コミュニティ版SONiCを AI/ML基盤適用できるか探る

NTT株式会社 ネットワークイノベーションセンタ
中野 寛二

2025/8/1

自己紹介

名前：中野 寛二

所属：NTT株式会社

　　ネットワークイノベーションセンタ

経歴：2021年入社

　　ホワイトボックススイッチを活用した研究開発

　　SONiC Workshop Japan運営 (2024年、2025年)

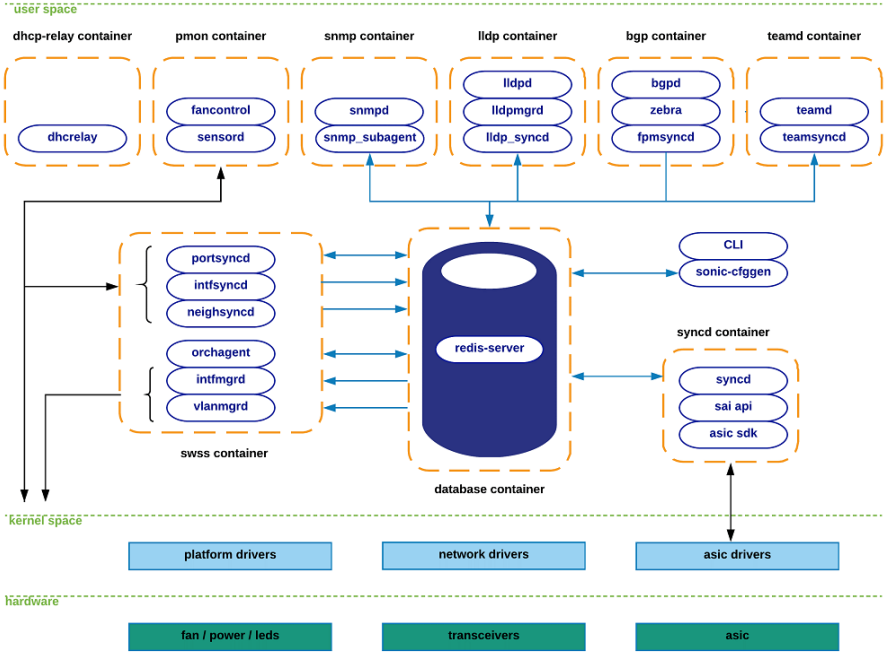
JANOG参加歴：J41, J42(若者支援), J45(若者支援), J50, J52, J53, J54, J55, J56(初登壇)

目次

SONiCの概要

SONiCとは

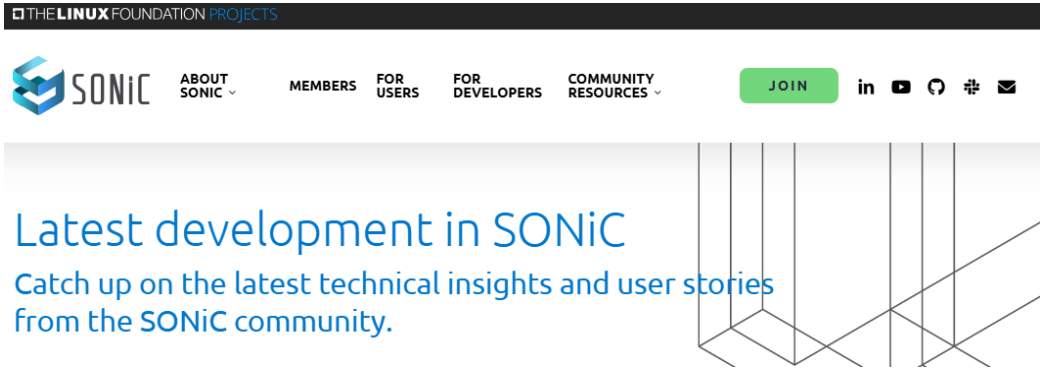
- SONiCはMicrosoft主導のOSSネットワークOSであり、ホワイトボックススイッチ向けに開発されている。
- OSSのNOSとしてはデファクトになっている。
- SAI (Switch Abstraction Inface)によりASICの抽象化をしている。

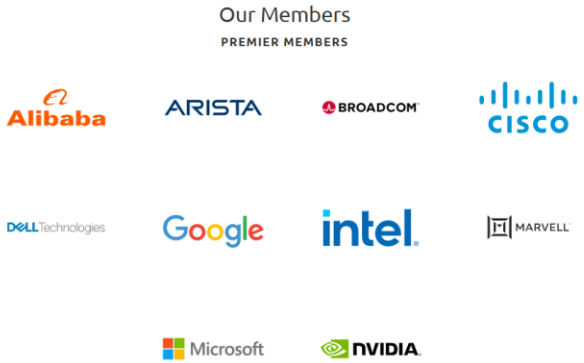


https://github.com/sonic-net/SONiC/wiki/Architecture

SONiC Foundation

- 2022年にOCPからLinux FoundationへSONiC Foundation運営が移管した。

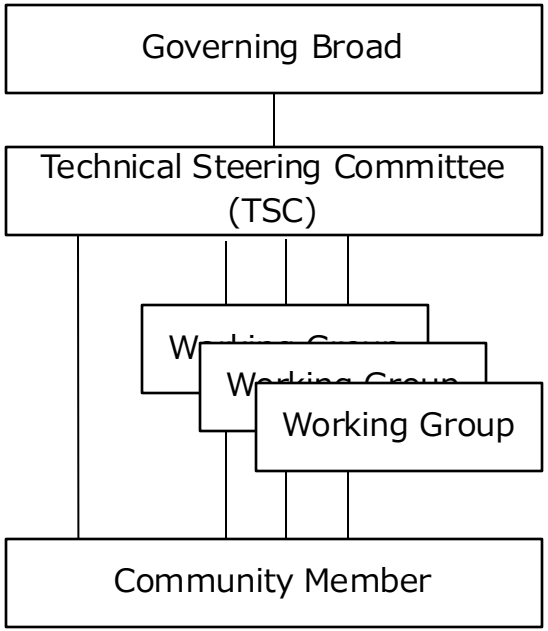




https://sonicfoundation.dev/

SONiC Working Group

- SONiCコミュニティは、ワーキンググループ内で技術議論をしている

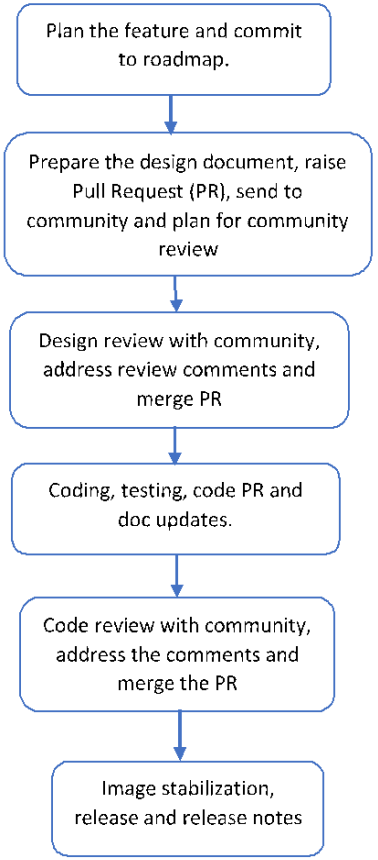


ワーキンググループ名	概要
Platform/OS Working Group	SONiCのPlatform機能およびOS統合に関する課題を議論するWG
sonic-smartswitch	SmartSwitch（ASIC + DPU/）に関する技術議論をするWG
SONiC AI Working Group	AIワークロードに適したEthernet技術をSONiCで実現するための技術検討を行うWG
sonic-wg-routing	ルーティングの機能や性能改善を議論。FRRoutingコミュニティからも開発者が参加しており、FRRoutingとの連携や改善を含めた議論が行われているWG
UMF Working Group	Unified Management Framework（UMF）に関する技術議論をするWG
SONiC-Scale-Up-WG	スケールアップ技術に関する技術議論を行うWG
SONiC Working Group for Optical Transport Network	光トランスポートネットワーク（OTN）に対応したSONiCの仕様や実装について議論するWG

