.131
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel6
 ip address 10.101.106.101 255.255.255.0
 load-interval 30
 tunnel source Loopback6
 tunnel mode ipsec ipv4
 tunnel destination 192.168.2.38
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel7
 ip address 10.101.107.101 255.255.255.0
```

```
 load-interval 30
 tunnel source Loopback7
 tunnel mode ipsec ipv4
 tunnel destination 192.168.2.130
 tunnel protection ipsec profile ipsec-vpn-tunnel
!


interface GigabitEthernet2
 mtu 9216
 ip address dhcp
 load-interval 30
 speed 25000
 no negotiation auto
 no mop enabled
 no mop sysid
!
interface GigabitEthernet3
 mtu 9216
 ip address dhcp
 load-interval 30
 speed 25000
 no negotiation auto
 no mop enabled
 no mop sysid
!

 !  ### IP route from servers to c8kv-uut

ip route 10.1.0.0 255.255.0.0 GigabitEthernet2 10.0.1.10

!  ### IP routes from c8kv-uut to clients on c8kv-peer side, routes are evenly distributed to all 8 TXQ

ip route 10.10.0.0 255.255.0.0 Tunnel0
ip route 10.10.0.0 255.255.0.0 Tunnel1
ip route 10.10.0.0 255.255.0.0 Tunnel2
ip route 10.10.0.0 255.255.0.0 Tunnel3
ip route 10.10.0.0 255.255.0.0 Tunnel4
ip route 10.10.0.0 255.255.0.0 Tunnel5
ip route 10.10.0.0 255.255.0.0 Tunnel6
ip route 10.10.0.0 255.255.0.0 Tunnel7

 !  ### IP route from c8kv-uut Loopback int tunnel endpoint to c8kv-peer Loopback int tunnel endpoints

ip route 192.168.2.0 255.255.255.0 GigabitEthernet3 10.0.2.30
```

루프백 인터페이스와 함께 12개의 TXQ를 사용하는 샘플 토폴로지 및 CLI 컨피그레이션

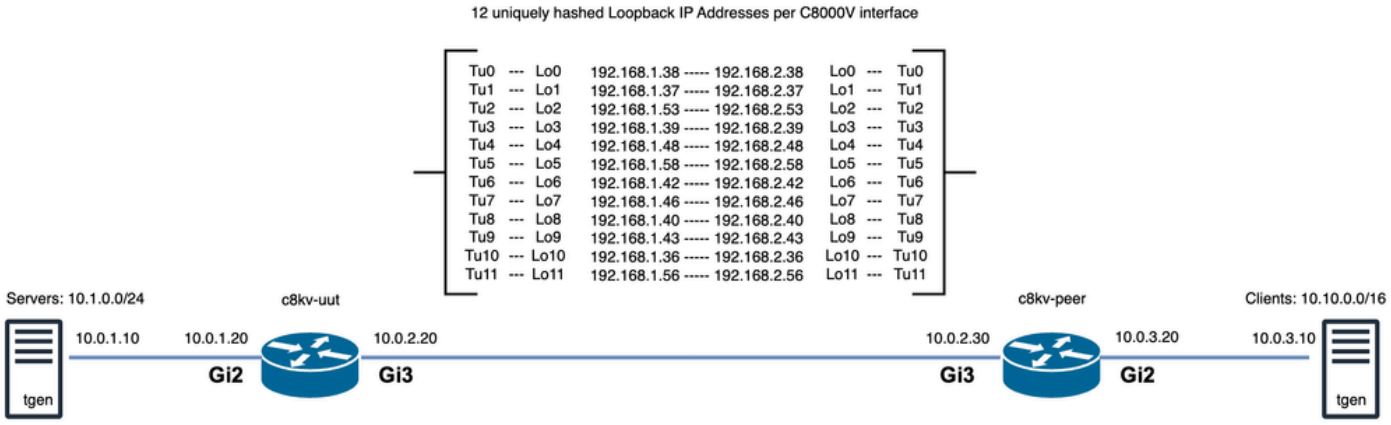


그림 4. 루프백 인터페이스를 사용하여 12개의 TxQ를 사용하는 샘플 토폴로지.

이는 이전 섹션에서 계산된 해시된 IP 주소(192.168.1.X)를 사용하여 루프백 인터페이스를 사용하여 12개의 IPsec 터널을 생성하는 'c8kv-ut'에 대한 샘플 CLI 컨피그레이션입니다(그림 4).

유사한 컨피그레이션이 나머지 8개의 계산된 해시된 IP 주소(192.168.2.X)와 함께 다른 라우터 엔드포인트(c8kv-peer)에 적용됩니다.

```
ip cef load-sharing algorithm include-ports source destination 00ABC123

crypto keyring tunnel0
  local-address Loopback0
  pre-shared-key address 192.168.2.38 key cisco
crypto keyring tunnel1
  local-address Loopback1
  pre-shared-key address 192.168.2.37 key cisco
crypto keyring tunnel2
  local-address Loopback2
  pre-shared-key address 192.168.2.53 key cisco
crypto keyring tunnel3
  local-address Loopback3
  pre-shared-key address 192.168.2.39 key cisco
crypto keyring tunnel4
  local-address Loopback4
  pre-shared-key address 192.168.2.48 key cisco
crypto keyring tunnel5
  local-address Loopback5
  pre-shared-key address 192.168.2.58 key cisco
crypto keyring tunnel6
  local-address Loopback6
  pre-shared-key address 192.168.2.42 key cisco
crypto keyring tunnel7
  local-address Loopback7
  pre-shared-key address 192.168.2.46 key cisco
crypto keyring tunnel8
  local-address Loopback8
  pre-shared-key address 192.168.2.40 key cisco
crypto keyring tunnel9
  local-address Loopback9
  pre-shared-key address 192.168.2.43 key cisco
crypto keyring tunnel10
  local-address Loopback10
```

```
  pre-shared-key address 192.168.2.36 key cisco
crypto keyring tunnel11
  local-address Loopback11
  pre-shared-key address 192.168.2.56 key cisco


crypto isakmp policy 200
 encryption aes
 hash sha
 authentication pre-share
 group 16
 lifetime 28800
crypto isakmp profile isakmp-tunnel0
   keyring tunnel0
   match identity address 0.0.0.0
   local-address Loopback0
crypto isakmp profile isakmp-tunnel1
   keyring tunnel1
   match identity address 0.0.0.0
   local-address Loopback1
crypto isakmp profile isakmp-tunnel2
   keyring tunnel2
   match identity address 0.0.0.0
   local-address Loopback2
crypto isakmp profile isakmp-tunnel3
   keyring tunnel3
   match identity address 0.0.0.0
   local-address Loopback3
crypto isakmp profile isakmp-tunnel4
   keyring tunnel4
   match identity address 0.0.0.0
   local-address Loopback4
crypto isakmp profile isakmp-tunnel5
   keyring tunnel5
   match identity address 0.0.0.0
   local-address Loopback5
crypto isakmp profile isakmp-tunnel6
   keyring tunnel6
   match identity address 0.0.0.0
   local-address Loopback6
crypto isakmp profile isakmp-tunnel7
   keyring tunnel7
   match identity address 0.0.0.0
   local-address Loopback7
crypto isakmp profile isakmp-tunnel8
   keyring tunnel8
   match identity address 0.0.0.0
   local-address Loopback8
crypto isakmp profile isakmp-tunnel9
   keyring tunnel9
   match identity address 0.0.0.0
   local-address Loopback9
crypto isakmp profile isakmp-tunnel10
   keyring tunnel10
   match identity address 0.0.0.0
   local-address Loopback10
crypto isakmp profile isakmp-tunnel11
   keyring tunnel11
   match identity address 0.0.0.0
   local-address Loopback11

crypto ipsec transform-set ipsec-prop-vpn-tunnel esp-gcm 256
```

```
 mode tunnel
crypto ipsec df-bit clear

crypto ipsec profile ipsec-vpn-tunnel
 set transform-set ipsec-prop-vpn-tunnel
 set pfs group16

interface Loopback0
 ip address 192.168.1.38 255.255.255.255
!
interface Loopback1
 ip address 192.168.1.37 255.255.255.255
!
interface Loopback2
 ip address 192.168.1.53 255.255.255.255
!
interface Loopback3
 ip address 192.168.1.39 255.255.255.255
!
interface Loopback4
 ip address 192.168.1.48 255.255.255.255
!
interface Loopback5
 ip address 192.168.1.58 255.255.255.255
!
interface Loopback6
 ip address 192.168.1.42 255.255.255.255
!
interface Loopback7
 ip address 192.168.1.46 255.255.255.255
!
interface Loopback8
 ip address 192.168.1.40 255.255.255.255
!
interface Loopback9
 ip address 192.168.1.43 255.255.255.255
!
interface Loopback10
 ip address 192.168.1.36 255.255.255.255
!
interface Loopback11
 ip address 192.168.1.56 255.255.255.255

interface Tunnel0
 ip address 10.101.100.101 255.255.255.0
 load-interval 30
 tunnel source Loopback0
 tunnel mode ipsec ipv4
 tunnel destination 192.168.2.38
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel1
 ip address 10.101.101.101 255.255.255.0
 load-interval 30
 tunnel source Loopback1
 tunnel mode ipsec ipv4
 tunnel destination 192.168.2.37
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel2
 ip address 10.101.102.101 255.255.255.0
 load-interval 30
```

