失敗例を成功に変える
AWSアンチパターンのご紹介

2015-06-09

アマゾンデータサービスジャパン

荒木靖宏

自己紹介

- 名前
  - 荒木 靖宏
- 所属
  - アマゾンデータサービスジャパン株式会社
  - 技術本部レディネスソリューション部シニアマネージャ
    プリンシパルソリューションアーキテクト
- 好きなAWSサービス
  - Amazon Virtual Private Cloud
  - AWS Direct Connect

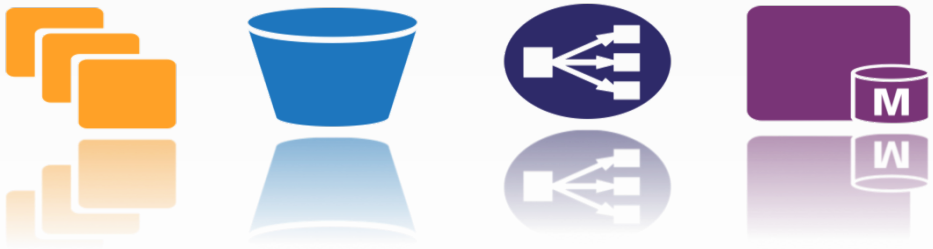
これまで、数多くのAWS成功例がうまれていった。。

その成功例は「パターン」として受け継がれ。。

そして、それらは時に「秘伝のたれ」「さわってはいけないもの」とされてきた。。

AWSクラウドデザインパターンとは...

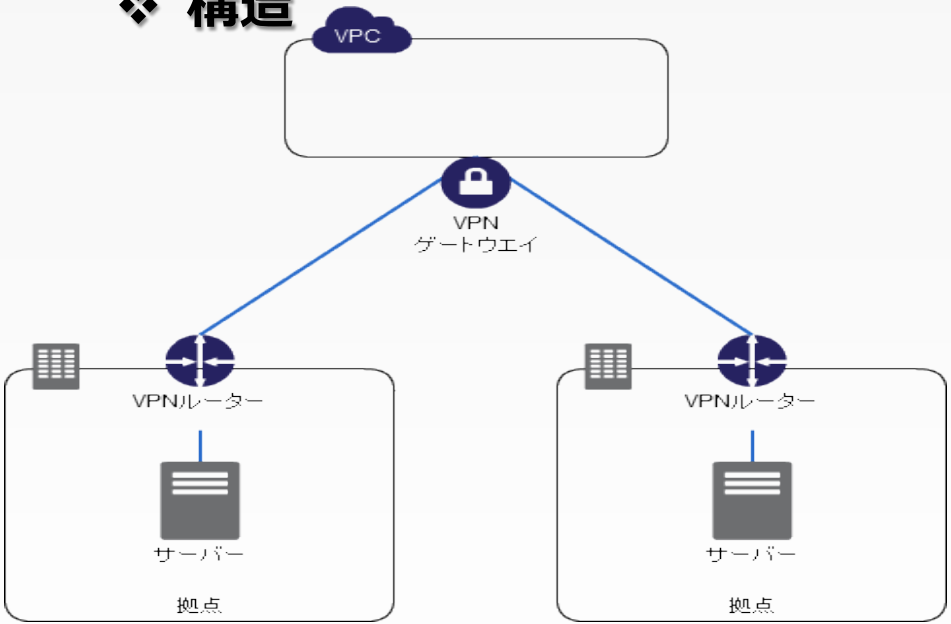
- AWSクラウドを使ったシステムアーキテクチャ設計を行う際に発生する、典型的な問題とそれに対する解決策・設計方法を、分かりやすく分類して、ノウハウとして利用できるように整理したもの。



例えば... (CloudHubパターン)

- **解決したい課題**
  VPNをハブ状に構成し、メンテナンスコストを下げる。この場合ハブとなるVPN機器の可用性と性能に対するコストが問題になる。
- **クラウドでの解決**
  ハブ構成に対応したVPN機能を安価に提供しているサービスを利用する。
- **実装**
  vpcのVPNゲートウェイに対して複数のカスタマーゲートウェイを設定する。
- **利点**
  管理コスト、初期費用の削減
- **注意点**
  対応可能なVPN数などの制約に注意する

❖ **構造**



アンチパターンの前に

アンチパターン

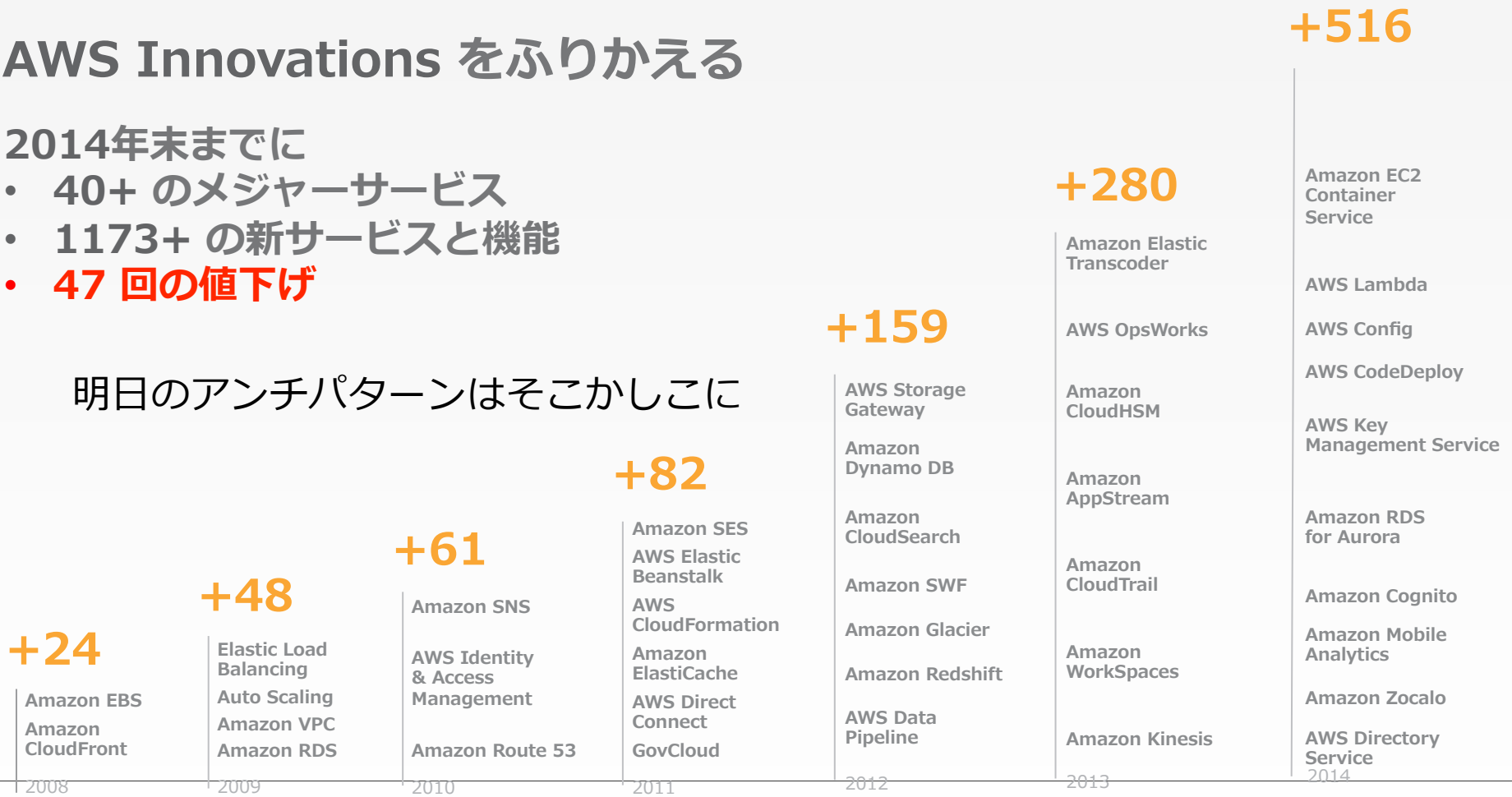
- 失敗に陥るパターンを類型化し、事例の早期発見と対応策に関しての提案を目的とする
- 動作やプロセス、構造について、当初は妥当であったのに、最終的に悪い結果が繰り返されるパターン
- **リファクタリングするための方法が存在**するパターン

AWS Innovations をふりかえる

2014年末までに

- 40+ のメジャーサービス
- 1173+ の新サービスと機能
- **47 回の値下げ**

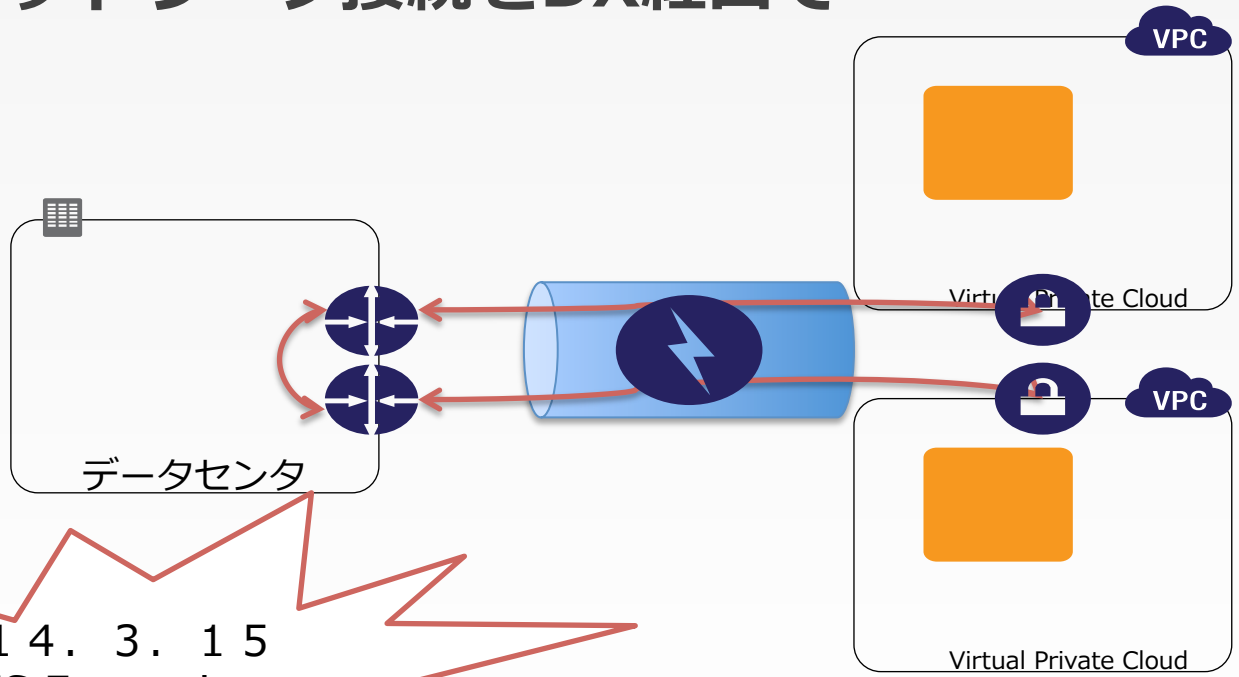
明日のアンチパターンはそこかしこに



amazon web services

アンチパターンとなった例

ヘアピンDXパターン
VPC間のネットワーク接続をDX経由で



２０１４．３．１５
JAWS Enterprise
CDPのセッションで
発表

VPC Peering

VPC Peering

- 登場
  - 2014年3月24日（AWS Summit San Francisco）
- 引導を渡したもの
  - ヘアピンDXパターン

アンチパターンを紹介します

EC2にまつわるアンチパターン

EC2一神教アンチパターン

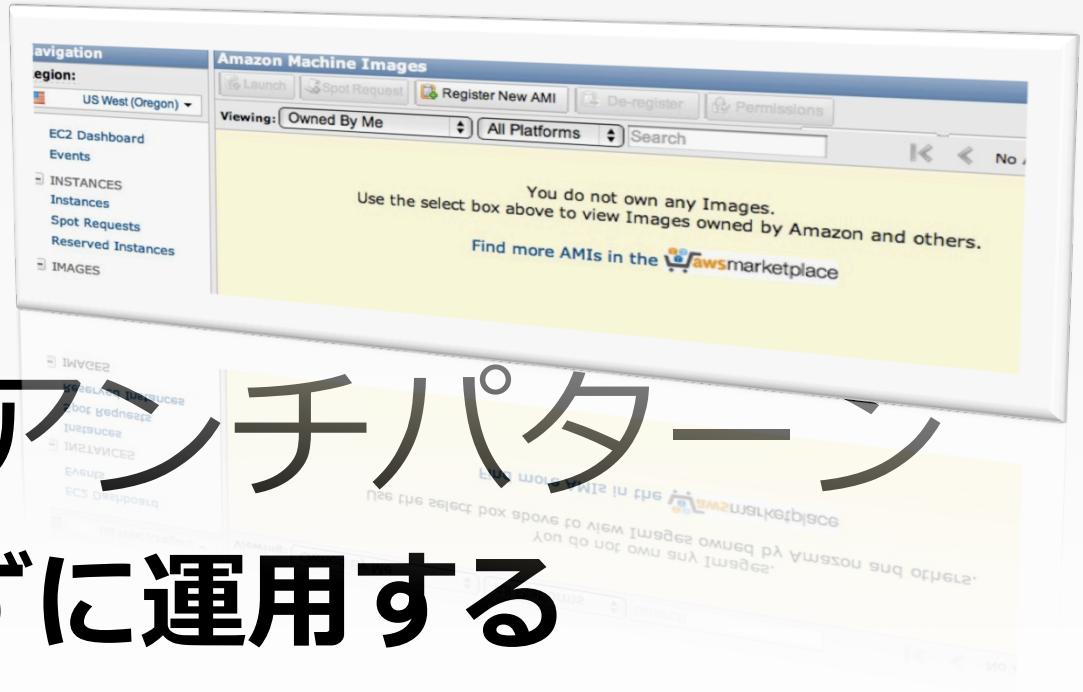
EC2は自由。

EC2一神教アンチパターン

- 原因
  - AWSの知識が古いまま止まっている
  - サーバを調達し、その上で作る方法に慣れ親しんでいるため
- 症状
  - 目的毎にEC2を用意するため、インスタンス数が増えすぎる
  - 可用性の担保にも手間がかかる