SONiCのリリースプロセス

- SONiCは年二回(5月と11月)リリースを行っている
- SONiCに機能提案を行う際に次の開発サイクルを行う。

Plan	: 次回リリースのロードマップに追加するように、コミュニティに機能提案する。 また、HLDの概要を提案するのとレビューの日付を決定する。
Design review	: HLDには詳細なデザインとSONiCアーキテクチャへの適合性を記載することが必要になる。HLDはコミュニティからレビューが行われる。
Coding & Test	: HLDに従ってコーディングとテスト行う。HLDレビュー結果によってコード修正が必要となる。
Release	: 機能を実装したブランチがリリース日に提供される。ブランチの動作検証/コード修正期間(1-2ヶ月)が設けてあり、その間に当該機能の検証/修正作業を行う。

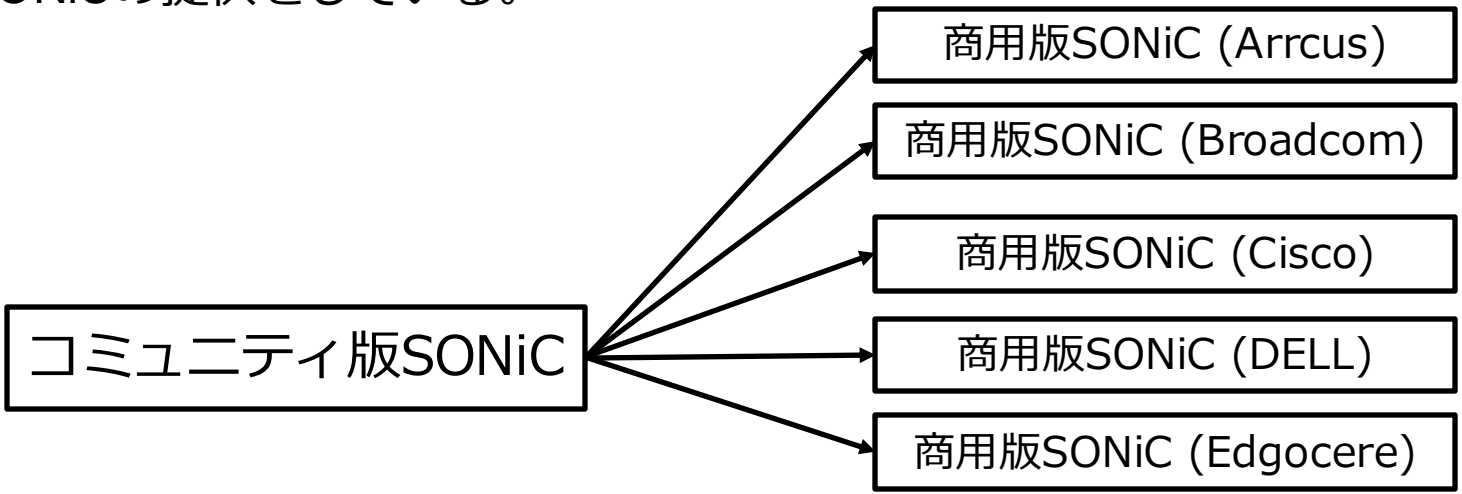


- 当該ブランチのリリース日までに完成しなかった機能が次のリリース以降に実装される。

https://github.com/sonic-net/SONiC/wiki/release_train

商用版SONiC

- コミュニティ版SONiCをベースにして、ベンダによるカスタマイズをしたSONiCの提供をしている。



https://mpls.jp/2022/presentations/mpls2022-Arrcus.pdf
https://jp.broadcom.com/products/ethernet-connectivity/software/enterprise-sonic
https://mpls.jp/2023/presentations/mpls2023-kamata.pdf
https://www.dell.com/ja-jp/dt/networking/sonic/index.htm
https://www.edge-core.com/jp/sonic/

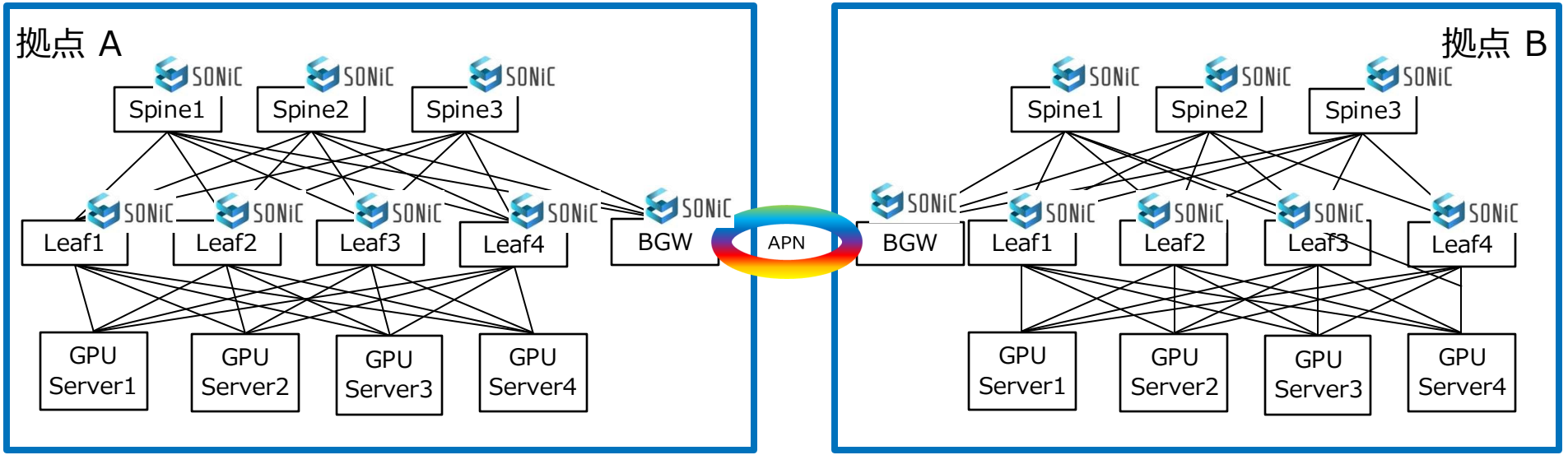
商用版SONiCの導入事例

- さくらインターネットはAI/ML基盤のネットワーク部分にSONiCの導入をしている。この基盤はHPLベンチマークによるコンピュータ性能のランキングであるTOP500 49位になった。
- 三井情報はAI/ML基盤のネットワーク部分にSONiCの導入をしている。この基盤は創薬支援サービス「Tokyo-1」として活用している。
- KDDIはHaaSのHW管理用NWにSONiCの導入をしている。
- NTT-PCは監視網にSONiCの導入をしている。

https://arxiv.org/abs/2507.02124
https://www.janog.gr.jp/meeting/janog54/sonic/
https://www.janog.gr.jp/meeting/janog52/sonicztp/
https://speakerdeck.com/sonic/jian-shi-wang-rutanisonicwocai-yong-siteji-qi-geng-gai-sitemita

AI/ML基盤へのSONiC適用

- 各社、AI/ML基盤のネットワーク装置にSONiCの導入している。
- NTTではAll-Photonics Network(APN)を活用した分散DCの検討をしている。