```
 tunnel source Loopback2
 tunnel mode ipsec ipv4
 tunnel destination 192.168.2.53
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel3
 ip address 10.101.103.101 255.255.255.0
 load-interval 30
 tunnel source Loopback3
 tunnel mode ipsec ipv4
 tunnel destination 192.168.2.39
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel4
 ip address 10.101.104.101 255.255.255.0
 load-interval 30
 tunnel source Loopback4
 tunnel mode ipsec ipv4
 tunnel destination 192.168.2.48
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel5
 ip address 10.101.105.101 255.255.255.0
 load-interval 30
 tunnel source Loopback5
 tunnel mode ipsec ipv4
 tunnel destination 192.168.2.58
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel6
 ip address 10.101.106.101 255.255.255.0
 load-interval 30
 tunnel source Loopback6
 tunnel mode ipsec ipv4
 tunnel destination 192.168.2.42
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel7
 ip address 10.101.107.101 255.255.255.0
 load-interval 30
 tunnel source Loopback7
 tunnel mode ipsec ipv4
 tunnel destination 192.168.2.46
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel8
 ip address 10.101.108.101 255.255.255.0
 load-interval 30
 tunnel source Loopback8
 tunnel mode ipsec ipv4
 tunnel destination 192.168.2.40
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel9
 ip address 10.101.109.101 255.255.255.0
 load-interval 30
 tunnel source Loopback9
 tunnel mode ipsec ipv4
 tunnel destination 192.168.2.43
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel10
```

```
 ip address 10.101.110.101 255.255.255.0
 load-interval 30
 tunnel source Loopback10
 tunnel mode ipsec ipv4
 tunnel destination 192.168.2.36
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel11
 ip address 10.101.111.101 255.255.255.0
 load-interval 30
 tunnel source Loopback11
 tunnel mode ipsec ipv4
 tunnel destination 192.168.2.56
 tunnel protection ipsec profile ipsec-vpn-tunnel


interface GigabitEthernet2
 mtu 9216
 ip address dhcp
 load-interval 30
 speed 25000
 no negotiation auto
 no mop enabled
 no mop sysid
!
interface GigabitEthernet3
 mtu 9216
 ip address dhcp
 load-interval 30
 speed 25000
 no negotiation auto
 no mop enabled
 no mop sysid
!

  !  ### IP route from c8kv-uut to local servers

ip route 10.1.0.0 255.255.0.0 GigabitEthernet2 10.0.1.10

!  ### IP routes from c8kv-uut to clients on c8kv-peer side, routes are evenly distributed to all 12 TX

ip route 10.10.0.0 255.255.0.0 Tunnel0
ip route 10.10.0.0 255.255.0.0 Tunnel1
ip route 10.10.0.0 255.255.0.0 Tunnel2
ip route 10.10.0.0 255.255.0.0 Tunnel3
ip route 10.10.0.0 255.255.0.0 Tunnel4
ip route 10.10.0.0 255.255.0.0 Tunnel5
ip route 10.10.0.0 255.255.0.0 Tunnel6
ip route 10.10.0.0 255.255.0.0 Tunnel7
ip route 10.10.0.0 255.255.0.0 Tunnel8
ip route 10.10.0.0 255.255.0.0 Tunnel9
ip route 10.10.0.0 255.255.0.0 Tunnel10
ip route 10.10.0.0 255.255.0.0 Tunnel11

 !  ### IP route from c8kv-uut Loopback int tunnel endpoint to c8kv-peer Loopback int tunnel endpoints

ip route 192.168.2.0 255.255.255.0 GigabitEthernet3 10.0.2.30
```

보조 IP 주소와 함께 12개의 TXQ를 사용하는 샘플 토폴로지 및 CLI 컨피그레이션

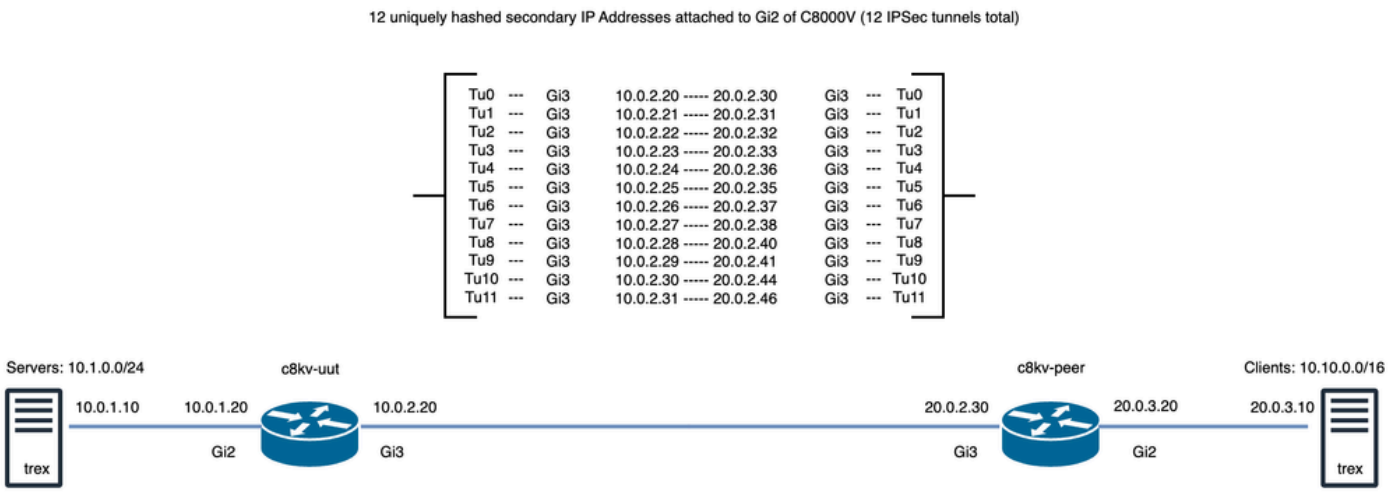
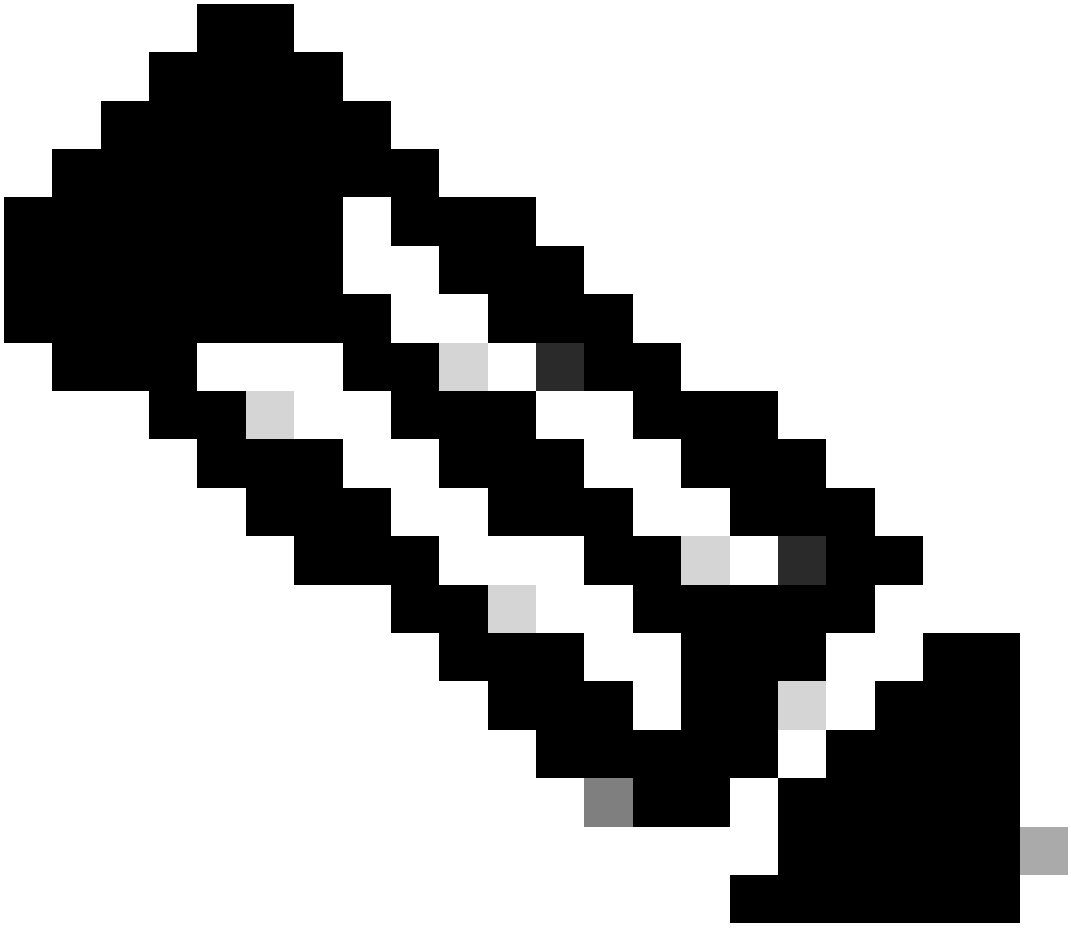


그림 5. 보조 IP 주소를 사용하여 12개의 TxQ를 사용하는 샘플 토폴로지.

AWS 환경에서 루프백 주소를 사용할 수 없는 경우 ENI에 연결된 보조 IP 주소를 대신 사용할 수 있습니다.

'c8kv-ut'에 대한 샘플 CLI 컨피그레이션입니다(그림 5). 이 컨피그레이션은 12개의 IPsec 터널을 생성하며, 소스는 계산된 해시된 IP 주소(10.0.2.X)를 사용하여 GigabitEthernet3 인터페이스에 연결된 1개의 기본 IP 주소 + 11개의 보조 IP 주소입니다. 유사한 컨피그레이션이 나머지 12개의 계산된 해시된 IP 주소(20.0.2.X)와 함께 다른 라우터 엔드포인트(c8kv-peer)에 적용됩니다.



참고: 이 예에서는 두 번째 C8000V를 터널 엔드포인트로 사용하지만 TGW 또는 DX와 같은 다른 클라우드 네트워킹 엔드포인트도 사용할 수 있습니다.