EC2一神教アンチパターン

- 解決法
  - SQS, Route53, RDS, S3, ELBなどEC2以外のサービスを活用する
    - 可用性の確保
    - 利用額の低下



AMIなしアンチパターン

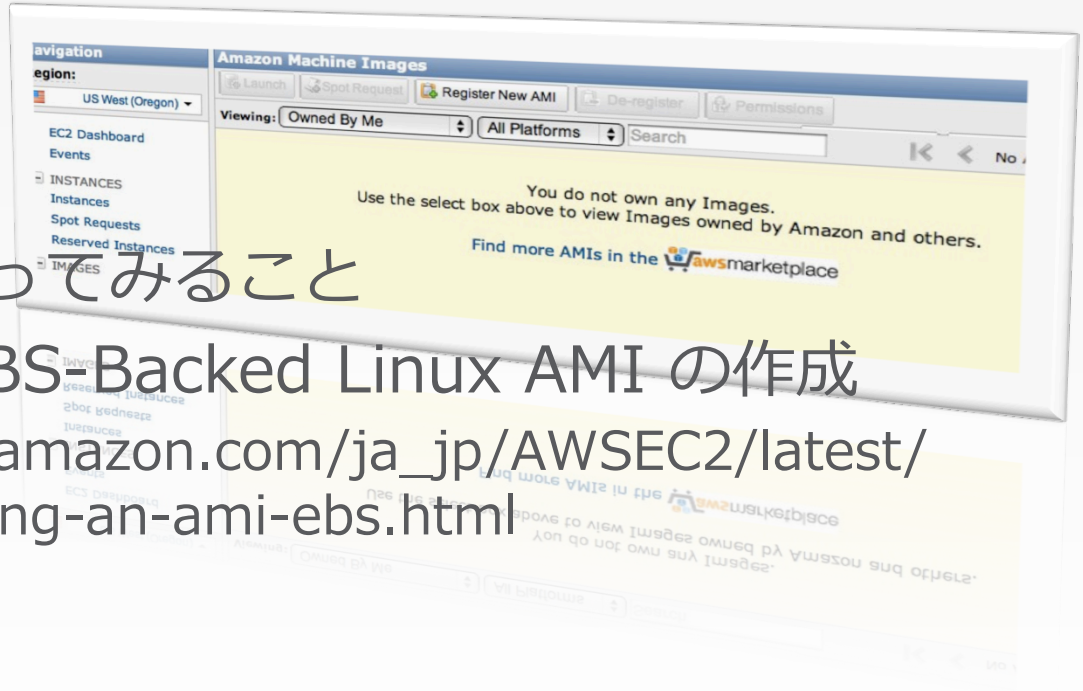
AMIを作らずに運用する

AMI無しアンチパターン

- 原因
  - AMI作りが難しいと思っている
  - オンプレミスのインストール手順に固執
- 症状
  - インスタンスをイチから用意するため、サービスインに時間がかかる
  - サービス拡張時も手順が残っていないと対処できない

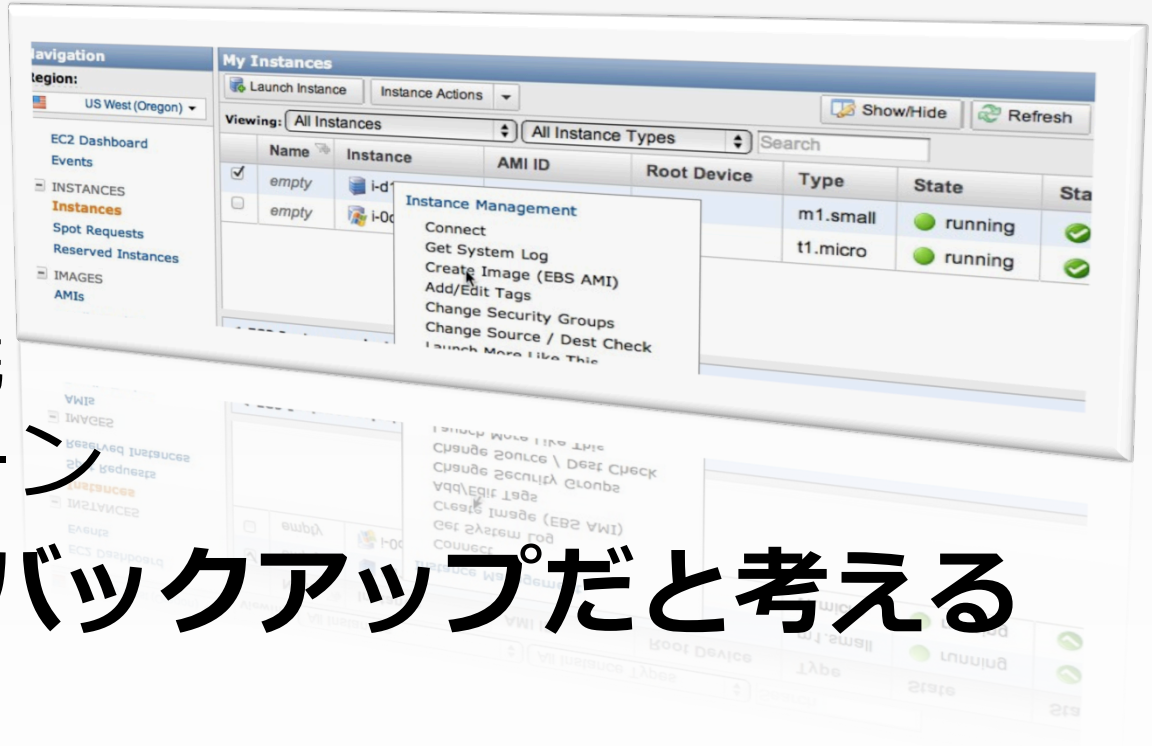
AMI無しアンチパターン

- 解決法
  - AMI作成をまずやってみること
  - 参考：Amazon EBS-Backed Linux AMI の作成
    - http://docs.aws.amazon.com/ja_jp/AWSEC2/latest/UserGuide/creating-an-ami-ebs.html



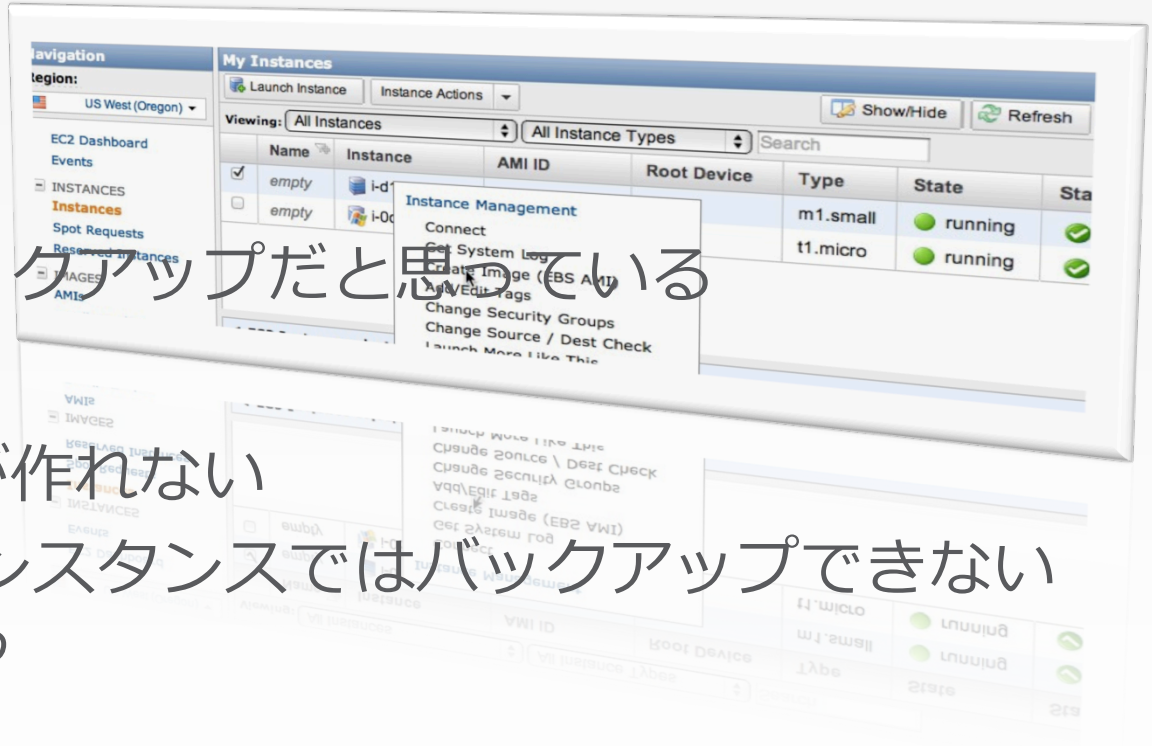
AMI至上主義
アンチパターン

AMI作成をバックアップだと考える



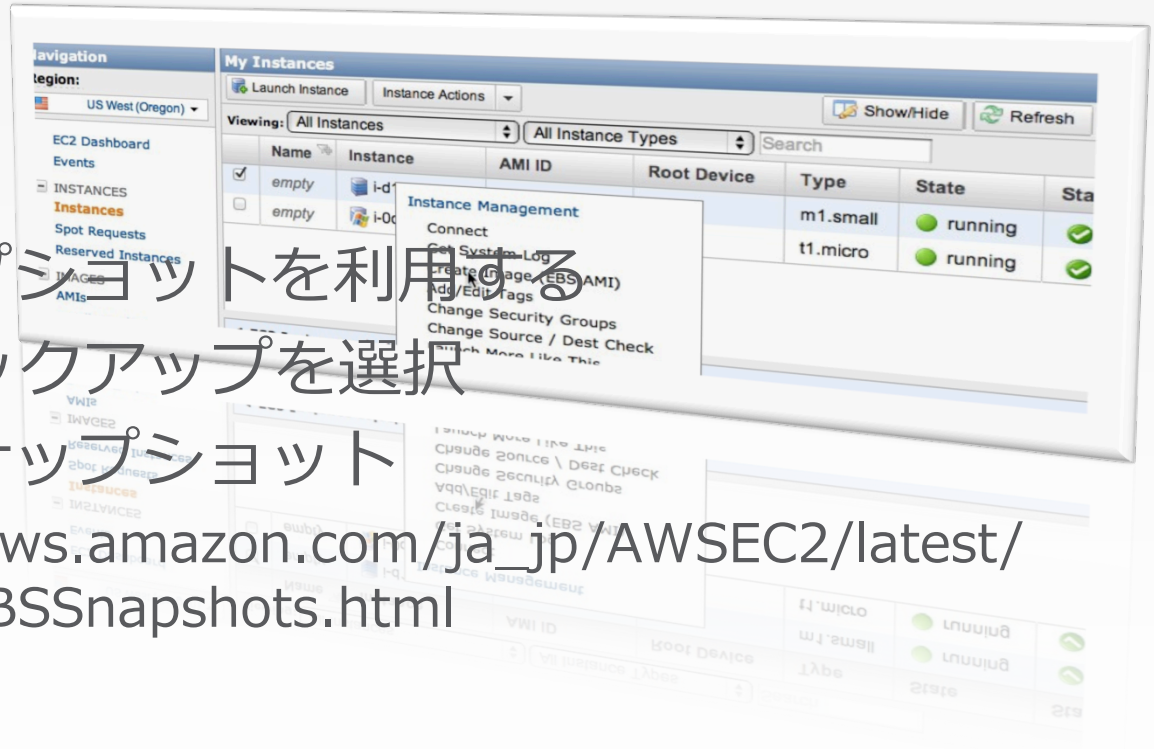
AMI至上主義アンチパターン

- 原因
  - AMI作成をバックアップだと思っている
- 症状
  - 動作中にAMIが作れない
  - S3バックのインスタンスではバックアップできないと思ってしまう



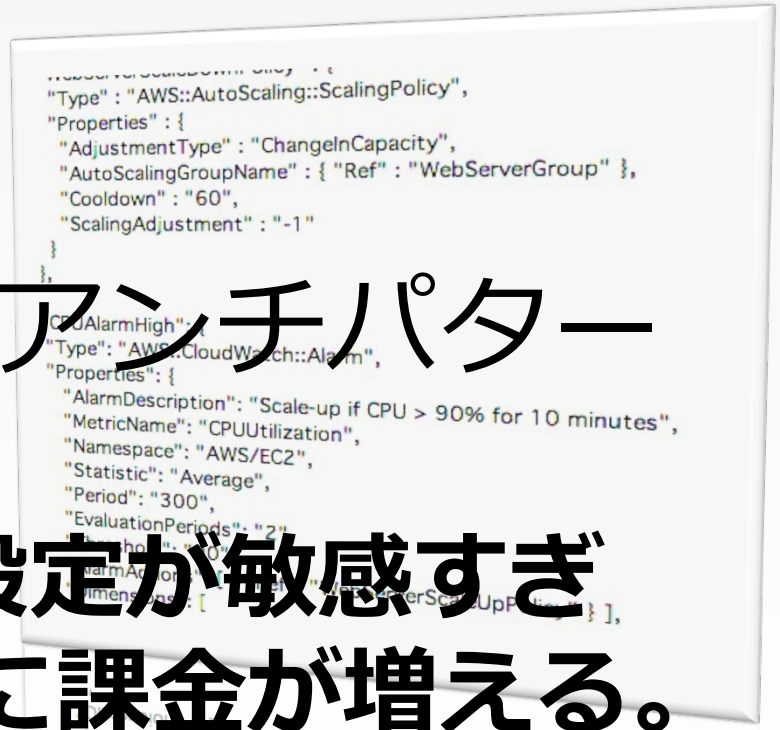
AMI至上主義アンチパターン

- 解決法
  - EBSのスナップショットを利用する
  - 適材適所のバックアップを選択
  - 参考：EBSスナップショット
    - http://docs.aws.amazon.com/ja_jp/AWSEC2/latest/UserGuide/EBSSnapshots.html



