JANOGerとクラウドのワクワクする関係

Yukihiro Kikuchi

2020.01.13

はじめに概要おさらい

クラウドの普及に伴い、JANOGerもクラウド接続に携わったり、サービスプラットフォームとして触れたりする機会が増えてきました。クラウドにも物理/論理のネットワークは必須であり、その内外ではBGPやIPSec VPNといったおなじみのプロトコルが広く利用されています。これらのデザインやオペレーションにJANOGerをはじめとするインフラエンジニアの腕の見せ所、そしてワクワクするポイントが数多くあります。

本プログラムでは、JANOG43,44「インフラエンジニアがクラウドについて語らうBoF」での議論内容もふまえ、インフラエンジニアがどのようにすればクラウドとワクワクする関係を作れるのか、そしてどうすればよりよく活用していいけるのか、運用の勘所やネットワーク技術を中心に、広く深く議論できればと考えております。

スピーカー

- ネットワークエンジニアの 自分がクラウドインフラとどう向き合ってきたか

  エクイニクス・ジャパン株式会社

  丸野達矢さんから

- いちクラウド提供者側からこれからの「ワクワク」できるネットワークを考えてみる

  アマゾン ウェブ サービス ジャパン株式会社

  菊池より

JANOG45: **JANOGerとクラウドのワクワクする関係**

いちクラウド提供者側からこれからの「ワクワク」できるネットワークを考えてみる

Amazon Web Service Japan K.K.
Yukihiro Kikuchi

2020.01.13

自己紹介

名前：菊池　之裕(きくち　ゆきひろ)

所属：アマゾン ウェブ サービス ジャパン株式会社
技術統括本部　レディネス＆テックソリューション本部
シニアソリューションアーキテクト ネットワークスペシャリスト

ロール：Network系サービスについてのご支援

経歴：ISP,IXP,VPN運用、開発を経てネットワーク機器、仮想ルータ販売会社のプリセールス、プロダクトSEからAWSへ

好きな AWS サービス:AWS Transit Gateway,
AWS Direct Connect, AWS Marketplace

クラウドはまったく新しい価値観の世界

クラウドでの一般設計原則(Design Principles)

- 必要なキャパシティを勘に頼らない
- 本番規模でのシステムテストを行う
- アーキテクチャ試行の回数を増やすために自動化を取り入れる
- 発展的なアーキテクチャを受け入れる
- データ計測に基づいてアーキテクチャを決定する
- 本番で想定されるトラブルをあらかじめテストし、対策する