```
ip cef load-sharing algorithm include-ports source destination 00ABC123

crypto keyring tunnel0
  local-address 10.0.2.20
  pre-shared-key address 20.0.2.30 key cisco
crypto keyring tunnel1
  local-address 10.0.2.21
  pre-shared-key address 20.0.2.31 key cisco
crypto keyring tunnel2
  local-address 10.0.2.22
  pre-shared-key address 20.0.2.32 key cisco
crypto keyring tunnel3
  local-address 10.0.2.23
  pre-shared-key address 20.0.2.33 key cisco
crypto keyring tunnel4
  local-address 10.0.2.24
  pre-shared-key address 20.0.2.36 key cisco
crypto keyring tunnel5
```

```
  local-address 10.0.2.25
  pre-shared-key address 20.0.2.35 key cisco
crypto keyring tunnel6
  local-address 10.0.2.26
  pre-shared-key address 20.0.2.37 key cisco
crypto keyring tunnel7
  local-address 10.0.2.27
  pre-shared-key address 20.0.2.38 key cisco
crypto keyring tunnel8
  local-address 10.0.2.28
  pre-shared-key address 20.0.2.40 key cisco
crypto keyring tunnel9
  local-address 10.0.2.29
  pre-shared-key address 20.0.2.41 key cisco
crypto keyring tunnel10
  local-address 10.0.2.30
  pre-shared-key address 20.0.2.44 key cisco
crypto keyring tunnel11
  local-address 10.0.2.31
  pre-shared-key address 20.0.2.46 key cisco


crypto isakmp policy 200
 encryption aes
 hash sha
 authentication pre-share
 group 16
 lifetime 28800
crypto isakmp profile isakmp-tunnel0
   keyring tunnel0
   match identity address 20.0.2.30 255.255.255.255
   local-address 10.0.2.20
crypto isakmp profile isakmp-tunnel1
   keyring tunnel1
   match identity address 20.0.2.31 255.255.255.255
   local-address 10.0.2.21
crypto isakmp profile isakmp-tunnel2
   keyring tunnel2
   match identity address 20.0.2.32 255.255.255.255
   local-address 10.0.2.22
crypto isakmp profile isakmp-tunnel3
   keyring tunnel3
   match identity address 20.0.2.33 255.255.255.255
   local-address 10.0.2.23
crypto isakmp profile isakmp-tunnel4
   keyring tunnel4
   match identity address 20.0.2.36 255.255.255.255
   local-address 10.0.2.24
crypto isakmp profile isakmp-tunnel5
   keyring tunnel5
   match identity address 20.0.2.35 255.255.255.255
   local-address 10.0.2.25
crypto isakmp profile isakmp-tunnel6
   keyring tunnel6
   match identity address 20.0.2.37 255.255.255.255
   local-address 10.0.2.26
crypto isakmp profile isakmp-tunnel7
   keyring tunnel7
   match identity address 20.0.2.38 255.255.255.255
   local-address 10.0.2.27
crypto isakmp profile isakmp-tunnel8
   keyring tunnel8
```

```
   match identity address 20.0.2.40 255.255.255.255
   local-address 10.0.2.28
crypto isakmp profile isakmp-tunnel9
   keyring tunnel9
   match identity address 20.0.2.41 255.255.255.255
   local-address 10.0.2.29
crypto isakmp profile isakmp-tunnel10
   keyring tunnel10
   match identity address 20.0.2.44 255.255.255.255
   local-address 10.0.2.30
crypto isakmp profile isakmp-tunnel11
   keyring tunnel11
   match identity address 20.0.2.46 255.255.255.255
   local-address 10.0.2.31


crypto ipsec transform-set ipsec-prop-vpn-tunnel esp-gcm 256
 mode tunnel
crypto ipsec df-bit clear

crypto ipsec profile ipsec-vpn-tunnel
 set transform-set ipsec-prop-vpn-tunnel
 set pfs group16

interface Tunnel0
 ip address 10.101.100.101 255.255.255.0
 load-interval 30
 tunnel source 10.0.2.20
 tunnel mode ipsec ipv4
 tunnel destination 20.0.2.30
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel1
 ip address 10.101.101.101 255.255.255.0
 load-interval 30
 tunnel source 10.0.2.21
 tunnel mode ipsec ipv4
 tunnel destination 20.0.2.31
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel2
 ip address 10.101.102.101 255.255.255.0
 load-interval 30
 tunnel source 10.0.2.22
 tunnel mode ipsec ipv4
 tunnel destination 20.0.2.32
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel3
 ip address 10.101.103.101 255.255.255.0
 load-interval 30
 tunnel source 10.0.2.23
 tunnel mode ipsec ipv4
 tunnel destination 20.0.2.33
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel4
 ip address 10.101.104.101 255.255.255.0
 load-interval 30
 tunnel source 10.0.2.24
 tunnel mode ipsec ipv4
 tunnel destination 20.0.2.36
```

```
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel5
 ip address 10.101.105.101 255.255.255.0
 load-interval 30
 tunnel source 10.0.2.25
 tunnel mode ipsec ipv4
 tunnel destination 20.0.2.35
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel6
 ip address 10.101.106.101 255.255.255.0
 load-interval 30
 tunnel source 10.0.2.26
 tunnel mode ipsec ipv4
 tunnel destination 20.0.2.37
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel7
 ip address 10.101.107.101 255.255.255.0
 load-interval 30
 tunnel source 10.0.2.27
 tunnel mode ipsec ipv4
 tunnel destination 20.0.2.38
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel8
 ip address 10.101.108.101 255.255.255.0
 load-interval 30
 tunnel source 10.0.2.28
 tunnel mode ipsec ipv4
 tunnel destination 20.0.2.40
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel9
 ip address 10.101.109.101 255.255.255.0
 load-interval 30
 tunnel source 10.0.2.29
 tunnel mode ipsec ipv4
 tunnel destination 20.0.2.41
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel10
 ip address 10.101.110.101 255.255.255.0
 load-interval 30
 tunnel source 10.0.2.30
 tunnel mode ipsec ipv4
 tunnel destination 20.0.2.44
 tunnel protection ipsec profile ipsec-vpn-tunnel
!
interface Tunnel11
 ip address 10.101.111.101 255.255.255.0
 load-interval 30
 tunnel source 10.0.2.31
 tunnel mode ipsec ipv4
 tunnel destination 20.0.2.46
 tunnel protection ipsec profile ipsec-vpn-tunnel
!


interface GigabitEthernet2
 mtu 9216
```

```
 ip address dhcp
 load-interval 30
 speed 25000
 no negotiation auto
 no mop enabled
 no mop sysid
!
interface GigabitEthernet3
 mtu 9216
 ip address 10.0.2.20 255.255.255.0
 ip address 10.0.2.21 255.255.255.0 secondary
 ip address 10.0.2.22 255.255.255.0 secondary
 ip address 10.0.2.23 255.255.255.0 secondary
 ip address 10.0.2.24 255.255.255.0 secondary
 ip address 10.0.2.25 255.255.255.0 secondary
 ip address 10.0.2.26 255.255.255.0 secondary
 ip address 10.0.2.27 255.255.255.0 secondary
 ip address 10.0.2.28 255.255.255.0 secondary
 ip address 10.0.2.29 255.255.255.0 secondary
 ip address 10.0.2.30 255.255.255.0 secondary
 ip address 10.0.2.31 255.255.255.0 secondary
 load-interval 30
 speed 25000
 no negotiation auto
 no mop enabled
 no mop sysid
!

