[AWS Black Belt Online Seminar] Elastic Load Balancing (ELB)

サービスカットシリーズ

Solutions Architect 保里 善太
2019/10/29

AWS 公式 Webinar
https://amzn.to/JPWebinar

過去資料
https://amzn.to/JPArchive

自己紹介

保里 善太(ほり ぜんた)

- 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 ソリューションアーキテクト

ゲーム業界のお客様を中心にご支援中

最近の関心事：

統計や機械学習を用いた不正検知やチート検出などのセキュリティの異常検知技術

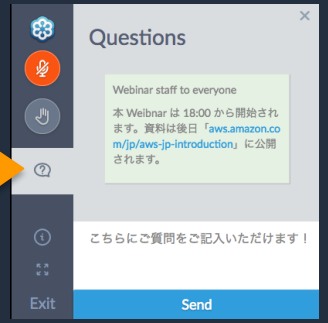
AWS Black Belt Online Seminar とは

「サービス別」「ソリューション別」「業種別」のそれぞれのテーマに分かれて、アマゾン ウェブ サービス ジャパン株式会社が主催するオンラインセミナーシリーズです。

質問を投げることができます！

- 書き込んだ質問は、主催者にしか見えません
- 今後のロードマップに関するご質問は
  お答えできませんのでご了承下さい

① 吹き出しをクリック
② 質問を入力
③ Sendをクリック



Twitter ハッシュタグは以下をご利用ください
#awsblackbelt

内容についての注意点

- 本資料では2019年10月29日時点のサービス内容および価格についてご説明しています。最新の情報はAWS公式ウェブサイト(http://aws.amazon.com)にてご確認ください。

- 資料作成には十分注意しておりますが、資料内の価格とAWS公式ウェブサイト記載の価格に相違があった場合、AWS公式ウェブサイトの価格を優先とさせていただきます。

- 価格は税抜表記となっています。日本居住者のお客様が東京リージョンを使用する場合、別途消費税をご請求させていただきます。

- AWS does not offer binding price quotes. AWS pricing is publicly available and is subject to change in accordance with the AWS Customer Agreement available at http://aws.amazon.com/agreement/. Any pricing information included in this document is provided only as an estimate of usage charges for AWS services based on certain information that you have provided. Monthly charges will be based on your actual use of AWS services, and may vary from the estimates provided.

本日のアジェンダ

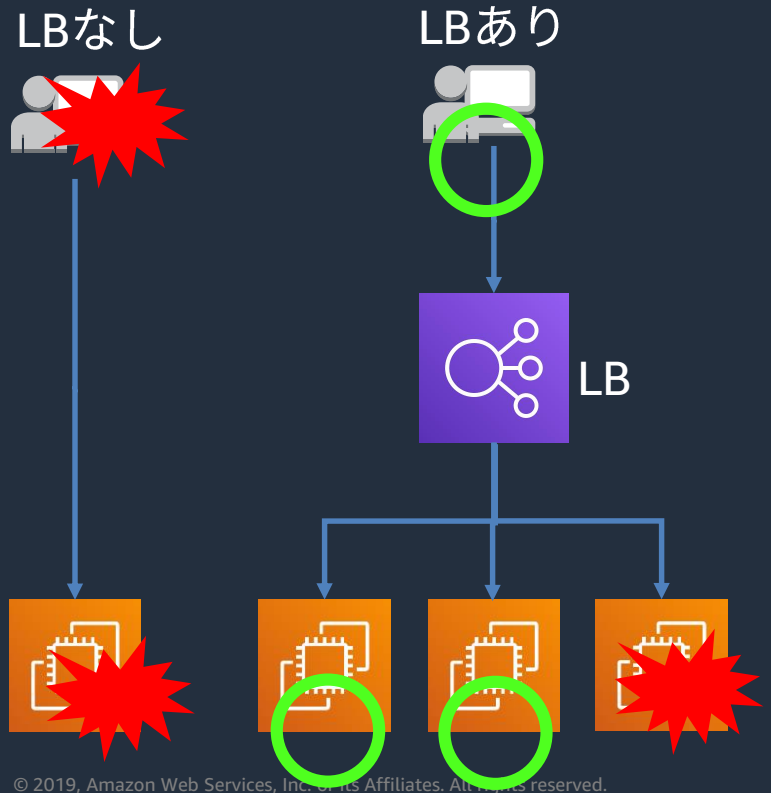
- ELBの基本
- ELBの各種機能
    - ELBの基本機能 (共通機能)
    - ALBの機能について
    - NLBの機能について
    - CLBの機能について
- ELBの応用と他サービスとの連携
- まとめ
- 補足資料

本日のアジェンダ

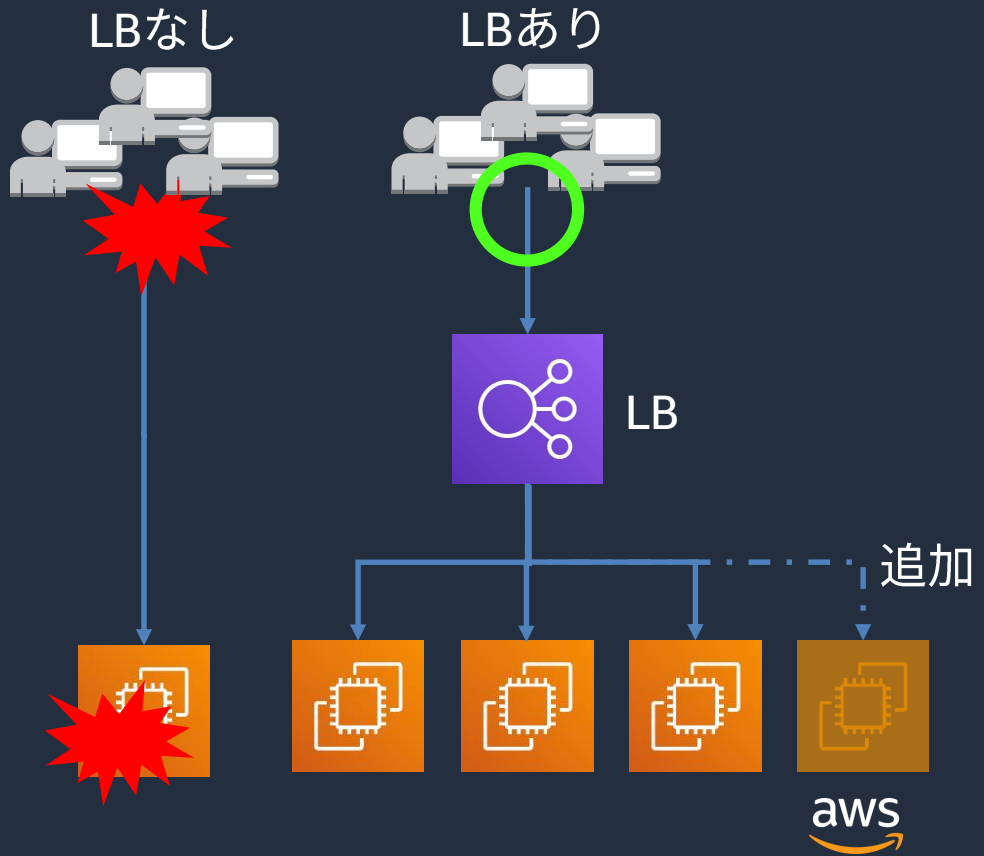
- ELBの基本
- ELBの各種機能
    - ELBの基本機能 (共通機能)
    - ALBの機能について
    - NLBの機能について
    - CLBの機能について
- ELBの応用と他サービスとの連携
- まとめ
- 補足資料

なぜロードバランサ(LB)が必要か？

1. 冗長化/可用性



2. 負荷分散/スケール



Elastic Load Balancing (ELB) とは？

～ AWSクラウド上のロードバランシングサービス ～

ELBで実現できるシステム

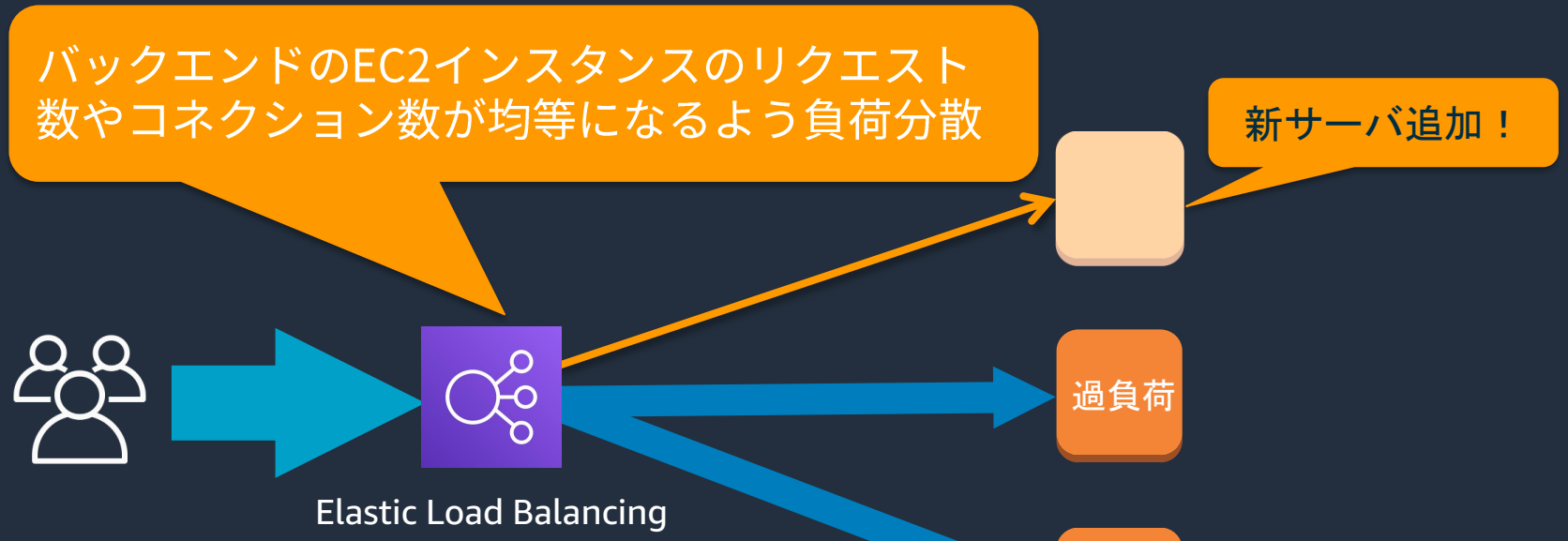
- **スケーラブル** : 複数のEC2インスタンス/ECSコンテナ..etc（ターゲット）に負荷分散
- **高い可用性**　: 複数のアベイラビリティゾーンにある複数のターゲットの中から正常なターゲットにのみ振り分け

ELB自体の特徴

- **スケーラブル**　: ELB自体も負荷に応じてキャパシティを自動増減
- **安価な従量課金** : 従量課金で利用可能
- **運用管理が楽**　: マネージドサービスなので管理が不要
- **豊富な連携機能** : Auto Scaling, Route 53, Cloud Formation… などと連携

aws

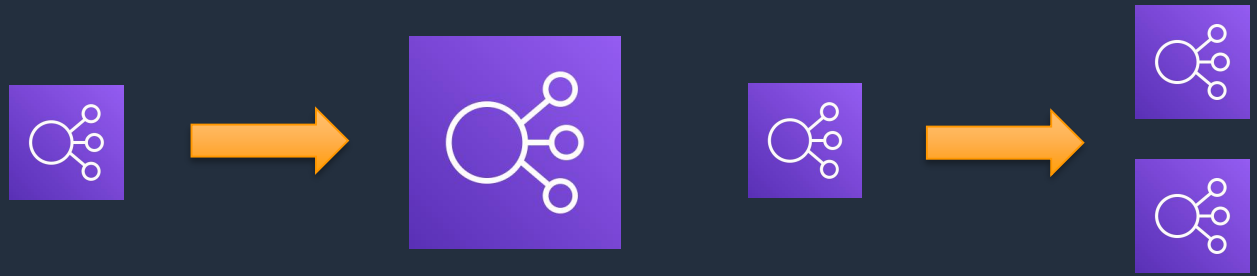
負荷分散してスケーラブルなシステムを



対応プロトコル
- L7： HTTP, HTTPS
- L4： TCP(1-65535番ポート), SSL

ELB自体もスケーラブル

ELB自体も負荷の増減に応じて自動でスケール
（キャパシティが自動で増加する）



スケールアップ　　スケールアウト

[注意]
NLB以外のELB(ALB/CLB)がスケールするときには、IPアドレスが変化します。
ELBへアクセスするときには必ずDNS名で！
DNSへ登録することで独自ドメインでのアクセスも可能。

ELBの種類

ELBの種類

Elastic Load Balancing (ELB)

以前の世代

Application Load Balancer
(ALB)

HTTP, HTTPS, HTTP/2

VPC

L7 のコンテントベースのロードバランサー

Network Load Balancer
(NLB)

TCP, UDP, TLS

VPC

L4機能を提供するロードバランサー

Classic Load Balancer
(CLB)

HTTP, HTTPS, TCP

EC2-Classic, VPC

EC2-Classic ネットワーク用ロードバランサー

aws

ELBの種類

Elastic Load Balancing (ELB)

Application Load Balancer
(ALB)

HTTP, HTTPS, HTTP/2

VPC

L7 のコンテントベースのロードバランサー

Network Load Balancer
(NLB)

TCP, UDP, TLS

VPC

L4機能を提供するロードバランサー

以前の世代

Classic Load Balancer
(CLB)

