畑 史彦
アマゾン ウェブ サービス ジャパン
ソリューションアーキテクト
2020/2/20作成

AWS INNOVATE 2020 オンラインカンファレンス

「クラウドで信頼性の高いシステムを構築するための障害や故障との向き合い方 - Face system failures with Chaos Engineering -」のクイズおよび解答

AWS INNOVATE 2020のセッションの視聴およびアンケートにご記入頂きありがとうございます。本資料が「**クラウドで信頼性の高いシステムを構築するための障害や故障との向き合い方 - Face system failures with Chaos Engineering -**」で出題されたクイズの解答になります。

Q1: 小さな障害の影響が想定以上に広範囲に波及してしまうことがあるのは、なぜでしょうか？

解答: 現代的なITシステムは分散システムとなる傾向が強いです。そして分散システムは本質的にその全体像を把握することが難しいです。また昨今では、ワークロードを適切な粒度に分割しその各々のコンポーネントを独立したサービスとして構築し相互にサービス間連携させることでシステム全体を構成する「**マイクロサービス・アーキテクチャ**」の採用が増えています。このマイクロサービス・アーキテクチャにおいては、一方のマイクロサービスから見て対向のマイクロサービスは、インターフェイスを除く内部アーキテクチャをブラックボックスとして捉えます。そのため、個々のサービスの障害やバグの影響が、連携するその他のマイクロサービスに対してどのように影響するか捉えるのがよりいっそう困難になりがちです。

「本番環境と開発やステージングの環境は本質的に異なる」という字義通りに解釈するとあたりまえですが、テストや障害への向き合い方においては目をそむけられがちな事実も重要なポイントです。開発環境で十分に結合テストされ障害テストもクリアしていたとしても、本番環境では扱うトラフィックも負荷状況も異なりますし、システム間連携するサービスの再現性（開発環境ではモックされていることが多い）はまちまちです。また、単純に本番環境とステージング環境の設定や構成がいつの間にかわずかに乖離してしまっている、といったこともよくある状況です。その結果、開発環境では限定的に確認された障害影響が本番環境では重大化してしまう、ということが起きます。

Q2: Chaos Engineering はどういった業界やシステムで採用が進んでいるでしょうか？
例）コンテンツ配信サービス、ECサイト、金融基幹システム、など。

解答: Chaos Engineeringは、業界やワークロードを問わず、その考え方とプラクティスが広く注目さ

畑 史彦
アマゾン ウェブ サービス ジャパン
ソリューションアーキテクト
2020/2/20作成

れ採用されています。セッション内でもご紹介した **Netflix** 社のようなコンテンツ配信の業界においては早くより注目され採用事例の公開も増えています。この例としてセッション内ではAWS re:Invent 2018でのDAZN様の登壇資料をご紹介致しました。またAmazon社内では、**Amazon Prime Video**や**Audible.com**においてシステムがResiliencyやScalabilityを確保するためにどのようにChaos Engineeringを活用しているかをAWS re:Inventにおいて情報発信しております。その他の業界に目を向けると、サービスのダウンタイムが収益にダイレクトに影響するECの業界での採用の公開情報が多いです。また2018年10月に金融庁が公開している金融機関が取り組むべき方針に「脅威ベースのペネトレーションテスト（**TLPT**: Threat Led Penetration Test）」が加えられました。ECでの採用事例の紹介や、このTLPTとChaos Engineeringの類似点などの考察は以下の記事や資料もご覧ください。

- Chaos Conf 2019 recap 勉強会 | Amazon Web Services ブログ
  https://aws.amazon.com/jp/blogs/news/chaos-conf-2019-recap-aws-loft-tokyo/
- What is suitable for Chaos Engineering? ~ChaosConf 2019 recap~ - Speaker Deck
  https://speakerdeck.com/fumihiko/what-is-suitable-for-chaos-engineering-chaosconf-2019-recap

参考資料URL

- AWS re:Invent 2018: Chaos Engineering and Scalability at Audible.com (ARC308) - YouTube （英語）
  https://www.youtube.com/watch?v=7uJG3oPw_AA

Q3: 本番環境に障害を注入（Failure Injection）しつつ本番環境の無事を担保するにはどうすればよいでしょうか？

解答: Chaos Engineeringでは、利用状況や負荷状況、データ量、外部システム連携などが本番環境と乖離する開発環境においてExperimentを実施するのではなく、本番環境自体でExperimentを実施することを推奨しています。その結果、本番環境に注入された障害が大きなサービス影響などをもたらす可能性があります。もちろんこの事実自体はリスクであり、これを完全に防ぐ絶対的な方法もありません。しかしながら、リスクは得られるメリットとのトレードオフにおいて評価されるため、このリスクを最小限に抑えるために考慮すべきポイントをいくつかご紹介します。まず１つ目は、対象システム自体の可用性や信頼性の設計が十分になされていることが重要です。確認すべき信頼性がそもそも始めから考慮されていないシステムで障害が注入すればあたりまえですが単純に障害が拡大するだけです。これについては、**AWS Well Architected Framework**や**AWS Builders Library**などが役に立ちま

畑 史彦
アマゾン ウェブ サービス ジャパン
ソリューションアーキテクト
2020/2/20作成

す。

- AWS による優れた設計 – 安全で効率的なクラウド対応アプリケーション
  https://aws.amazon.com/jp/architecture/well-architected/

- The Amazon Builders' Library
  https://aws.amazon.com/jp/builders-library/

次に、障害の影響範囲 (**Blast Radius**: 直訳すると「爆発半径」) の最小化という考え方が知られています。例えば、障害を注入する際にその対象トラフィックを限定したり、対象インスタンスやコンテナを限定したりします。３つ目は、何かあった場合に障害の注入をすぐに中断できるようにしておくことです。障害の注入自体が想定以上の影響を出してしまう場合はもちろんですし、関係ないところで障害やバグが顕在化している状況でも、障害の注入が中止できるような包括的な対応が望ましいです。

- Principles of Chaos Engineering
  https://principlesofchaos.org/?lang=JAcontent

４つ目は、目的が障害全般への対応にそれてしまいますが、仮に大きな障害が起きてしまっても速やかに対処できるように日頃から訓練しておく、というものです。障害への対応手順などは整備され確認されていることも多いですが、障害とは常に想定の範囲を超えることがありうるものです。実践的に複合的な障害への対応を訓練しておくことは非常に重要です。AWSでは**Gameday**という未知の障害への対応や運用自体の模擬訓練のワークショップをAWS re:Inventにおいて開催しています。AWS LOFT TOKYOにおいても不定期で開催しておりますので興味がある方はぜ一度参加をご検討ください。

- AWS GameDay - Tokyo
  https://awsgamedaymicroservicestokyo.splashthat.com/
- イベント・セミナー一覧 - AWS Loft Tokyo | AWS
  https://aws.amazon.com/jp/start-ups/loft/tokyo/events/