インスタンス振動アンチパターン

AUTOSCALINGの設定が敏感すぎて、増減過多。ムダに課金が増える。

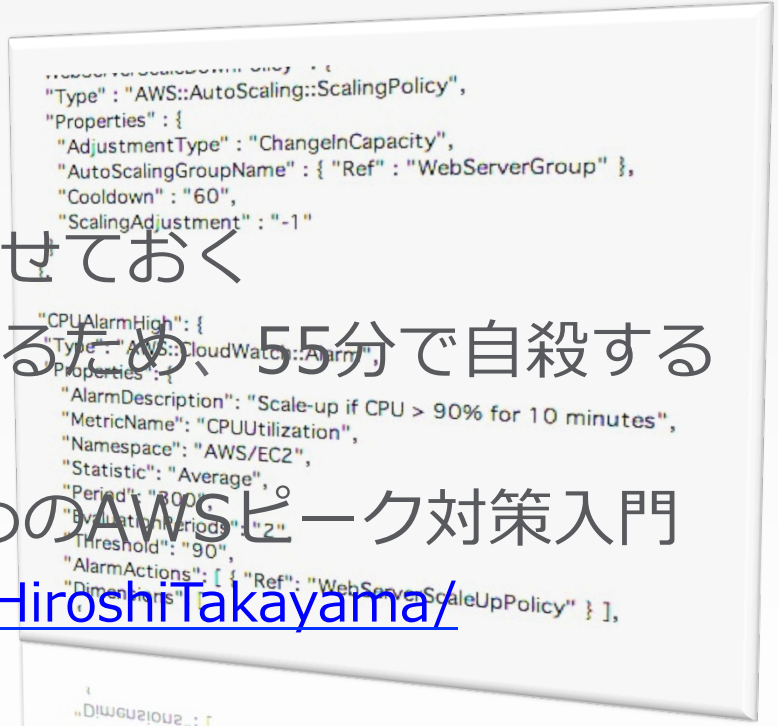


インスタンス振動
アンチパターン

- 原因
  - AutoScalingの設定が敏感すぎる。
  - CloudWatchの条件ソースが不適切
- 症状
  - 増減にともなう課金増加

インスタンス振動
アンチパターン

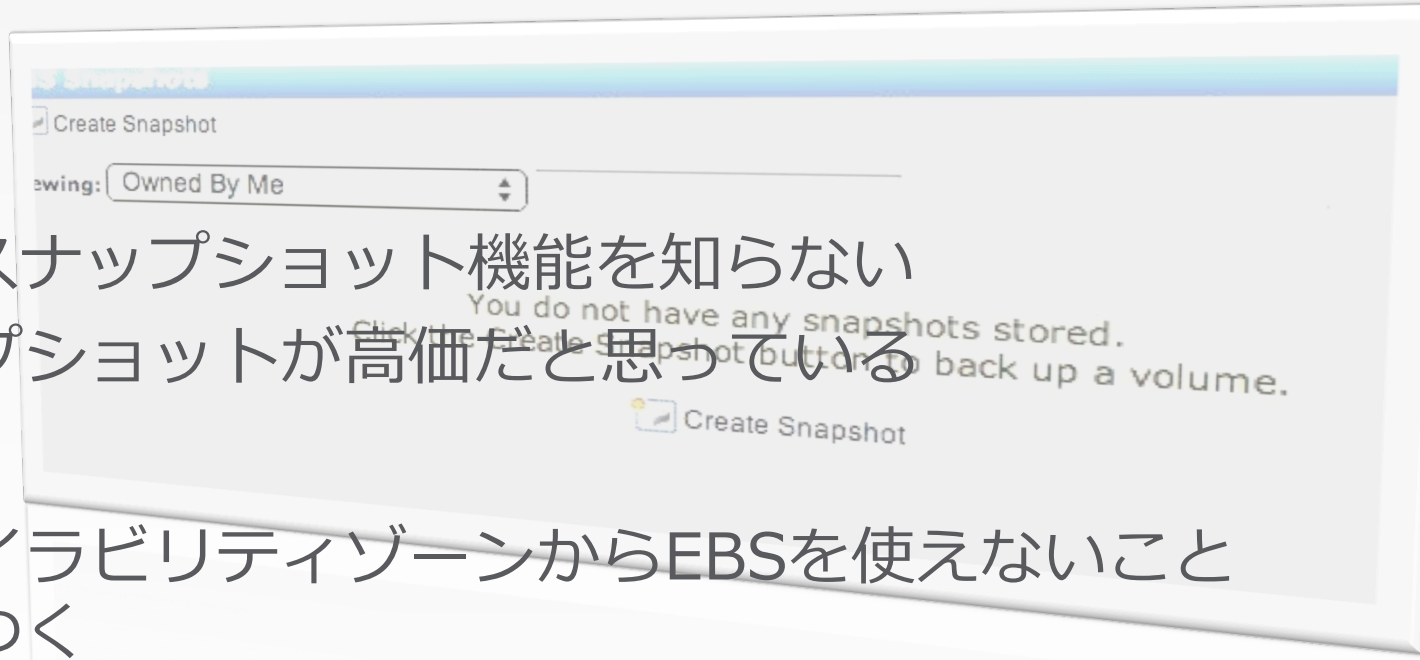
- 解決法
  - 起動条件の4倍程度に緩和させておく
  - 1時間に切り上げて課金されるため、55分で自殺する方法も
  - 参考：スタートアップのためのAWSピーク対策入門
    - http://www.slideshare.net/HiroshiTakayama/jawschiba20140405public



ノースナップショットアンチパターン

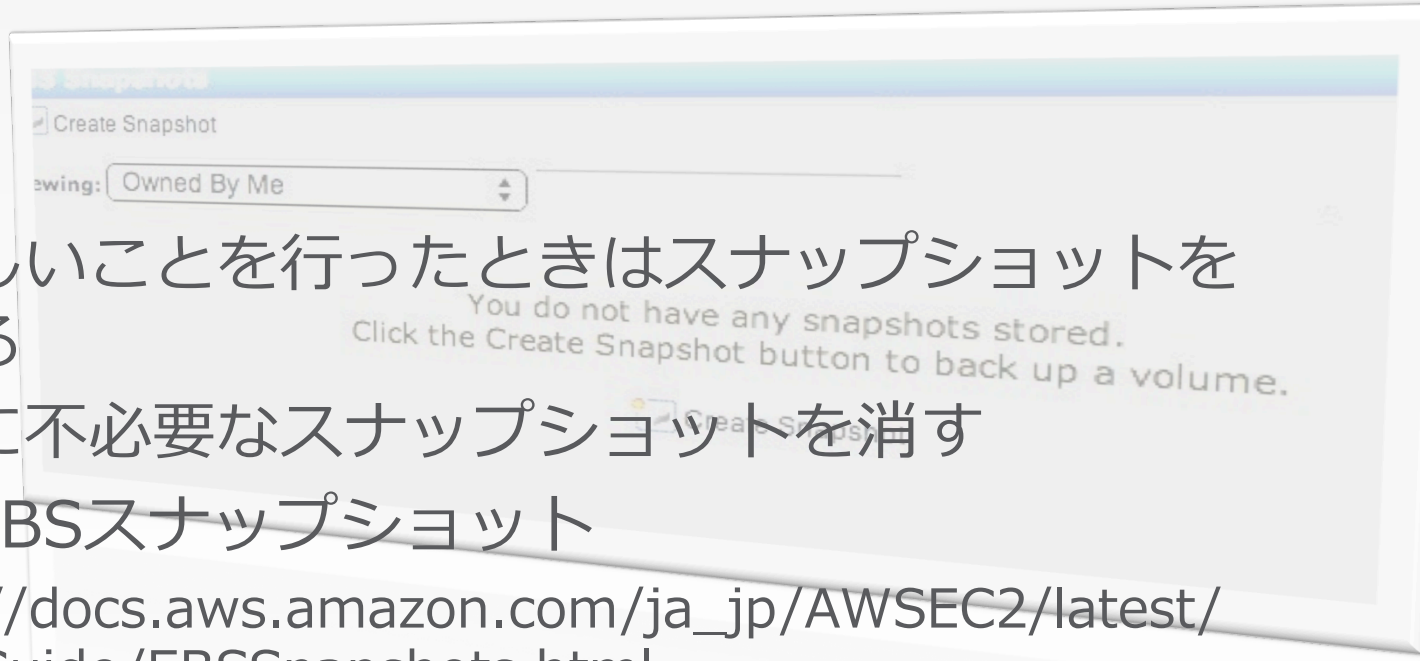
楽なバックアップです

ノースナップショット アンチパターン

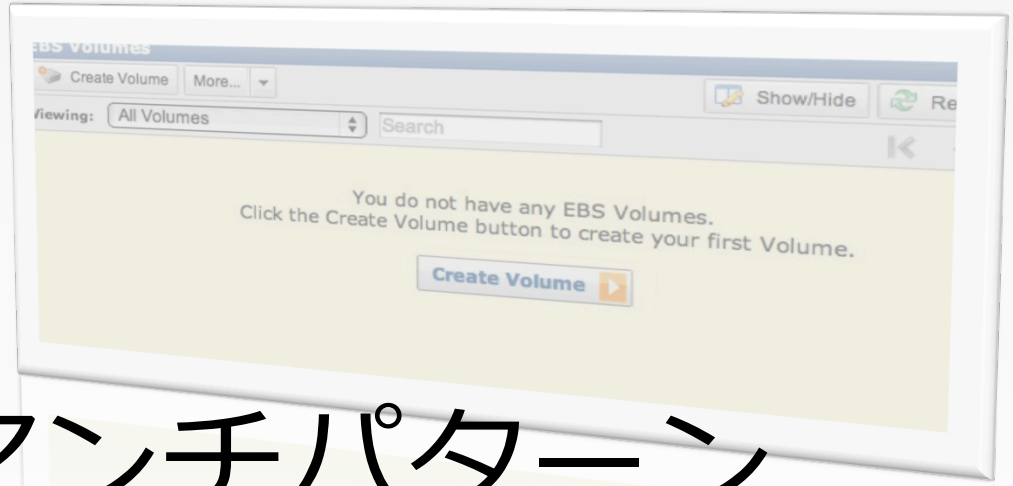


- 原因
  - EBSのスナップショット機能を知らない
  - スナップショットが高価だと思っている
- 症状
  - 別アベイラビリティゾーンからEBSを使えないことで気がつく
  - 操作ミスでファイルを消してから泣きつく

ノースナップショット アンチパターン



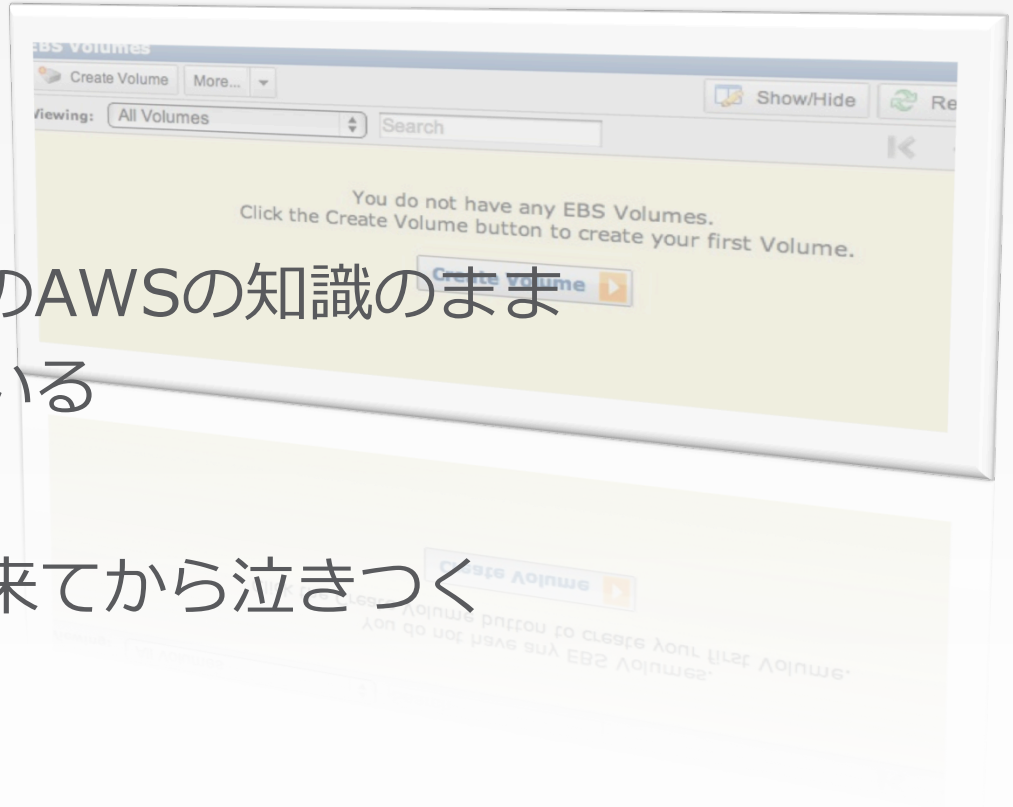
- 解決法
  - 何か新しいことを行ったときはスナップショットを作成する
  - 定期的に不必要なスナップショットを消す
  - 参考：EBSスナップショット
    - http://docs.aws.amazon.com/ja_jp/AWSEC2/latest/UserGuide/EBSSnapshots.html