HTTP, HTTPS, TCP

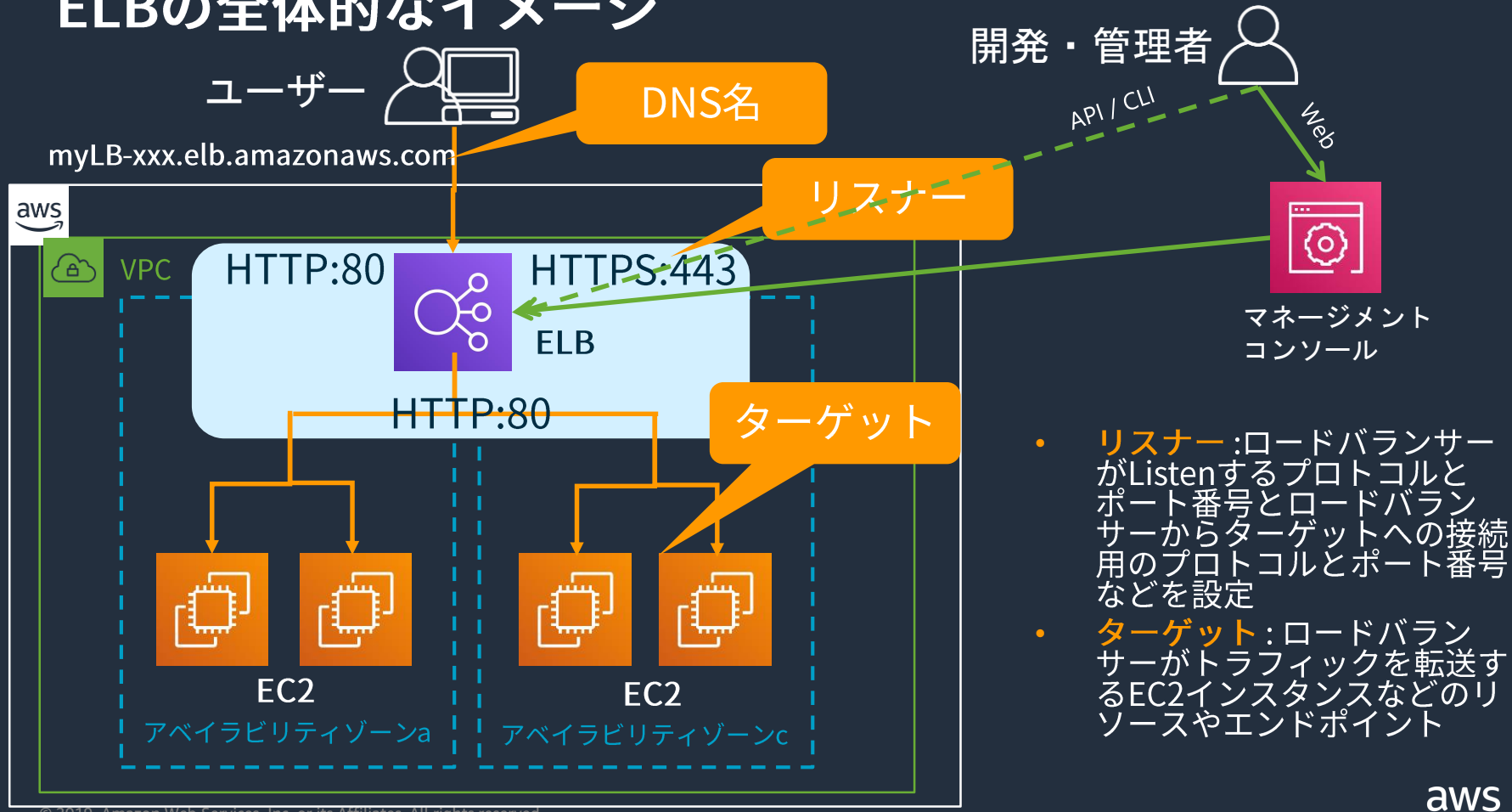
EC2-Classic, VPC

EC2-Classic ネットワーク用ロードバランサー

互換性などの理由がなければALBまたはNLBを選ぶのが現在のベスト

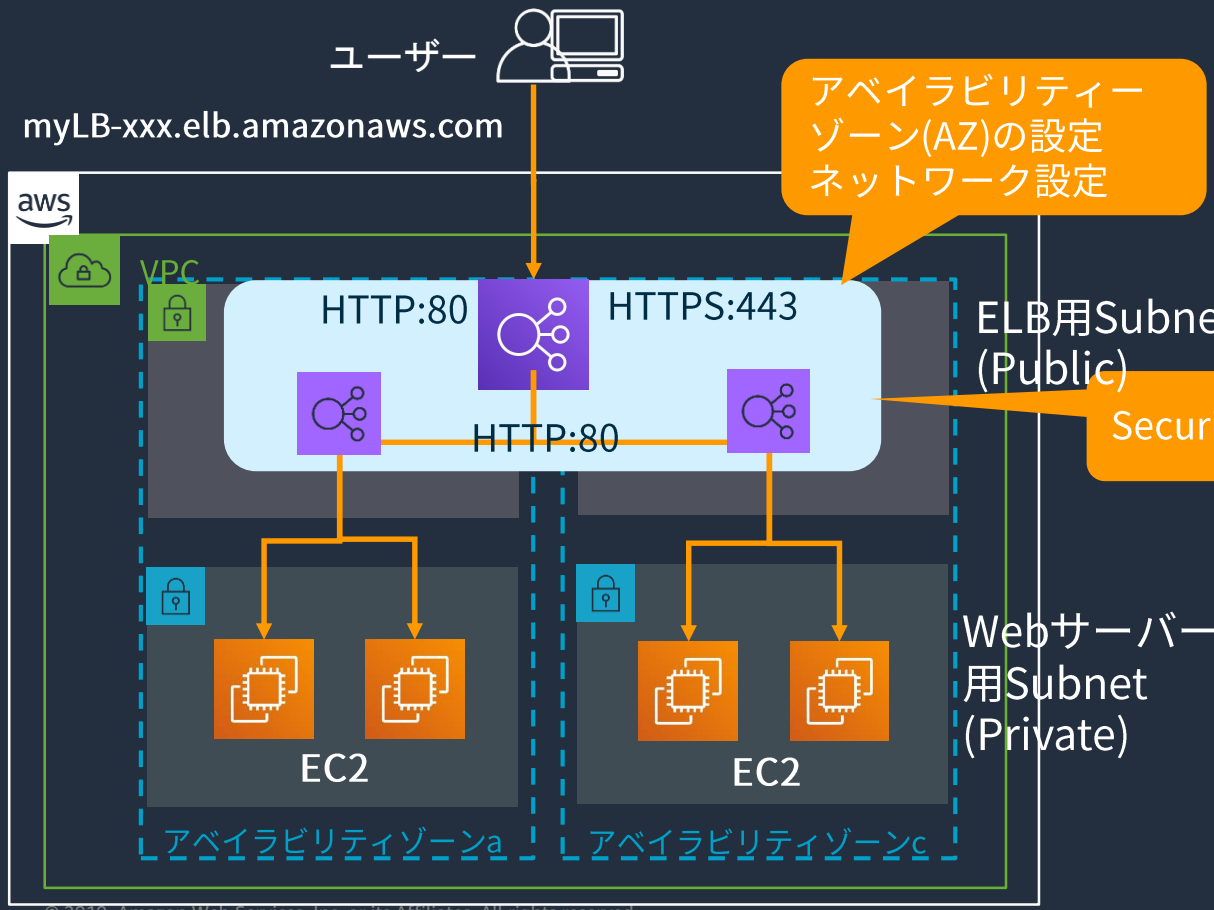
aws

ELBの全体的なイメージ



- リスナー :ロードバランサーがListenするプロトコルとポート番号とロードバランサーからターゲットへの接続用のプロトコルとポート番号などを設定
- ターゲット : ロードバランサーがトラフィックを転送するEC2インスタンスなどのリソースやエンドポイント

ELBの使い方



アベイラビリティーゾーン(AZ)の設定
ネットワーク設定

ELB用Subnet
(Public)

Security Groupの設定

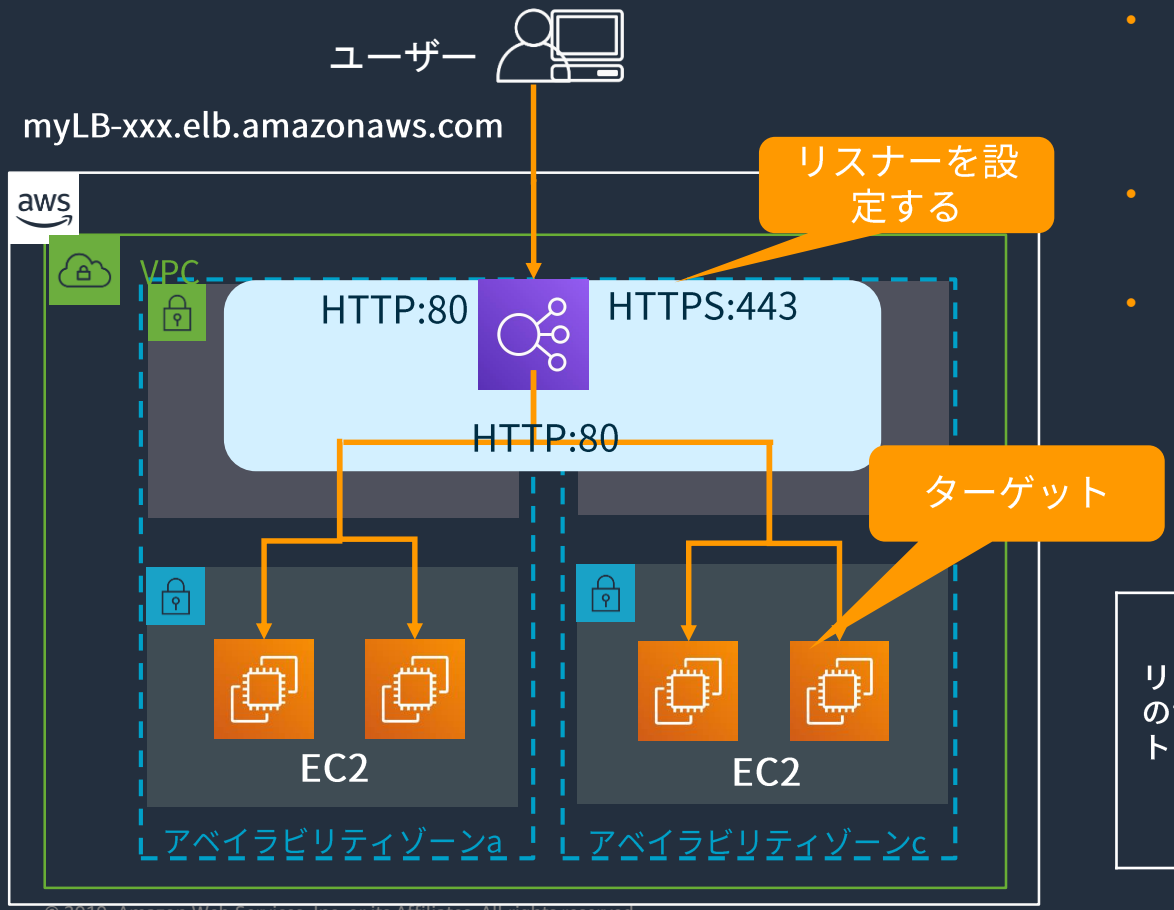
Webサーバー用Subnet
(Private)

- ELB自体をVPC内、アベイラビリティーゾーン(AZ)に設置
- AZごとに１つのサブネットを指定
- ロードバランサーの可用性を高めるには、２つ以上のAZからサブネットを指定する必要がある

- ELBに任意のSecurity Groupを指定可能
- ただしNLBはSecurity Groupと関連づけられない
- ICMP Echo Request/Replyを許可 すれば、ELBがpingにも応答
- バックエンドのEC2インスタンスは ELBからのみリクエストを受け付ける設定を推奨

https://aws.amazon.com/jp/premiumsupport/knowledge-center/security-group-load-balancer/