  !  ### IP route from c8kv-uut to local servers

ip route 10.1.0.0 255.255.255.0 GigabitEthernet2 10.0.1.10

!  ### IP routes from c8kv-uut to clients on c8kv-peer side, routes are evenly distributed to all 12 TX

ip route 10.10.0.0 255.255.0.0 Tunnel0
ip route 10.10.0.0 255.255.0.0 Tunnel1
ip route 10.10.0.0 255.255.0.0 Tunnel2
ip route 10.10.0.0 255.255.0.0 Tunnel3
ip route 10.10.0.0 255.255.0.0 Tunnel4
ip route 10.10.0.0 255.255.0.0 Tunnel5
ip route 10.10.0.0 255.255.0.0 Tunnel6
ip route 10.10.0.0 255.255.0.0 Tunnel7
ip route 10.10.0.0 255.255.0.0 Tunnel8
ip route 10.10.0.0 255.255.0.0 Tunnel9
ip route 10.10.0.0 255.255.0.0 Tunnel10
ip route 10.10.0.0 255.255.0.0 Tunnel11


!  ### IP route from c8kv-uut Gi3 int tunnel endpoint to c8kv-peer Gi3

int tunnel endpoints (secondary IP addresses on c8kv-peer side)

ip route 20.0.2.30 255.255.255.255 10.0.2.1
ip route 20.0.2.31 255.255.255.255 10.0.2.1
ip route 20.0.2.32 255.255.255.255 10.0.2.1
ip route 20.0.2.33 255.255.255.255 10.0.2.1
ip route 20.0.2.36 255.255.255.255 10.0.2.1
ip route 20.0.2.35 255.255.255.255 10.0.2.1
ip route 20.0.2.37 255.255.255.255 10.0.2.1
ip route 20.0.2.38 255.255.255.255 10.0.2.1
ip route 20.0.2.40 255.255.255.255 10.0.2.1
ip route 20.0.2.41 255.255.255.255 10.0.2.1
```

```
ip route 20.0.2.44 255.255.255.255 10.0.2.1
ip route 20.0.2.46 255.255.255.255 10.0.2.1
```

AWS에서 일반적인 Catalyst 8000V 구축

자동 모드

이전 샘플 CLI 컨피그레이션 및 토폴로지를 참조하십시오. 네트워크 주소 지정 체계 및 생성된 해시된 IP 주소를 기반으로 CLI 컨피그레이션을 복사하고 수정할 수 있습니다.

성공적인 터널 생성을 위해서는 C8000V 및 AWS VPC에서 모두 IP 경로를 생성해야 합니다.

SD-WAN 모드

이는 AWS VPC에 위치한 C8000V에서 루프백 인터페이스를 사용하여 TLOC를 생성하는 토폴로지 및 SD-WAN 구성의 예입니다.

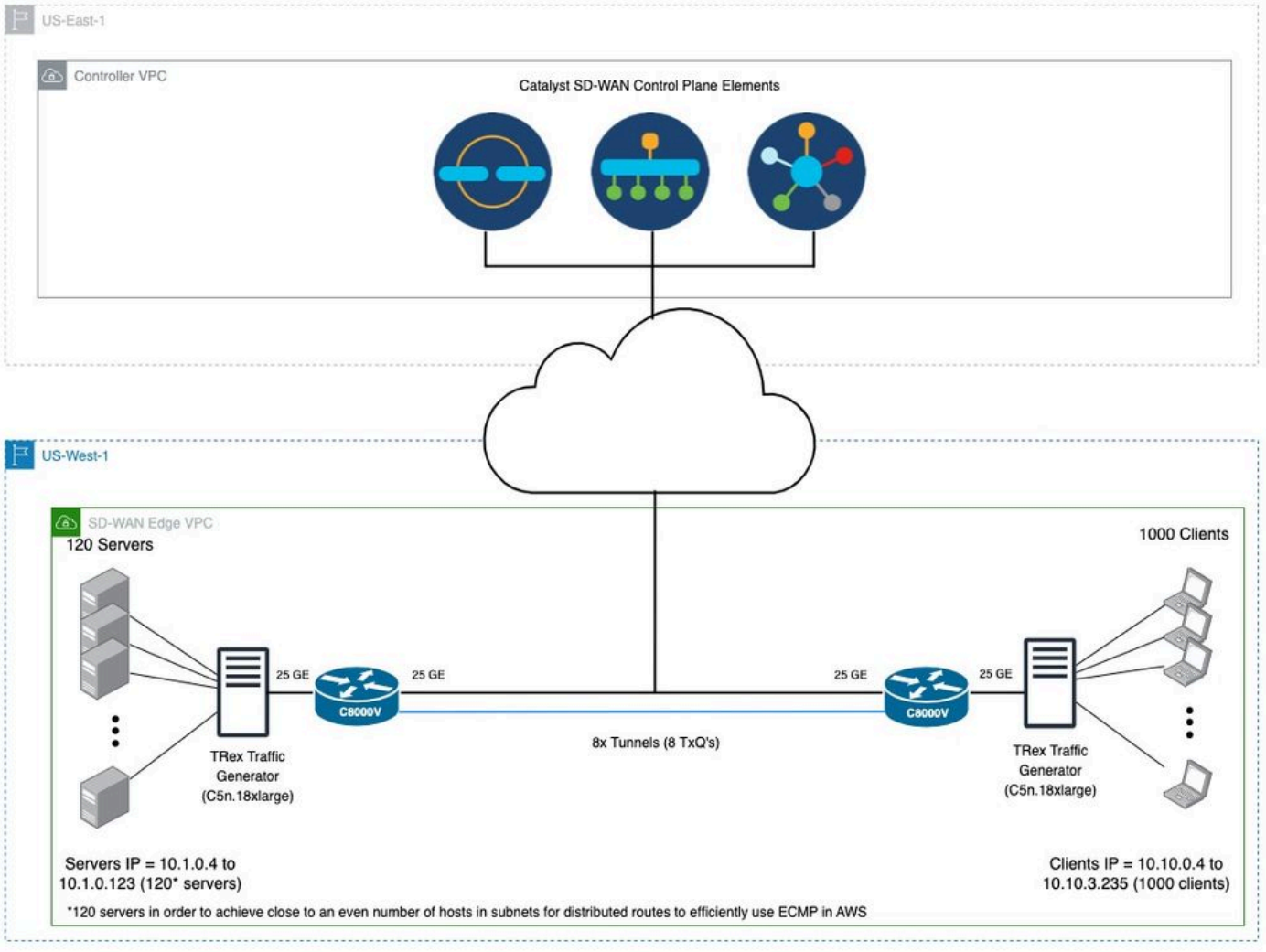
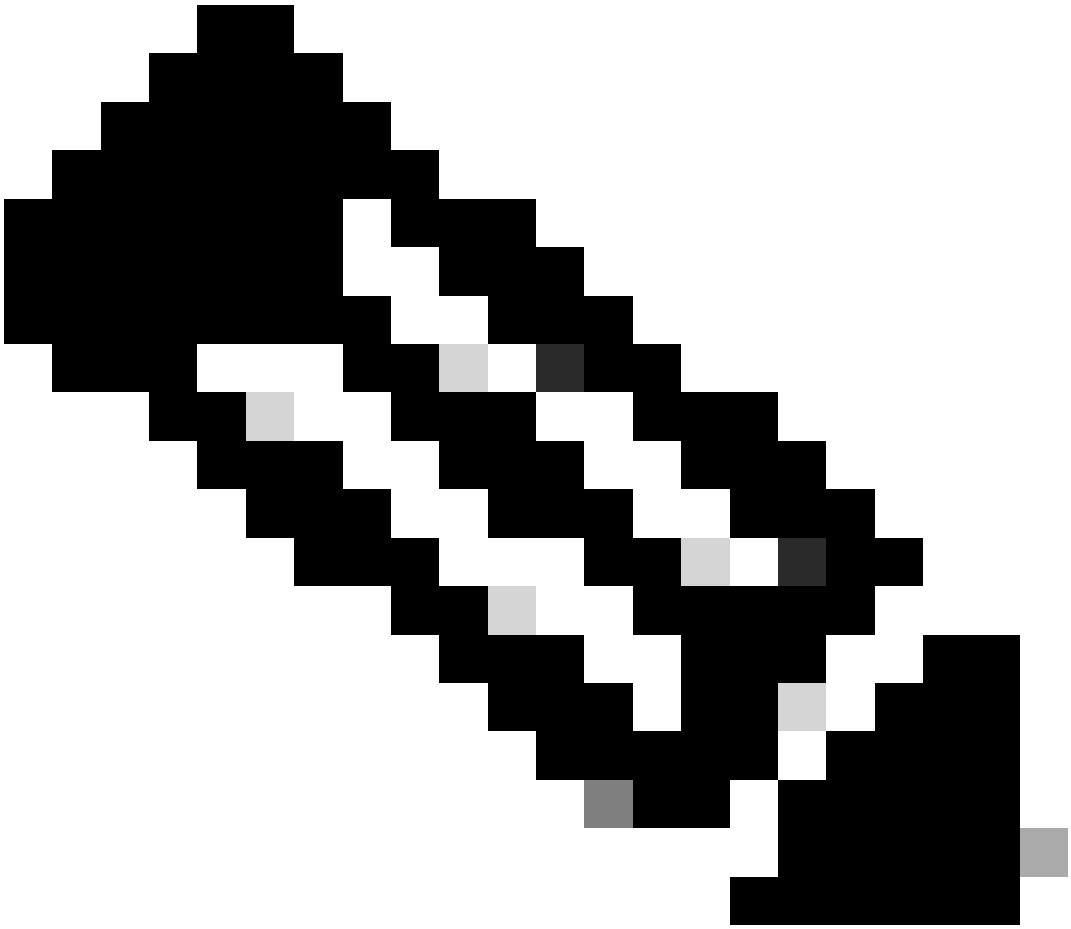


그림 6. AWS VPC에 위치한 C8000V에서 루프백 인터페이스와 함께 TLOC를 사용하는 샘플 SD-

WAN 토폴로지.



참고: 그림 6에서 검은색 연결은 SD-WAN 제어 평면 요소와 SD-WAN 에지 장치 간의 Control(VPN0) 연결을 나타냅니다. 파란색 연결은 TLOC를 사용하는 두 SD-WAN 에지 장치 간의 터널을 나타냅니다.

그림 6(여기)의 샘플 SD-WAN CLI 컨피그레이션을 찾을 수 있습니다.