データ揮発アンチパターン

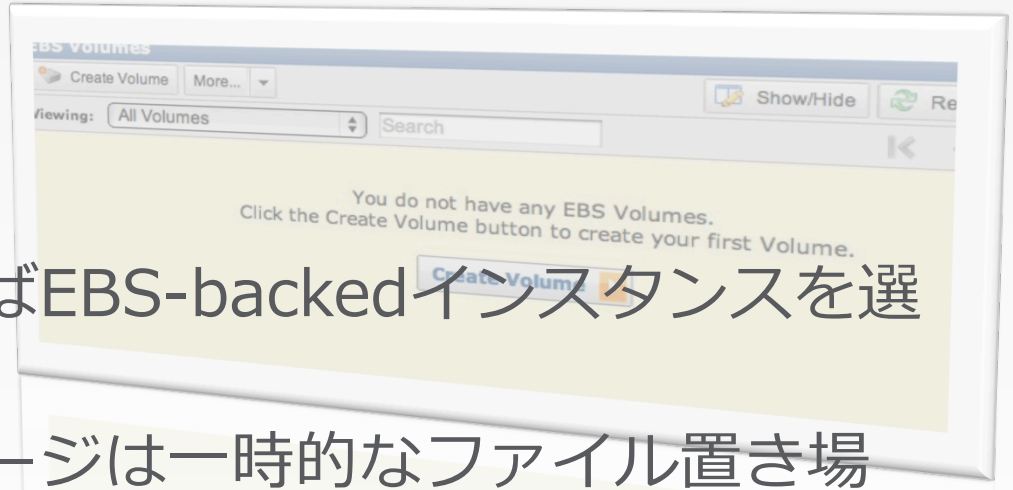
薄氷。

データ揮発
アンチパターン



- 原因
  - EBSがなかったころのAWSの知識のまま
  - EBS費用を心配している
- 症状
  - メンテナンス通知が来てから泣きつく

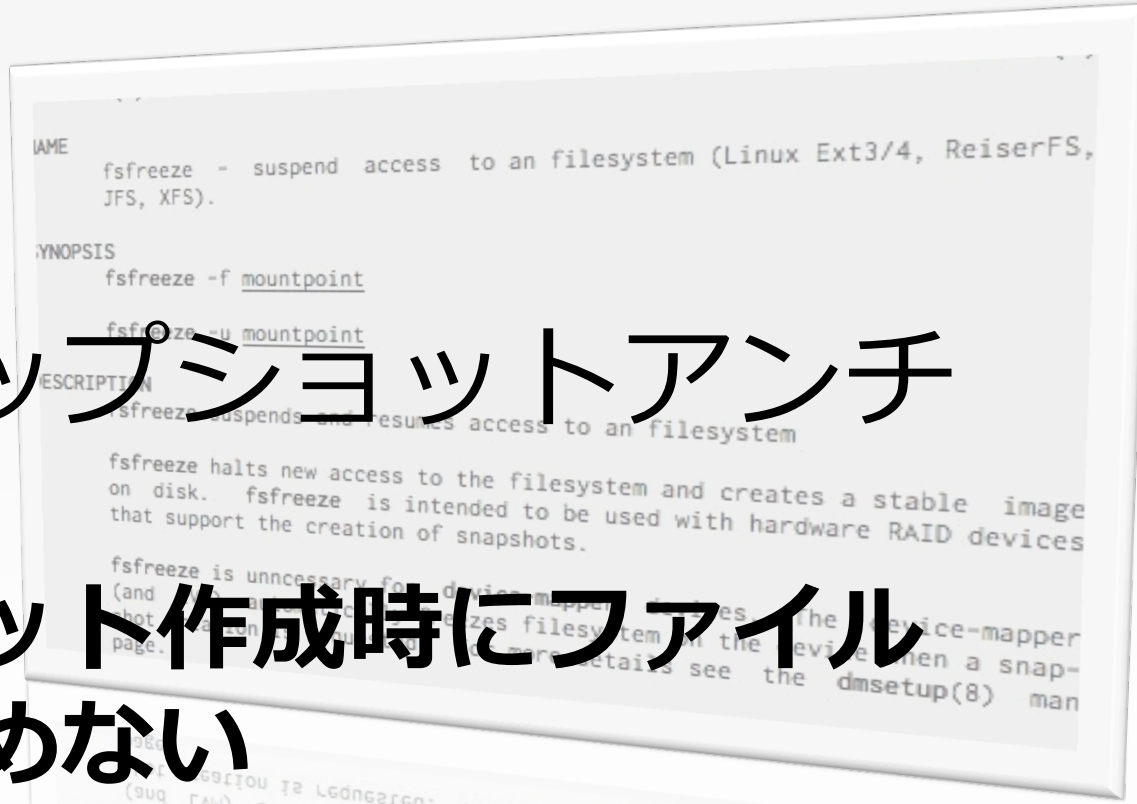
データ揮発
アンチパターン



- 解決法
  - 特に理由がないならばEBS-backedインスタンスを選択する
  - インスタンスストレージは一時的なファイル置き場として使う
  - 参考：EBSについて
    - http://aws.amazon.com/jp/ebs/

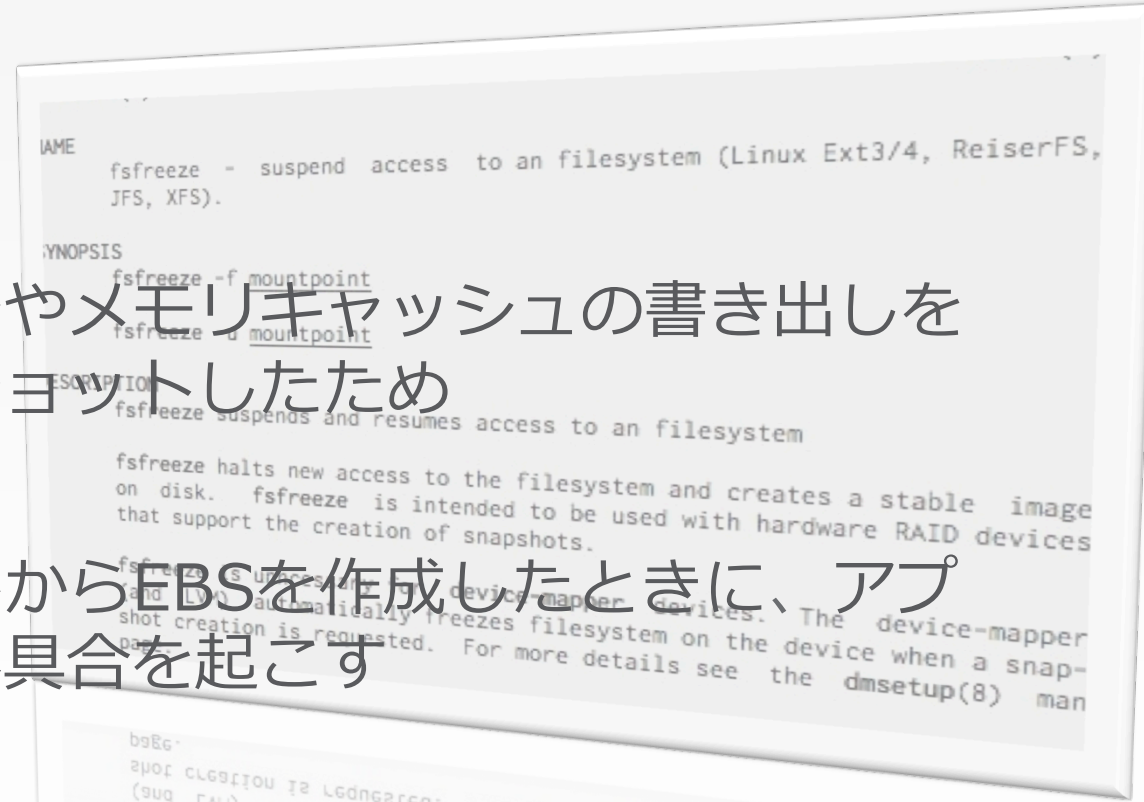
無鉄砲スナップショットアンチパターン

スナップショット作成時にファイルシステムを止めない



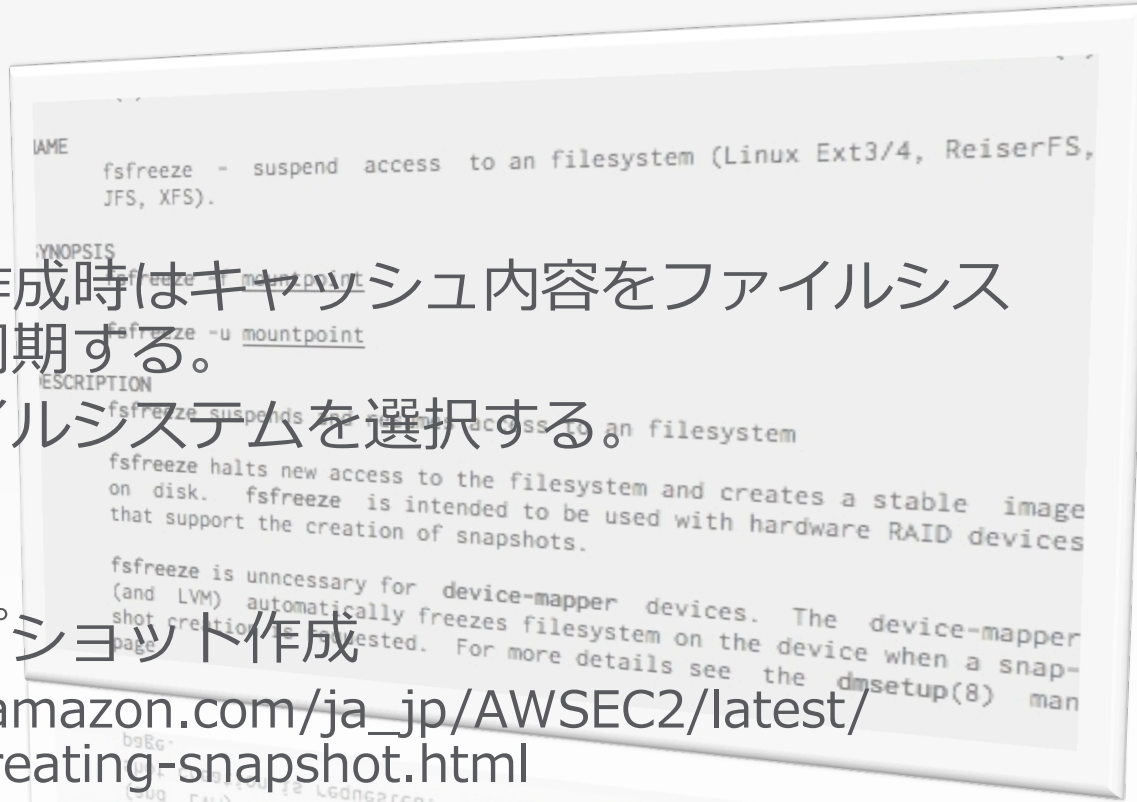
無鉄砲スナップショット アンチパターン

- 原因
  - アプリケーションやメモリキャッシュの書き出しをせずにスナップショットしたため
- 症状
  - スナップショットからEBSを作成したときに、アプリケーションが不具合を起こす



無鉄砲スナップショット アンチパターン

- 解決法
  - スナップショット作成時はキャッシュ内容をファイルシステムに書き出し、同期する。
  - freeze対応のファイルシステムを選択する。
    - XFS: xfs_freeze
    - ext4: fsfreeze
  - 参考：EBSスナップショット作成
    - http://docs.aws.amazon.com/ja_jp/AWSEC2/latest/UserGuide/ebs-creating-snapshot.html



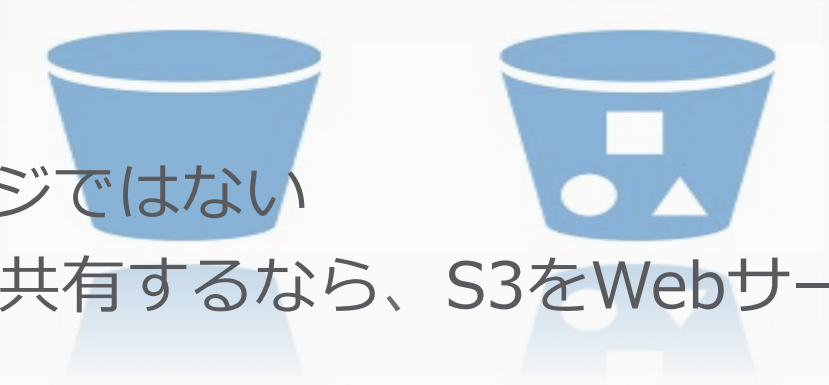
S3勘違いアンチパターン

S3は万能のストレージか？

S3勘違いアンチパターン

- 原因
  - S3を共有ストレージ、ブロックストレージとして使ってしまう
- 症状
  - S3FS経由でログやウェブコンテンツをS3に配置して共有しようとする。
  - S3FS経由でデータベースファイルを配置する

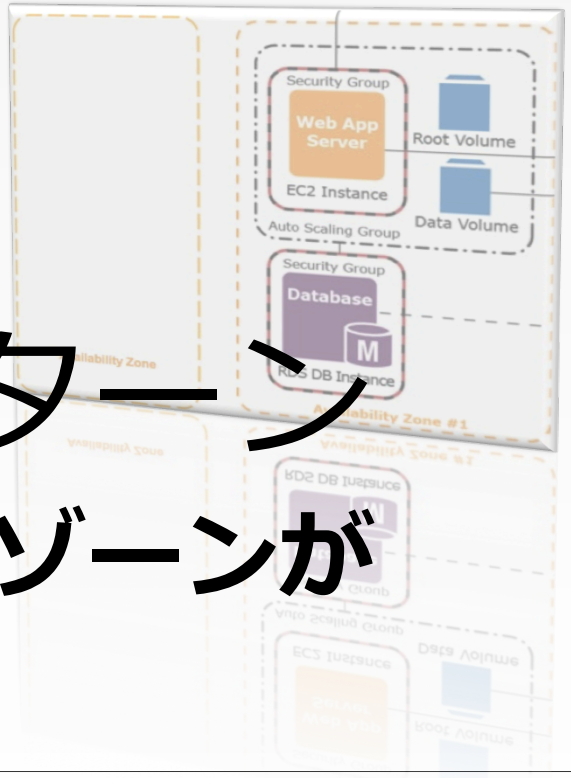
S3勘違いアンチパターン



- 解決法
  - S3はブロックストレージではない
  - S3でWebコンテンツを共有するなら、S3をWebサーバとして利用する
  - (Coming soon) Elastic File System (EFS)を利用する

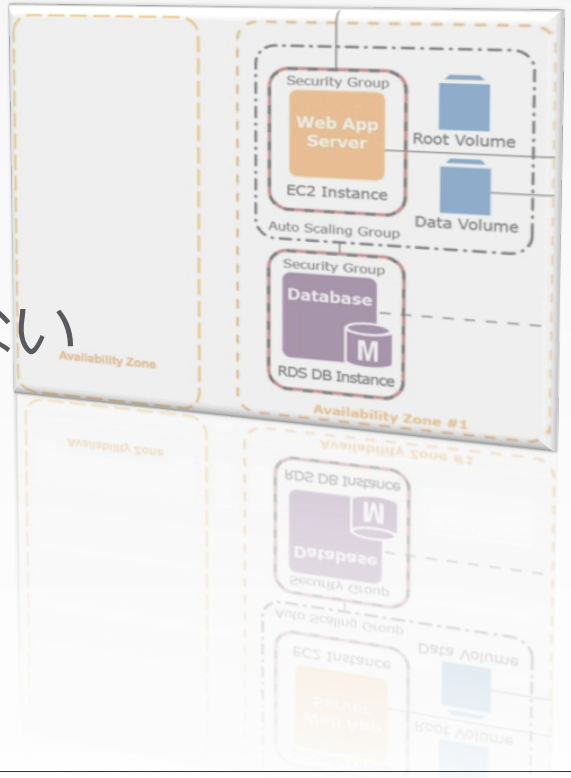
片寄せアンチパターン

マルチアベイラビリティゾーンがAWSの基本です。



片寄せアンチパターン

- 原因
  - AZ間の通信料金を気にする
  - 複数AZ運用ノウハウを単に知らない
  - レイテンシを過剰に気にする
- 症状
  - トラブル時にあわてることになる