今日もって帰っていただきたいこと

信頼性について考えてみる
　　Design for failure

クラウドに必要な考え方と技術
　　本質的なことは変わらない

クラウドを使った自動化と高信頼性について
　　絶えず壊し続ける手法（カオスエンジニアリング）のご紹介

Agenda

- 信頼性について考えてみる
- オンプレミス／クラウド間接続を高信頼にするには
- クラウドに必要な考え方と技術
- クラウドを使った自動化と高信頼性について

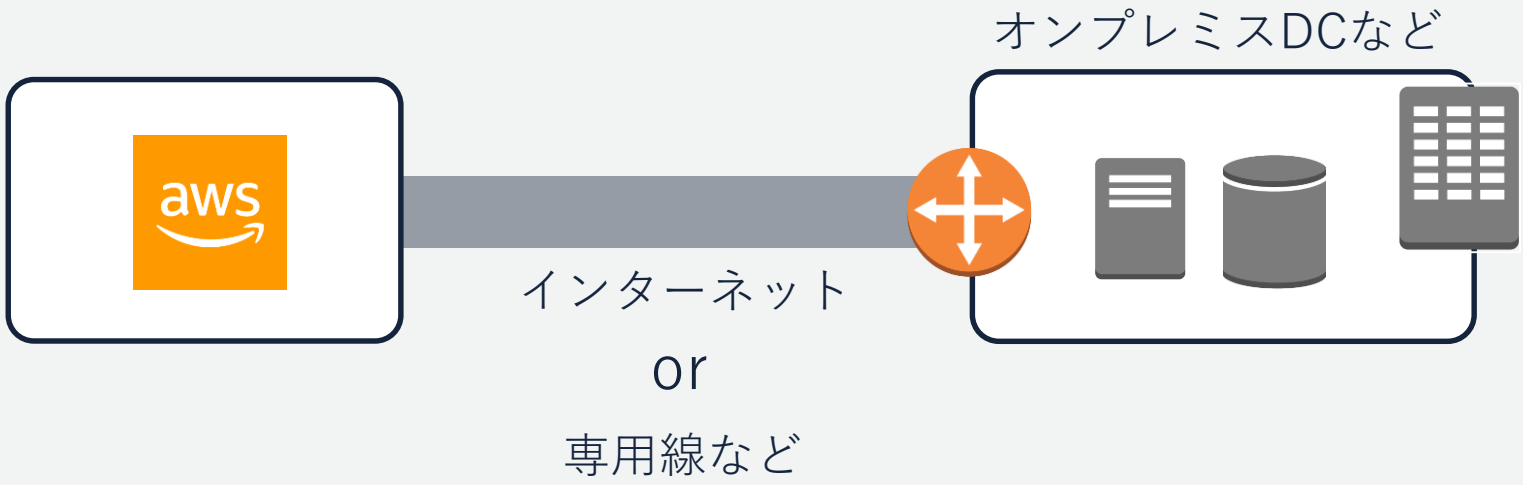
Agenda

- 信頼性について考えてみる
- オンプレミス／クラウド間接続を高信頼にするには
- クラウドに必要な考え方と技術
- クラウドを使った自動化と高信頼性について

信頼性について考えてみる

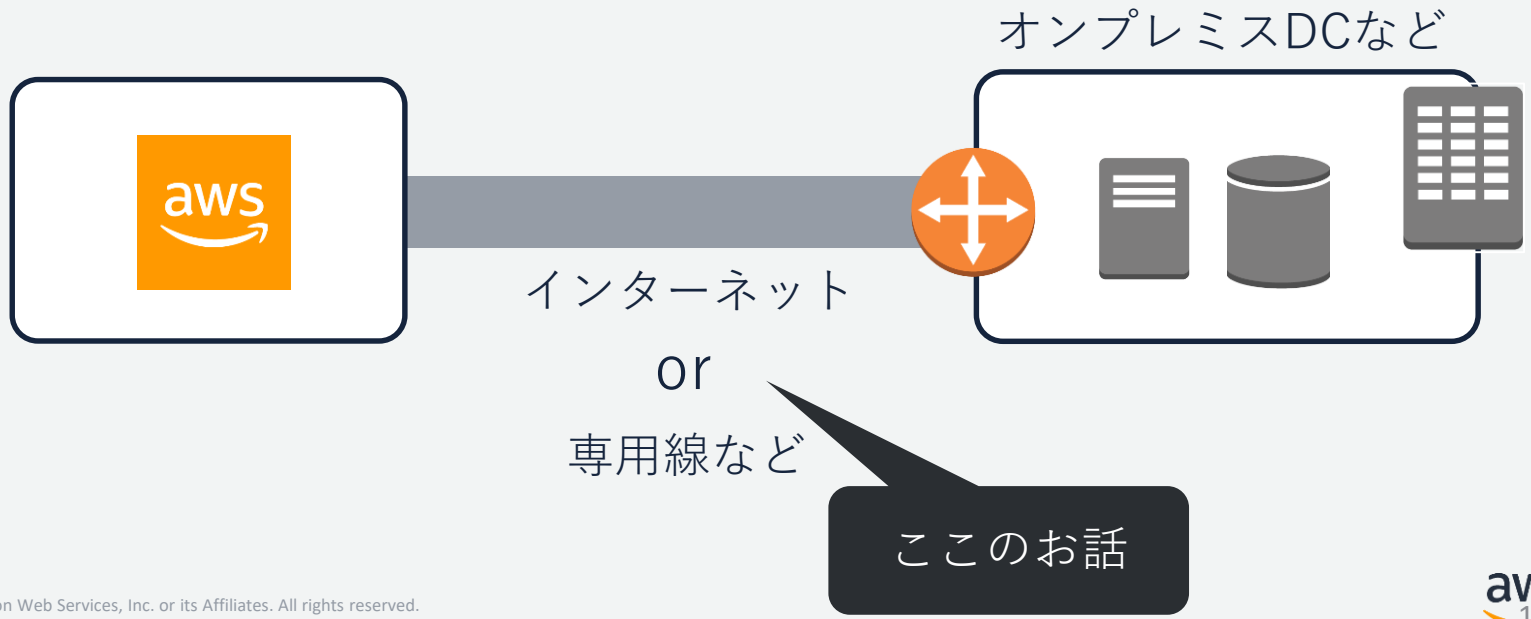
AWSとオンプレミスを接続するためには

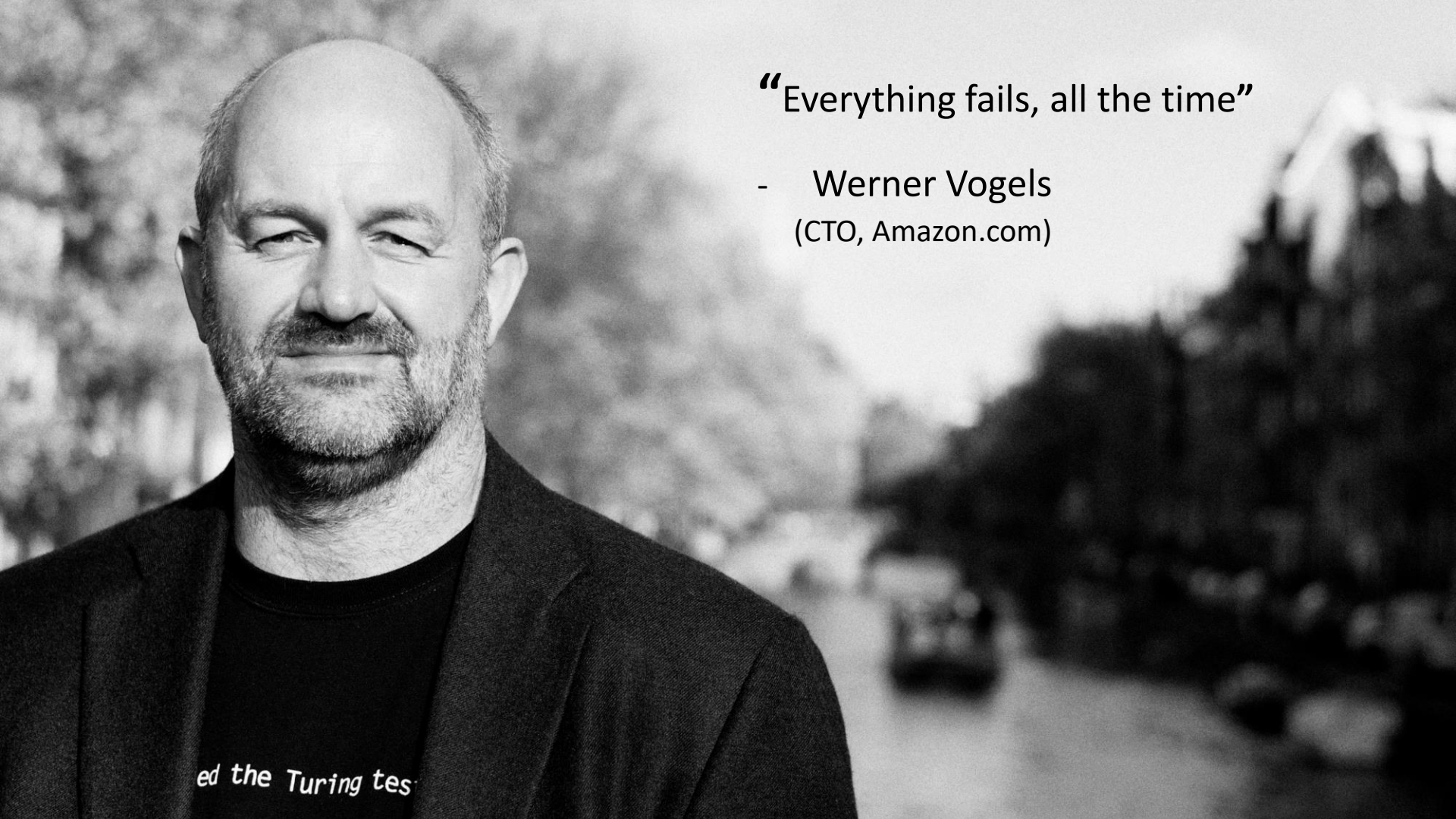
- オンプレミス環境をAWSクラウドに接続するためにはネットワークが必要
- ネットワーク＝インターネットまたは専用線など