```
csr_uut#show sdwan run
system
system-ip             29.173.249.161
site-id               5172
admin-tech-on-failure
sp-organization-name  SP_ORG_NAME
organization-name     ORG_NAME
upgrade-confirm       15
vbond X.X.X.X
!
```

```
memory free low-watermark processor 68484
service timestamps debug datetime msec
service timestamps log datetime msec
no service tcp-small-servers
no service udp-small-servers
platform console virtual
platform qfp utilization monitor load 80
platform punt-keepalive disable-kernel-core
hostname csr_uut
username ec2-user privilege 15 secret 5 $1$4P16$..ag88eFsOMLIemjNcWSt0
vrf definition 11
address-family ipv4
exit-address-family
!
address-family ipv6
exit-address-family
!
!
vrf definition Mgmt-intf
address-family ipv4
exit-address-family
!
address-family ipv6
exit-address-family
!
!
no ip finger
no ip rcmd rcp-enable
no ip rcmd rsh-enable
no ip dhcp use class
ip route 0.0.0.0 0.0.0.0 X.X.X.X
ip route 0.0.0.0 0.0.0.0 X.X.X.X
ip route 0.0.0.0 0.0.0.0 X.X.X.X
ip route vrf 11 10.1.0.0 255.255.0.0 X.X.X.X
ip route vrf Mgmt-intf 0.0.0.0 0.0.0.0 X.X.X.X
no ip source-route
ip ssh pubkey-chain
username ec2-user
key-hash ssh-rsa 353158c28c7649710b3c933da02e384b ec2-user
!
!
!
no ip http server
ip http secure-server
ip nat settings central-policy
ip nat settings gatekeeper-size 1024
ipv6 unicast-routing
class-map match-any class0
match dscp 1
!
class-map match-any class1
match dscp 2
!
class-map match-any class2
match dscp 3
!
class-map match-any class3
match dscp 4
!
class-map match-any class4
match dscp 5
!
```

```
class-map match-any class5
match dscp 6
!
class-map match-any class6
match dscp 7
!
class-map match-any class7
match dscp 8
!
policy-map qos_map1
class class0
priority percent 20
!
class class1
bandwidth percent 18
random-detect
!
class class2
bandwidth percent 15
random-detect
!
class class3
bandwidth percent 12
random-detect
!
class class4
bandwidth percent 10
random-detect
!
class class5
bandwidth percent 10
random-detect
!
class class6
bandwidth percent 10
random-detect
!
class class7
bandwidth percent 5
random-detect
!
!
interface GigabitEthernet1
no shutdown
ip address dhcp
no mop enabled
no mop sysid
negotiation auto
exit
interface GigabitEthernet2
no shutdown
ip address dhcp
load-interval 30
speed 10000
no negotiation auto
service-policy output qos_map1
exit
interface GigabitEthernet3
shutdown
ip address dhcp
load-interval 30
speed 10000
```

```
no negotiation auto
exit
interface GigabitEthernet4
no shutdown
vrf forwarding 11
ip address X.X.X.X 255.255.255.0
load-interval 30
speed 10000
no negotiation auto
exit
interface Loopback1
no shutdown
ip address 192.168.1.21 255.255.255.255
exit
interface Loopback2
no shutdown
ip address 192.168.1.129 255.255.255.255
exit
interface Loopback3
no shutdown
ip address 192.168.1.20 255.255.255.255
exit
interface Loopback4
no shutdown
ip address 192.168.1.128 255.255.255.255
exit
interface Loopback5
no shutdown
ip address 192.168.1.23 255.255.255.255
exit
interface Loopback6
no shutdown
ip address 192.168.1.131 255.255.255.255
exit
interface Loopback7
no shutdown
ip address 192.168.1.22 255.255.255.255
exit
interface Loopback8
no shutdown
ip address 192.168.1.130 255.255.255.255
exit
interface Tunnel1
no shutdown
ip unnumbered GigabitEthernet1
tunnel source GigabitEthernet1
tunnel mode sdwan
exit
interface Tunnel14095001
no shutdown
ip unnumbered Loopback1
no ip redirects
ipv6 unnumbered Loopback1
no ipv6 redirects
tunnel source Loopback1
tunnel mode sdwan
exit
interface Tunnel14095002
no shutdown
ip unnumbered Loopback2
no ip redirects
ipv6 unnumbered Loopback2
```

```
no ipv6 redirects
tunnel source Loopback2
tunnel mode sdwan
exit
interface Tunnel14095003
no shutdown
ip unnumbered Loopback3
no ip redirects
ipv6 unnumbered Loopback3
no ipv6 redirects
tunnel source Loopback3
tunnel mode sdwan
exit
interface Tunnel14095004
no shutdown
ip unnumbered Loopback4
no ip redirects
ipv6 unnumbered Loopback4
no ipv6 redirects
tunnel source Loopback4
tunnel mode sdwan
exit
interface Tunnel14095005
no shutdown
ip unnumbered Loopback5
no ip redirects
ipv6 unnumbered Loopback5
no ipv6 redirects
tunnel source Loopback5
tunnel mode sdwan
exit
interface Tunnel14095006
no shutdown
ip unnumbered Loopback6
no ip redirects
ipv6 unnumbered Loopback6
no ipv6 redirects
tunnel source Loopback6
tunnel mode sdwan
exit
interface Tunnel14095007
no shutdown
ip unnumbered Loopback7
no ip redirects
ipv6 unnumbered Loopback7
no ipv6 redirects
tunnel source Loopback7
tunnel mode sdwan
exit
interface Tunnel14095008
no shutdown
ip unnumbered Loopback8
no ip redirects
ipv6 unnumbered Loopback8
no ipv6 redirects
tunnel source Loopback8
tunnel mode sdwan
exit
no logging console
aaa authentication enable default enable
aaa authentication login default local
aaa authorization console
```

```
aaa authorization exec default local none
login on-success log
license smart transport smart
license smart url https://smartreceiver.cisco.com/licservice/license
line aux 0
!
line con 0
stopbits 1
!
line vty 0 4
transport input ssh
!
line vty 5 80
transport input ssh
!
sdwan
interface GigabitEthernet1
tunnel-interface
encapsulation ipsec
color private1 restrict
allow-service all
no allow-service bgp
allow-service dhcp
allow-service dns
allow-service icmp
no allow-service sshd
no allow-service netconf
no allow-service ntp
no allow-service ospf
no allow-service stun
allow-service https
no allow-service snmp
no allow-service bfd
exit
exit
interface GigabitEthernet2
exit
interface GigabitEthernet3
exit
interface Loopback1
tunnel-interface
encapsulation ipsec preference 150 weight 1
no border
color private2 restrict
no last-resort-circuit
no low-bandwidth-link
max-control-connections       0
no vbond-as-stun-server
vmanage-connection-preference 0
port-hop
carrier                       default
nat-refresh-interval          5
hello-interval                1000
hello-tolerance               12
bind                          GigabitEthernet2
no allow-service all
no allow-service bgp
allow-service dhcp
allow-service dns
allow-service icmp
no allow-service sshd
no allow-service netconf
```