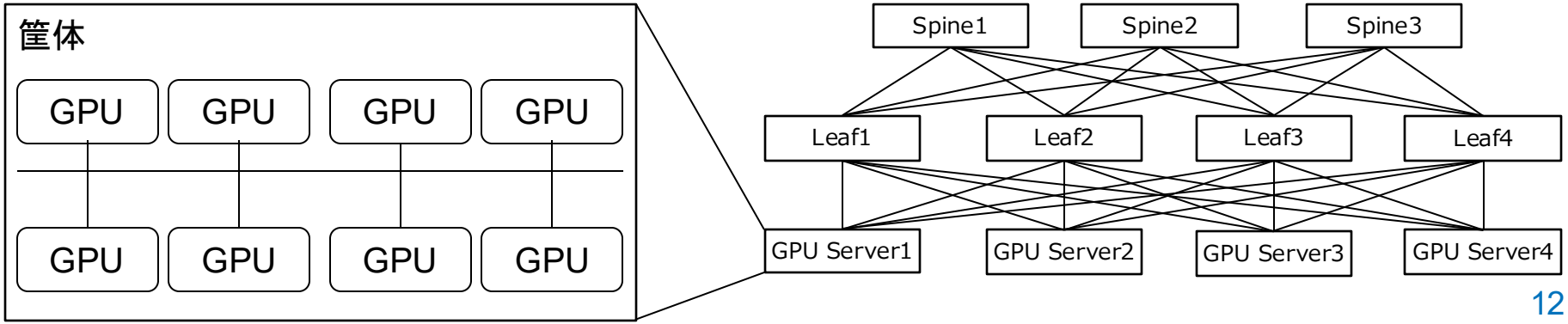


AI/ML基盤へのSONiC適用

AI/ML基盤の特徴

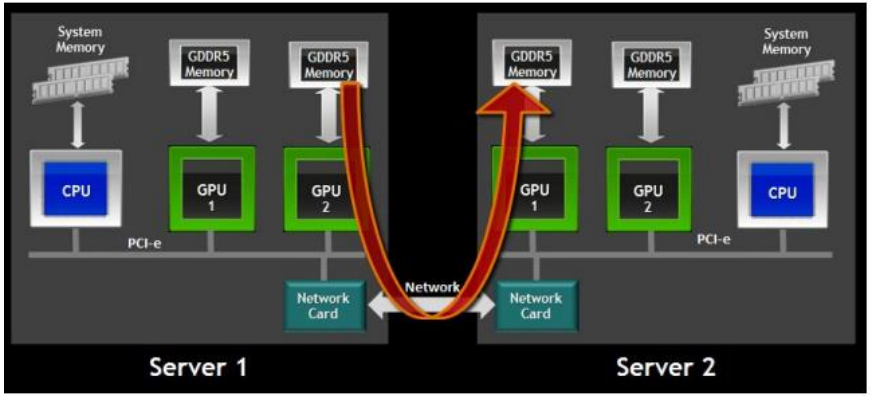
- AI/ML学習などの演算を高速化するために筐体のGPUサーバを増やして分散・並列化する
- 筐体内のGPU間の通信や筐体間の通信がボトルネックになる
- GPUの内部バスに近い性能の通信をEthernetで実現するかが課題となる

AI/ML基盤におけるGPU間通信は従来のTCP通信を主とするLossyな環境ではなく
GPUのための超広帯域、低遅延、ロスレスなネットワークが必要

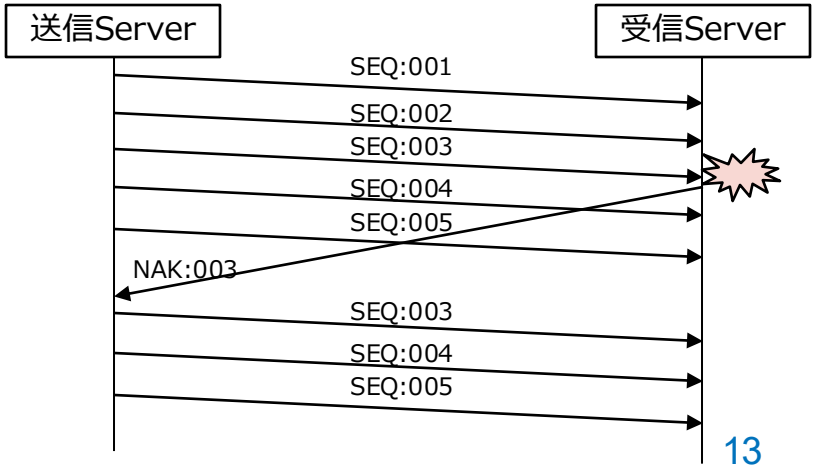


膨大なデータを高速に処理する技術

- RDMA(Remote Direct Memory Access)
  - ネットワークを通してリモートホストのGPUのメモリに直接データを送る
  - RDMAのために使用される通信規格としてInfiniBand、RoCEv2がある
  - InfiniBandとRoCEv2はロスレスを基本としているので再送に弱い(Go-Back-N)



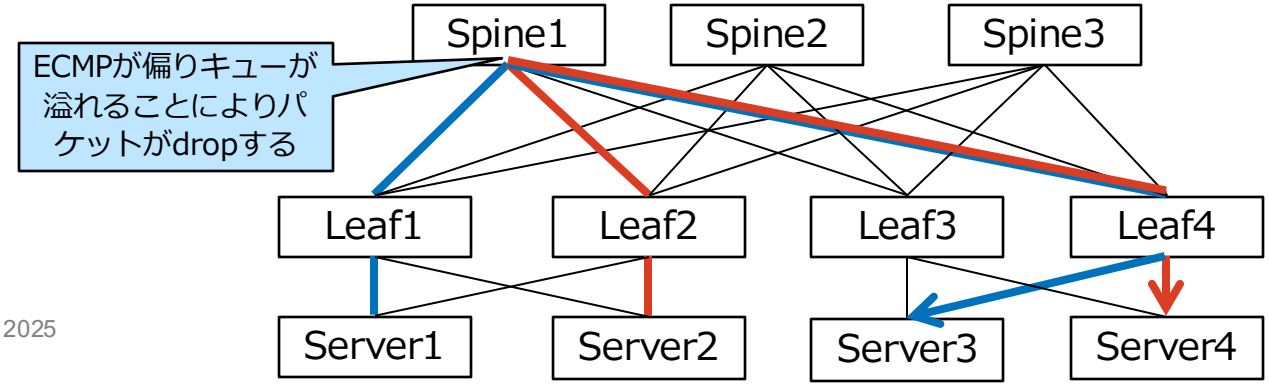
https://www.anandtech.com/show/4198/nvidia-announces-cuda-40



EthernetでAI/ML学習用ネットワークをつくるには

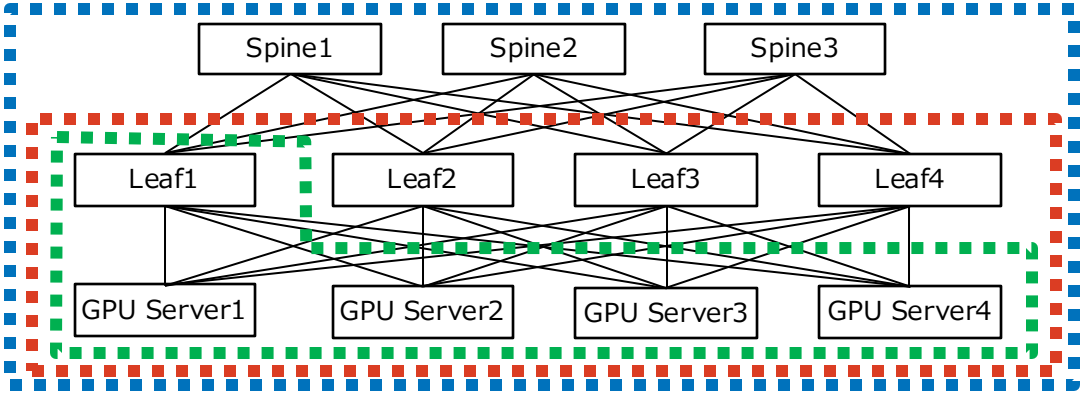
AI/MLクラスタにおける分散学習や推論処理では、複数GPUノード間で大量のデータが同時に転送されるため、一部のリンクに通信が集中し、ネットワーク輻輳が性能ボトルネックとなることがある。この課題に対して、以下の2つの観点から対策がとられている：