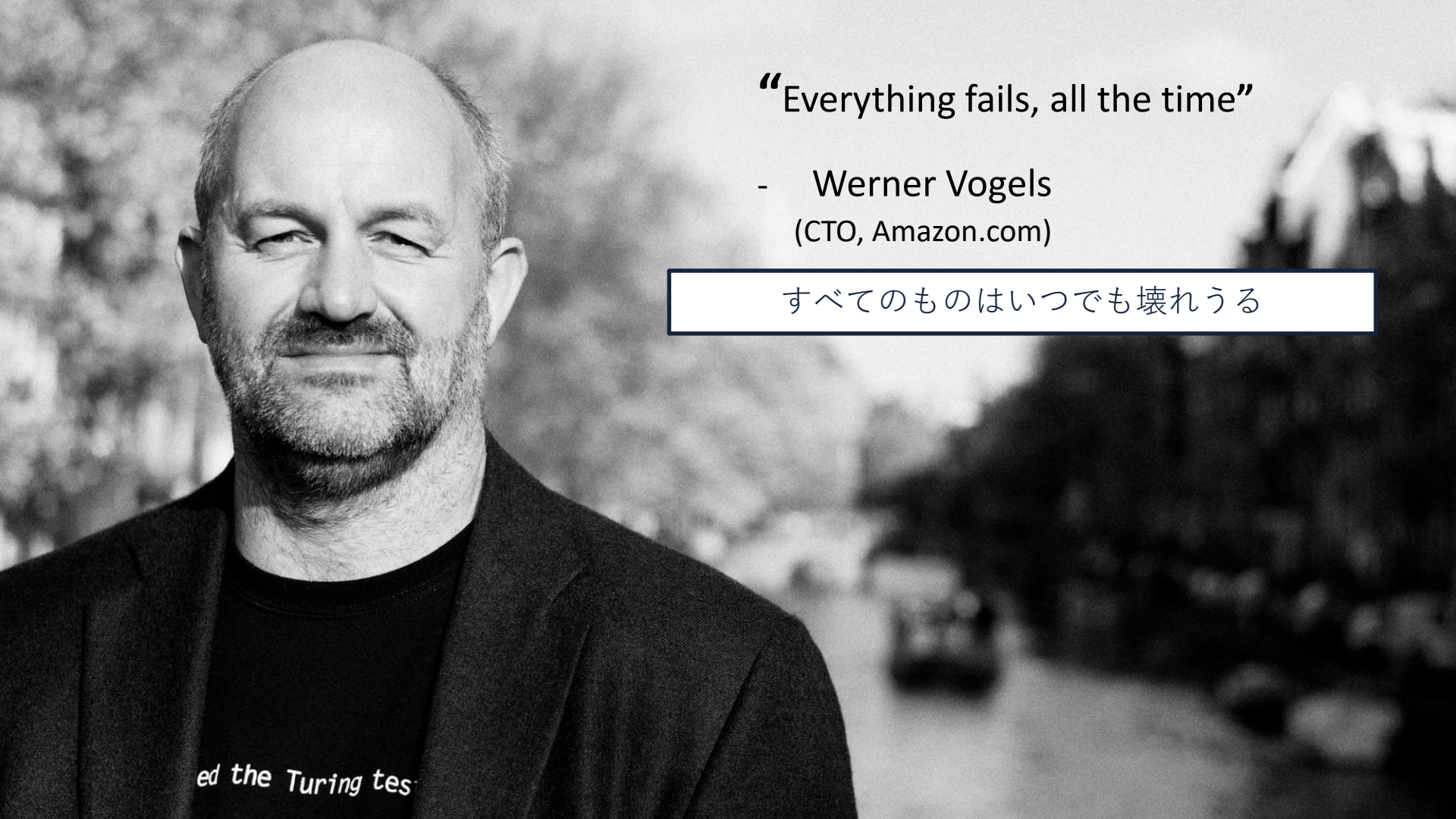


AWSとオンプレミスを接続するためには

- オンプレミス環境をAWSクラウドに接続するためにはネットワークが必要
- ネットワーク＝インターネットまたは専用線など




“Everything fails, all the time”
- Werner Vogels
(CTO, Amazon.com)
ed the Turing tes


“Everything fails, all the time”
- Werner Vogels
(CTO, Amazon.com)
すべてのものはいつでも壊れうる
ed the Turing tes

障害事例

米国
工事中の事故で
光ファイバー切断

13

パケットドロップ

障害事例

ブラジル
ゴミ捨て場火災で
光ファイバー切断

お客様より
インパクトの申告なし

リージョン内ネットワークの可用性

ダークファイバー網

低遅延と経路の
多様性に最適化

自社による運用

ファイバーの経路を
定常的に点検

位置のトラッキング

地理空間座標を
用いた配置の分析

キャパシティ拡張

光波長多重化を活用

Agenda

- 信頼性について考えてみる
- オンプレミス／クラウド間接続を高信頼にするには
- クラウドに必要な考え方と技術
- クラウドを使った自動化と高信頼性について

オンプレミス／クラウド間接続を
高信頼にするには

オンプレミス／AWS間接続を高信頼にしたい

解決したい課題

・ミッションクリティカルなワークロードを扱いたい

・オンプレミスとAWS間の接続を高信頼で構築したい

解決パターン

・SPOF(Single Point of Failure）の排除

・費用対効果の観点で冗長化する箇所を優先付けして設計

オンプレミス／AWS間の冗長化・高信頼設計

その前に・・・冗長の考え方を変えてみる

【よくある要求仕様】

TCPセッションは障害時に即座にバックアップ機に引き継がれる事

➡ データセンター／Availability Zoneを跨いだ時点で不可能

システム全体の正常性の定義を考え直してみる
1パケット落としてもTCPでは再送がかかる
システム全体でリカバリができていれば良しとする

Design For Failure の考え方

”ゼッタイに落ちないシステムはない”