```
no allow-service ntp
no allow-service ospf
no allow-service stun
allow-service https
no allow-service snmp
no allow-service bfd
exit
exit
interface Loopback2
tunnel-interface
encapsulation ipsec preference 150 weight 1
no border
color private3 restrict
no last-resort-circuit
no low-bandwidth-link
max-control-connections       0
no vbond-as-stun-server
vmanage-connection-preference 0
port-hop
carrier                       default
nat-refresh-interval          5
hello-interval                1000
hello-tolerance               12
bind                          GigabitEthernet2
no allow-service all
no allow-service bgp
allow-service dhcp
allow-service dns
allow-service icmp
no allow-service sshd
no allow-service netconf
no allow-service ntp
no allow-service ospf
no allow-service stun
allow-service https
no allow-service snmp
no allow-service bfd
exit
exit
interface Loopback3
tunnel-interface
encapsulation ipsec preference 150 weight 1
no border
color private4 restrict
no last-resort-circuit
no low-bandwidth-link
max-control-connections       0
no vbond-as-stun-server
vmanage-connection-preference 0
port-hop
carrier                       default
nat-refresh-interval          5
hello-interval                1000
hello-tolerance               12
bind                          GigabitEthernet2
allow-service all
no allow-service bgp
allow-service dhcp
allow-service dns
allow-service icmp
no allow-service sshd
no allow-service netconf
```

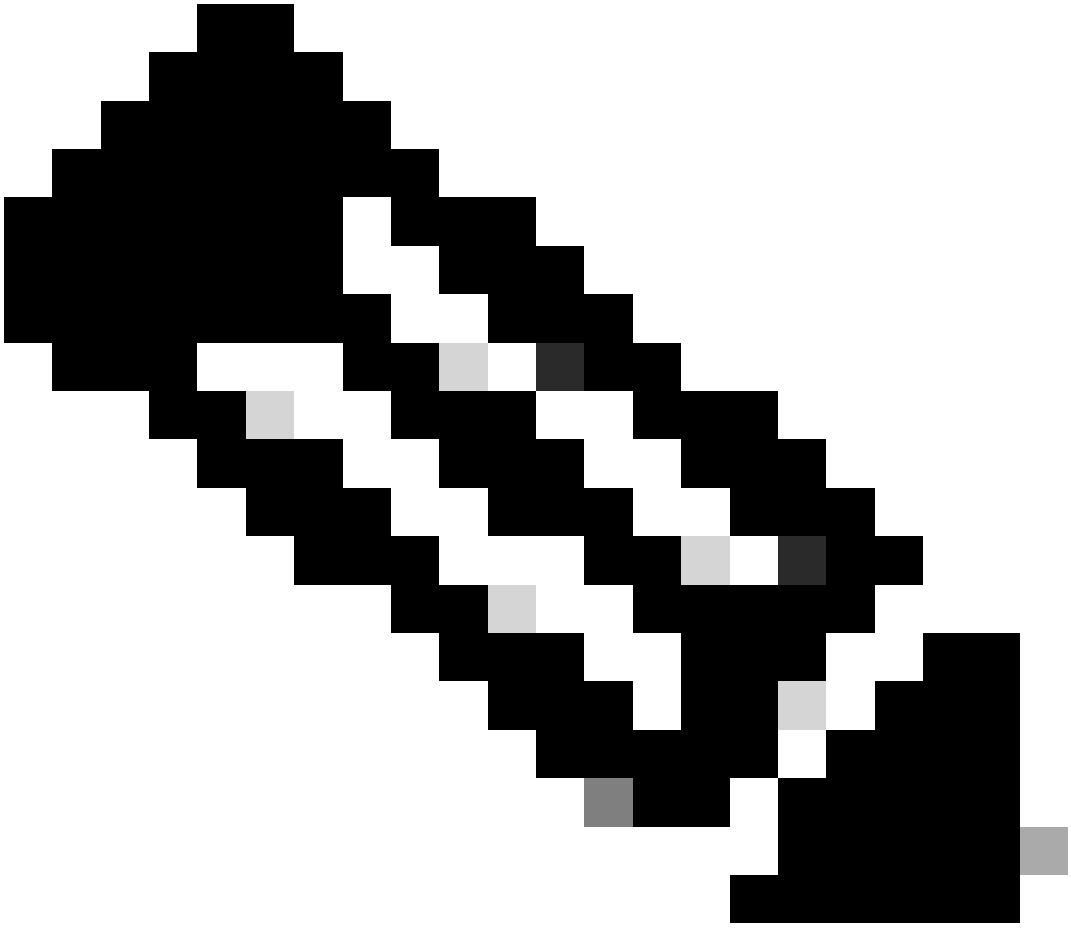
```
no allow-service ntp
no allow-service ospf
no allow-service stun
allow-service https
no allow-service snmp
no allow-service bfd
exit
exit
interface Loopback4
tunnel-interface
encapsulation ipsec preference 150 weight 1
no border
color private5 restrict
no last-resort-circuit
no low-bandwidth-link
max-control-connections       0
no vbond-as-stun-server
vmanage-connection-preference 0
port-hop
carrier                       default
nat-refresh-interval          5
hello-interval                1000
hello-tolerance               12
bind                          GigabitEthernet2
no allow-service all
no allow-service bgp
allow-service dhcp
allow-service dns
allow-service icmp
no allow-service sshd
no allow-service netconf
no allow-service ntp
no allow-service ospf
no allow-service stun
allow-service https
no allow-service snmp
no allow-service bfd
exit
exit
interface Loopback5
tunnel-interface
encapsulation ipsec preference 150 weight 1
no border
color private6 restrict
no last-resort-circuit
no low-bandwidth-link
max-control-connections       0
no vbond-as-stun-server
vmanage-connection-preference 0
port-hop
carrier                       default
nat-refresh-interval          5
hello-interval                1000
hello-tolerance               12
bind                          GigabitEthernet2
no allow-service all
no allow-service bgp
allow-service dhcp
allow-service dns
allow-service icmp
no allow-service sshd
no allow-service netconf
```

```
no allow-service ntp
no allow-service ospf
no allow-service stun
allow-service https
no allow-service snmp
no allow-service bfd
exit
exit
interface Loopback6
tunnel-interface
encapsulation ipsec preference 150 weight 1
no border
color red restrict
no last-resort-circuit
no low-bandwidth-link
max-control-connections       0
no vbond-as-stun-server
vmanage-connection-preference 0
port-hop
carrier                       default
nat-refresh-interval          5
hello-interval                1000
hello-tolerance               12
bind                          GigabitEthernet2
no allow-service all
no allow-service bgp
allow-service dhcp
allow-service dns
allow-service icmp
no allow-service sshd
no allow-service netconf
no allow-service ntp
no allow-service ospf
no allow-service stun
allow-service https
no allow-service snmp
no allow-service bfd
exit
exit
interface Loopback7
tunnel-interface
encapsulation ipsec preference 150 weight 1
no border
color blue restrict
no last-resort-circuit
no low-bandwidth-link
max-control-connections       0
no vbond-as-stun-server
vmanage-connection-preference 0
port-hop
carrier                       default
nat-refresh-interval          5
hello-interval                1000
hello-tolerance               12
bind                          GigabitEthernet2
no allow-service all
no allow-service bgp
allow-service dhcp
allow-service dns
allow-service icmp
no allow-service sshd
no allow-service netconf
```

```
no allow-service ntp
no allow-service ospf
no allow-service stun
allow-service https
no allow-service snmp
no allow-service bfd
exit
exit
interface Loopback8
tunnel-interface
encapsulation ipsec preference 150 weight 1
no border
color green restrict
no last-resort-circuit
no low-bandwidth-link
max-control-connections       0
no vbond-as-stun-server
vmanage-connection-preference 0
port-hop
carrier                       default
nat-refresh-interval          5
hello-interval                1000
hello-tolerance               12
bind                          GigabitEthernet2
no allow-service all
no allow-service bgp
allow-service dhcp
allow-service dns
allow-service icmp
no allow-service sshd
no allow-service netconf
no allow-service ntp
no allow-service ospf
no allow-service stun
allow-service https
no allow-service snmp
no allow-service bfd
exit
exit
appqoe
no tcpopt enable
no dreopt enable
no httpopt enable
!
omp
no shutdown
send-path-limit  16
ecmp-limit       16
graceful-restart
no as-dot-notation
timers
graceful-restart-timer 43200
exit
address-family ipv4
advertise connected
advertise static
!
address-family ipv6
advertise connected
advertise static
!
!
```