片寄せアンチパターン

- 解決法
  - ELB,RDSなど複数AZ便利に使うサービスの導入
  - 同様の機能をもう一つのAZにコピー

単機能AZアンチパターン

マルチAZになっているようでなっていないパターン

単機能AZアンチパターン

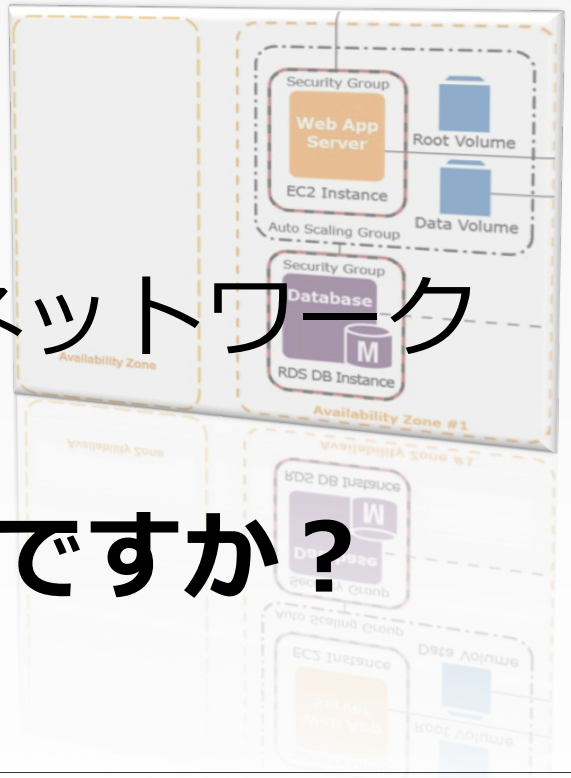
- 原因
  - サブネット毎に「DB用」「アプリ用」などと分けてしまい、それぞれを複数設置しわすれる
- 症状
  - 特定の機能が単一のAZにしか無いので、そのAZが障害を起こすとシステム全断

単機能AZアンチパターン

- 解決法
  - ELB,RDSなど複数AZ便利に使うサービスの導入
  - 機能別にサブネットを分けたら、そのサブネットは同じものを別AZにも作成する

オンプレ構成なぞりすぎネットワーク
アンチパターン

そのネットワーク、必要ですか？

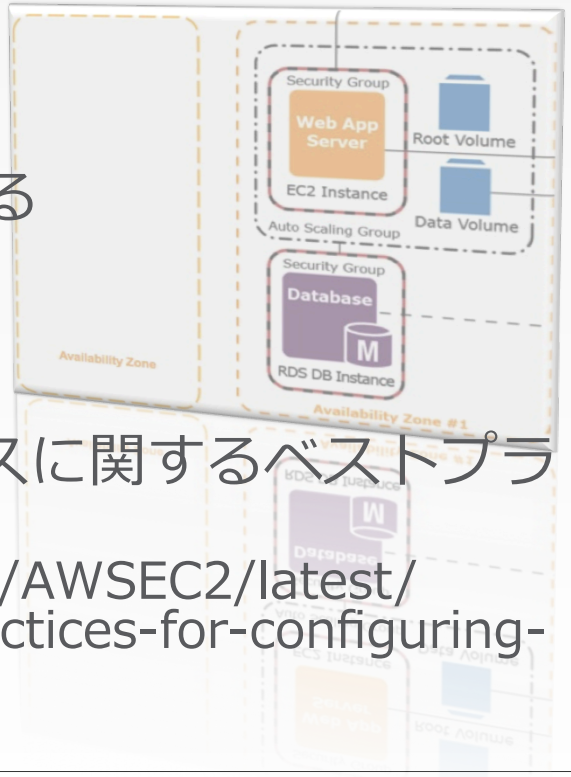


オンプレ構成なぞりすぎネットワークアンチパターン

- 原因
  - バックアップLAN、監視LANなどをENIを複数使ってオンプレと同じことを再現試行
  - OS側設定の不十分
- 症状
  - 通信出来ず
  - パフォーマンス不足

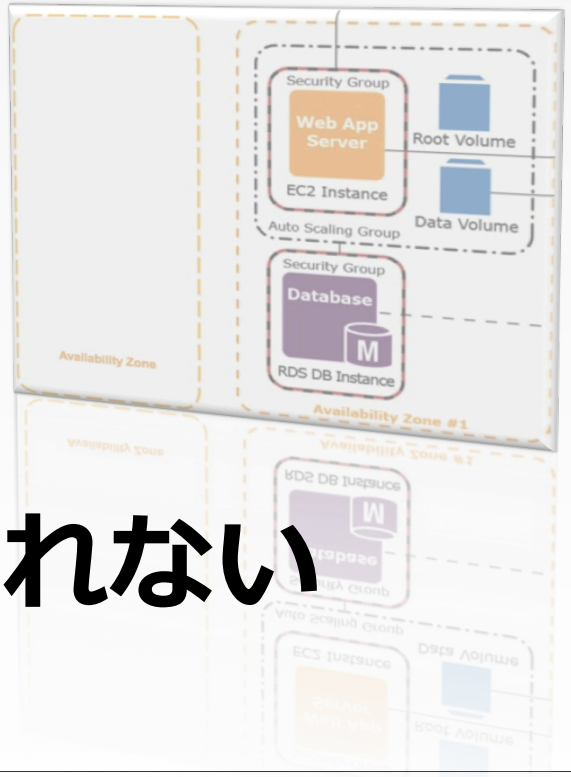
オンプレ構成なぞりすぎネットワークアンチパターン

- 解決法
  - ENI追加時はOS側も相応の設定をする
  - パフォーマンス対応
    - EBS Optimizedオプション
    - インスタンスタイプの変更
  - 参考：ネットワークインターフェースに関するベストプラクティス
    - http://docs.aws.amazon.com/ja_jp/AWSEC2/latest/UserGuide/using-eni.html#best-practices-for-configuring-network-interfaces



メール送信
アンチパターン

メールだけがうまく送れない

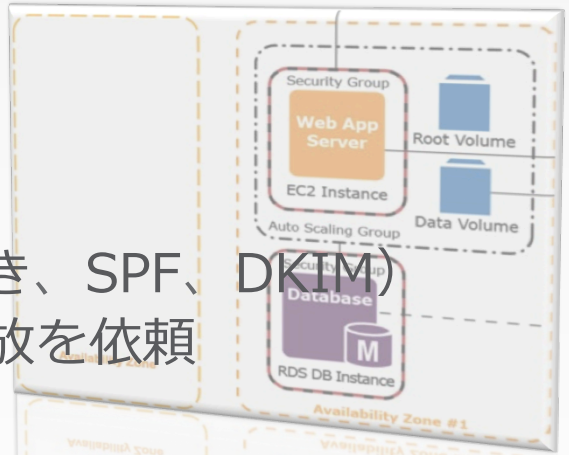


メール送信アンチパターン

- 原因
  - SPAMメールリレー防止機能
  - DNS設定不足
  - ブラックリスト
- 症状
  - メールの送信だけができない

メール送信アンチパターン

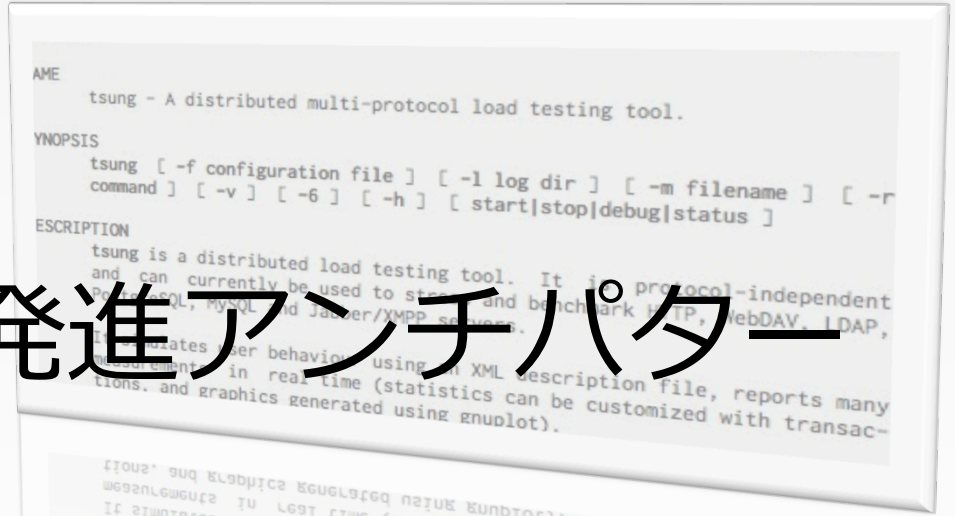
- 解決法
  - Amazon Simple Email Serviceの使用
  - EIPを取得し、DNS設定（正引き、逆引き、SPF、DKIM）
  - DNS逆引きとSMTPアウトバウンドの開放を依頼
  - 参考：
    - SES http://www.slideshare.net/AmazonWebServicesJapan/aws-black-belt-tech-amazon-ses
    - AWSからのメール送信 http://www.slideshare.net/AmazonWebServicesJapan/aws-30934799



キャパシティにまつわるアンチパターン

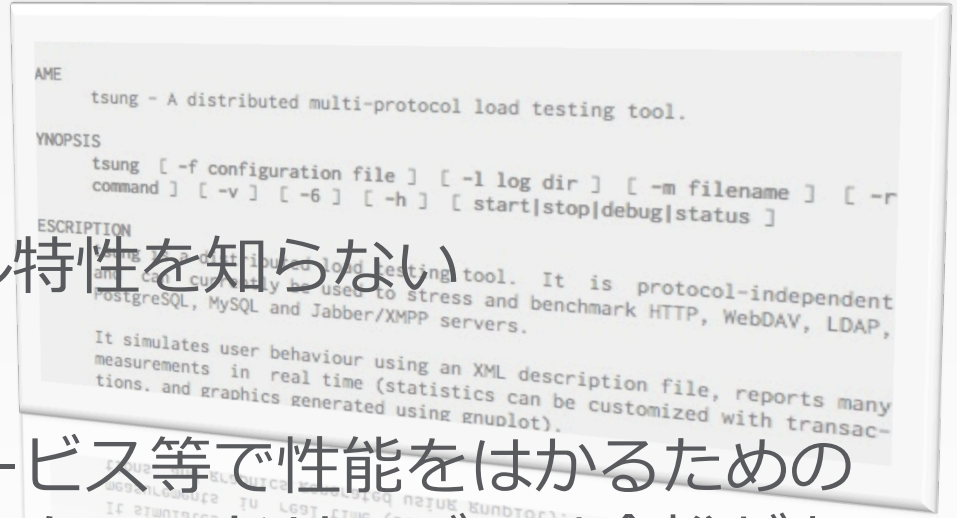
負荷テスト急発進アンチパターン

ELBはELBのあるように動作



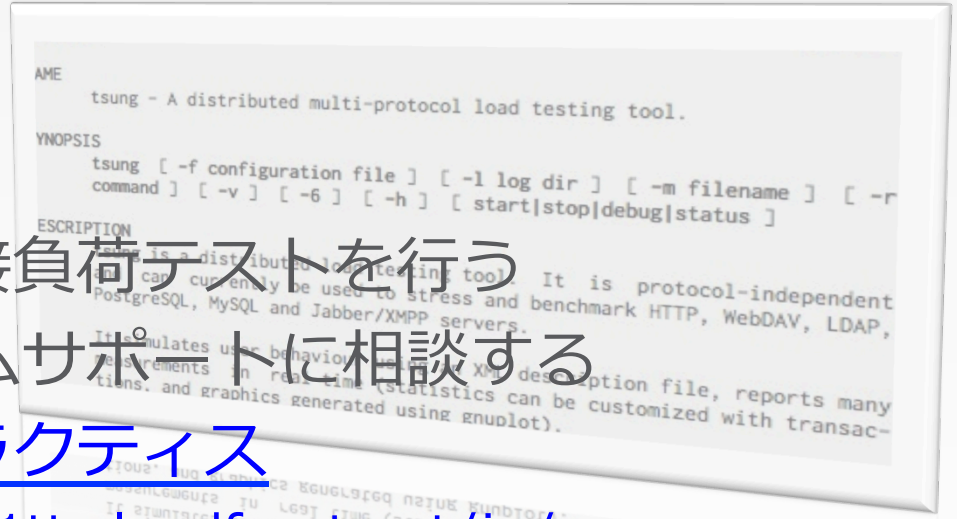
負荷テスト急発進アンチパターン

- 原因
  - ELBのオートスケール特性を知らない
- 症状
  - ELBを使ったWebサービス等で性能をはかるための試験を行う際に、バックエンドサーバには余裕があるのに、クライアントには503が届く



負荷テスト急発進アンチパターン

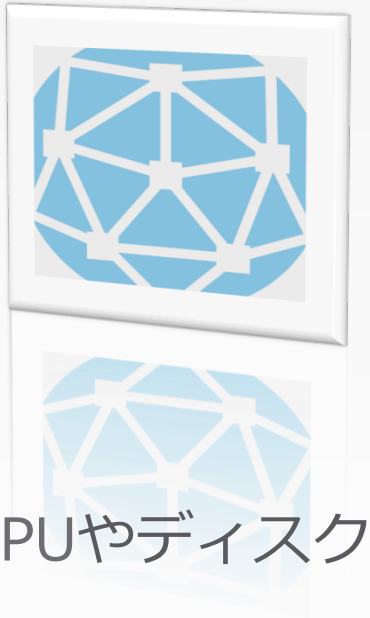
- 解決法
  - ELB下のサーバに直接負荷テストを行う
  - テスト前にプレミアムサポートに相談する
  - 参考：ELBベストプラクティス
    - http://d36cz9buwru1tt.cloudfront.net/jp/documentation/BestPracticesInEvaluatingELB-ja-final.pdf



CloudFront使わないアンチパターン

日本だけでも使います

CloudFront使わない
アンチパターン

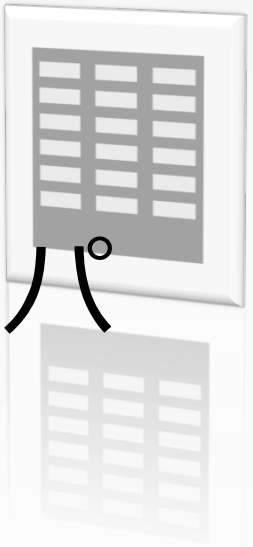


- 原因
  - 配信先が日本だけなのでいらないと考える
  - キャッシュ設定を嫌う
- 症状
  - HTTPアクセスピーク時に帯域が律速に。CPUやディスクに余裕があるのにサービスできず
  - S3のレスポンス（200-500msec）が遅いと文句を言う