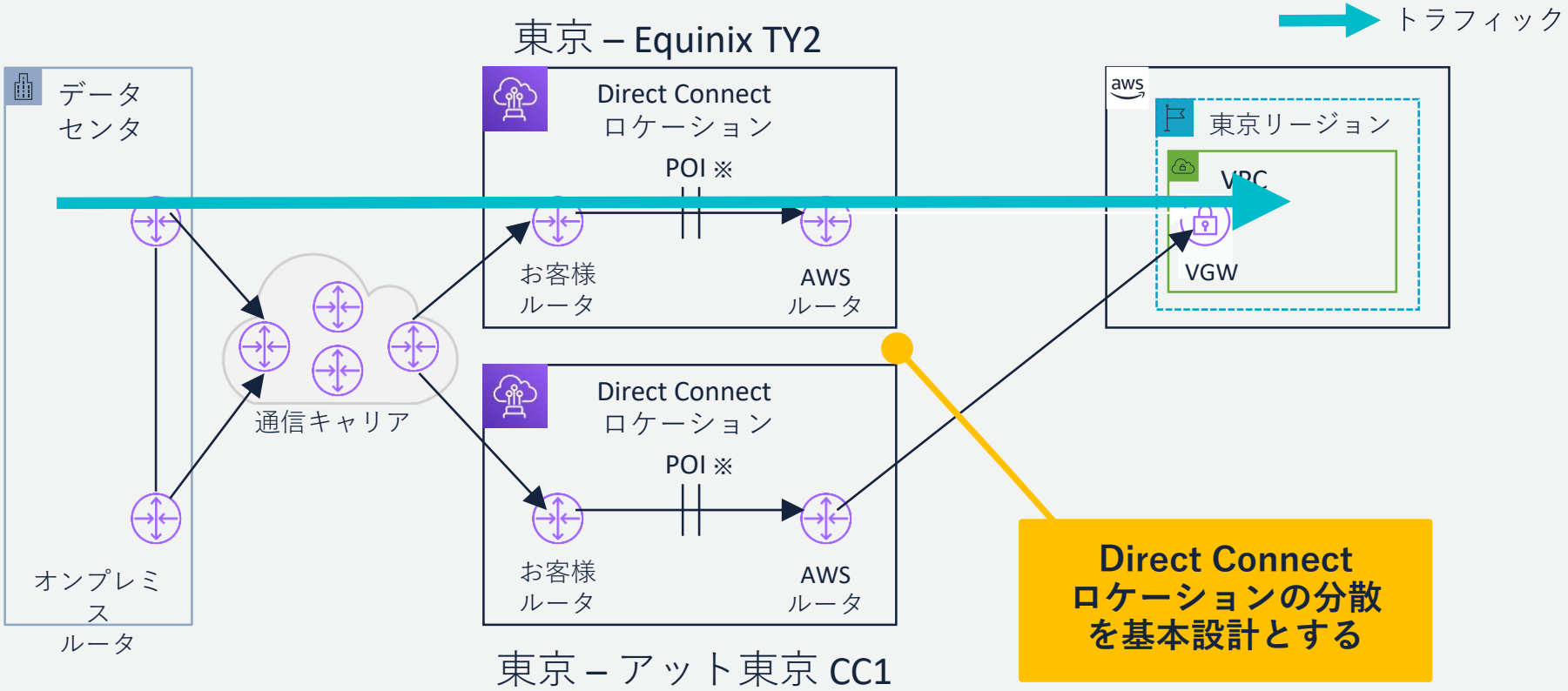
疎結合や、水平に展開できるシステムを目指す
IPアドレスに依存しない。DNSを活用
DNSで複数エントリを書くようにする