```
!
security
ipsec
replay-window  8192
integrity-type ip-udp-esp esp
!
!
sslproxy
no enable
rsa-key-modulus      2048
certificate-lifetime 730
eckey-type           P256
ca-tp-label          PROXY-SIGNING-CA
settings expired-certificate  drop
settings untrusted-certificate drop
settings unknown-status       drop
settings certificate-revocation-check none
settings unsupported-protocol-versions drop
settings unsupported-cipher-suites drop
settings failure-mode         close
settings minimum-tls-ver      TLSv1
dual-side optimization enable
!
policy
app-visibility
flow-visibility
!
```

AWS의 처리량 성능 문제 해결



참고: 퍼블릭 클라우드 환경에서 성능 테스트를 수행하면 처리량 성능에 영향을 줄 수 있는 새로운 변수가 도입됩니다. 이러한 종류의 테스트를 수행하는 동안 고려해야 할 몇 가지 사항은 다음과 같습니다.

- 테스트 실행 시 피어의 기본 리소스 사용량
- 전용 호스트를 사용하지 않음(전용 호스트를 사용하면 클라우드 비용이 16배 증가함)
- 클라우드는 여러 지역에서 실행되므로 성능이 달라질 수 있습니다.
- 기능 프로필에 관계없이 숫자가 비슷한 경우도 있습니다. 이는 인스턴스 크기당 AWS의 인터페이스 조절 때문일 수 있습니다
- AWS는 패킷 삭제를 유발할 수 있는 EC2 인스턴스의 초당 패킷 비율도 조절합니다
- AWS는 전송률 조절 속도를 공개하지 않지만, 전송률 조절로 인해 감소하는 것을 'pps_allowance_exceeded' 카운터를 통해 관찰할 수 있습니다

유용한 CLI 문제 해결 명령

처리량 성능 테스트를 수행할 때 이러한 문제 해결 명령을 사용하여 병목 지점이나 성능 저하의 원

인을 찾아낼 수 있습니다.

"show platform hardware qfp active statistics drop" - c8kv에 드롭이 있는지 확인할 수 있습니다. 우리는 상당한 테일 드롭이나 관련 카운터가 증가하지 않도록 해야 합니다.

"show platform hardware qfp active statistics drop clear" - 이 명령은 카운터를 지웁니다.

"show platform hardware qfp active datapath infrastructure sw-cio" - 이 명령은 성능 실행 중에 활용되는 PP(Packet Processor), TM(Traffic Manager)의 비율에 대한 자세한 정보를 제공합니다. 이를 통해 c8kv로부터 충분한 처리 능력이 있는지 여부를 판단할 수 있다.

"show platform hardware qfp active datapath util summary" - 이 명령은 c8kv가 모든 포트에서 전송/수신하는 입력/출력의 전체 정보를 제공합니다.

입력/출력 속도를 확인하고 누락이 있는지 확인합니다. 또한 처리 로드 비율을 확인합니다. 100%에 달할 경우 이는 c8kv가 용량에 도달했음을 의미합니다.

"show plat hardware qfp active infrastructure bqs interface GigabitEthernetX" - 이 명령을 사용하면 대기열 번호, 대역폭, taildrops 측면에서 인터페이스 수준 통계를 확인할 수 있습니다.

"show controller" - 이 명령은 rx/tx good packets, missed packets에 대한 보다 세부적인 정보를 제공합니다.

이 명령은 어떤 테일 드롭도 표시되지 않지만 트래픽 생성기에서 여전히 드롭을 표시하는 시나리오에서 사용할 수 있습니다.

이는 데이터 활용률이 이미 100%에 도달하고 PP도 100%에 도달하는 시나리오에서 발생할 수 있습니다.

rx_missed_errors 카운터가 계속 증가하는 경우, CSR이 더 이상 트래픽을 처리할 수 없으므로 클라우드 인프라에 부담을 주고 있음을 의미합니다.

"show platform hardware qfp active datapath infrastructure sw-hqf" - AWS의 역압력으로 인해 발생하는 정체 현상을 확인하는 데 사용할 수 있습니다.

"show plat hardware qfp active datapath infrastructure sw-nic" - 여러 큐에서 트래픽이 로드 밸런싱되는 방법을 결정합니다. 17.7 이후에는 8개의 Multi-TXQ가 있습니다.

또한 모든 트래픽을 처리하거나 로드 밸런싱이 제대로 되고 있는 특정 대기열이 있는지 확인할 수 있습니다.

"컨트롤러 표시 | in errors|exceeded|Giga" - pps_allowance_exceeded 카운터를 통해 관찰할 수 있는 AWS 측에서 수행한 pps 조절로 인한 패킷 삭제를 표시합니다.

샘플 CLI 출력

Tail drops 카운터가 계속 증가하는 샘플 출력 - 카운터가 증가하고 있는지 확인하기 위해 명령을 여

러 번 실행하여 실제로 tail drops인지 확인할 수 있습니다.

<#root>

```
csr_uut#show platform hardware qfp active statistics drop
Last clearing of QFP drops statistics : never
-----------------------------------------------------------------------
Global Drop Stats Packets Octets
-----------------------------------------------------------------------
Disabled 30 3693
IpFragErr 192 290976
Ipv4NoRoute 43 3626
Ipv6NoRoute 4 224
SdwanImplicitAclDrop 31 3899
```

TailDrop 19099700 22213834441

```
UnconfiguredIpv6Fia 3816 419760
```

여기에 표시되는 샘플 출력 - 30초마다 명령을 실행하여 실시간 데이터를 가져옵니다.

<#root>

```
csr_uut#show platform hardware qfp active datapath infrastructure sw-cio
Credits Usage:

ID Port Wght Global WRKR0 WRKR1 WRKR2 WRKR3 WRKR4 WRKR5 WRKR6 WRKR7 WRKR8 WRKR9 WRKR10 WRKR11 WRKR12 WR
1 rcl0 16: 455 0 4 1 2 3 2 2 4 4 4 4 0 4 23 512
1 rcl0 32: 496 0 0 0 0 0 0 0 0 0 0 0 0 0 16 512
2 ipc 1: 468 4 2 4 3 0 1 1 4 0 2 0 4 0 18 511
3 vxe_punti 4: 481 0 0 0 0 0 0 0 0 0 0 0 0 0 31 512
4 Gi1 4: 446 0 0 1 1 0 2 3 0 3 2 0 1 1 52 512
5 Gi2 4: 440 4 4 4 3 2 1 1 3 2 4 4 3 2 59 504
6 Gi3 4: 428 1 1 1 0 4 4 1 0 4 4 0 0 2 43 494
7 Gi4 4: 427 1 1 0 1 4 2 0 4 3 4 1 1 7 56 512

Core Utilization over preceding 12819.5863 seconds
-------------------------------------------------
ID: 0 1 2 3 4 5 6 7 8 9 10 11 12 13
```

% PP

```
: 6.11 6.23 6.09 6.09 6.04 6.05 6.06 6.07 6.05 6.03 6.04 6.06 0.00 0.00
% RX: 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 2.23
```

% TM:

```
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 4.79 0.00
% IDLE: 93.89 93.77 93.91 93.91 93.96 93.95 93.94 93.93 93.95 93.97 93.96 93.94 95.21 97.77
```

여기에 표시된 샘플 출력 - 입력/출력 속도를 확인하고 누락이 있는지 확인합니다. 또한 처리 로드 비율을 확인합니다. 100%에 달할 경우 노드가 용량에 도달했음을 의미합니다.

<#root>

```
csr_uut#show platform hardware qfp active datapath util summary
CPP 0: 5 secs 1 min 5 min 60 min

Input: Total (pps)

900215 980887 903176 75623
(bps) 10276623992 11197595912 10310265440 863067008

Output: Total (pps)

900216 937459 865930 72522
(bps) 10276642720 10712432752 9894215928 828417104

Processing: Load (pct)

56 58 54 4
```

인터페이스 레벨 통계에 대한 샘플 출력이 여기에 표시됩니다.

<#root>

```
csr_uut#sh plat hardware qfp active infrastructure bqs interface GigabitEthernet2
Interface: GigabitEthernet2, QFP interface: 7
Queue: QID: 111 (0x6f)
bandwidth (cfg) : 0 , bandwidth (hw) : 1050000000
shape (cfg) : 0 , shape (hw) : 0
prio level (cfg) : 0 , prio level (hw) : n/a
limit (pkts ) : 1043
Statistics:
depth (pkts ) : 0

tail drops (bytes): 0 , (packets) : 0


total enqs (bytes): 459322360227 , (packets) : 374613901
licensed throughput oversubscription drops:
(bytes): 0 , (packets) : 0
Schedule: (SID:0x8a)
Schedule FCID : n/a
bandwidth (cfg) : 10500000000 , bandwidth (hw) : 10500000000
shape (cfg) : 10500000000 , shape (hw) : 10500000000
Schedule: (SID:0x87)
Schedule FCID : n/a
bandwidth (cfg) : 200000000000 , bandwidth (hw) : 200000000000
shape (cfg) : 200000000000 , shape (hw) : 200000000000
Schedule: (SID:0x86)
Schedule FCID : n/a
bandwidth (cfg) : 500000000000 , bandwidth (hw) : 500000000000
shape (cfg) : 500000000000 , shape (hw) : 500000000000



csr_uut#sh plat hardware qfp active infrastructure bqs interface GigabitEthernet3 | inc tail
tail drops (bytes): 55815791988 , (packets) : 43177643
```

RX/TX 정상 패킷, 누락된 패킷 통계에 대한 샘플 출력

<#root>