ELBの使い方

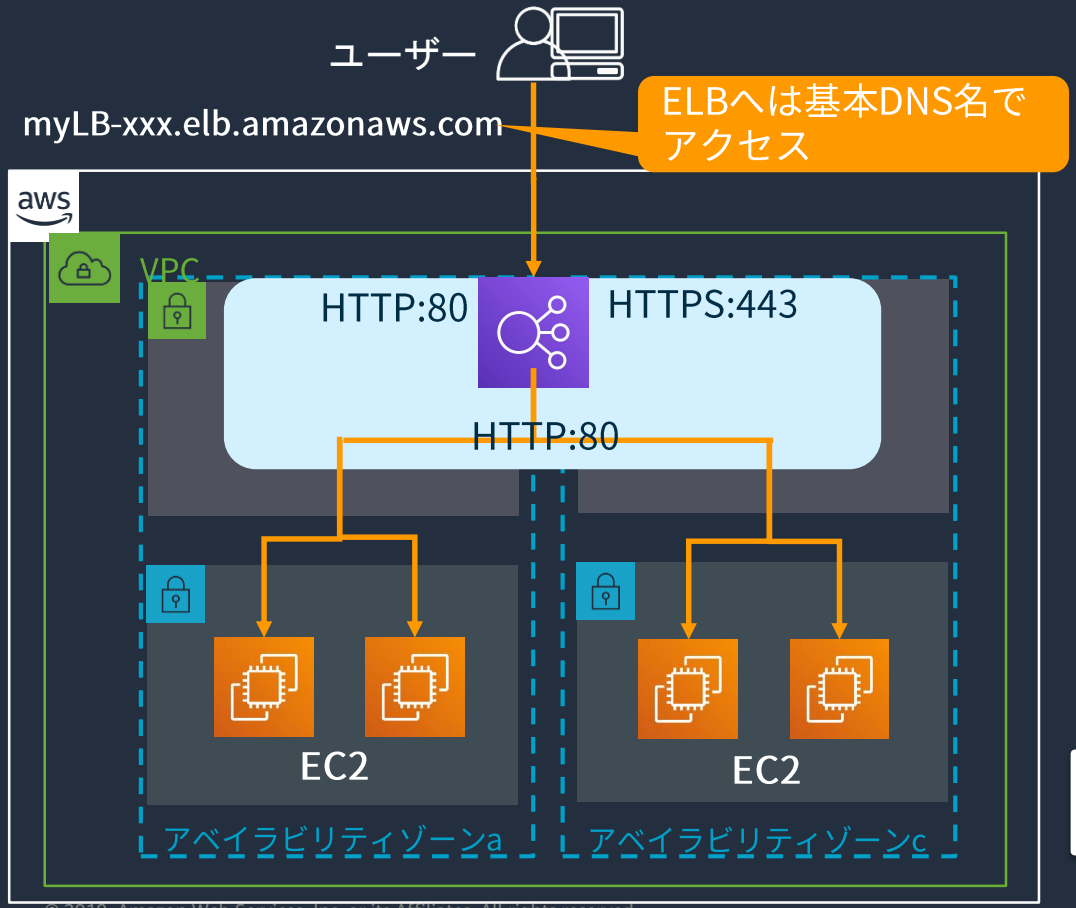


- リスナー :ロードバランサーがListenするプロトコルとポート番号(1 ～ 65535)とロードバランサーからターゲットへの接続用のプロトコルとポート番号などを設定
- ターゲット : ロードバランサーがトラフィックを転送するEC2インスタンスなどのリソースやエンドポイント
- ターゲットの種類
  - インスタンス ID でターゲットを指定
  - IPアドレスでターゲットを指定 (CLB以外)
  - Lambda関数をターゲットに指定 (ALBのみ)

	Application Load Balancer	Network Load Balancer	Classic Load Balancer
リスナーのサポートプロトコル	HTTP, HTTPS	TCP, TLS, UDP, TCP_UDP	HTTP, HTTPS, TCP, SSL

https://docs.aws.amazon.com/ja_jp/elasticloadbalancing/latest/classic/elb-listener-config.html

ELBの使い方



独自ドメイン(カスタムドメイン)を利用する場合

- Route 53を使用する場合は、Route 53エイリアスレコードで登録
  - CNAMEでの登録も可能
- Route 53以外のDNSを使用する場合はCNAMEで登録
- Zone Apex (www.exapmple.com ではなく example.com を指定) の場合
  - 通常のDNSサーバではCNAME設定不可
  - Route 53のエイリアスレコードを 使うことで設定可能

www.example.com CNAME myLB-xxxx.ap-northeast-1.elb.amazonaws.com

https://docs.aws.amazon.com/ja_jp/elasticloadbalancing/latest/classic/using-domain-names-with-elb.html

aws

本日のアジェンダ

- ELBの基本
- **ELBの各種機能**
    - **ELBの基本機能 (共通機能)**
    - ALBの機能について
    - NLBの機能について
    - CLBの機能について
- ELBの応用と他サービスとの連携
- まとめ
- 補足資料

ELBの基本機能 (ALB/NLB/CLB共通機能)

- 高可用性と負荷分散
  - ゾーンごとのフェイルオーバー
  - クロスゾーン負荷分散
  - 同一のインスタンスで複数ポートに負荷分散
  - IP アドレスをターゲットに設定
- モニタリング・ログ
  - ヘルスチェック
  - 運用のモニタリング
  - アクセスログの記録
- セキュリティ関連
  - SSL/TLS サポート
  - Server Name Indication (SNI)
  - バックエンドサーバーの暗号化
- コネクション
  - Connection Draining (登録解除の遅延)
  - スティッキーセッション
  - WebSocket 対応

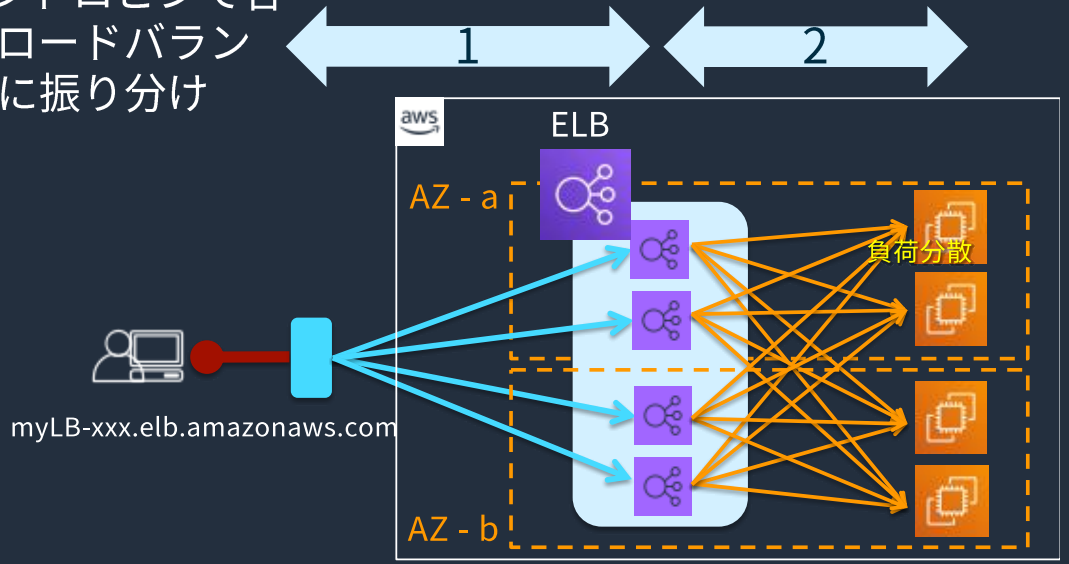
高可用性と負荷分散

複数アベイラビリティゾーン(AZ)に分散

ALB NLB CLB

1) DNSラウンドロビンで各AZ内のELB(ロードバランサーノード)に振り分け

2段階での負荷分散



2) 負荷が均等になるようにバックエンドのEC2にそれぞれのルーティングアルゴリズムで振り分け

	Application Load Balancer	Network Load Balancer	Classic Load Balancer
バックエンドへのルーティングアルゴリズム	ラウンドロビンルーティング	フローハッシュアルゴリズムによるルーティング	TCP : ラウンドロビンルーティング HTTP/HTTPS : The least outstanding requests routing algorithm

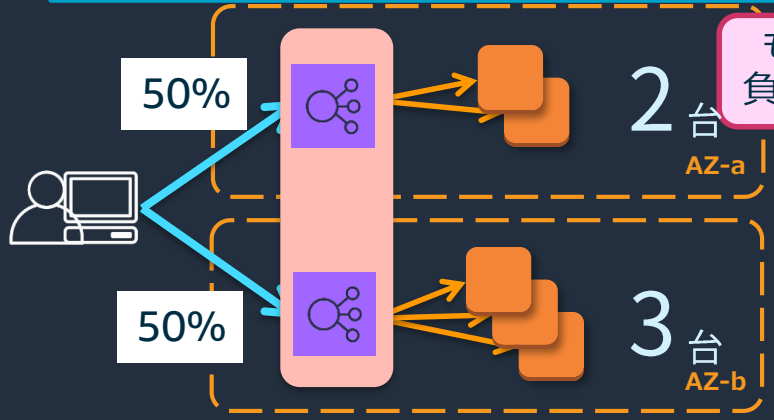
https://docs.aws.amazon.com/ja_jp/elasticloadbalancing/latest/userguide/how-elastic-load-balancing-works.html#request-routing

aws

AZとバックエンドキャパシティの関係

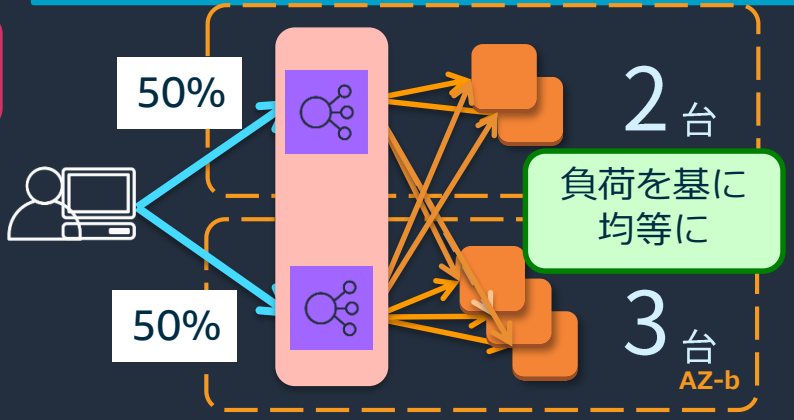
ALB NLB CLB

良くない例：AZ間でキャパシティが不均等



もう一方より
負荷が高くなる

クロスゾーン負荷分散が有効であれば



負荷を基に
均等に

クロスゾーン負荷分散	Application Load Balancer	Network Load Balancer	Classic Load Balancer
	デフォルトで有効	デフォルトで無効	デフォルトで有効

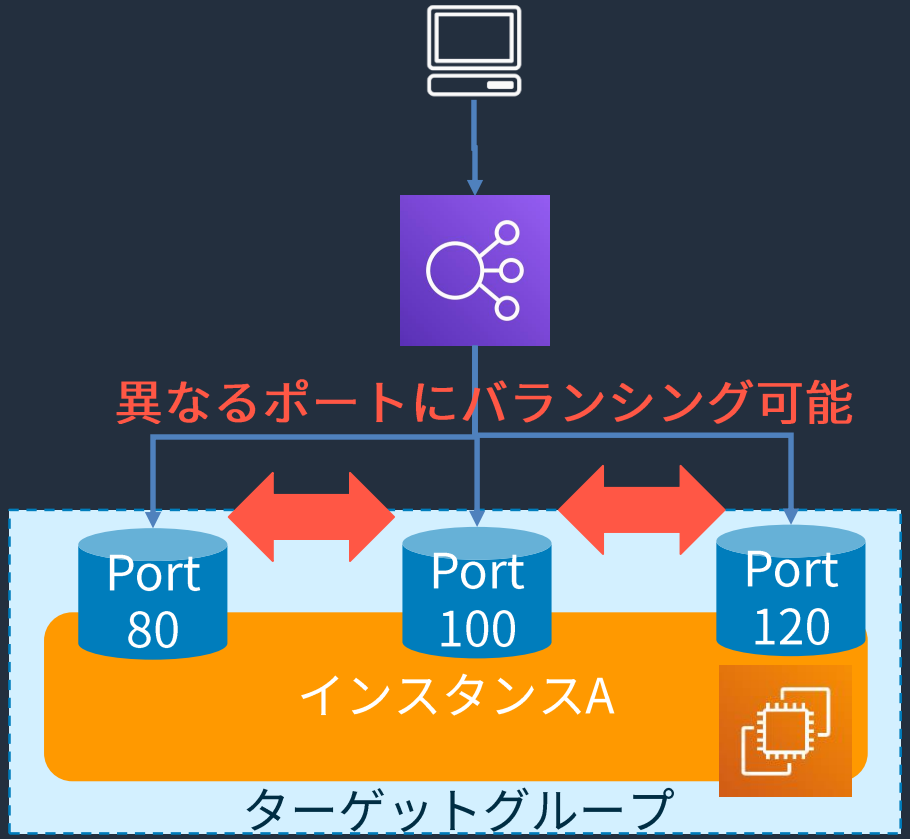
- リージョン内の複数AZに負荷分散可能
  - 複数リージョンへの分散にはRoute 53を併用できる
- AZごとのEC2インスタンス数が異なってもクロスゾーン負荷分散により全ての負荷を均等にできる
- 極力AZごとのEC2インスタンス数は均等にし、各EC2インスタンスのタイプは同じにする