詳しくはこちらのホワイトペーパーを
「AWSを用いた耐障害性の高いアプリケーションの構築 」

https://aws.amazon.com/jp/whitepapers/designing-fault-tolerant-applications/

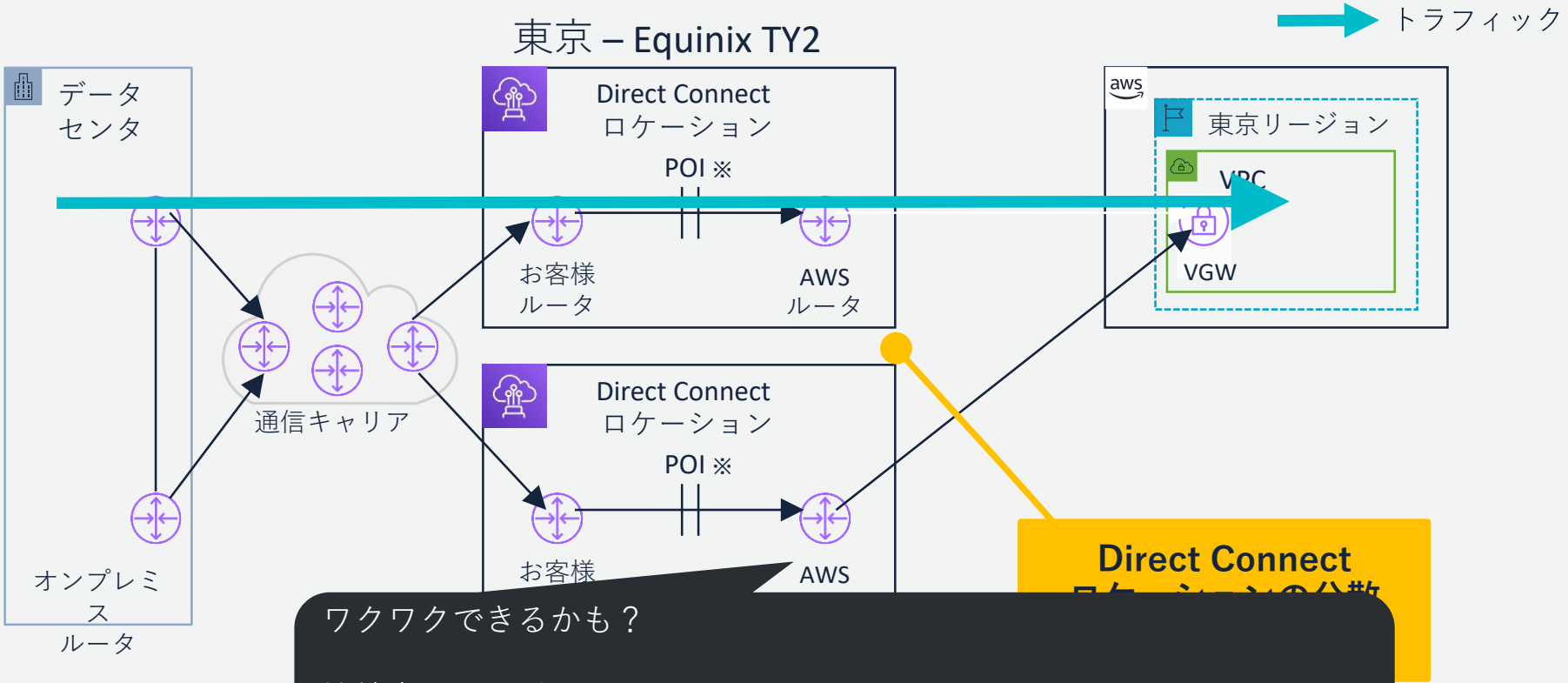
専用線接続における信頼性

可用性のためのデュアルDirect Connectロケーション



※ POI：Point Of Interface
責任分界点

可用性のためのデュアルDirect Connectロケーション



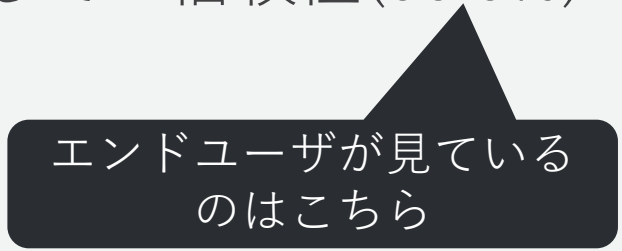
信頼性についての定義

- 網としての信頼性（99.99% ↑）
- アプリケーションとしての信頼性(99.9%)

エンドユーザが見ているのはこちら

信頼性についての考え方

- 網としての信頼性（99.99% ↑）
- アプリケーションとしての信頼性(99.9%)



がむしゃらに網の信頼性を上げるよりかは
システム全体の信頼性を見てみては？
アプリケーションの人とお話するとよりワクワクできる関係ができる

Agenda

- 信頼性について考えてみる
- オンプレミス／クラウド間接続を高信頼にするには
- クラウドに必要な考え方と技術
- クラウドを使った自動化と高信頼性について

クラウドに必要な考え方と技術

クラウドはまったく新しい価値観の世界

クラウドでの一般設計原則(Design Principles)

- 必要なキャパシティを勘に頼らない
- 本番規模でのシステムテストを行う
- アーキテクチャ試行の回数を増やすために自動化を取り入れる
- 発展的なアーキテクチャを受け入れる
- データ計測に基づいてアーキテクチャを決定する
- 本番で想定されるトラブルをあらかじめテストし、対策する

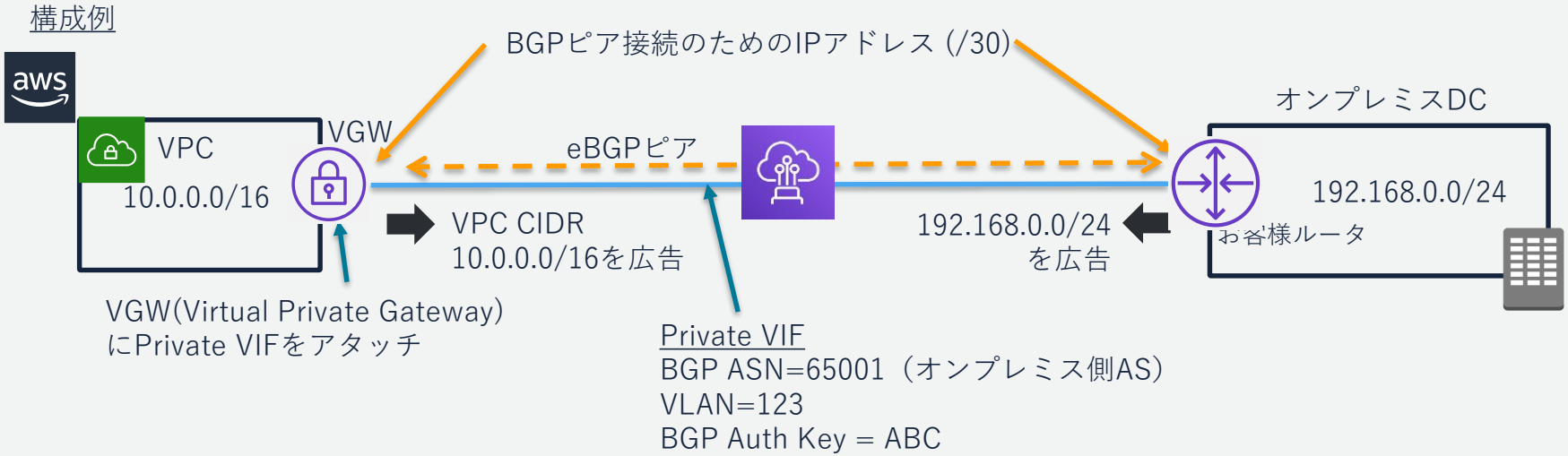
ネットワークエンジニアはいらない？でも、、

- 本質は変化しない
- 例えば、、

Direct Connect（専用線接続）の場合

- Private VIFを使用してVPCへの接続を提供
- お客様ルータでBGP, MD5認証, IEEE802.1q VLANのサポートが必要
- VPCのCIDR(IPv4,IPv6)がAWSから広告される
- Jumbo Frame(MTU=9001)をサポート