CloudFront使わない アンチパターン

- 解決法
  - CloudFrontのレスポンスを体験してみる
  - オリジンサーバのキャッシュ制御を適切に設定する
  - 参考：CloudFrontのAWSマイスターシリーズ
    - http://www.slideshare.net/AmazonWebServicesJapan/20131120-aws-medisterregeneratecloudfrontetspublic/11　（キャッシュコントロールについて）

オンプレキャパプラアンチパターン



5年分の確保はいりません

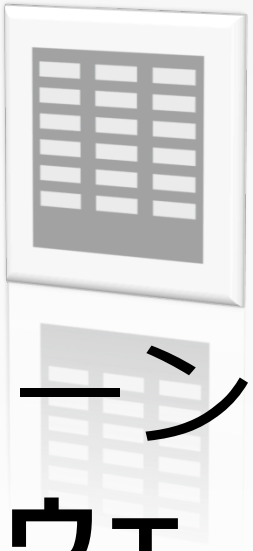
オンプレキャパプラアンチパターン

- 原因
  - システムの利用予定期間内ずっと使えるだけのキャパシティを確保
- 症状
  - 5年先に必要だからといって。。
    - EBSのボリュームを最大の16TB確保する
    - Provisioned IOPSを20000にする
    - 大きすぎるインスタンスを使用する

オンプレキャパプラアンチパターン

- 解決法
  - 使用するリソースを順次拡大することを設計に盛り込む
  - 使用するリソースは定期的に見直しする

ベンチマークアンチパターン



実アプリとベンチマークソフトウェアの乖離に注意

ベンチマークソフトウェアアンチパターン

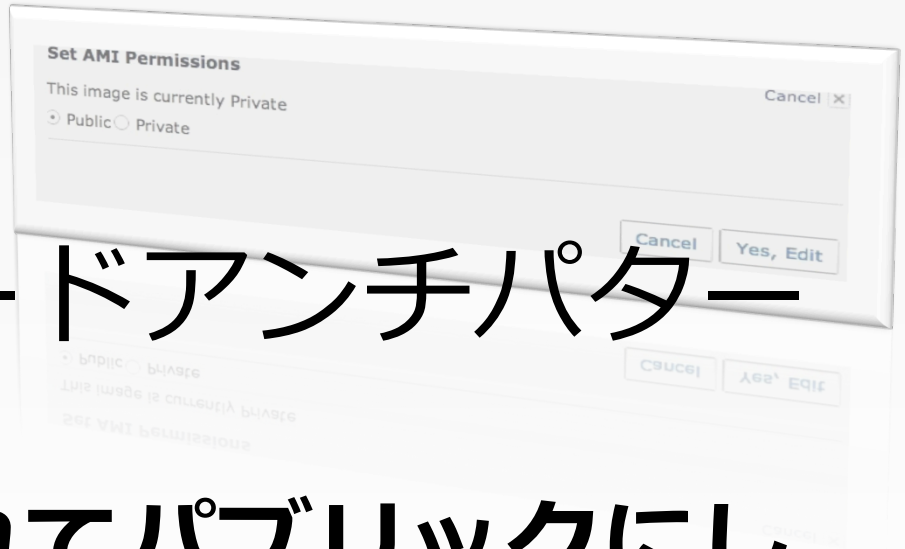
- 原因
  - システム実態と違うベンチマークソフトウェアによる測定値を使ったサイジング
  - 本番と測定時の規模の差
- 症状
  - パフォーマンス不足
  - コスト過大

ベンチマークソフトウェアアンチパターン

- 解決法
  - 本番システムと全く同じシステムを一時期的にAWSでは確保できるので、それでパフォーマンスを測定する

セキュリティにまつわるアンチパターン

キーインクルードアンチパターン

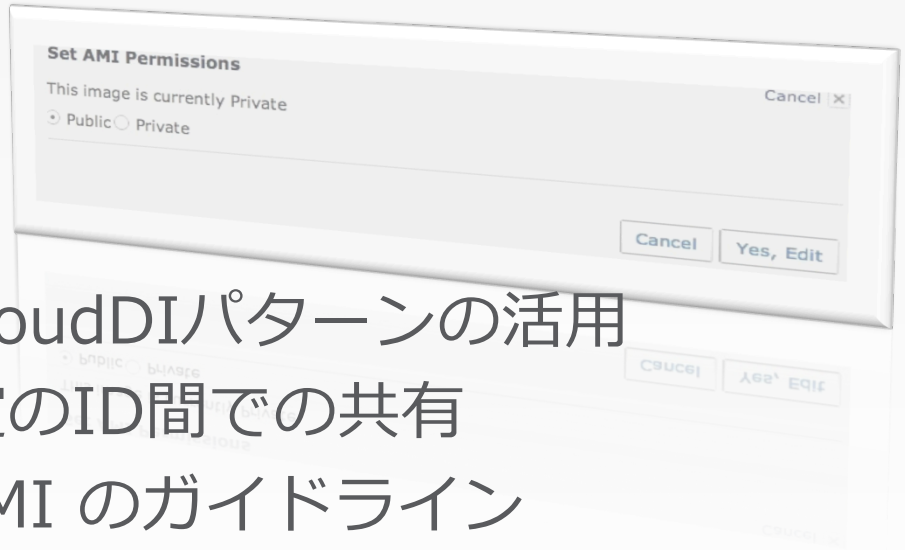


AMIにキーをいれてパブリックにしてしまう

キーインクルード アンチパターン

- 原因
  - AMI作成時の削除忘れ
  - 特定IDのみと共有できることを知らない
- 症状
  - ssh鍵がのこっている
  - Credentialがのこっている

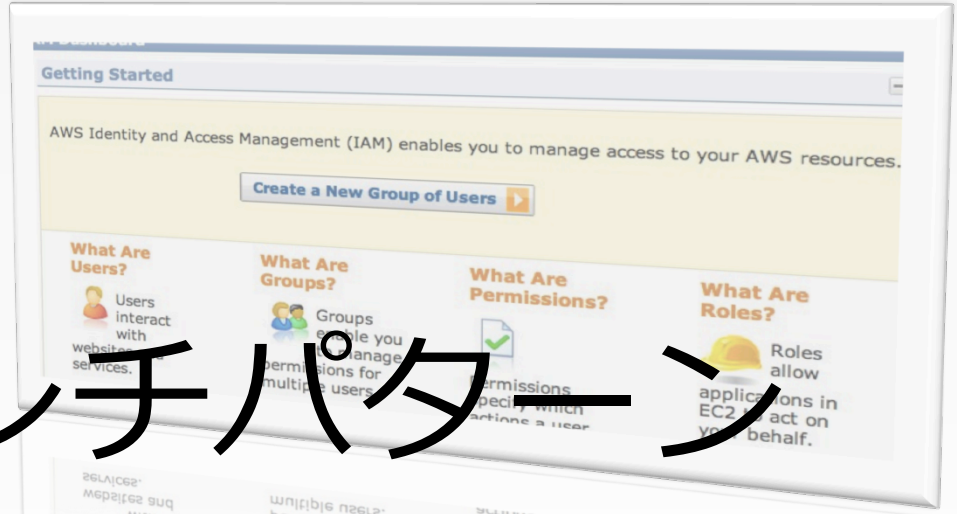
キーインクルード
アンチパターン



- 解決法
  - IAM roleの活用
  - S3から反映させる：CloudDIパターンの活用
  - AWSIDを指定した特定のID間での共有
  - 参考：　共有 Linux AMI のガイドライン
    - http://docs.aws.amazon.com/ja_jp/AWSEC2/latest/UserGuide/building-shared-amis.html

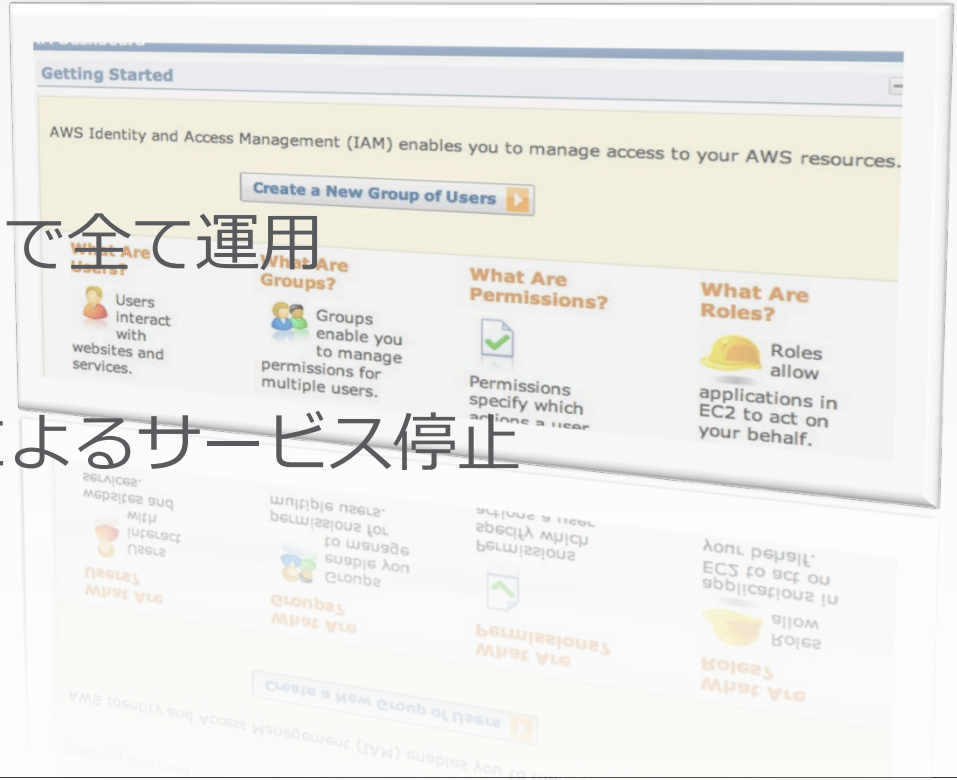
フルrootアンチパターン

昔はこれが当たり前でした



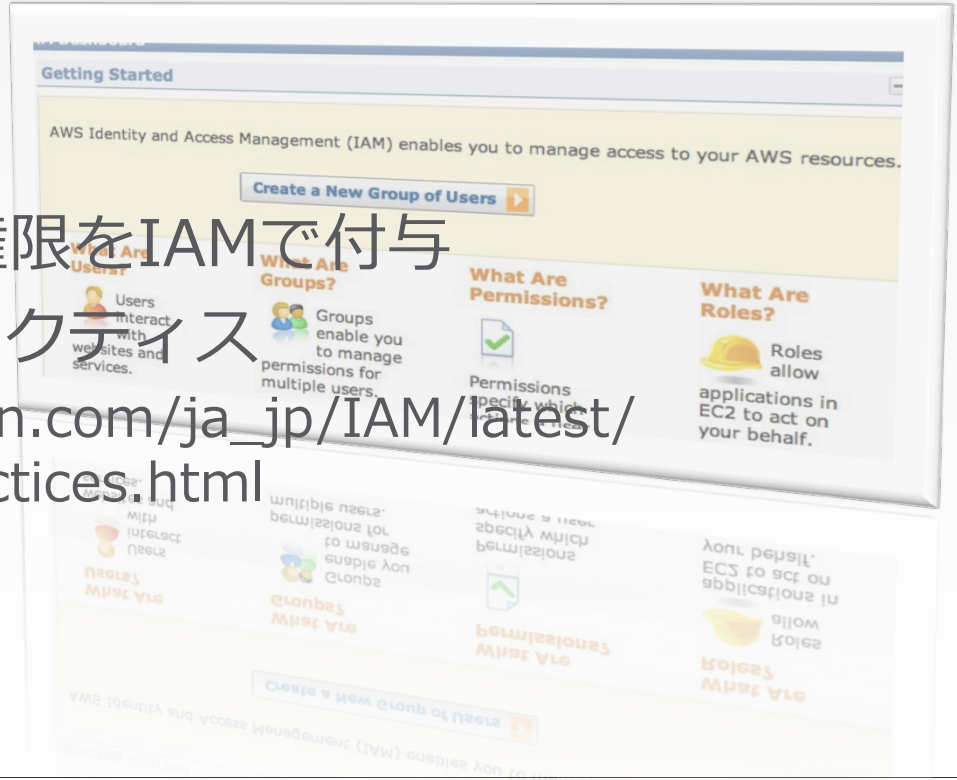
フルrootアンチパターン

- 原因
  - IAMを使わずに同一キーで全て運用
- 症状
  - （意図しない）ユーザによるサービス停止

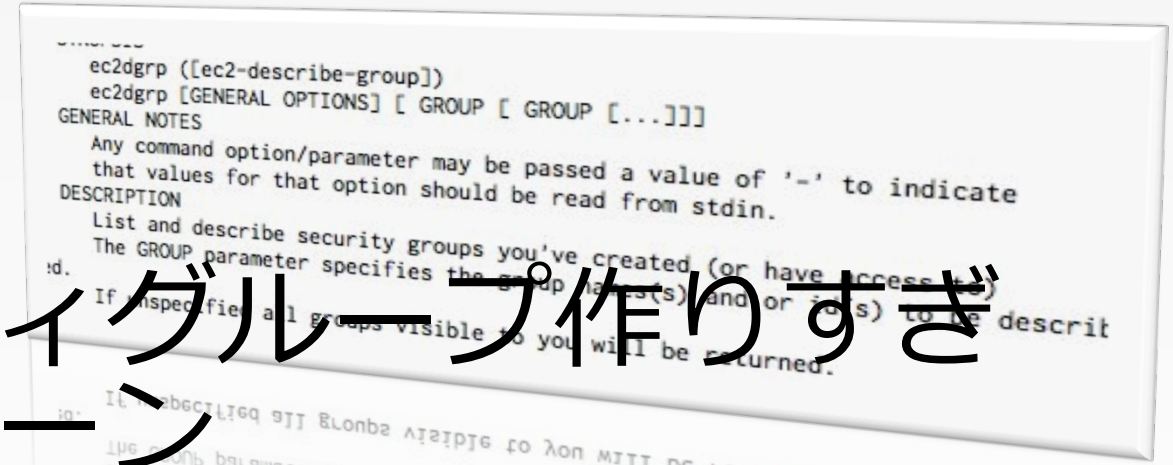


フルrootアンチパターン

- 解決法
  - 目的に合わせて必要な権限をIAMで付与
  - 参考：IAMのベストプラクティス
    - http://docs.aws.amazon.com/ja_jp/IAM/latest/UserGuide/IAMBestPractices.html