参照 http://docs.aws.amazon.com/ja_jp/ElasticLoadBalancing/latest/DeveloperGuide/enable-disable-crosszone-lb.html

aws

同一のインスタンスで複数ポートに負荷分散可能

ALB NLB

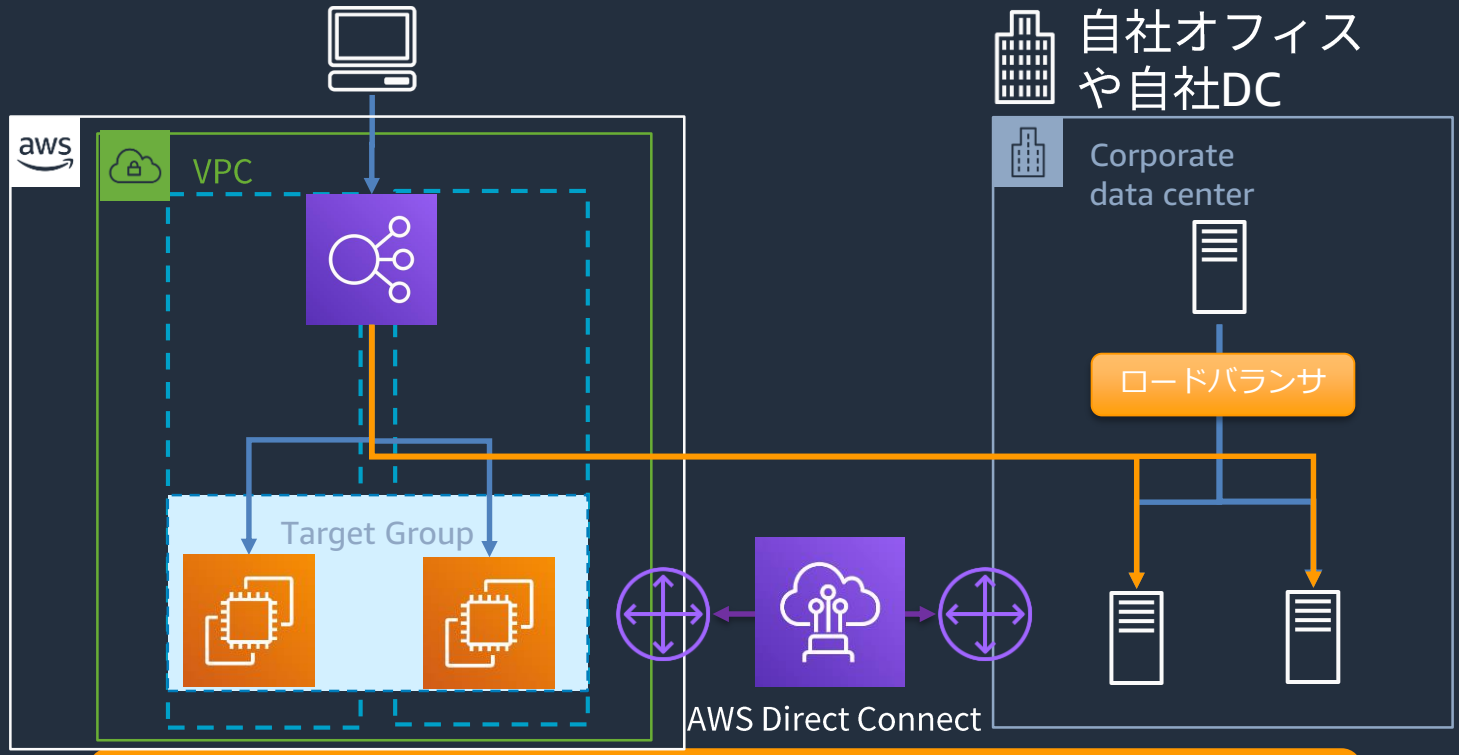


- EC2インスタンスをターゲットグループに割り当てる際に複数ポートを個別のターゲットとして登録することが可能
- 1つのインスタンスに対して、複数ポートで負荷分散が可能
- コンテナを用いる際には、同一インスタンス上で複数のコンテナ(サーバー)を運用する必要があるので有用
- ECSを利用の場合には動的ポートマッピング機能を利用できる(後述)

aws

IP アドレスをターゲットに設定 (IP target)

ALB NLB



ALB / NLBではインスタンス名以外に
オンプレミスのサーバなどをIP targetとして指定できる

aws

ELB自体のスケーリング

ELBは負荷に応じて自動でスケールする

- 但し、ALB/CLBにおいては接続・リクエストが瞬間的に急増したために、ELBのスケーリングが間に合わない場合、HTTP 503を返す
  - 新サービス開始
  - TVやメディアによるサービス紹介
  - 負荷テスト　　等
- 回避方法は事前にALB/CLBをスケールさせておく
  - Pre-Warming（暖気運転）の申請をサポートケースにて行う
    ※Business/Enterpriseサポート要
  - 自前で負荷を段階的にかけてスケールさせておく
- NLBは暖気不要で突発的な数百万リクエスト/秒のトラフィックも捌ける

aws

ELBのモニタリング・ログ

ヘルスチェック

ALB NLB CLB

- ELBは正常なターゲットにのみトラフィックをルーティングする
- ELBは設定値に基づき、ターゲットに対してヘルスチェックを定期的に行い、正常なターゲットかを判定する
- 正常判定が厳しすぎるとインスタンスが 使えるまでに時間がかかり、逆に異常との判定が厳しすぎても、 過負荷時に処理できるインスタンスを減らしてしまうことにもなる

設定	説明	設定例
プロトコル (HealthCheckProtocol)	ターゲットでヘルスチェックを実行するときにロードバランサーが使用するプロトコル (ELBの種類によってHTTP/HTTPS, TCPなど)	HTTP (Status 200 が返るのを確認)
ポート (HealthCheckPort)	ターゲットでヘルスチェックを実行するときにロードバランサーが使用するポート デフォルトは、各ターゲットがロードバランサーからトラフィックを受信するポート	トラフィックポート
パス (HealthCheckPath)	ヘルスチェックのターゲットの送信先である ping パス デフォルトは /	/index.html
タイムアウト時間	ヘルスチェックを失敗と見なす、ターゲットからレスポンスがない時間 (秒単位) 範囲は 2〜120 秒	5秒
正常のしきい値 (HealthyThresholdCount)	異常なターゲットが正常であると見なされるまでに必要なヘルスチェックの連続成功回数	5
非正常のしきい値 (UnhealthyThresholdCount)	ターゲットが異常であると見なされるまでに必要なヘルスチェックの連続失敗回数	2
間隔	個々のターゲットのヘルスチェックの概算間隔 (秒単位)	30秒
成功コード (Matcher)	ターゲットからの正常なレスポンスを確認するために使用する HTTP コード	200

運用のモニタリング

ALB NLB CLB

- CloudWatchによりELBのメトリクスを60 秒間隔で監視可能
- 統計情報として平均(average)、合計(sum)、最大値(max)、最小値(min)等を表示できるが各メトリクスによってどの統計で監視するのが適切かは異なるので注意

<メトリクスの例>

設定値	説明
HealthyHostCount	正常なバックエンドのホスト数
UnHealthyHostCount	異常なバックエンドのホスト数
RequestCount	リクエスト数
Latency	遅延時間
HTTPCode_ELB_4XX HTTPCode_ELB_5XX	ELBが返した4xx, 5xxのレスポンス数
HTTPCode_Backend_2XX、 HTTPCode_Backend_3XX、 HTTPCode_Backend_4XX、 HTTPCode_Backend_5XX	バックエンドが返した2xx,3xx,4xx,5xxレスポンス数
BackendConnectionErrors	バックエンドへの接続エラー回数
SurgeQueueLength	バックエンドへの送信保留中の件数
SpilloverCount	キュー溢れのため拒否した件数

UnHealthyHostCount のモニタリングには**Average** および **Minimum** の統計情報を利用するのが有用

参照 http://docs.aws.amazon.com/ja_jp/ElasticLoadBalancing/latest/DeveloperGuide/US_MonitoringLoadBalancerWithCW.html

aws

アクセスログの記録

ALB NLB CLB

- 最短5分間隔でELBのアクセスログを取得可能
- 指定したS3バケットに簡単にログを自動保管
- ELBの種類によってアクセスログの出力フィールドも異なる

例) ALBのHTTPSリスナーのログ

```
https 2018-07-02T22:23:00.186641Z app/my-loadbalancer/50dc6c495c0c9188 192.168.131.39:2817 10.0.0.1:80 0.086
0.048 0.037 200 200 0 57 "GET https://www.example.com:443/ HTTP/1.1" "curl/7.46.0" ECDHE-RSA-AES128-GCM-
SHA256 TLSv1.2 arn:aws:elasticloadbalancing:us-east-2:123456789012:targetgroup/my-targets/73e2d6bc24d8a067
"Root=1-58337281-1d84f3d73c47ec4e58577259" "www.example.com" "arn:aws:acm:us-east-
2:123456789012:certificate/12345678-1234-1234-1234-123456789012" 1 2018-07-02T22:22:48.364000Z
"authenticate,forward" "-" "-"
```

[リクエストタイプ] [timestamp] [elbのリソースID] [クライアントのIPとポート番号] [ターゲットのIPとポート番号] [ELBがリクエストを受け取った時点からターゲットに送信するまでの合計経過時間] [ELBがターゲットにリクエストを送信した時点から、そのターゲットが応答ヘッダーの送信を開始した時点までの合計経過時間] [ELBがターゲットから応答ヘッダーを受け取った時点から、クライアントへの応答の送信を開始した時点までの合計経過時間] .....

参照 https://docs.aws.amazon.com/ja_jp/elasticloadbalancing/latest/application/load-balancer-access-logs.html
https://docs.aws.amazon.com/ja_jp/elasticloadbalancing/latest/network/load-balancer-access-logs.html
http://docs.aws.amazon.com/ja_jp/ElasticLoadBalancing/latest/DeveloperGuide/access-log-collection.html

aws

コネクションについて

ELBのコネクションタイムアウト

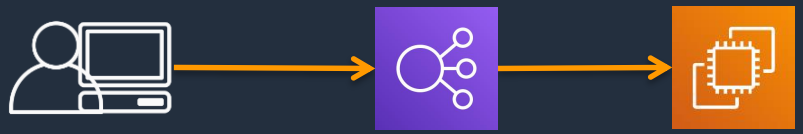
ALB NLB CLB

無通信状態が続くとそのコネクションを自動で切断する

- ALB/CLBのデフォルトではコネクションタイムアウト値は60秒
- NLBのデフォルトのコネクションタイムアウト値は350秒 (固定)

ALB/CLBのコネクションタイムアウト値は変更可能

- 1～4,000秒の間で自由に設定可能
- NLBは350 秒で固定
- 設定方法は※参照



参照
https://docs.aws.amazon.com/ja_jp/elasticloadbalancing/latest/application/application-load-balancers.html#connection-idle-timeout

aws

Connection Draining (登録解除の遅延)

ALB NLB CLB

バックエンドのEC2インスタンスをELBから登録解除したり、ヘルスチェックが失敗した時に、新規リクエストの割り振りは中止して、処理中のリクエストは終わるまで一定期間待つ

- 全てのELBでデフォルトで有効、タイムアウト 300秒
- タイムアウト最大 3600秒
- Connection Draining 動作中はターゲットのヘルスステータスに「draining」と表示される
  - CLBではInService: Instance deregistration currently in progressと表示される

https://docs.aws.amazon.com/ja_jp/elasticloadbalancing/latest/classic/config-conn-drain.html

aws

スティッキー セッション (stickiness)

ALB CLB

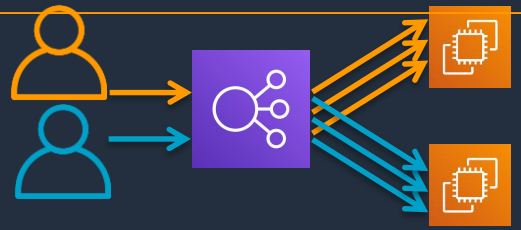
同じユーザから来たリクエストを全て同じEC2インスタンスに送信

- アプリケーションでのセッション情報、一時ファイルなどをEC2インスタンスが保持する構成の場合に必要
- デフォルトで無効、利用するためには有効にする
- HTTP/HTTPSでのみ利用可能

スティッキーセッションの有効期限について

- ロードバランサーによって生成されるクッキーを使用したスティキーセッション (ALB/CLB)
  - セッション開始からの有効期間を指定してELBで制御
  - 無期限にする事も可（無期限でもブラウザを閉じれば終了）
- アプリケーション生成のクッキーを使用したスティッキーセッション (CLBのみサポート)
  - アプリケーションが作成したCookieにあわせる
  - アプリケーションが作成するCookie名を指定

EC2インスタンスの増減を柔軟にできるように、セッション情報などは別のDBサーバやキャッシュサーバに持たせるのが望ましい。この場合スティッキー セッションは不要。



aws

セキュリティ関連

SSL / TLS Termination

ALB NLB CLB

ELB側でSSL / TLS 認証 ができる (基本的にはTLSプロトコルを使用)
以下の通信パターンが考えられる

a. ELBでSSL Terminationし、バックエンドとはSSLなし
   - バックエンドのEC2インスタンスでSSL処理せずに済むため負荷をオフロードできる
b. ELBでSSL Terminationし、バックエンドとは別途SSL
c. SSLをバイパスしてバックエンドにTCPで送信
   - クライアント証明書認証などを利用するためにはTCPとして扱う

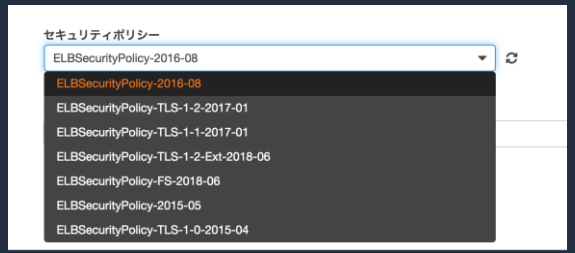
a)	HTTPS	HTTP
b)	HTTPS / SSL	HTTPS / SSL
c)	TCP	TCP

参照 http://docs.aws.amazon.com/ja_jp/ElasticLoadBalancing/latest/DeveloperGuide/elb-listener-config.html#using-elb-listenerconfig-quickref

事前定義されたセキュリティポリシー

ALB NLB CLB

- SSL/TLS利用時には事前定義されたセキュリティポリシーを利用する
- 事前定義されたセキュリティポリシーはSSL/TLSプロトコルの設定+暗号スイート(Cypher)の設定が定義されている
- TLS v1.0, v1.1, v1.2をサポート
- CLBのみセキュリティポリシーのカスタマイズができる(カスタムセキュリティポリシー)
- Perfect Forward Secrecy (PFS) のサポート
- Server Order Preference
- 新しく作ったELBではELBSecurityPolicy-2016-08 がデフォルト
  - 既存のELBには互換性確認の上 ELBSecurityPolicy-2016-08 の適用を