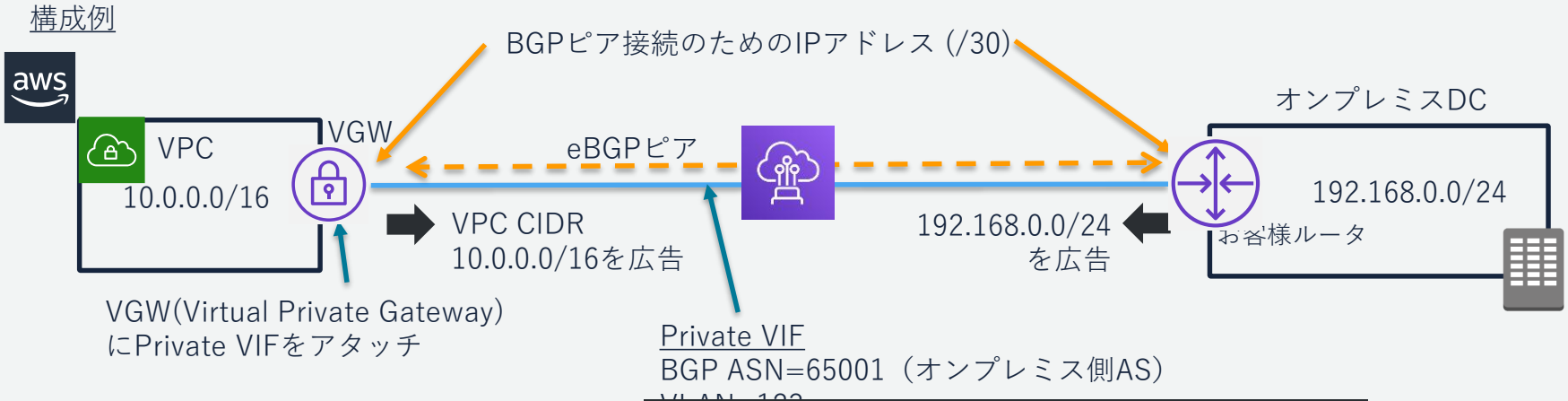
  https://docs.aws.amazon.com/ja_jp/directconnect/latest/UserGuide/set-jumbo-frames-vif.html



Direct Connect（専用線接続）の場合

- Private VIFを使用してVPCへの接続を提供
- お客様ルータでBGP, MD5認証, IEEE802.1q VLANのサポートが必要
- VPCのCIDR(IPv4,IPv6)がAWSから広告される
- Jumbo Frame(MTU=9001)をサポート

  https://docs.aws.amazon.com/ja_jp/directconnect/latest/UserGuide/set-jumbo-frames-vif.html