- Lossless（輻輳時のパケットロスを防ぐ）
- Load Balancing（輻輳の発生を回避）



AI/ML基盤の構成

- 小～中規模構成では現行SONiCの実装で対応可能
- 大規模構成では帯域を有効活用するためにECMPの最適化やARS導入が必要になる



Large Scale構成

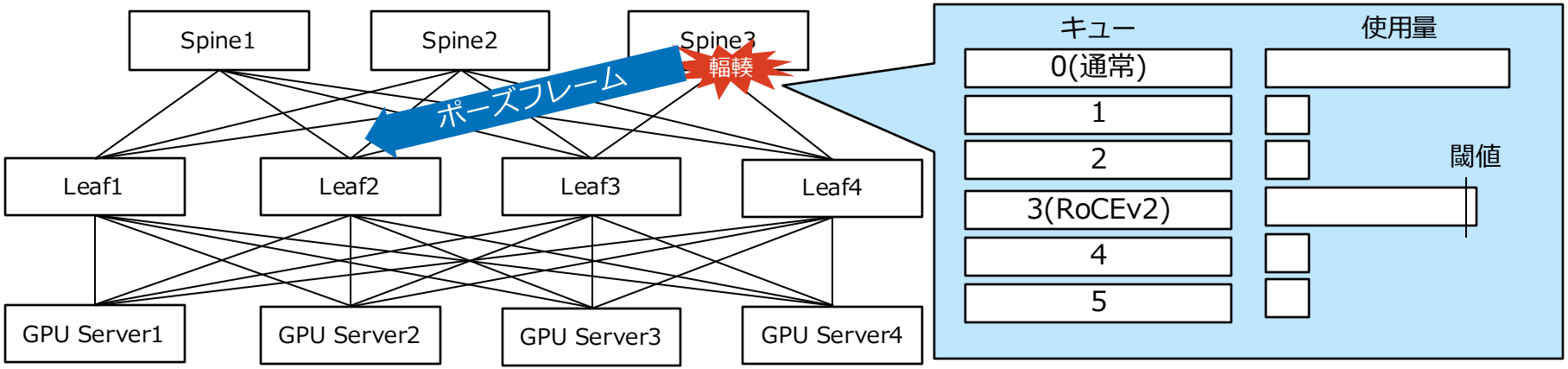
Medium Scale構成

Small Scale構成

規模	GPU数の目安	想定ユースケース	ネットワーク構成	ネットワーク要件	現状の実装状況
Small Scale	1 ～ 256	推論/RAG等	- 単一プレーン - スイッチ1台	- ECN/PFC	○ ECN/PFC実装済み
Medium Scale	～ 2,000	軽い学習・推論	- レール構成 - 8 Leafスイッチ構成 - 単一 or デュアルプレーン	- Smallと同様	○ ECN/PFC実装済み
Large Scale	2,000 ～ 16,000	大規模学習	- 2ティア構成 - 16 SU（スケーラブルユニット） - 単一またはデュアルプレーン	- ECN/PFC - Enhanced ECMP - ARS（Adaptive Routing Switching）	○ ECN/PFC実装済み × Enhanced ECMP/ARSは未実装

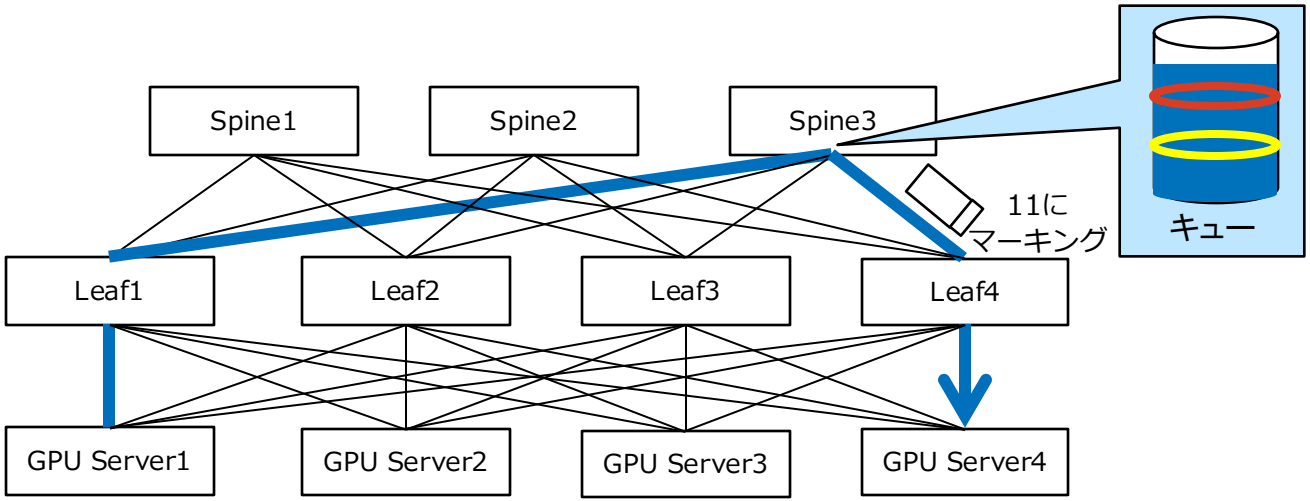
PFC(Priority Flow Control)

- トラフィック集中時にポーズフレームで前段のスイッチの送信を止める輻輳制御アルゴリズム。
- ポーズフレームを使ったキュー単位の輻輳制御アルゴリズム
- ロスレス対象のトラフィックをDSCPで分類して対象のキューでPFCが動作することでロスを防止



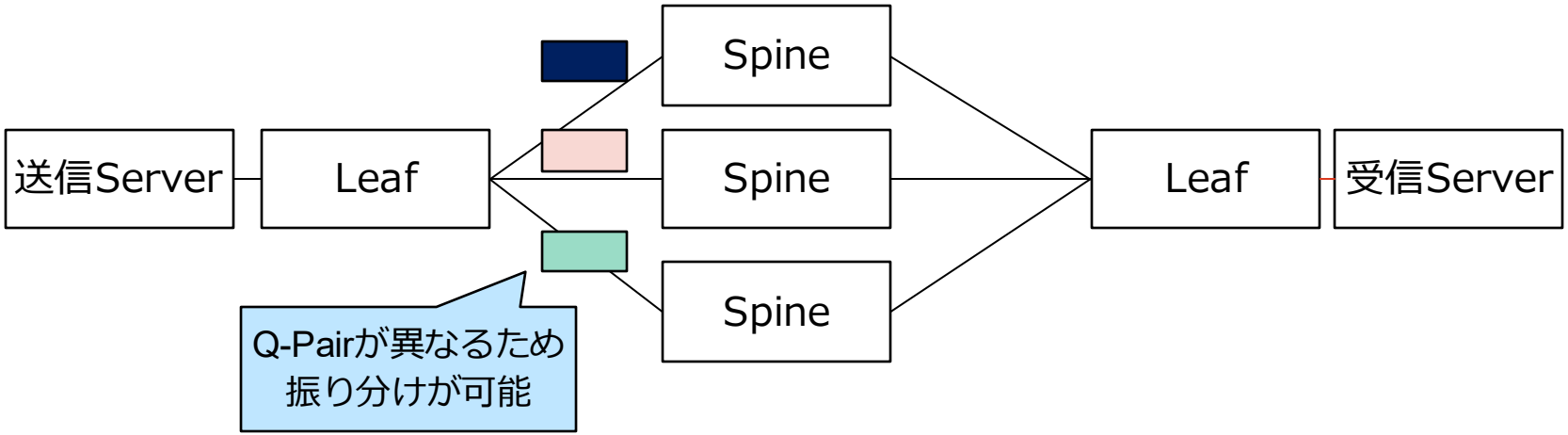
ECN(Explicit Congestion Notification)

- 輻輳しているときに送信サーバに通知することで輻輳制御するアルゴリズム
- IP ToSの2bitで輻輳制御
- 輻輳時はすべてのパケットにECN CE(11)をマーキング
- 受信側ホストは送信側ホストにCNPを送信して送信側はレート制御を実施