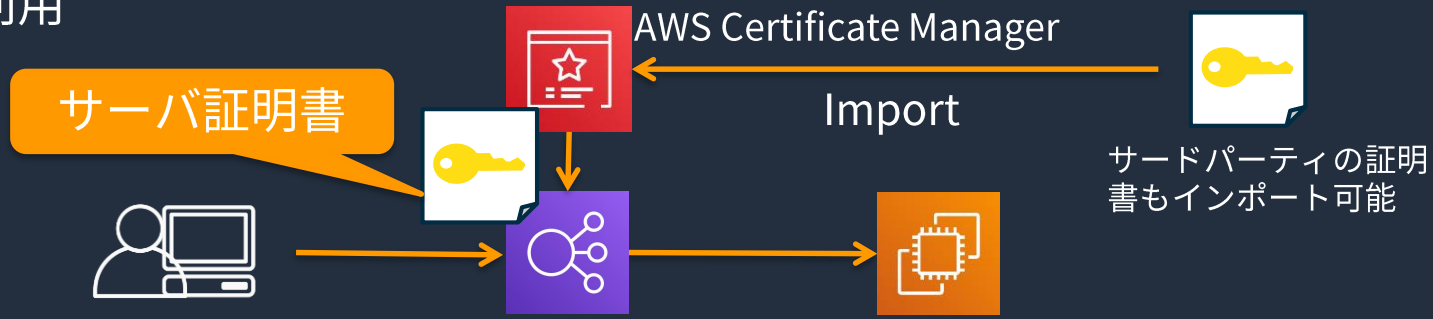
参照
https://docs.aws.amazon.com/ja_jp/ElasticLoadBalancing/latest/DeveloperGuide/ssl-config-update.html

HTTPS/SSL利用時のTLSサーバ証明書

ALB NLB CLB

AWS Certificate Manager(ACM)を使用すれば証明書のリクエスト、管理、更新、プロビジョニングが容易に実行可能

- 無料で証明書を利用可能（ACMと統合されているELB, Amazon CloudFront , Amazon API Gatewayに対してのみ）
- ELB に対する証明書の設定を数クリックで完了
- 証明書は自動更新されるので、失効の心配がない
- ACMで発行される証明書はドメイン認証タイプ(DV)の証明書なのでより上位の証明書(OV, EV)を利用する場合はサードパーティの証明書を取得してインポートして利用



参照
http://docs.aws.amazon.com/ja_jp/ElasticLoadBalancing/latest/DeveloperGuide/ssl-server-cert.html

SNIでの複数TLS証明書のスマートセレクション

ALB NLB

- 複数のTLS証明書を1つのALB/NLBのListenerに設定可能に
  - SNIをサポートするクライアントには、適切な証明書を選択してTLSで通信をできる
    - SNI非サポートのクライアントにはデフォルト証明書が使われる
  - ドメインはもちろんサポートする鍵交換方式や暗号、署名アルゴリズムを元に証明書を選択するスマートセレクション

- ALB毎に最大25証明書まで (デフォルト証明書を除く)
  - ACMまたはIAMの全ての証明書が利用可能

https://aws.amazon.com/jp/blogs/news/new-application-load-balancer-sni/
https://aws.amazon.com/about-aws/whats-new/2017/10/elastic-load-balancing-application-load-balancers-now-support-multiple-ssl-certificates-and-smart-certificate-selection-using-server-name-indication-sni/

aws

本日のアジェンダ

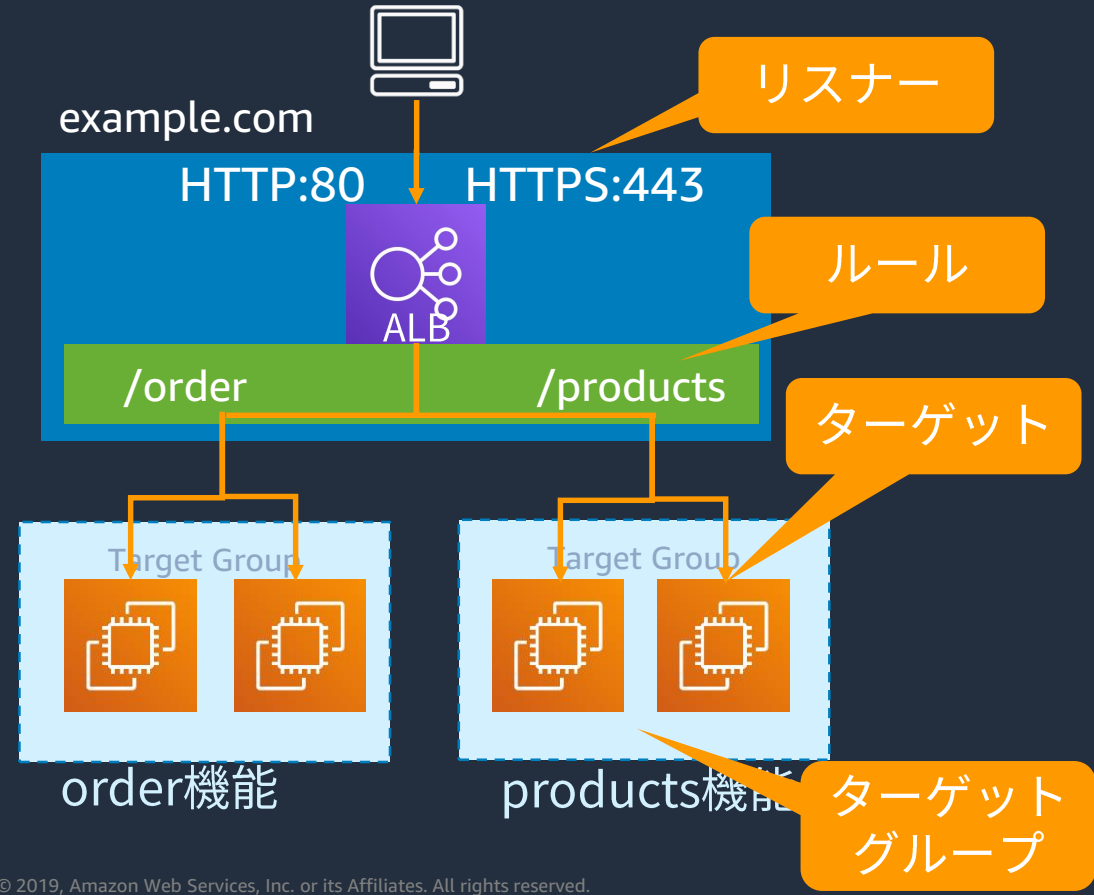
- ELBの基本
- **ELBの各種機能**
    - ELBの基本機能 (共通機能)
    - **ALBの機能について**
    - NLBの機能について
    - CLBの機能について
- ELBの応用と他サービスとの連携
- まとめ
- 補足資料

ALBの特徴と固有の機能

- L7ロードバランサー
- HTTP/HTTPSのみ対応 (TCPには対応していない)
- コンテンツベースのルーティング (高度なリクエストルーティング)
- ユーザー認証機能
- 暖気申請は必要
- その他
  - ネイティブ HTTP/2 対応
  - ターゲットとしての Lambda 関数
  - クライアントのIPアドレス取得について (ALB / NLBの違い)
    Client IP Preservation
  - AWS WAFとの連携について
  - Websocketに対応

aws

ALBのコンポーネント



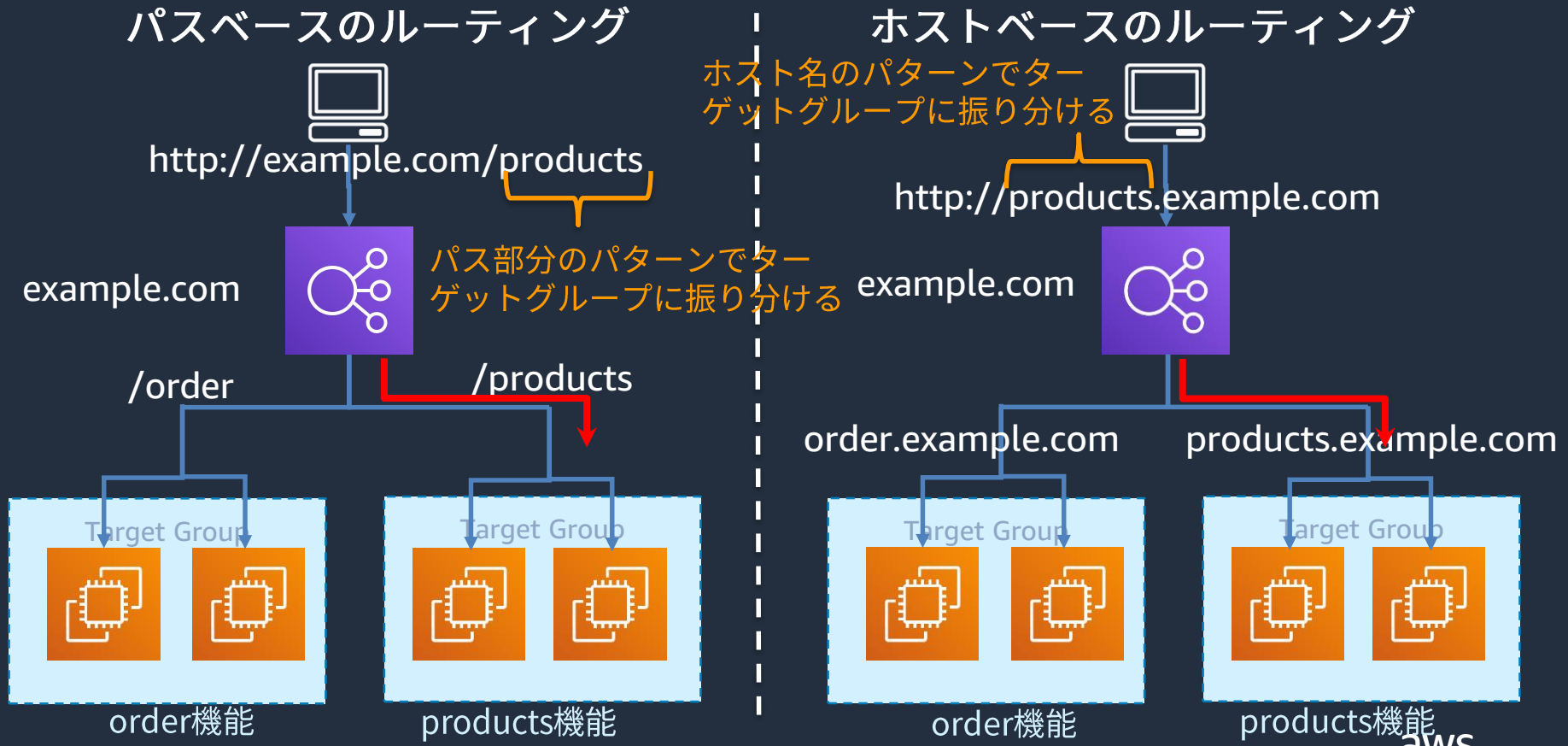
- リスナー：ロードバランサでListenするポートとプロトコルとユーザーが定義したルールを含む設定
- ルール：リクエストがどのように転送されるかを条件とアクションで定義
- ターゲット：ロードバランサーがトラフィックを転送するEC2などのリソースやエンドポイント
- ターゲットグループ：EC2インスタンスなどのターゲットの集合。ターゲットグループ内でリクエストは負荷分散される。インスタンス側の設定として、インスタンスで公開するポート、プロトコル、設定を含む

L7ロードバランサー

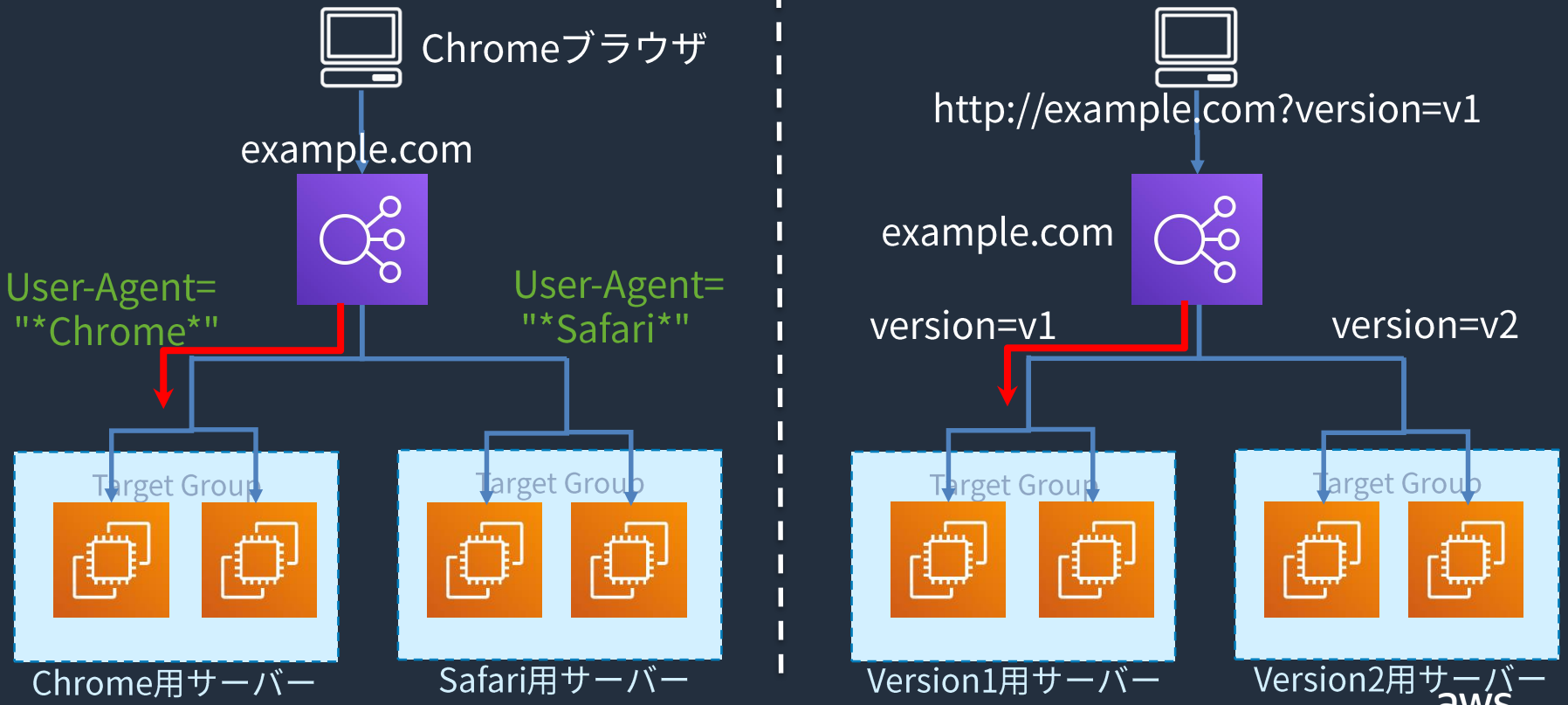
コンテンツベースのルーティング (高度なリクエストルーティング)	
パスベースのルーティング	リクエスト URL のパスパターンに基づいたルーティング
ホストベースのルーティング	HTTP ヘッダーの Host フィールドに基づいてクライアントのリクエストをルーティング 同一のロードバランサーから複数のドメインへのルーティングが可能
HTTP ヘッダーベースのルーティング	各リクエストの HTTP ヘッダーに基づいたルーティング
HTTP メソッドベースのルーティング	標準またはカスタムの HTTP メソッドに基づいてクライアントのリクエストをルーティング
クエリ文字列パラメータベースのルーティング	キーと値のペアまたはクエリストリングの値に基づいたルーティング
送信元 IPアドレスCIDRベースのルーティング	リクエスト元のソース IP アドレス CIDR に基づいてクライアントのリクエストをルーティング