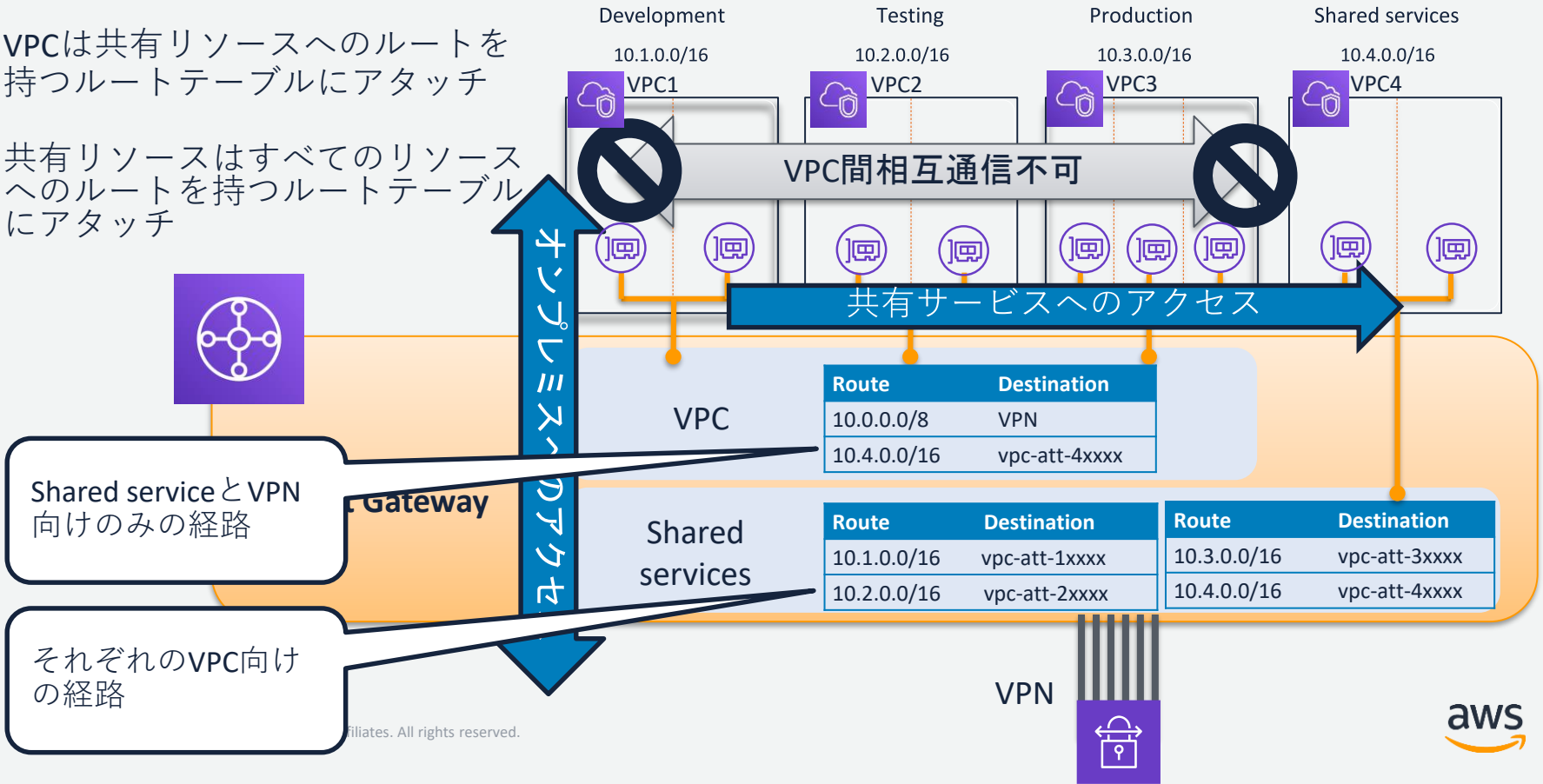


BGPに関する知識が必要

Transit Gatewayの場合

VPCは共有リソースへのルートを持つルートテーブルにアタッチ

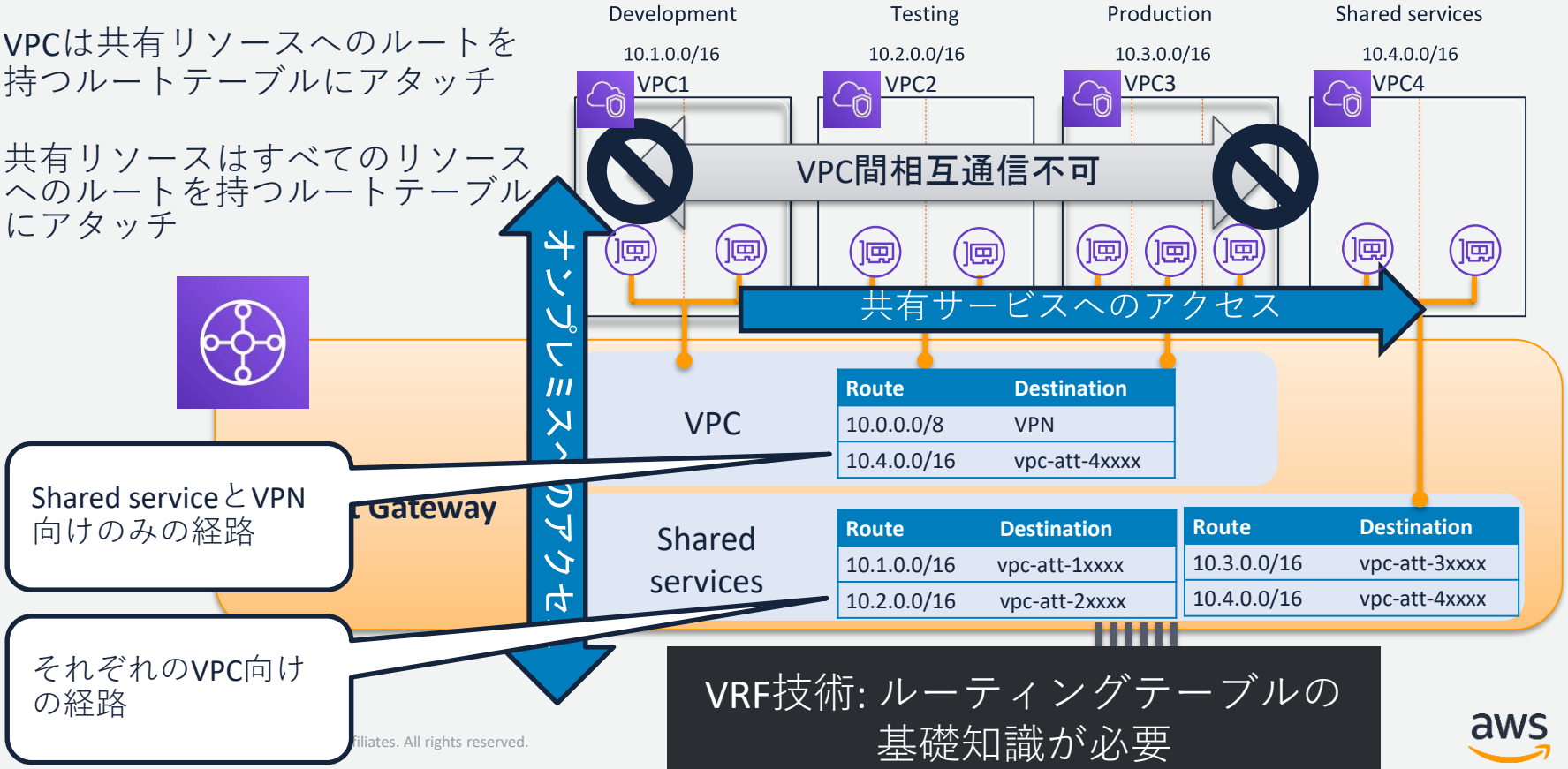
共有リソースはすべてのリソースへのルートを持つルートテーブルにアタッチ



Transit Gatewayの場合

VPCは共有リソースへのルートを持つルートテーブルにアタッチ

共有リソースはすべてのリソースへのルートを持つルートテーブルにアタッチ



よく見てみると

- 抽象化はされているが、使っている技術は変わらない
  - IPアドレス,ルーティングテーブル、ルーティングプロトコル(BGP)
- クラウドでも引き続きネットワークエンジニアは必要
  - 抽象化されることにより本質の理解が必要
  - 物理機器もAPI（Ansible等)でたたくように

クラウドはまったく新しい価値観の世界（再掲）

クラウドでの一般設計原則(Design Principles)

- 必要なキャパシティを勘に頼らない
- 本番規模でのシステムテストを行う
- アーキテクチャ試行の回数を増やすために自動化を取り入れる
- 発展的なアーキテクチャを受け入れる
- データ計測に基づいてアーキテクチャを決定する
- 本番で想定されるトラブルをあらかじめテストし、対策する

クラウドはまったく新しい価値観の世界（再掲）

クラウドでの一般設計原則(Design Principles)

- 必要なキャパシティを勘に頼らない
- 本番規模でのシステムテストを行う
- アーキテクチャ試行の回数を増やすために自動化を取り入れる
-
-
- る

Network環境を作って壊すのには最適では？
（わりと）テストの中で失敗を繰り返しやすいのでは？
試行してみることがやりやすい

ある外資系ハードウェア企業での検証の例

- 実際に機材をラックに積んで配線
- 設定を入れて確認

場合によっては機材がないのでサンノゼに出張して現地で検証

ある外資系ハードウェア企業での検証の例

- 実際に機材をラックに積んで配線
- 設定を入れて確認

場合によっては機材がないのでサンノゼに出張して現地で検証

これが、APIコールやマネージドコンソールですべてが解決できる
何度でも試行できる。
TerraformやAnsibleなどで自動化できる。

Agenda

- 信頼性について考えてみる
- オンプレミス／クラウド間接続を高信頼にするには
- クラウドに必要な考え方と技術
- クラウドを使った自動化と高信頼性について

クラウドを使った自動化と高信頼性について

自動化とカオスエンジニアリング
ユースケース:Netflix

Project Nimble/Chaos/Failover

- 8 Minute Failover
- Chaos Kong
- DX Failure in IAD
- Titus Running Wild

“Chaos doesn’t cause problems, it reveals them.”