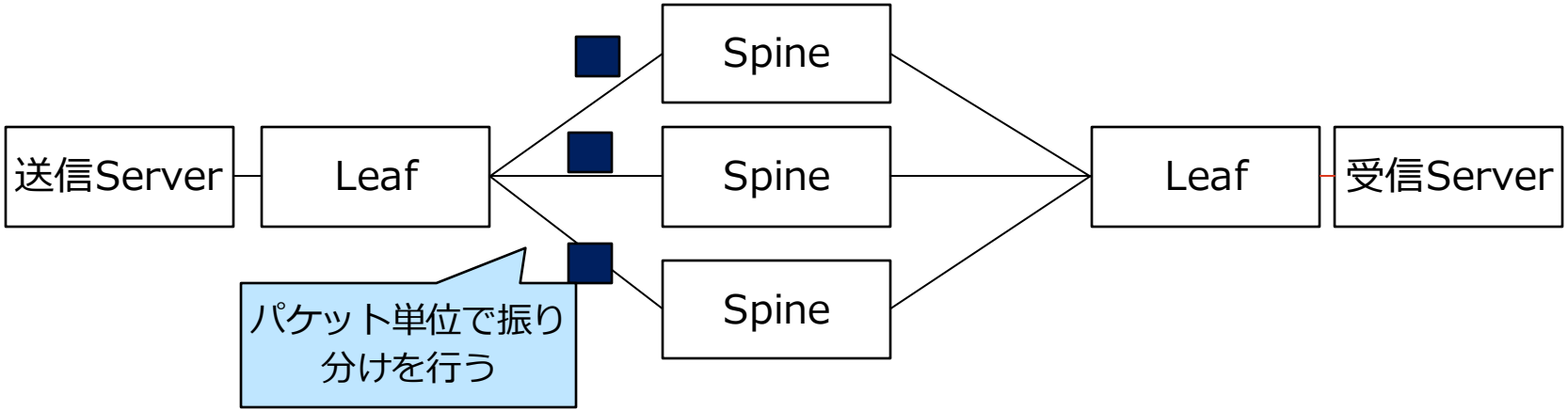
負荷分散技術

- Enhanced ECMP: RoCEv2のIB BTHヘッダーに含まれるQ-Pairをハッシュキーに組み込み、5-tuple + Q-Pair でハッシュ計算を行うことで、トラフィックの偏り問題を解決する方式。



負荷分散技術

- ARS（Adaptive Routing Switching）:従来のフロー単位ではなく、空いている経路にパケット単位で振り分ける方式。パケットの順序入れ替えは受信サーバ側で行う。



AI/ML基盤のトラフィック特徴

- AllReduce などの処理により、GPU間通信が一斉に発生し、バースト的にトラフィックが急増
- 一部スイッチやリンクに負荷が集中し、輻輳やパケットロスの原因となる
- バースト自体は抑制できないが、リアルタイム監視により早期検知と対処が可能
- SNMPのようなポーリング型では不十分であり、gNMIによるPush型テレメトリが有効

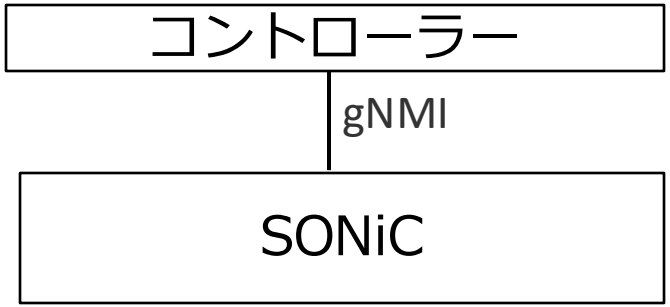
トラフィックの監視が重要になる。

gNMIによるテレメトリ

gNMIは次世代のネットワーク監視技術であり、各社による検討や実装・OpenConfig等による標準化が進められている。

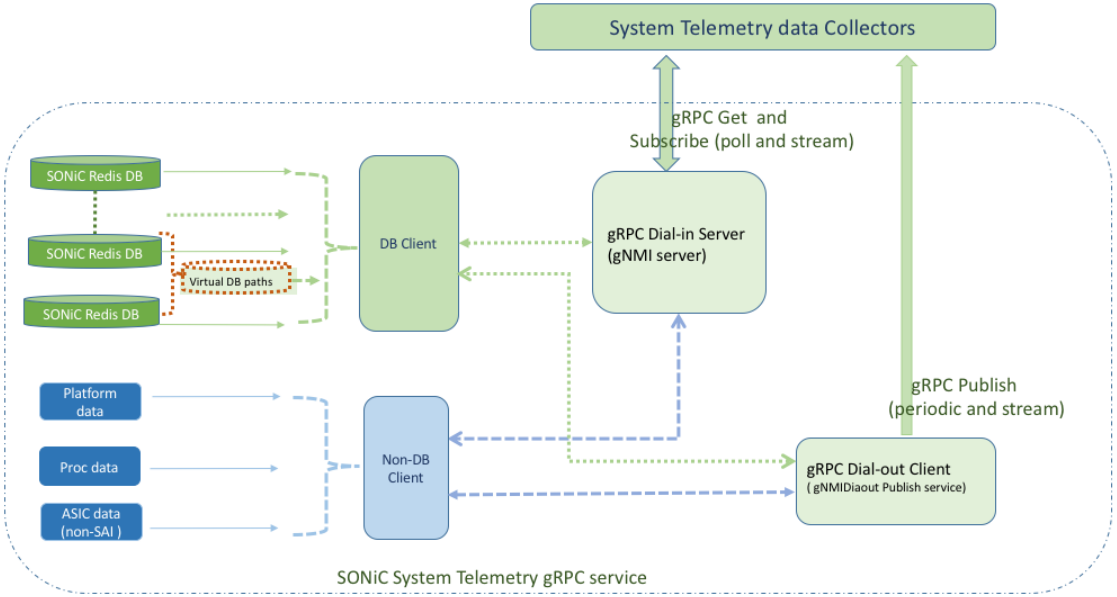
gNMIは、一般的には下記3つの特徴を持つよう設計されている。

1. SNMPと比較し低負荷なプロトコル設計による高精細でリアルタイムな情報取得
2. OpenConfigによるベンダフリーな情報取得
3. Pub/Subモデルの導入が容易な設計による高いスケール性



SONiCのテレメトリ機能 (1/2)

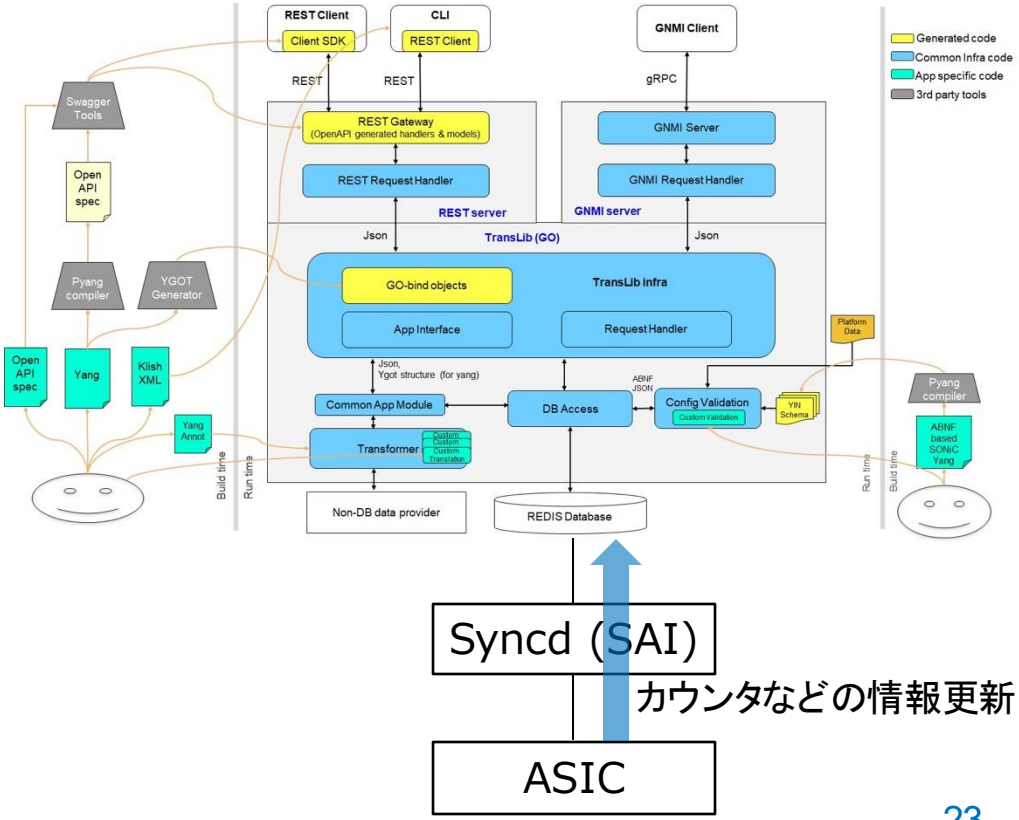
- SONiC のテレメトリは、OpenConfigによるデータ取得以外にSONiCのDBからデータ取得とDBに格納されていない装置情報の取得が可能である。
  - DB クライアントは SONiCのRedis データベースからのデータの取得をする
  - Non-DB クライアントは、装置情報などDBに格納していないデータの取得をする