aws

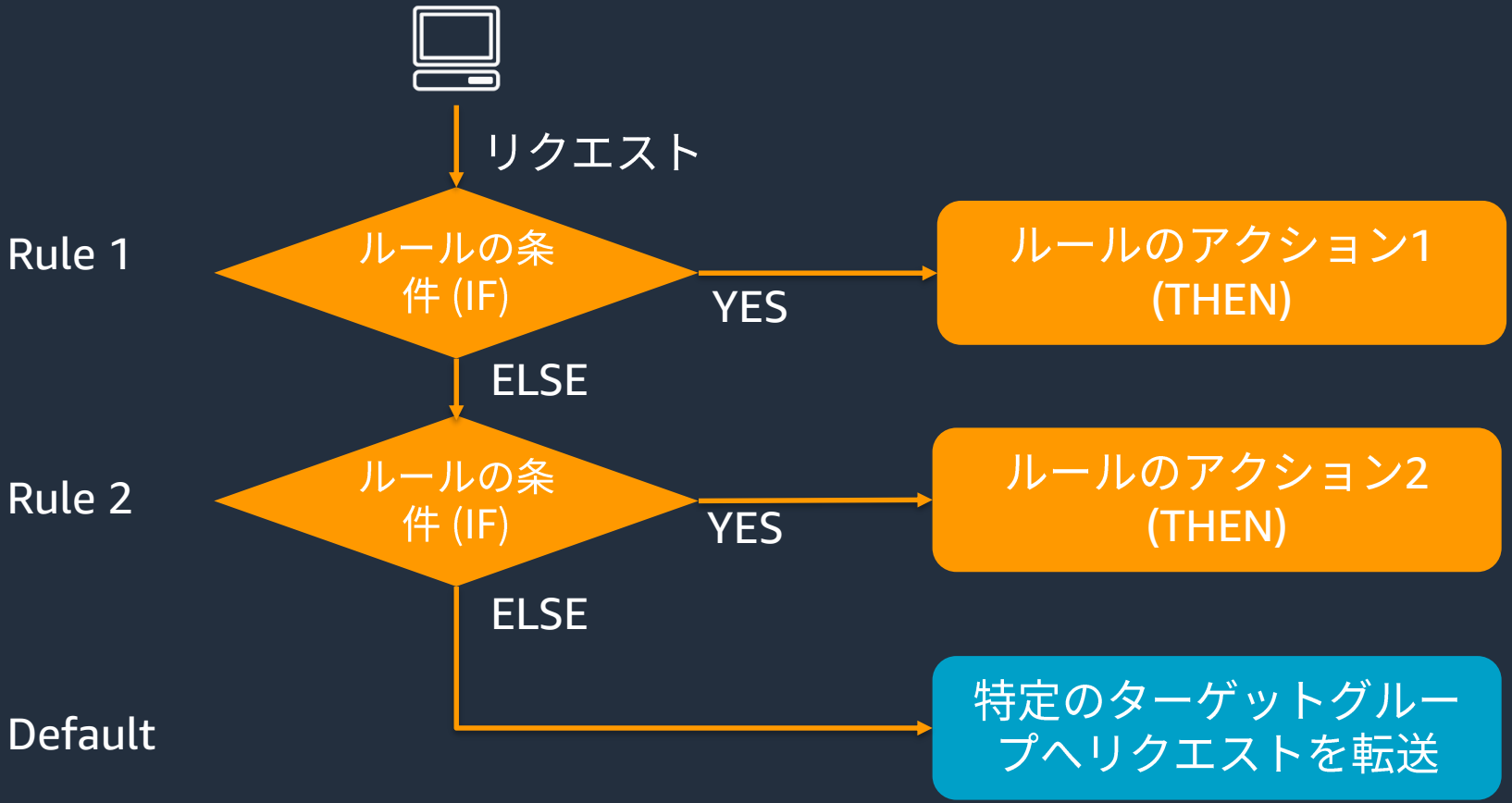
ALBのコンテンツベースのルーティング



ALBのコンテンツベースのルーティング



ALBのコンテンツベースのルーティング



ALBのコンテンツベースのルーティング

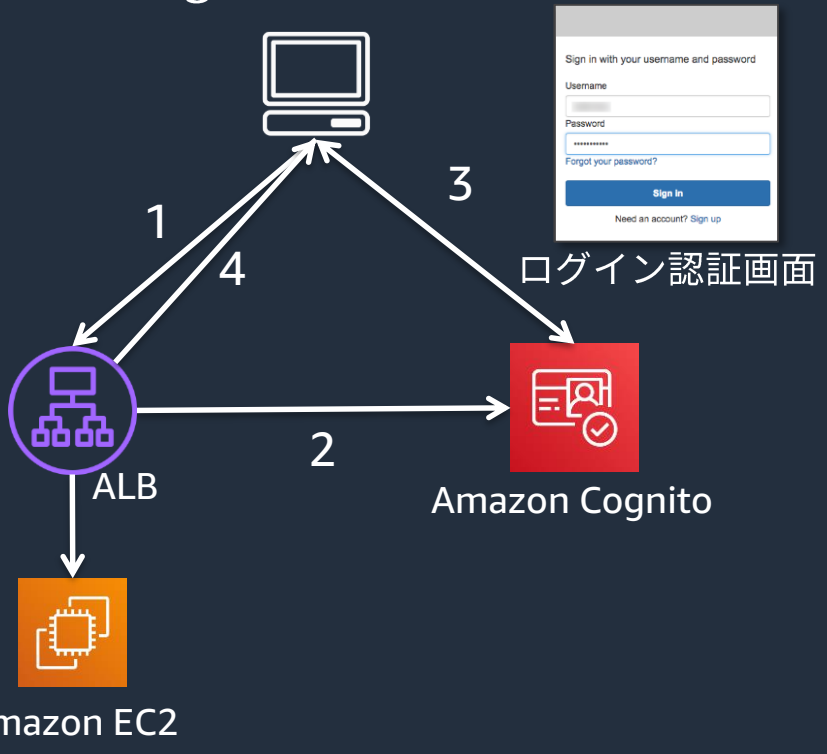
ルールの条件 (IF)	
path-pattern (パスベースのルーティング)	リクエスト URL のパスパターンに基づいたルーティング
host-header (ホストベースのルーティング)	各リクエストのホスト名に基づいたルーティング
http-header (HTTP ヘッダーベースのルーティング)	標準またはカスタムの HTTP メソッドに基づいてクライアントのリクエストをルーティング
http-request-method(HTTP メソッドベースのルーティング)	各リクエストの HTTP リクエストメソッドに基づいたルーティング
query-string (クエリ文字列パラメータベースのルーティング)	キーと値のペアまたはクエリストリングの値に基づいたルーティング
source-ip (送信元 IPアドレスベースのルーティング)	リクエスト元のソース IP アドレス CIDR に基づいてクライアントのリクエストをルーティング

ルールのアクション (THEN)	
転送アクション(forward)	指定されたターゲットグループにリクエストを転送
リダイレクト (redirect)	各リクエストのホスト名に基づいたルーティング
固定レスポンスアクション (fixed-response)	クライアントリクエストを破棄し、2XX、4XX、5XX などのカスタムの HTTP レスポンスを返す
Cognito認証アクション (authenticate-cognito)	[HTTPS リスナーのみ] Amazon Cognito を使用してユーザー認証を実施
OIDC認証アクション (authenticate-oidc)	[HTTPS リスナーのみ] OpenID Connect (OIDC) に準拠する ID プロバイダーを使用してユーザー認証を実施

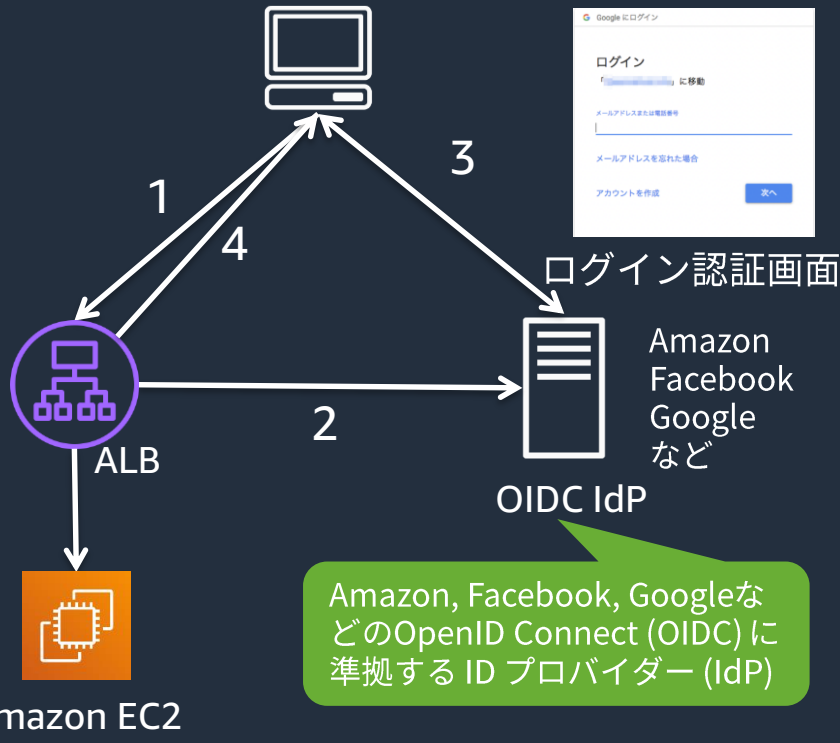
aws

ALBのユーザー認証機能

Cognitoによる認証



OIDC IdPによる認証



最初にALBにアクセスするとCognitoやIdPの認証画面にリダイレクトされる

aws

ALB(CLB)利用時のクライアントのIPアドレス取得

- HTTP/HTTPS リスナーを使用する Application Load Balancer と Classic Load Balancer の場合、クライアントの IP アドレスをキャプチャするには X-Forwarded-For ヘッダーを使用する必要あり
  - L7ロードバランサーのため、TCP等の処理をALBで終端しているため
  - ターゲット側のアクセスログにはALB/CLBのIPアドレスが記録される

- Network Load Balancer の場合は、何も設定せずに透過的にクライアント IP アドレスを取得可能 (インスタンス ID を使用したターゲット登録の場合)

参照 https://aws.amazon.com/jp/premiumsupport/knowledge-center/elb-capture-client-ip-addresses/

そのほかの機能

- ネイティブ HTTP/2 対応
  - HTTP 1.1プロトコルから多数の改善が行われ、1つのHTTP/2コネクションで最大 128 のリクエストを並行して送信可能
  - ただし、ALBからターゲットまでの通信はHTTP/1.1になるので注意
  - HTTP/2 のサーバープッシュ機能は使用不可
- ターゲットとしての Lambda 関数
  - 他サービスとの連携の章で後述する
- Websocketに対応

本日のアジェンダ

- ELBの基本
- **ELBの各種機能**
    - ELBの基本機能 (共通機能)
    - ALBの機能について
    - **NLBの機能について**
    - CLBの機能について
- ELBの応用と他サービスとの連携
- まとめ
- 補足資料

NLBの特徴と機能

- TCP(L4)のバランサとして機能
- 固定IPアドレス: AZ毎に1つ、既に持っているEIPも利用可能
- 送信元IPアドレスの保持: X-Forwarded-ForやProxy Protocolが不要
- 暖機なしに急激なスパイクにも対応可能
- VPC エンドポイントサービス(AWS PrivateLink)のサポート
- NLBにはセキュリティーグループの設定がない