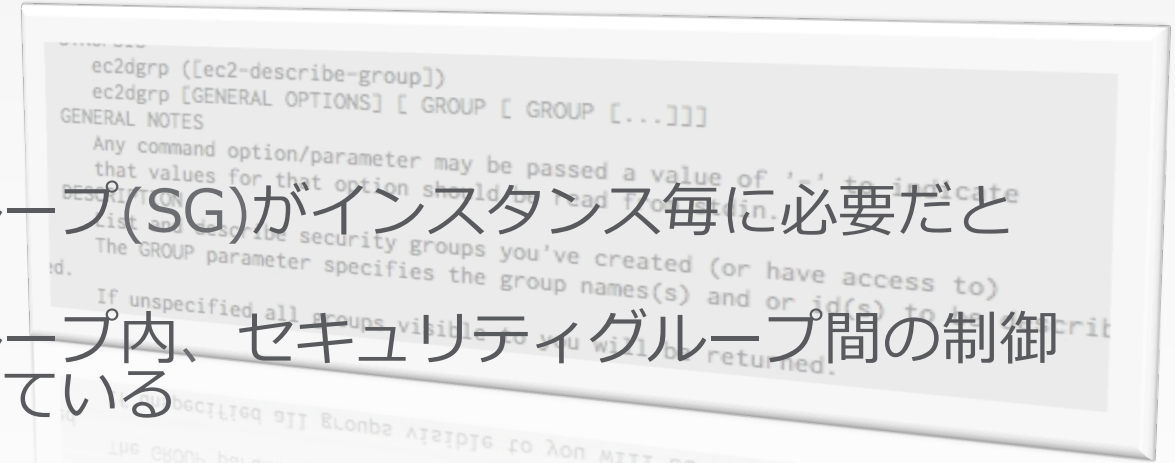


セキュリティグループ作りすぎアンチパターン



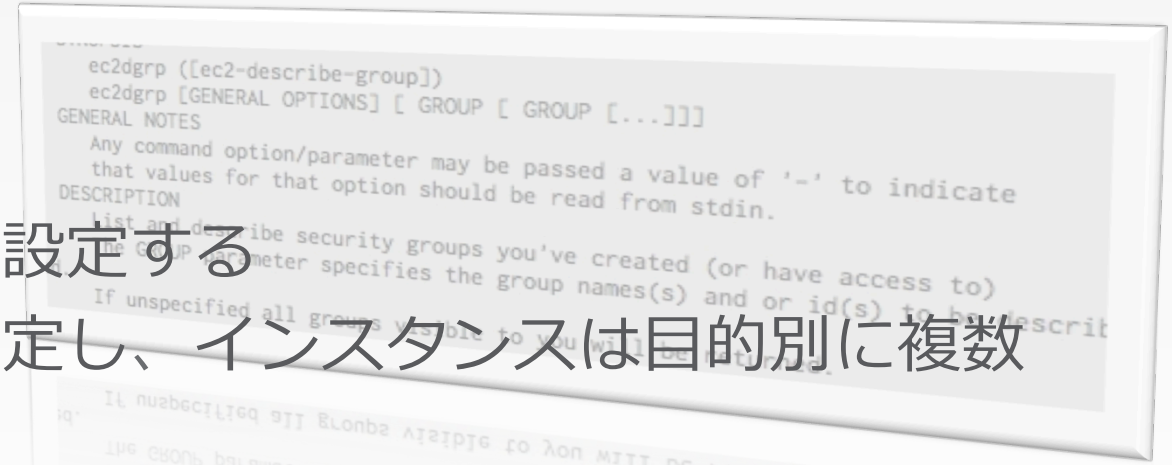
VPCだと特に楽です。

セキュリティグループ作りすぎ アンチパターン

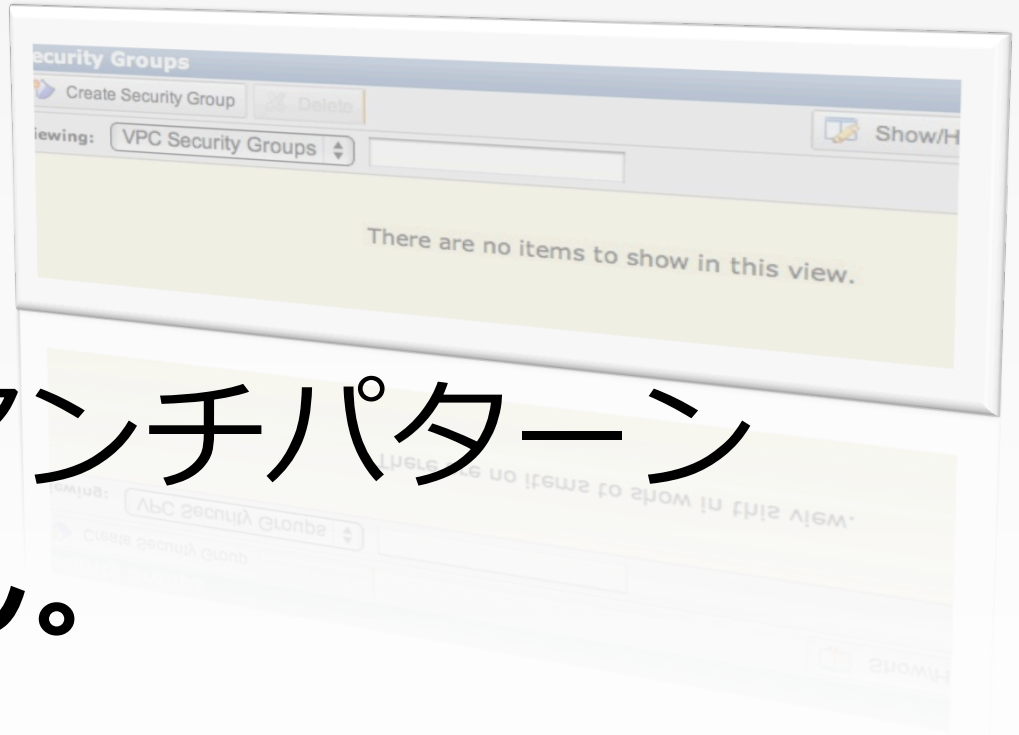


- 原因
  - セキュリティグループ(SG)がインスタンス毎に必要だと思っている
  - セキュリティグループ内、セキュリティグループ間の制御ができないと思っている
- 症状
  - SG数が多すぎて整理できなくなり、意図する通信を阻害する
  - あるいは意図せぬ漏洩するような設定となる。

セキュリティグループ作りすぎアンチパターン



- 解決法
  - SG名を通信元に設定する
  - 目的別にSGを設定し、インスタンスは目的別に複数SGを選択する
  - 参考：VPCのセキュリティグループ
    - http://docs.aws.amazon.com/ja_jp/AmazonVPC/latest/UserGuide/VPC_SecurityGroups.html

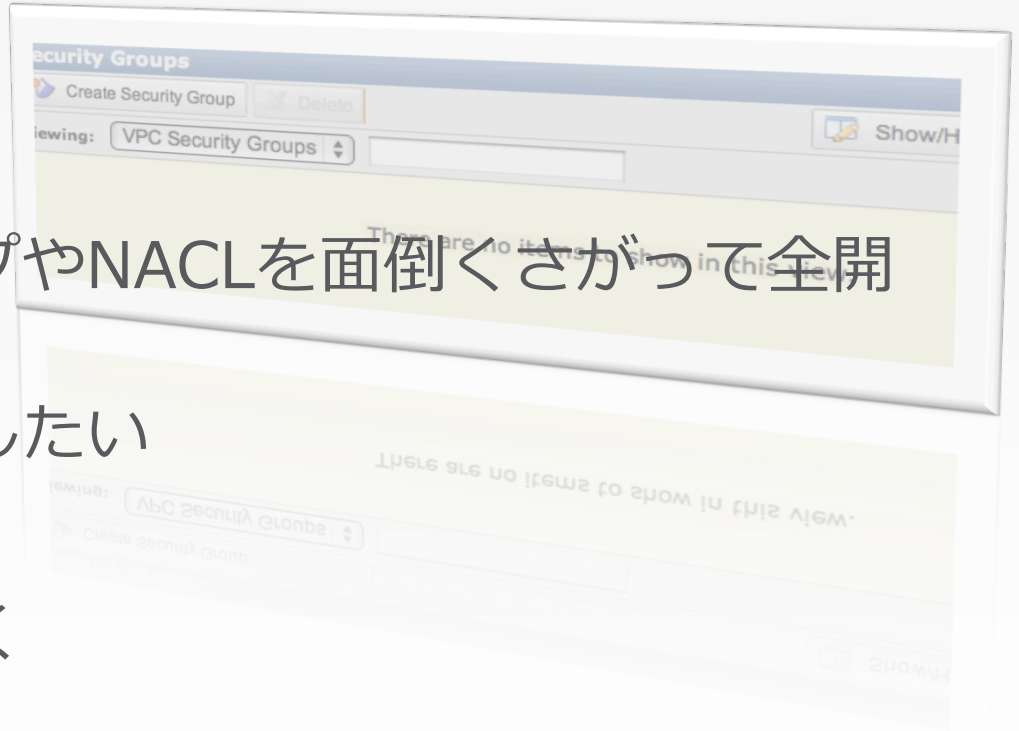


ノーガードアンチパターン

作戦ならばよし。

ノーガードアンチパターン

- 原因
  - セキュリティグループやNACLを面倒くさがって全開けにしてしまう
  - アタックログを記録したい
- 症状
  - 思わぬ攻撃で気がつく



ノーガードアンチパターン

- 解決法
  - 本番環境、開発環境を分離する
  - 本番環境で必要な通信をみなおす
  - Trusted Advisorによってサポートも
  - 参考：VPCのセキュリティグループ
    - http://docs.aws.amazon.com/ja_jp/AmazonVPC/latest/UserGuide/VPC_SecurityGroups.html

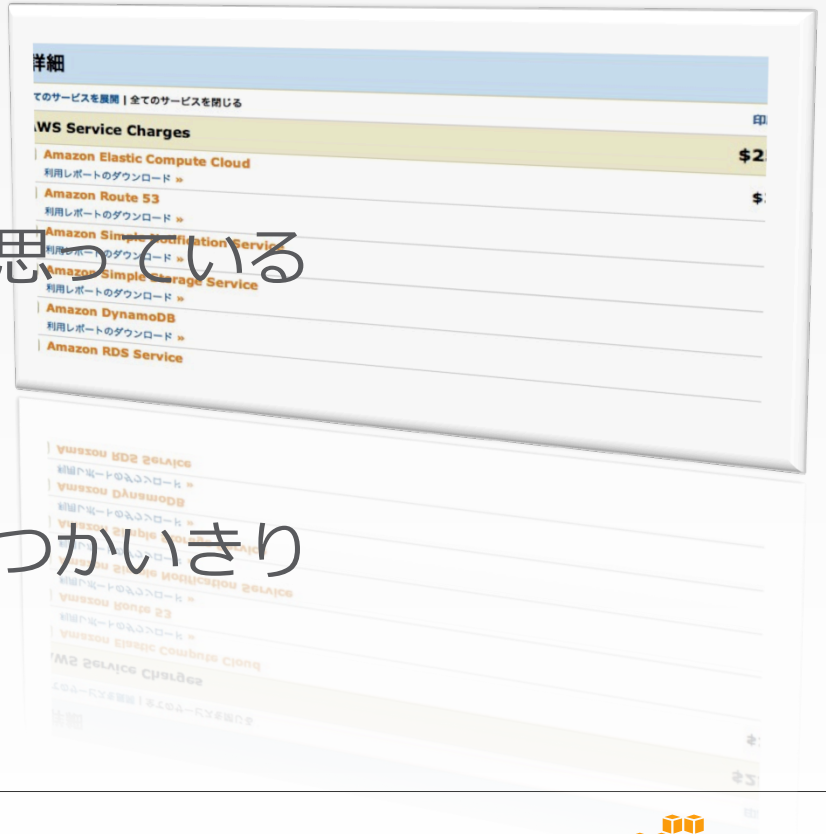
その他のアンチパターン

ノールック明細アンチパターン

明細は適宜確認しましょう

ノールック明細アンチパターン

- 原因
  - 月末の料金請求しかないと思っている
- 症状
  - 支払い周期のずれ
  - クレジットカード与信額のつかいきり



ノールック明細アンチパターン

- 解決法
  - 「AWSサポート新規契約」「RI購入」の際に請求書がわかれる
  - RI購入は契約時の購入費用分のみ / サポート契約は初月分のみ請求書が分かれる
  - Consolidated Billing 使用の場合、子アカウントがRIを購入しても親アカウント宛に請求書が届くことになるので注意

消化不良アンチパターン

つまみぐいサイコーです。

消化不良アンチパターン

- 原因
  - 全機能を使おうとして消化不良になる
- 症状
  - AutoScalingを無理やり使おうとする
  - ENIを設定したもののOSの設定をしない

消化不良アンチパターン

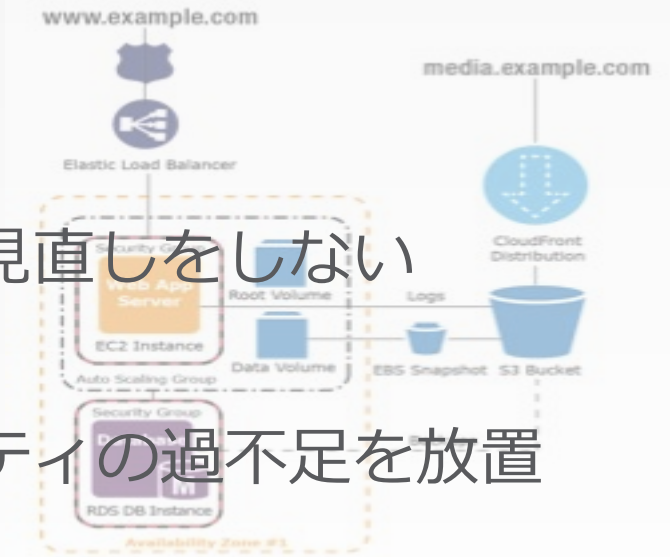
- 解決法
  - システムが動くことが最優先であることを思い出す
  - AWSが提供する機能は、その機能を要求のうちの最大公約数的なもの
    - 自分のシステムに最適かどうかはわからない
    - そもそも、その機能いりますか？