aws

- 8分間（以内）のマルチリージョンフェイルオーバー
- カオスエンジニアリングによる意図的なリージョン障害 (Chaos Kong)
- Direct Connect障害の高信頼性対応

カオスエンジニアリングとは

http://principlesofchaos.org/

本番環境の分散システムが過酷な状況でも耐えれるとの確信を得るために、実験するという取り組み

From:
https://techlife.cookpad.com/entry/2018/08/02/110000

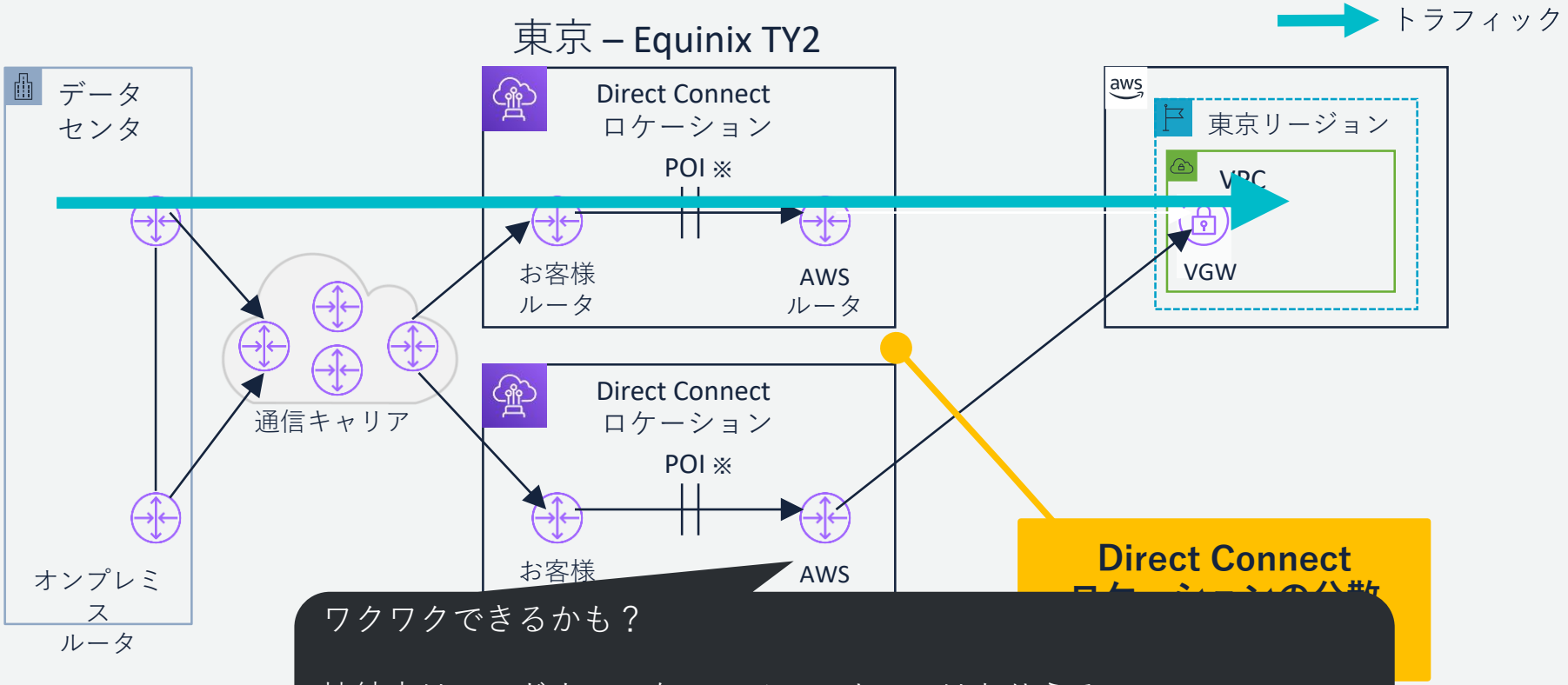
本番系で絶えず起こしているのがNetflix

日本企業でも

https://techlife.cookpad.com/entry/2018/08/02/110000

- Chaos Engineering やっていく宣言 by Cookpad

（再掲）可用性のためのデュアルDirect Connectロケーション



問いかけ

- メンテナンス通知時に張り付き対応というフローが変わらない
- メンテナンスも障害の一部、とみなせばメンテナンス張り付きという対応も変わるかも。
- 自動化の取り組みの中でなくせないか？

まとめ

今日もって帰っていただきたいこと

信頼性について考えてみる
　　Design for failure

クラウドに必要な考え方と技術
　　基本的なことは変わらない

クラウドを使った自動化と高信頼性について
　　絶えず壊し続ける手法（カオスエンジニアリング）のご紹介

今日もって帰っていただきたいこと

信頼性について考えてみる
　　Design for failure

クラウドに必要な考え方と技術
　　基本的なことは変わらない

クラウドを使った自動化と高信頼性について
　　絶えず壊し続ける手法（カオスエンジニアリング）のご紹介

考え方は変わっても本質は変わらない

Thank You!

kyukihi@amazon.co.jp

ディスカッション

会場に聞きたいこと

- こんな部分でクラウドをうまく利用してます。などの
  例があれば知りたい

- 使いたいけど使えない or あえて使ってない。などあれば
  その理由も含めて教えてほしい

- クラウドを使う際CLIベースの運用は必要、ただなかなか教育や文化作り
  をするの難しい。
  うまくやっている例などあれば知りたい

問いかけ

- メンテナンス通知時に張り付き対応というフローが変わらない
- メンテナンスも障害の一部、とみなせばメンテナンス張り付きという対応も変わるかも。
- 自動化の取り組みの中でなくせないか？

補足資料

References

今さら聞けない耐障害性の話 ～クラウドで信頼性を高めるためには～
https://www.youtube.com/watch?v=9RD7hCZlSG0
https://pages.awscloud.com/rs/112-TZM-766/images/E3-06.pdf

AWS re:Invent 2019: Designing for failure: Architecting resilient systems on AWS (ARC335-R1)
https://www.youtube.com/watch?v=BJVzwaTiOdk

AWS の日本語資料の場所「AWS 資料」で検索

https://amzn.to/JPArchive