特に重要なのは下記の3つ

1. 高可用性、高スループット、低レイテンシ
2. Source IP/Portがターゲットまで保持される
3. 固定IP

https://aws.amazon.com/jp/blogs/news/new-network-load-balancer-effortless-scaling-to-millions-of-requests-per-second/
https://aws.amazon.com/about-aws/whats-new/2017/09/announcing-network-load-balancer-for-elastic-load-balancing/

aws

1. 高可用性、高スループット、低レイテンシ

- 高い可用性を実現
  - DNS名なら、UnhealthyなAZのIPアドレスが自動削除される
  - 長時間セッションも維持が可能

- 暖機不要で突発的な数百万リクエスト/秒のトラフィックも捌ける
  - ELBは動的にキャパシティが拡張されるが、突発的なアクセス上昇の場合、ALB/CLBの拡張が間に合わないことがある。その場合は暖機申請が必要
  - 固定IPのまま動的にスケールする

- TCP負荷分散を同一AZ内で行うので、レイテンシが小さい
  - 単一AZ構成も可能 (ALBは複数AZ構成が必須)

aws

2. Source IP/Portがターゲットまで保持

- クライアントのSource IPとPortが、そのままTargetまで届く
  - Targetはクライアントと直接通信しているかの様に見える
  - 実際は、行きも帰りもNLBを通っている (DSRではない)
  - IP Target（後述）やPrivateLink経由の場合は保持されず、NLBからの通信となる
    - Direct Connectは接続されているVPCからのみ通信可能なので、こちらで回避
- TargetのSecurity Groupでクライアントの接続を許可する必要あり
  - インターネット向けに広く公開する場合は0.0.0.0/0で公開が必要
  - ある程度制限をする場合は加えて、Health checkのためにVPC CIDRかNLB ENIからのアクセスも許可する必要あり
  - VPC内からのアクセスの場合でもターゲットへのアクセス許可はセキュリティグループ IDの指定ではなくクライアントIPの指定が必要
- Targetの選択は5-tupleなのでStickyになる
  - src ip, src port, dst ip, dst port, protocol

aws

3. 固定IP

- Internet-facing、Internal共にIPアドレスが固定
  - AZ毎に1つのIPアドレスを利用、DNSはAレコードでも設定可能
  - ALB, CLBではIPアドレスは不定（DNSで同定可能）

- NLB作成時に自動割当されたIPアドレス、又はNLB作成時に指定した自分が持っているElastic IPのいずれか
  - 自動割り当てされたIPアドレス以外の自前のElastic IPを使う際にはNLB作成前にあらかじめElastic IPを用意しておく必要あり（重要）
  - NLB作成後に変更は不可能

- よくあるユースケース
  - Firewallの制約等で、ELBのIPアドレスの固定が必要な時

注意すること

- Instance Targetには一部古い世代が利用不可
  - C1, CC1, CC2, CG1, CG2, CR1, CS1, G1, G2, HI1, HS1, M1, M2, M3, T1
  - まとめるとI2,C3を除く2013年以前のインスタンスタイプ
- Idle Connection Timeoutは350秒固定
  - アイドルタイムアウト期間の経過後にクライアントまたはターゲットがデータを送信した場合、TCP RST パケットが返されて接続が無効になったことを示す
- Health Checkの設定に、あまり柔軟性がない
  - Timeoutは固定(TCPとHTTPSは10秒、HTTPは6秒)
  - Intervalは10秒または30秒のみで、後から変更不可
- TLSリスナーではアクセスログが取得可能だが、TCPの場合はVPC Flow Logで代替
- NLB自体にセキュリティーグループの設定はない

細かい比較は下記URLを参照
https://aws.amazon.com/jp/elasticloadbalancing/details/#compare

その他特徴

- Websocketに対応
- NLB APIはALBと互換
  - Load Balancer=>Listener=>Rule=>Target Group
  - ALB同様、1つのTargetをポートを変えて同じTarget Groupに複数登録可能
- TargetはInstance又はIPアドレスが利用可能
  - IPアドレスはプライベートセグメントのみ、Direct Connect越しのIPアドレスも利用可能 (VPC peeringとVPNは不可)
- 関連サービスでNLB対応済のもの一覧
  - Amazon ECS, AWS CloudFormation, AWS CodeDeploy, AWS Elastic Beanstalk

本日のアジェンダ

- ELBの基本
- **ELBの各種機能**
    - ELBの基本機能 (共通機能)
    - ALBの機能について
    - NLBの機能について
    - **CLBの機能について**
- ELBの応用と他サービスとの連携
- まとめ
- 補足資料

CLBが必要となるケース

ほとんどのケースにおいては、ALBとNLBを利用することでカバーできるが、
下記の特殊なケースにおいてのみCLBを利用することが可能

- EC2-Classic のサポート
- TCP および SSL リスナーのサポート
- アプリケーション生成のクッキーを使用したスティッキーセッションのサポート
- カスタムセキュリティポリシー

参照
https://docs.aws.amazon.com/ja_jp/elasticloadbalancing/latest/classic/introduction.html#classic-load-balancer-benefits

aws

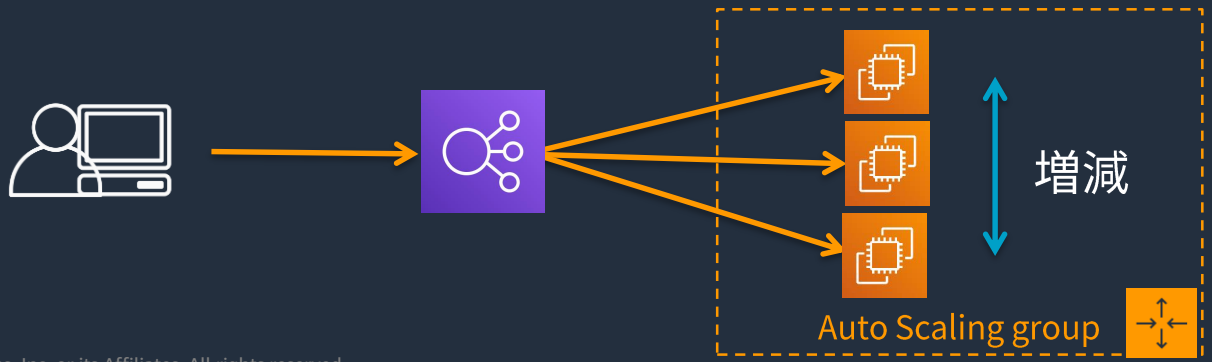
本日のアジェンダ

- ELBの基本
- ELBの各種機能
    - ELBの基本機能 (共通機能)
    - ALBの機能について
    - NLBの機能について
    - CLBの機能について
- **ELBの応用と他サービスとの連携**
- まとめ
- 補足資料

Auto Scalingとの連携

ALB NLB CLB

- Auto Scalingによるインスタンス増減時にELBへの追加・削除が可能
- ELBのヘルスチェックの結果をAuto Scalingに反映可能
- インスタンス削減時は、Connection Drainingによる処理中の接続を待つ
- 利用例
  - 一定間隔でレスポンスをチェックし、遅延が増加したらインスタンスを自動追加
  - ELBのヘルスチェックが成功したEC2インスタンスを常にX台以上



aws

ECSとの連携

ALB NLB

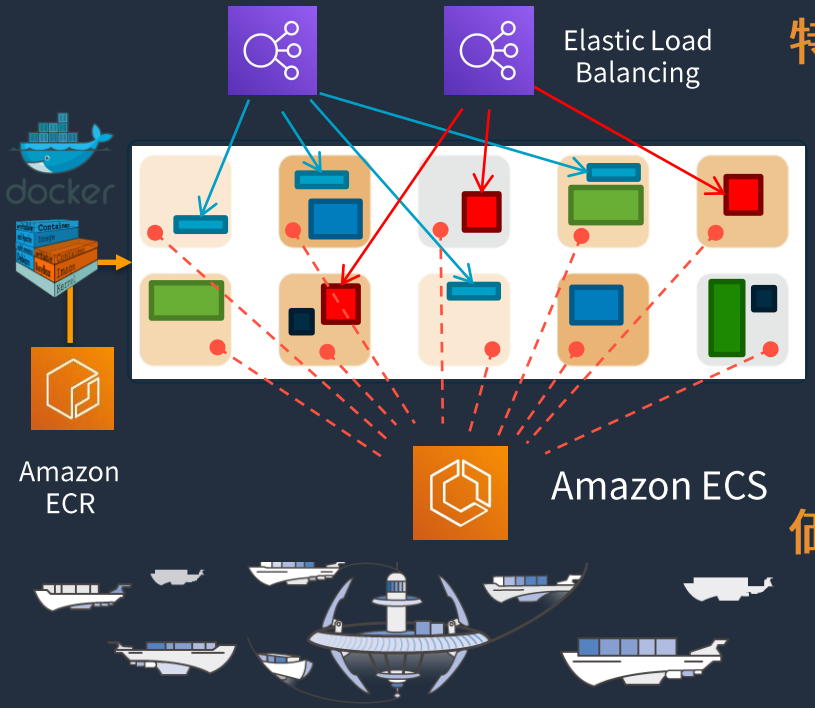
ECS のタスク（サービス）を ALB またはNLBのターゲットグループに登録することで、ECS の各タスクに対して負荷分散を実施できる

動的ポートマッピングにより各タスクに割り振られたECSクラスタ上のポートをターゲットグループに自動的に登録

aws

Amazon EC2 Container Service (ECS)

管理されたEC2クラスタ上に、コンテナを自在に配置できる



特徴 (https://aws.amazon.com/ecs/)

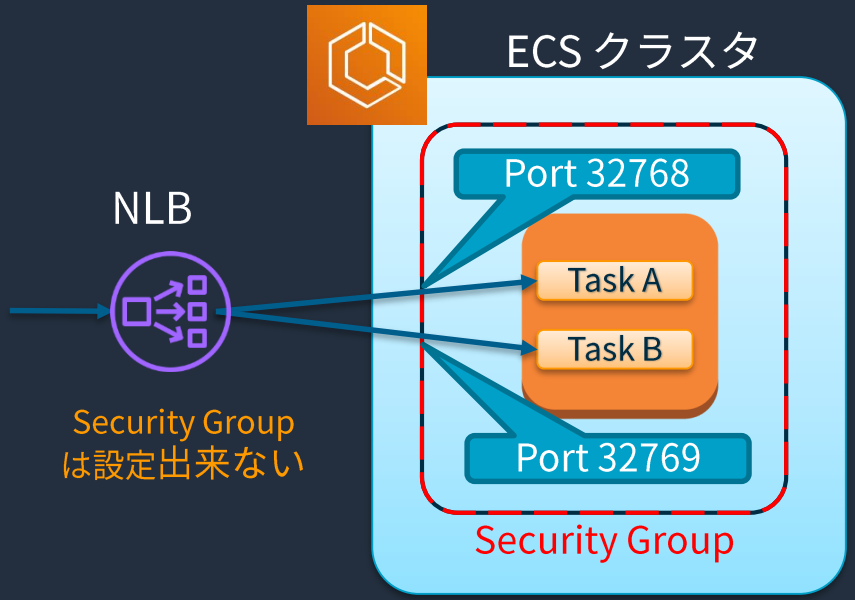
- 管理ノード不要の、安定かつ高パフォーマンスなクラスタ管理サービス
- Serviceスケジューラで多様なロングランニングプロセスを実行する基盤に
  - コンテナを必要な台数稼働させる
  - ELB連携で、デプロイも簡単に
- Run Taskでバッチジョブを実行する基盤に
  - どこかのEC2でコンテナを起動して処理させる

価格体系 (https://aws.amazon.com/ecs/pricing/)

- 無料
  - 利用するEC2, EBS, ELBなどの料金のみ発生

Security Group と動的ポートマッピング

NLB には Security Group が設定できないため、ECS コンテナインスタンス側で Security Group の設定を行う。ECS タスクに動的に設定されるポートの範囲を意識する必要がある。

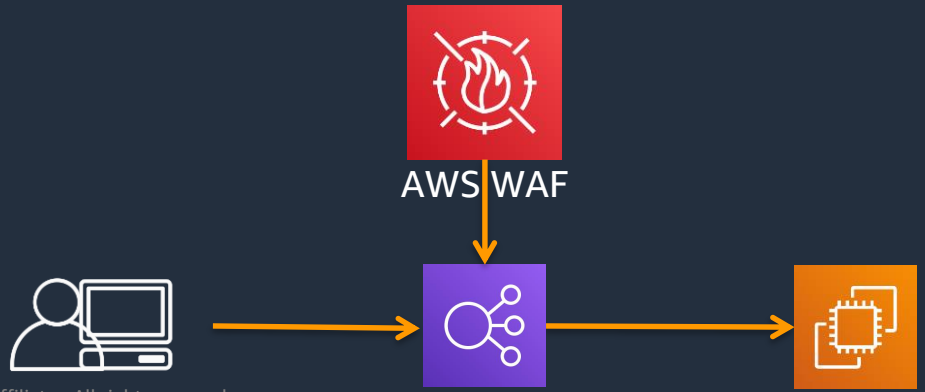


- ECS コンテナインスタンス側で Security Group を設定
- ECS の動的ホストポートマッピングの範囲は /proc/sys/net/ipv4/ip_local_port_range で設定