SONiCのマネジメント (2/2)

gNMIはRedis DBの値を出力する

ASICからRedis DBに更新される間隔が重要になる

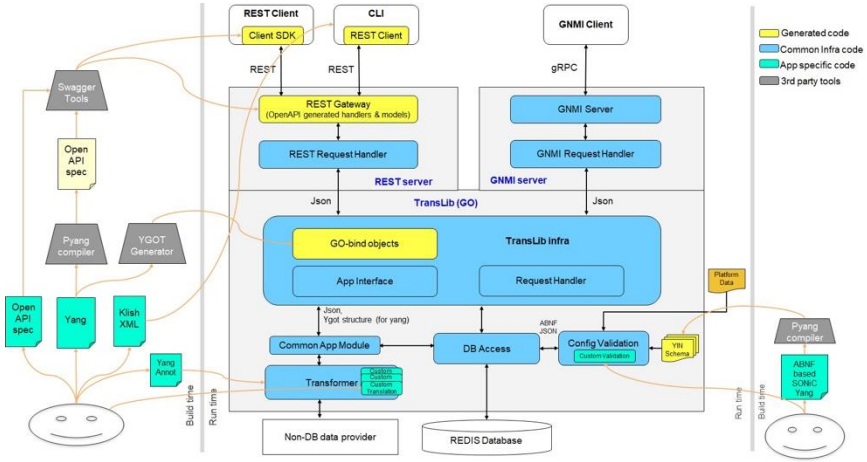


SONiCのマネジメント (2/2)

gNMIはRedis DBの値を出力する

ASICからRedis DBに更新される間隔が重要になる

Redis DBの更新間隔を確認したところ1秒間隔であった



実行コマンド: sonic-db-dump -n COUNTERS_DB | jq -r --arg oid "oid:0x1000000000002" --arg field "SAI_PORT_STAT_IF_IN_OCTETS" '.["COUNTERS:"+$oid].value[$field]'

15:05:25.164 oid:0x1000000000002 SAI_PORT_STAT_IF_IN_OCTETS: 23837641
15:05:26.737 oid:0x1000000000002 SAI_PORT_STAT_IF_IN_OCTETS: 23837845
15:05:28.313 oid:0x1000000000002 SAI_PORT_STAT_IF_IN_OCTETS: 23838049
15:05:29.904 oid:0x1000000000002 SAI_PORT_STAT_IF_IN_OCTETS: 23838049
15:05:31.480 oid:0x1000000000002 SAI_PORT_STAT_IF_IN_OCTETS: 23838253

Redis DBの更新間隔

Syncd (SAI)

ASIC

SONiCのテレメトリ機能動作確認 (DB クライアント) NTT

- DBに格納されているインターフェース情報の取得

Redis DBに格納されている値

```
admin@as7726:~$ redis-dump -d 4 -y -k
"INTERFACE|*"
{
  "INTERFACE|Ethernet16": {
    "expireat": 1722523250.1734788,
    "ttl": -0.001,
    "type": "hash",
    "value": {
      "NULL": "NULL"
    }
  },
  "INTERFACE|Ethernet16|10.0.0.1/30": {
    "expireat": 1722523250.1734843,
    "ttl": -0.001,
    "type": "hash",
    "value": {
      "NULL": "NULL"
    }
  }
}
```

```
ntt@gnmi_vm:~$ ./gnmi_get -notls -prefix
"CONFIG_DB" -xpath "INTERFACE" -
target_addr 172.27.254.214:8080 -
target_name target.com
== GetRequest:
 prefix: <
  target: "CONFIG_DB"
>
path: <
  elem: <
    name: "INTERFACE"
  >
>
encoding: JSON_IETF
```

gNMIで取得した結果

```
== GetResponse:
 notification: <
  prefix: <
    target: "CONFIG_DB"
  >
  update: <
    path: <
      elem: <
        name: "INTERFACE"
      >
    >
    val: <
      json_ietf_val: "
{
    "Ethernet16": {
        "NULL": "NULL"
    },
    "Ethernet16|10.0.0.1/30": {
        "NULL": "NULL"
    }
}
```

SONiCのテレメトリ機能動作確認 (Non-DB クライアント)

- 装置のCPU情報の取得

gNMIで取得した結果

```
ntt@gnmi_vm:~$ ./gnmi_get -notls -prefix "OTHERS" -
xpath "platform/cpu" -target_addr 172.27.254.214:8080
-target_name target.com
== GetRequest:
 prefix: <
  target: "OTHERS"
>
path: <
  elem: <
    name: "platform"
  >
  elem: <
    name: "cpu"
  >
>
encoding: JSON_IETF
```

```
== GetResponse:
 notification: <
  prefix: <
    target: "OTHERS"
  >
  update: <
    path: <
      elem: <
        name: "platform"
      >
      elem: <
        name: "cpu"
      >
    >
    val: <
      json_ietf_val: "
```

```
{
   "cpu_all": {
      "id": "cpu",
      "100ms": 0,
      "1s": 3,
      "5s": 3,
      "1min": 3,
      "5min": 3
   }
}
```

SONiCのテレメトリ機能動作確認 (OpenConfig) (1/2)

- OpenConfigでインターフェース情報の取得

gNMIで取得した結果


```
{
  "openconfig-interfaces:interface": [
    {
      "config": {
        "description": "",
        "mtu": 9100,
        "name": "Ethernet16"
      },
      "name": "Ethernet16",
      "openconfig-if-ethernet:ethernet": {
        "config": {
          "port-speed": "openconfig-if-
ethernet:SPEED_100GB"
        },
        "state": {
          "port-speed": "openconfig-if-
ethernet:SPEED_100GB"
        }
      },
      "state": {
        "admin-status": "UP",
        "counters": {
          "in-broadcast-pkts": "0",
          "in-discards": "2",
          "in-errors": "0",
          "in-multicast-pkts": "6389",
          "in-octets": "1597018",
          "in-pkts": "6389",
          "in-unicast-pkts": "0",
          "out-broadcast-pkts": "0",
          "out-discards": "0",
          "out-errors": "0",
          "out-multicast-pkts": "6391",
          "out-octets": "1641995",
          "out-pkts": "6391",
          "out-unicast-pkts": "0"
        },
        "description": "",
        "ifindex": 5,
        "mtu": 9100,
        "name": "Ethernet16",
        "oper-status": "UP"
      },
      "subinterfaces": {
        "subinterface": [
          {
            "index": 0
          }
        ]
      }
    }
  ]
}
```

SONiCのテレメトリ機能動作確認 (OpenConfig) (2/2)

- NTTとしてL1光情報取得の実装が必要な
- SONiCで実装済みのOpenConfigを取得した結果は正常に応答されることの確認をした

No	xpath	yang	結果
1	"/openconfig-acl:acl"	openconfig-acl.yang	正常応答
2	"/openconfig-interfaces:interfaces/interface[name="Ethernet16"]"	openconfig-interfaces.yang	正常応答
3	"/openconfig-lldp:lldp/interfaces/interface[name="Ethernet16"]"	openconfig-lldp.yang	正常応答
4	"/openconfig-platform:components"	openconfig-platform.yang	正常応答
5	"/openconfig-sampling-sflow:sampling"	openconfig-sampling-sflow.yang	正常応答
6	"ietf-yang-library:modules-state"	ietf-yang-library.yang	正常応答
7	"sonic-interface:sonic-interface"	sonic-interface.yang	正常応答
8	"sonic-port:sonic-port"	sonic-port.yang	正常応答