インフラ塩漬けアンチパターン

成長するクラウド。成長するビジネス。

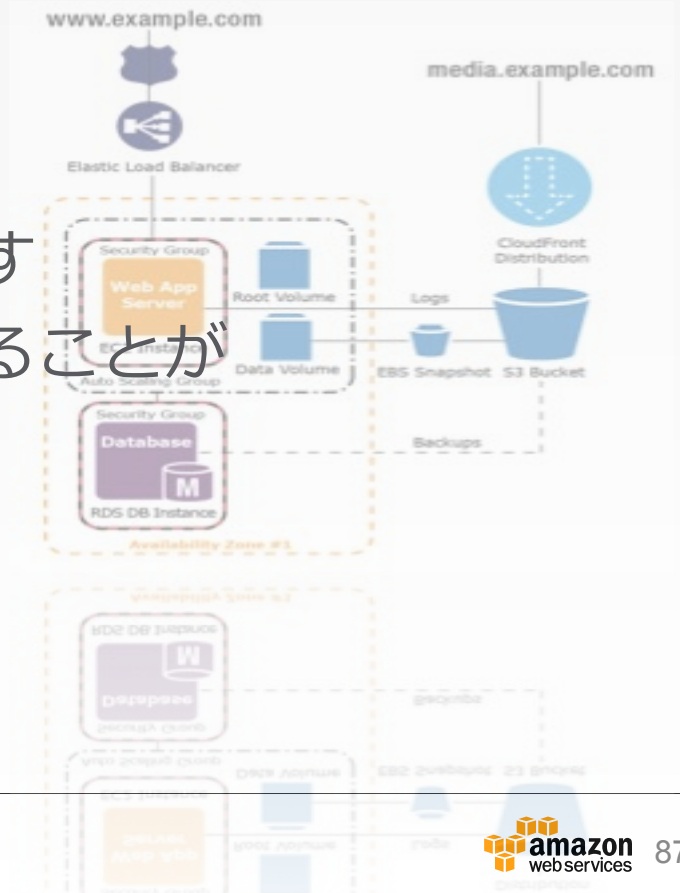
インフラ塩漬けアンチパターン

- 原因
  - 構築した当初のままインフラの見直しをしない
- 症状
  - 実際の利用にくらべてキャパシティの過不足を放置したまま利用している
  - 一時凌ぎで選んだサービスをそのまま使い続けている



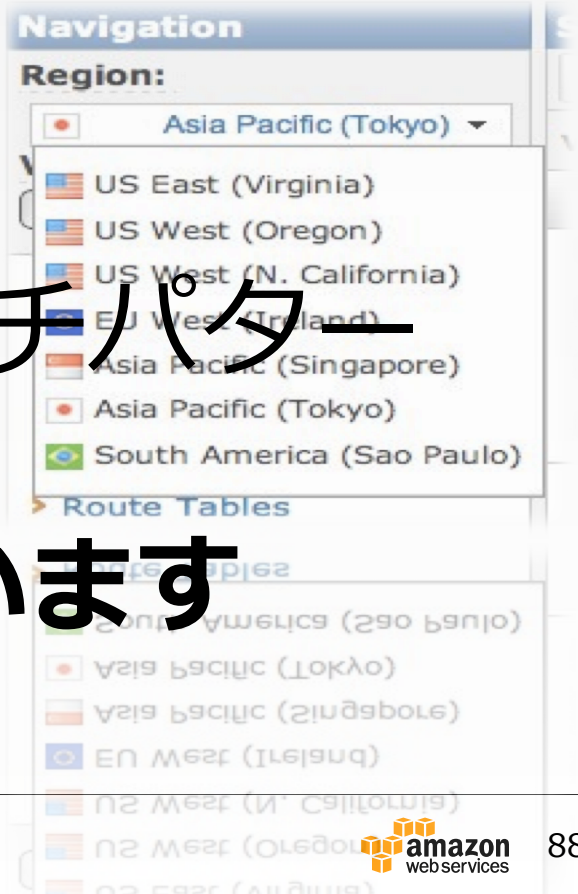
インフラ塩漬けアンチパターン

- 解決法
  - サービスは四半期に一度は見直す
  - 新サービスや新機能が助けになることが

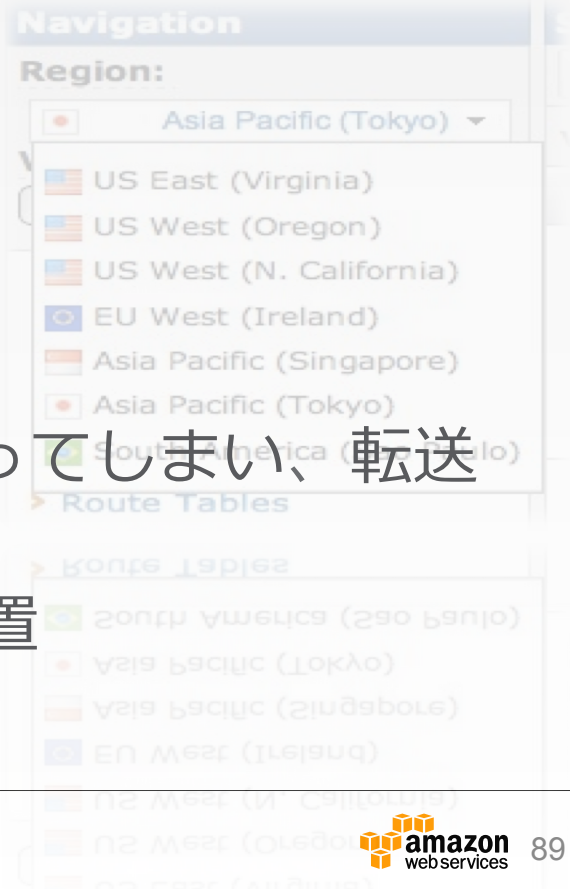


リージョン勘違いアンチパターン

リージョンは独立しています



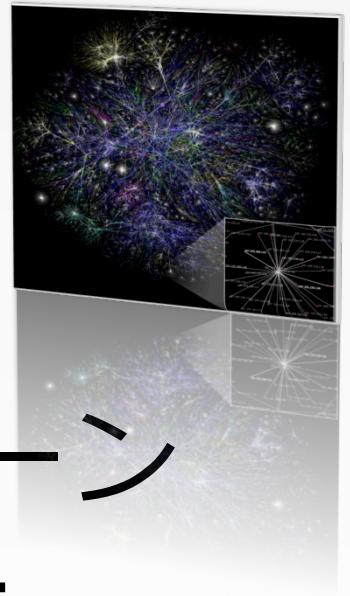
リージョン間違いアンチパターン



- 原因
  - リージョン選択を意識していない
- 症状
  - S3のバケットをUS–Standardに作ってしまい、転送速度や遅延に悩む
  - 不要なサービス、インスタンスの放置
  - sshの鍵が無くてログインできない

リージョン間違いアンチパターン

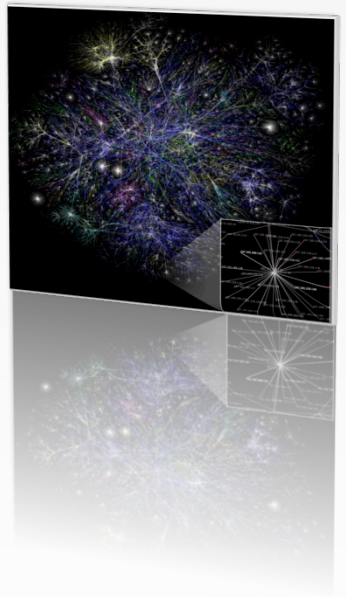
- 解決法
  - マネージメントコンソールではよく使うサービスをリージョン名を含めてブックマークしておく
  - 明細をときどきチェックする
  - 課金アラートを設定する
  - S3バケットは作りなおす
    - COPYすればバケット間で直接移行できる。



IPアドレス信者アンチパターン

4倍に増えたらどうするんでしょう？

IPアドレス信者アンチパターン

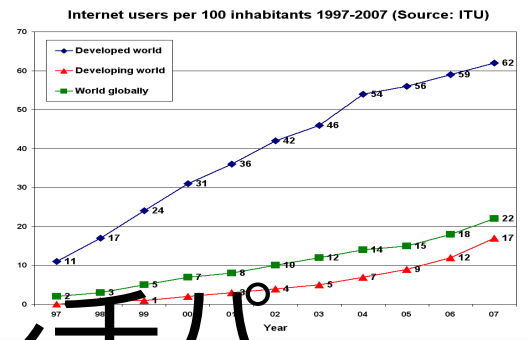


- 原因
  - IPアドレス固定環境への慣れ
- 症状
  - 全てのインスタンスにEIP
  - ELBでCNAMEのFQDNを使わずIPを指定する
  - RDSのエンドポイントを使わない
  - VPCでプライベートアドレスが固定されることを知らない

IPアドレス信者アンチパターン

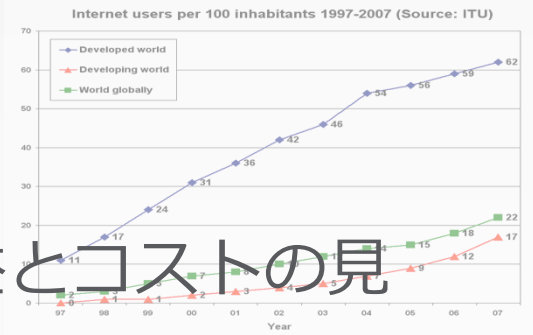
- 解決法
  - EC2以外のサービスを使うときはIPアドレスではないことを理解する
  - 利点を理解する

トラフィック心配性アンチパターン



AWSに向かうトラフィックは関係ありません

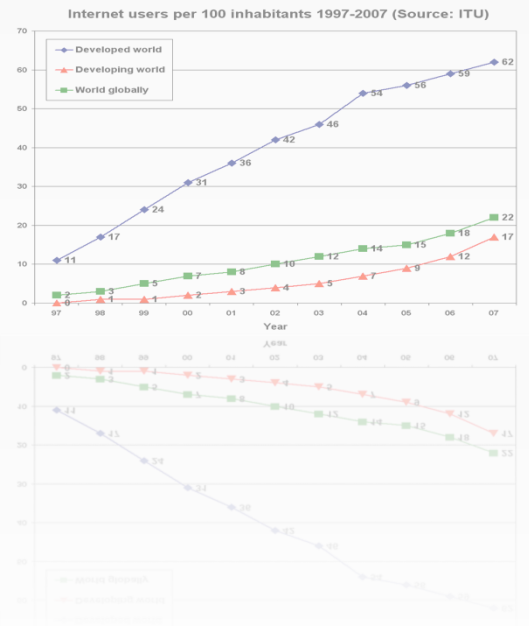
トラフィック料金心配性アンチパターン



- 原因
  - トラフィック、リクエストによる利益とコストの見積りができない
- 症状
  - 料金青天井だと心配しすぎる
  - ログを確認しない

トラフィック料金心配性アンチパターン

- 解決法
  - システムの特性をまず洗い出す
  - 料金の相談ができる会社の利用



机上の空論アンチパターン

JUST DO IT!

机上の空論アンチパターン

- 原因
  - サーバ発注、システムデプロイ、納品の硬直したループにはまっている
- 症状
  - 動作確認をしない
  - 事前のキャパシティプランニングに時間をかけすぎる

机上の空論アンチパターン

- 解決法
  - ともかく小さく試してみること

デザインを塗り替えるAWSの新機能には注意

まとめ

- CDPアンチパターンを活用し

システム規模に合わせた可用性を持つシステムを構築が可能に

低コストで耐障害性の高いシステムを簡単に構築することが可能に

システムが拡大しても、運用者の負担を削減する仕組みづくりが可能に

まとめ　(改善・革新)

改善

今まで**できていたこと**を、
より**早く**、**簡単に**、**安く**実現できる

革新

今まで**できなかったこと**が
実現できる

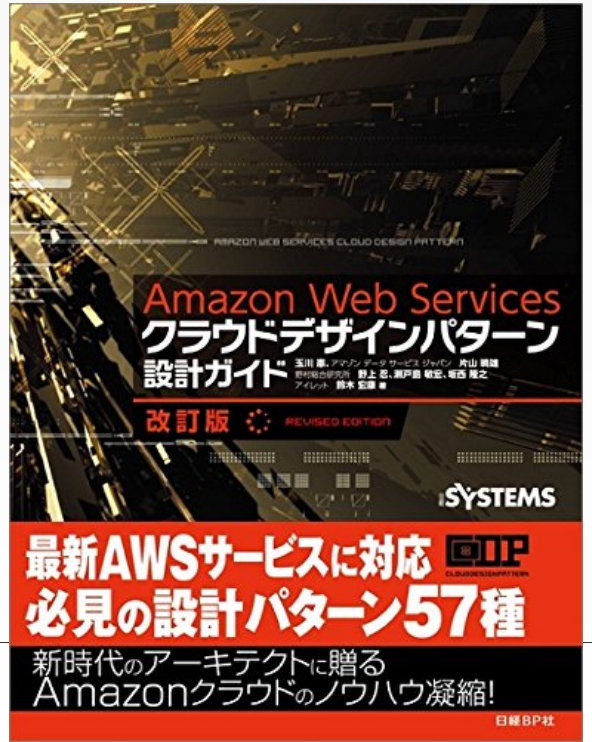
Webでノウハウを共有

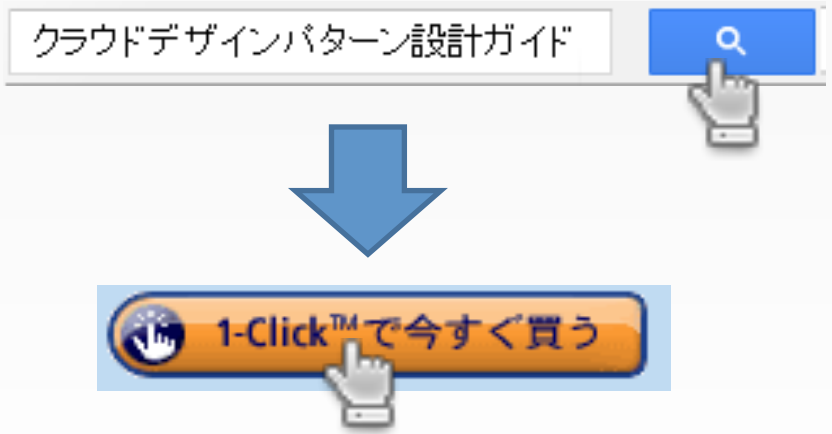
[WIKI http://aws.clouddesignpattern.org/index.php](http://aws.clouddesignpattern.org/index.php)

[FACEBOOK https://www.facebook.com/awscdp](https://www.facebook.com/awscdp)

書籍でノウハウを共有

Amazon Web Services クラウドデザインパターン 設計ガイド





http://www.amazon.co.jp/dp/4822277372/