```
c8kv-aws-1#show controller
GigabitEthernet1 - Gi1 is mapped to UIO on VXE
rx_good_packets 346
tx_good_packets 243
rx_good_bytes 26440
tx_good_bytes 31813
rx_missed_errors 0
rx_errors 0
tx_errors 0
rx_mbuf_allocation_errors 0
rx_q0packets 0
rx_q0bytes 0
rx_q0errors 0
tx_q0packets 0
tx_q0bytes 0
GigabitEthernet2 - Gi2 is mapped to UIO on VXE
rx_good_packets 96019317
tx_good_packets 85808651
rx_good_bytes 12483293931
tx_good_bytes 11174853219
```

rx_missed_errors 522036

```
rx_errors 0
tx_errors 0
rx_mbuf_allocation_errors 0
rx_q0packets 0
rx_q0bytes 0
rx_q0errors 0
tx_q0packets 0
tx_q0bytes 0
GigabitEthernet3 - Gi3 is mapped to UIO on VXE
rx_good_packets 171596935
tx_good_packets 191911304
rx_good_bytes 11668588022
tx_good_bytes 13049984257
```

rx_missed_errors 21356065

```
rx_errors 0
tx_errors 0
rx_mbuf_allocation_errors 0
rx_q0packets 0
rx_q0bytes 0
rx_q0errors 0
tx_q0packets 0
tx_q0bytes 0
GigabitEthernet4 - Gi4 is mapped to UIO on VXE
rx_good_packets 95922932
tx_good_packets 85831238
rx_good_bytes 12470124252
tx_good_bytes 11158486786
```

rx_missed_errors 520328

```
rx_errors 46
tx_errors 0
rx_mbuf_allocation_errors 0
```

```
rx_q0packets 0
rx_q0bytes 0
rx_q0errors 0
tx_q0packets 0
tx_q0bytes 0
```

AWS의 배압으로 인해 발생하는 혼잡이 있는지 확인하기 위한 샘플 출력:

<#root>

```
csr_uut#show platform hardware qfp active datapath infrastructure sw-hqf
Name : Pri1 Pri2 None / Inflight pkts
GigabitEthernet4 : XON XON XOFF / 43732

HQF[0] IPC: send 514809 fc 0 congested_cnt 0
HQF[0] recycle: send hi 0 send lo 228030112
fc hi 0 fc lo 0
cong hi 0 cong lo 0
HQF[0] pkt: send hi 433634 send lo 2996661158
fc/full hi 0 fc/full lo 34567275
```

cong hi 0 cong lo 4572971630********Congestion counters keep incrementing**

```
HQF[0] aggr send stats 3225639713 aggr send lo state 3225206079
aggr send hi stats 433634
max_tx_burst_sz_hi 0 max_tx_burst_sz_lo 0
HQF[0] gather: failed_to_alloc_b4q 0
HQF[0] ticks 662109543, max ticks accumulated 348
HQF[0] mpsc stats: count: 0
enq 3225683472 enq_spin 0 enq_post 0 enq_flush 0
sig_cnt:0 enq_cancel 0
deq 3225683472 deq_wait 0 deq_fail 0 deq_cancel 0
deq_wait_timeout
```

다중 대기열에서 트래픽을 로드 밸런싱하는 방법에 대한 샘플 출력:

```
um-csr-uut#sh plat hardware qfp active datapath infrastructure sw-nic
pmd b1c5a400 device Gi1
RX: pkts 50258 bytes 4477620 return 0 badlen 0
pkts/burst 1 cycl/pkt 579 ext_cycl/pkt 996
Total ring read 786244055, empty 786197491
TX: pkts 57860 bytes 6546349
pri-0: pkts 7139 bytes 709042
pkts/send 1
pri-1: pkts 3868 bytes 451352
pkts/send 1
pri-2: pkts 1875 bytes 219403
pkts/send 1
pri-3: pkts 2417 bytes 242527
pkts/send 1
pri-4: pkts 8301 bytes 984022
pkts/send 1
```

```
pri-5: pkts 10268 bytes 1114859
pkts/send 1
pri-6: pkts 1740 bytes 175353
pkts/send 1
pri-7: pkts 22252 bytes 2649791
pkts/send 1
Total: pkts/send 1 cycl/pkt 1091
send 56756 sendnow 0
forced 56756 poll 0 thd_poll 0
blocked 0 retries 0 mbuf alloc err 0
TX Queue 0: full 0 current index 0 hiwater 0
TX Queue 1: full 0 current index 0 hiwater 0
TX Queue 2: full 0 current index 0 hiwater 0
TX Queue 3: full 0 current index 0 hiwater 0
TX Queue 4: full 0 current index 0 hiwater 0
TX Queue 5: full 0 current index 0 hiwater 0
TX Queue 6: full 0 current index 0 hiwater 0
TX Queue 7: full 0 current index 0 hiwater 0
pmd b1990b00 device Gi2
RX: pkts 1254741010 bytes 511773562848 return 0 badlen 0
pkts/burst 16 cycl/pkt 792 ext_cycl/pkt 1342
Total ring read 1012256968, empty 937570790
TX: pkts 1385120320 bytes 564465308380
pri-0: pkts 168172786 bytes 68650796972
pkts/send 1
pri-1: pkts 177653235 bytes 72542203822
pkts/send 1
pri-2: pkts 22541430O bytes 91947701824
pkts/send 1
pri-3: pkts 136817435 bytes 55908224442
pkts/send 1
pri-4: pkts 256461818 bytes 104687120554
pkts/send 1
pri-5: pkts 176043289 bytes 71879529606
pkts/send 1
pri-6: pkts 83920827 bytes 34264110122
pkts/send 1
pri-7: pkts 160636635 bytes 64585622696
pkts/send 1
Total: pkts/send 1 cycl/pkt 442
send 1033104466 sendnow 41250092
forced 1776500651 poll 244223290 thd_poll 0
blocked 1060879040 retries 3499069 mbuf alloc err 0
TX Queue 0: full 0 current index 0 hiwater 31
TX Queue 1: full 718680 current index 0 hiwater 255
TX Queue 2: full 0 current index 0 hiwater 31
TX Queue 3: full 0 current index 0 hiwater 31
TX Queue 4: full 15232240 current index 0 hiwater 255
TX Queue 5: full 0 current index 0 hiwater 31
TX Queue 6: full 0 current index 0 hiwater 31
TX Queue 7: full 230668 current index 0 hiwater 224
pmd b1712d00 device Gi3
RX: pkts 1410702537 bytes 498597093510 return 0 badlen 0
pkts/burst 18 cycl/pkt 269 ext_cycl/pkt 321
Total ring read 1011915032, empty 934750846
TX: pkts 754803798 bytes 266331910366
pri-0: pkts 46992577 bytes 16616415156
pkts/send 1
pri-1: pkts 49194201 bytes 17379760716
pkts/send 1
pri-2: pkts 46991555 bytes 16616509252
pkts/send 1
```

```
pri-3: pkts 49195026 bytes 17381741474
pkts/send 1
pri-4: pkts 48875656 bytes 17283423414
pkts/send 1
pri-5: pkts 417370776 bytes 147056906106
pkts/send 6
pri-6: pkts 46992860 bytes 16617923068
pkts/send 1
pri-7: pkts 49191147 bytes 17379231180
pkts/send 1
Total: pkts/send 2 cycl/pkt 0
send 339705775 sendnow 366141927
forced 3138709511 poll 2888466204 thd_poll 0
blocked 1758644571 retries 27927046 mbuf alloc err 0
TX Queue 0: full 0 current index 0 hiwater 0
TX Queue 1: full 0 current index 0 hiwater 0
TX Queue 2: full 0 current index 0 hiwater 0
TX Queue 3: full 0 current index 0 hiwater 0
TX Queue 4: full 0 current index 1 hiwater 0
TX Queue 5: full 27077270 current index 0 hiwater 224
TX Queue 6: full 0 current index 0 hiwater 0
TX Queue 7: full 0 current index 0 hiwater 0
```

pps_allowance_exceeded 카운터를 통해 관찰할 수 있는 AWS측의 pps 스로틀링으로 인한 패킷 삭제를 보여주는 샘플 출력입니다.

```
C8k-AWS-2#show controllers | in errors|exceeded|Giga

GigabitEthernet1 - Gi1 is mapped to UIO on VXE
  rx_missed_errors 1750262
  rx_errors 0
  tx_errors 0
  rx_mbuf_allocation_errors 0
  rx_q0_errors 0
  rx_q1_errors 0
  rx_q2_errors 0
  rx_q3_errors 0
  bw_in_allowance_exceeded 0
  bw_out_allowance_exceeded 0
  pps_allowance_exceeded 11750
  conntrack_allowance_exceeded 0
  linklocal_allowance_exceeded 0
```

이 번역에 관하여

Cisco는 전 세계 사용자에게 다양한 언어로 지원 콘텐츠를 제공하기 위해 기계 번역 기술과 수작업 번역을 병행하여 이 문서를 번역했습니다. 아무리 품질이 높은 기계 번역이라도 전문 번역가의 번역 결과물만큼 정확하지는 않습니다. Cisco Systems, Inc.는 이 같은 번역에 대해 어떠한 책임도 지지 않으며 항상 원본 영문 문서(링크 제공됨)를 참조할 것을 권장합니다.