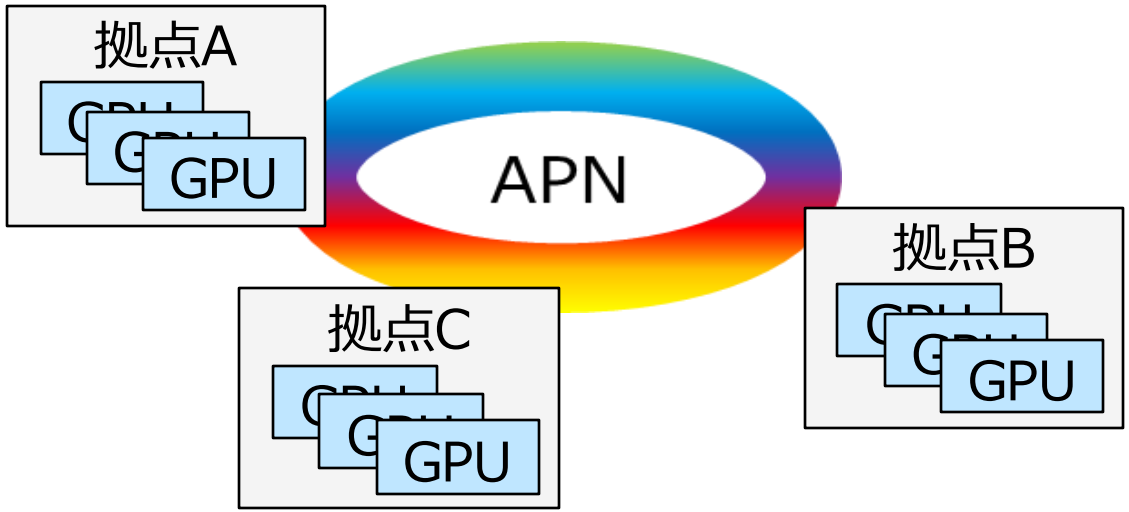
コミュニティ提案中のAI基板関連の機能

機能	提案企業	URL
High frequency telemetry	Microsoft	https://github.com/sonic-net/SONiC/pull/1795
PFC Historical Statistics	Arista	https://github.com/sonic-net/SONiC/issues/1904
Packet Trimming	Mellanox	https://github.com/sonic-net/SONiC/issues/1850
Local ARS	Marvell	https://github.com/sonic-net/SONiC/pull/1958
RoCEv2 support in SONiC Generic Hash (E-ECMP)	Marvell	https://github.com/sonic-net/SONiC/issues/2027
UEC Credit Based Flow Control	Marvell	https://github.com/sonic-net/SONiC/issues/2017
UEC Link Layer Retry	Marvell	https://github.com/sonic-net/SONiC/issues/2012

分散DC型AI/ML基盤へのSONiC適用

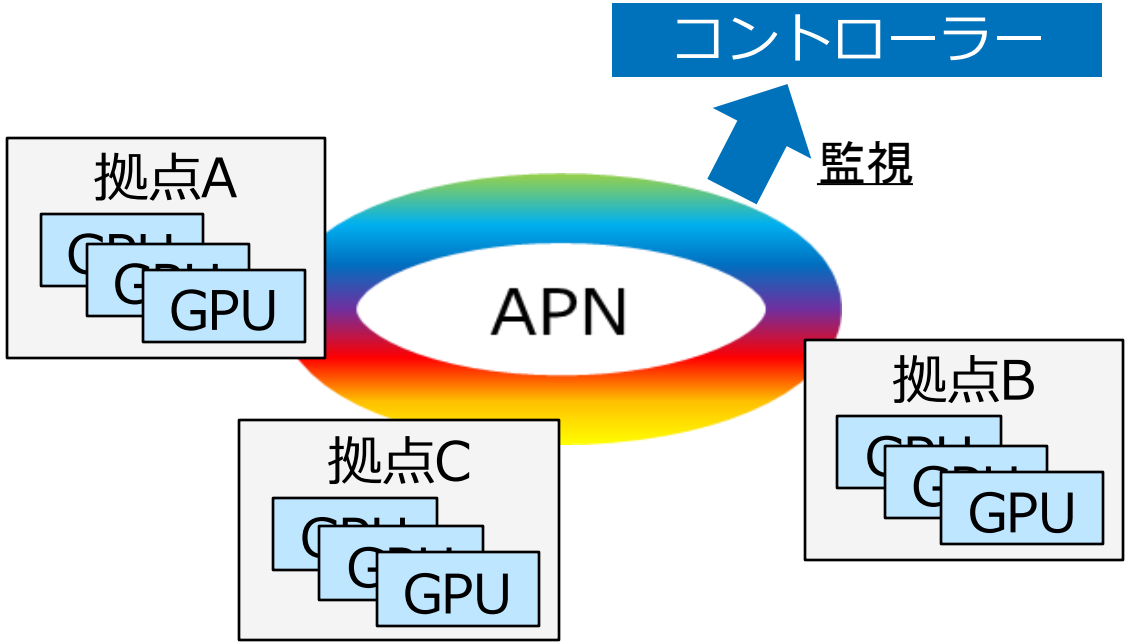
APNを活用した分散DC

- APNの長距離伝送・低遅延により、離れた拠点間でリソースを活用可能
- 災害リスクや電力・土地制約により、DCの分散化を検討中



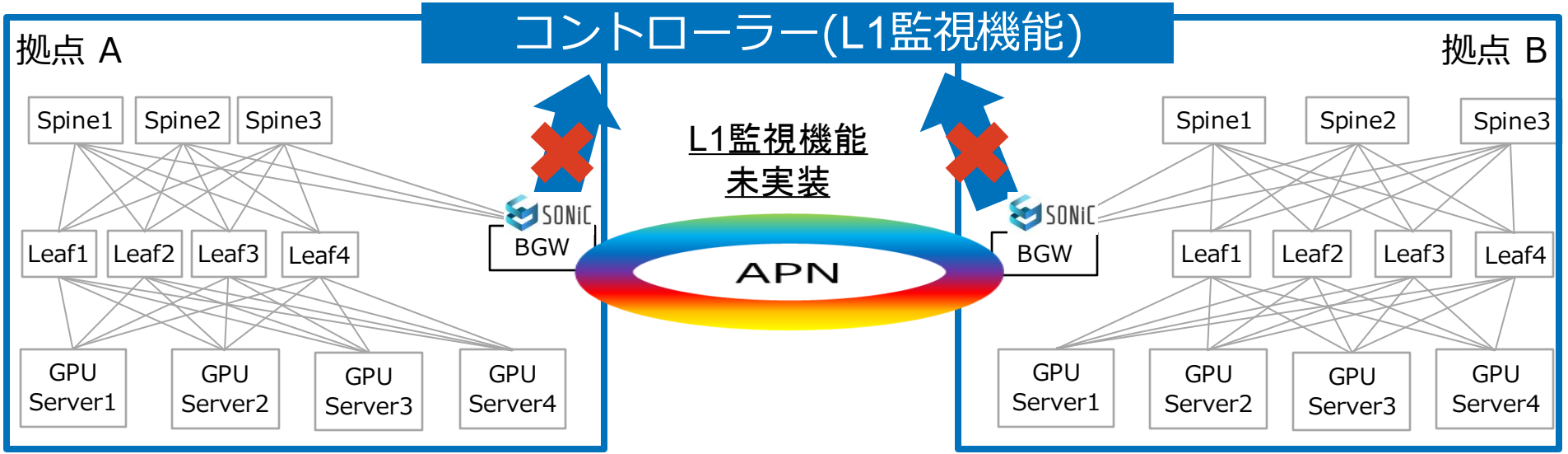
分散DCの監視

- APNは広域にまたがるため、遠隔からの可視化が不可欠である
- 障害の兆候を早期に捉え、迅速な保守対応につなげる必要がある
- ネットワーク状態のリアルタイム監視と通知機能の強化が求められる