ALBとAWS WAFの連携

ALB

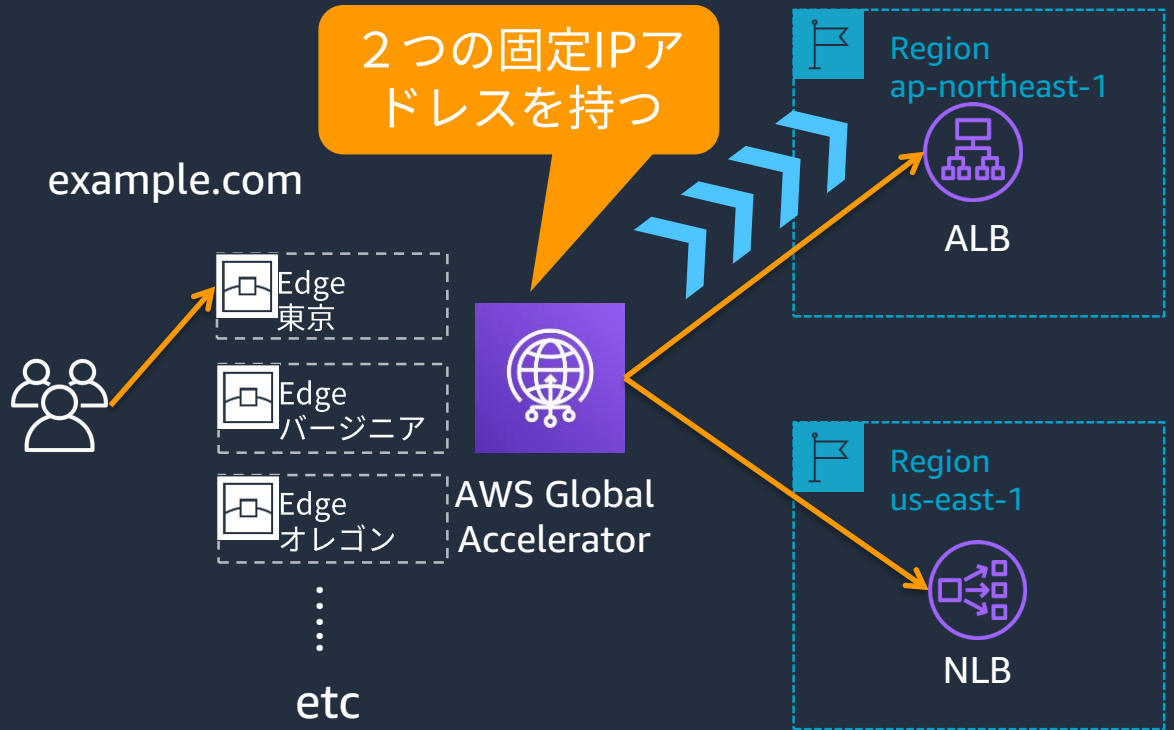
- ALBに対してAWS WAFを適用可能
- AWS WAFを用いると以下の条件を利用してALBを保護可能
  - リクエストレートによるアクセス権限(Rate Limit)
  - 特定のIPアドレスや地域からのアクセスを制限
  - クロスサイトスクリプティングやSQLインジェクションからの保護
  - HTTP ヘッダー、HTTP 本文、URI 文字列に対するサイズ制約や正規表現でのマッチング



aws

Global Acceleratorとの連携

ALB NLB

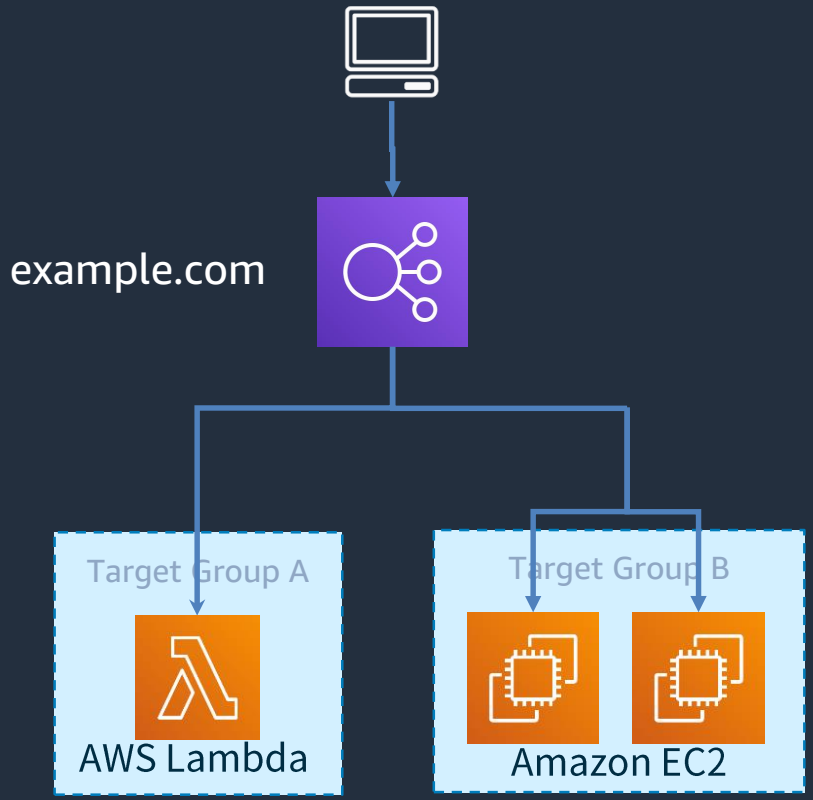


- Global AcceleratorのエンドポイントにALB, NLB (及びElastic IP)を指定できる
- ユーザーは近いエッジロケーションからAmazon Global Networkを経由して最も近いリージョンにアクセス可能
- ユーザーはマルチリージョンアプリケーションへのアクセスに対して固定IPでアクセス可能
- ALBのIPを固定にできる

aws

Lambda 関数をターゲットにする

ALB



- Lambda 関数をターゲットとするターゲットグループを作成し、リクエストを転送するリスナールールを設定可能
- ターゲットグループにリクエストを転送すると、Lambda 関数を呼び出し、リクエストのコンテンツを JSON 形式で Lambda 関数に渡す
- VPCが同一であれば、１つのALBに対してLambdaのターゲトグループにもEC2またはECSのターゲットグループにもリクエストを転送可能

aws

本日のアジェンダ

- ELBの基本
- ELBの各種機能
    - ELBの基本機能 (共通機能)
    - ALBの機能について
    - NLBの機能について
    - CLBの機能について
- ELBの応用と他サービスとの連携
- まとめ
- 補足資料

まとめ

- Application Load Balancer (ALB)
  - HTTP/HTTPSのみに対応した (TCPには対応していない) L7ロードバランサー
  - コンテンツベースのルーティング (高度なリクエストルーティング) が可能
- Network Load Balancer (NLB)
  - TCP/UDPに対応したL4ロードバランサー
  - 暖気不要で高可用性、高スループット、低レイテンシ
  - Source IP/Portがターゲットまで保持される
  - 固定IPを持つ
- Classic Load Balancer (CLB)
  - 特殊なケースのみに利用し、多くのケースではALBかNLBで十分
  - 既存のCLBはALBかNLBへの移行を推奨

aws

ELBの料金

ALB

- ALB毎の料金＋LCU消費の料金が1時間単位
  - ALB 1つにつき1時間毎に$0.0243 (東京リージョン)
    - 参考: NLBも$0.0243
  - 1 LCU 1時間毎に$0.008 (全リージョン)

ALBの1 LCUの単位

- 新規接続数: 25/sec
- Active接続数: 3,000/min
- 帯域: 2.22 Mbps (1 GB/hour)
- Ruleの評価数 1,000/sec

NLB

- NLB毎の料金＋LCU消費の料金が1時間単位
  - NLB 1つにつき1時間毎に$0.0243 (東京リージョン)
  - 1 LCU 1時間毎に$0.006 (全リージョン)
    - 参考: ALBは$0.008
    - LCUの定義が異なるのでALB/NLBどちらが安いかは比較すること。

NLBの1 LCUの単位

- 新規TCP接続又はフロー数: 800/sec
- Active接続又はフロー数: 10,000/min
- 帯域: 2.22 Mbps (1 GB/hour)

CLB

- CLB毎の料金＋1時間単位で処理されるデータのGB 単位で課金
  - $0.027/hour
  - $0.008/GB

多くのケースでNLB/ALBのほうがCLBよりも低コストとなります

詳細 https://aws.amazon.com/jp/elasticloadbalancing/pricing/

aws

本日のアジェンダ

- ELBの基本
- ELBの各種機能
    - ELBの基本機能 (共通機能)
    - ALBの機能について
    - NLBの機能について
    - CLBの機能について
- ELBの応用と他サービスとの連携
- まとめ
- **補足資料**

補足) CLBからALBへの移行を助けるメトリクス

CLBを、Load Balancer Capacity Unit (LCU)ベースで課金されるALBへ移行する際、参考になるおおよその値がCloudWatchメトリクスに出力される

4つの見積もり用メトリクス

- EstimatedALBActiveConnectionCount
- EstimatedALBNewConnectionCount
- EstimatedProcessBytes
- EstimatedALBConsumedLCUs – ALBにおける課金対象の1つ

メトリクスはどれも移行の見積もり用で、CloudWatchアラームでは利用しないこと

https://docs.aws.amazon.com/ja_jp/elasticloadbalancing/latest/classic/elb-cloudwatch-metrics.html

aws

補足) CLBからワンステップでALB/NLBへ移行できるマイグレーションウィザード

ウィザード形式でCLBからALB/NLBへワンステップで移行できるCLBの機能

https://docs.aws.amazon.com/ja_jp/elasticloadbalancing/latest/userguide/migrate-to-application-load-balancer.html

ELBの負荷テストについて

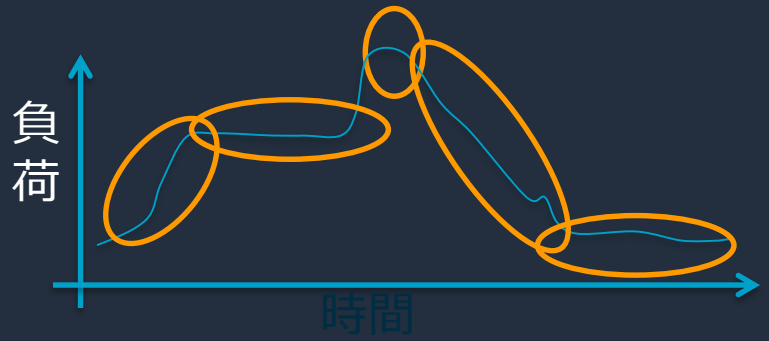
ELBの負荷テストについて

ELBのいくつかの特長がテストシナリオに影響を与える可能性がある

- ELB のスケーリング
- ELB の初期キャパシティ
- アイドル時のコネクションタイムアウト
- バックエンドインスタンスのヘルスチェック
- Sticky セッション 等

ご利用内容に合わせたシナリオでテストが必要

推奨テストアプローチ



- 想定する最大負荷のテスト
- 通常のトラフィック時のテスト
  - トラフィックの多い時
  - トラフィックの少ない時
  - トラフィックの傾向に変化がある時（朝や昼の時間帯など）
- 短い時間でトラフィックが大きく変化する場合のテスト

ELB以外にも負荷生成クライアント、バックエンドEC2インスタンスも監視すべき

  - アプリケーション内部の動作も要確認
  - どこかボトルネックになっているか把握しておく

負荷テストの注意事項

- ALB/CLBの初期スケールに注意
  - NLB以外のELB(ALB/CLB)ではスケールするまでに、HTTP 503レスポンスを返す期間があり得る
  - 回避策：
    - ALB/CLBの暖気運転（Pre-Warming）申請をする
    - 5分間隔で50%以上のトラフィック増加をしないよう負荷テストを設定
- DNSクエリの仕方に注意
  - テストクライアント側で少なくとも1分に1回DNSの再解決をする
- スティッキーセッション利用時の割り振り方
  - 同じCookieでリクエストを続けた場合などは振り分けに偏りが発生
- バックエンドインスタンスのアイドルタイムアウト
  - ELBのタイムアウト値以上に設定しないとELBが誤って不健全なホストと見なす可能性あり

詳細 https://aws.amazon.com/jp/articles/best-practices-in-evaluating-elastic-load-balancing/

ELB基本機能の比較 (1)

	Application Load Balancer	Network Load Balancer	Classic Load Balancer
プロトコル	HTTP, HTTPS, HTTP/2	TCP, UDP	TCP, SSL, HTTP, HTTPS
ヘルスチェック	✓	✓	✓
CloudWatch メトリクス	✓	✓	✓
ログ記録	✓	✓	✓
ゾーンごとのフェイルオーバー	✓	✓	✓
Connection Draining (登録解除の遅延)	✓	✓	✓
同一のインスタンスで複数ポートに負荷分散	✓	✓	
IP アドレスをターゲットに設定	✓	✓	

ELB基本機能の比較 (2)

	Application Load Balancer	Network Load Balancer	Classic Load Balancer
クロスゾーン負荷分散	✓	✓	✓
スティッキーセッション	✓		✓
パスベースルーティング/ホストベースのルーティング	✓		
静的 IP アドレス		✓	
送信元 IP アドレスの保持		✓	
WebSocket 対応	✓	✓	
Container Support	✓	✓	
PrivateLink のサポート		✓	

aws

ELBセキュリティ関連の比較

	Application Load Balancer	Network Load Balancer	Classic Load Balancer
SSL のオフロード	✓	✓	✓
Server Name Indication (SNI)	✓	✓	
バックエンドサーバーの暗号化	✓	✓	✓
ユーザー認証	✓		
カスタムセキュリティポリシー			✓

Q&A

お答えできなかったご質問については
AWS Japan Blog 「https://aws.amazon.com/jp/blogs/news/」にて
後日掲載します。

AWS の日本語資料の場所「AWS 資料」で検索

https://amzn.to/JPArchive

AWS Well-Architected 個別技術相談会

毎週”W-A個別技術相談会”を実施中

- AWSのソリューションアーキテクト(SA)に対策などを相談することも可能
- 申込みはイベント告知サイトから

(https://aws.amazon.com/jp/about-aws/events/)

AWS イベント で[検索]

ご視聴ありがとうございました

AWS 公式 Webinar
https://amzn.to/JPWebinar

過去資料
https://amzn.to/JPArchive