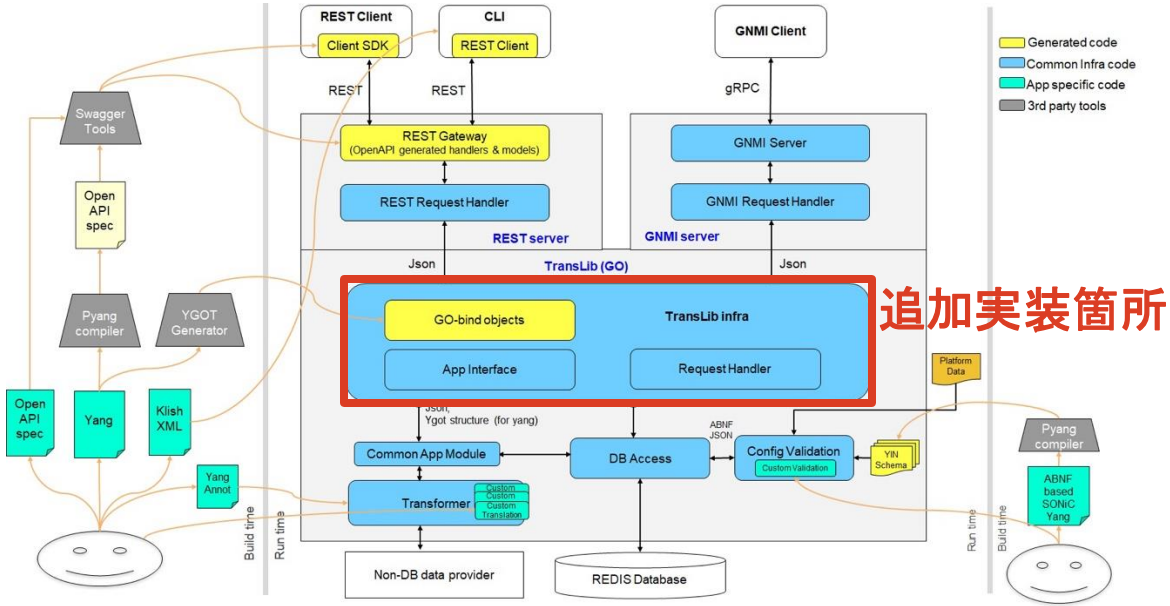
SONiCのL1情報取得に関する課題

- SONiCは光レベルやトランシーバー温度などL1情報を外部出力する
  インタフェースを備えていない
- 異常の兆候を検知・通知できず、障害の未然防止が困難



SONiCのgNMIにL1情報取得機能を実装

- L1情報をgNMI経由で取得するためにOpenConfig(openconfig-platform-transceiver.yang)モデルに対応する追加実装をした



https://github.com/sonic-net/SONiC/blob/master/doc/mgmt./Management%20Framework.md

L1情報取得の動作確認 (1/2)

- OpenConfigでL1情報の温度、受信パワー、送信バイアス、送信パワー、Pre-FEC BER、トランシーバー情報の取得できることの確認

```
ntt@gnmi_vm:~$ ./gnmi_get -insecure -xpath
"/openconfig-
platform:components/component[name=transcei
ver_Ethernet0]" -target_addr
172.27.254.24:8080 -target_name target.com
{
  "openconfig-platform:component": [
    {
     "name": "transceiver_Ethernet0",
      "openconfig-platform-
transceiver:transceiver": {
        "physical-channels": {
          "channel": [
            {
              "index": 0,
              "state": {
                "laser-temperature": {
                  "instant": "31.5"
                }
              }
            },
```

```
          {
            "index": 1,
            "state": {
              "input-power": {
                "instant": "-0.96"
              },
              "laser-bias-current": {
                "instant": "7.19"
              },
              "output-power": {
                "instant": "-1.09"
              }
            }
          },
```

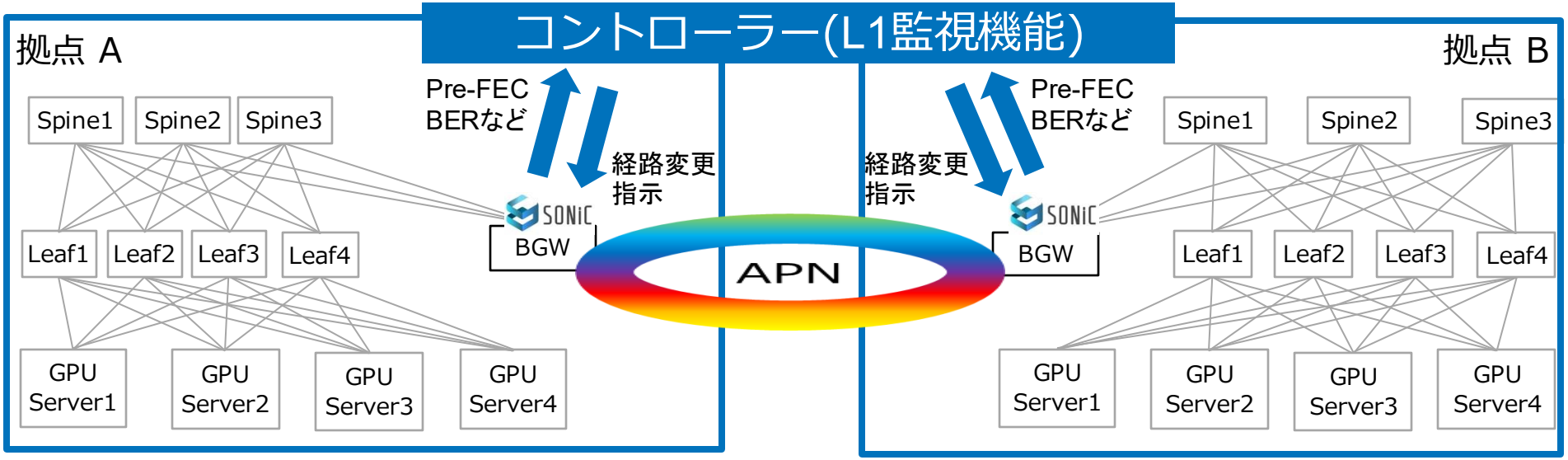
L1情報取得の動作確認 (2/2)

- OpenConfigでL1情報の温度、受信パワー、送信バイアス、送信パワー、Pre-FEC BER、トランシーバー情報の取得できることの確認


```
 "state": {
        "connector-type": "openconfig-
transport-types:LC_CONNECTOR",
        "date-code": "2023-07-
21T00:00:00.000Z",
        "input-power": {
          "avg": "-9.85",
          "instant": "-9.86",
          "max": "-9.82",
          "min": "-9.89"
        },
        "output-power": {
          "avg": "-10",
          "instant": "-10.01",
          "max": "-9.98",
          "min": "-10.04"
        },
        "pre-fec-ber": {
          "avg": "0.001077326126088339",
          "max": "0.000145771484073107",
          "min": "0.000141007542558746"
        },
        "serial-no": "N32329370049    ",
        "supply-voltage": {
          "instant": "3.26"
        },
        "vendor": "Edgecore        ",
        "vendor-part": "3c-2c-99",
        "vendor-rev": ""
      }
    },
    "state": {
      "firmware-version": "1.1.257",
      "hardware-version": "2.0",
      "oper-status": "openconfig-platform-types:ACTIVE",
      "part-no": "ECPO-QDDZR400G  ",
      "serial-no": "F72324000002    "
    }
  }
 ]
}
```

L1品質に基づく経路制御へ

- 劣化リンクをgNMIで検知し、L3ルーティングで回避
- Pre-FEC BERをもとに経路を切り替える仕組みの検討中



まとめと議論したいこと

まとめ

- コミュニティ版 SONiC における AI/ML 基盤向けの輻輳制御（PFC/ECN）はすでに実装済み。RoCEv2 向けのロードバランス技術（E-ECMP / ARS）は未実装であった。
- APN を活用した分散 DC 構成を想定し、遠隔地にある複数 DC 間の L1 光レイヤ状態（温度・光パワー等）を可視化・通知する仕組みを SONiC 上で新たに実装した。

議論したいこと

1. AI/ML基盤における輻輳制御・負荷分散へのSONiCの対応状況と課題
   - 現行の実装で十分か、今後必要となる拡張は何か？

2. トラフィック監視におけるテレメトリ機能の有効性
   - gNMIベースの監視でどこまで対応可能か、他の手段は必要か？

3. SONiCを商用利用する際の実現性と導入障壁
   - 商用導入における技術的・運用的な課題ついて。

Innovating a Sustainable Future for People and Planet

本研究成果は、国立研究開発法人情報通信研究機構（NICT）の助成事業（JPJ012368G60401）により得られたものです。


NTT