# はてなにおける
# メール基盤とDMARC対応

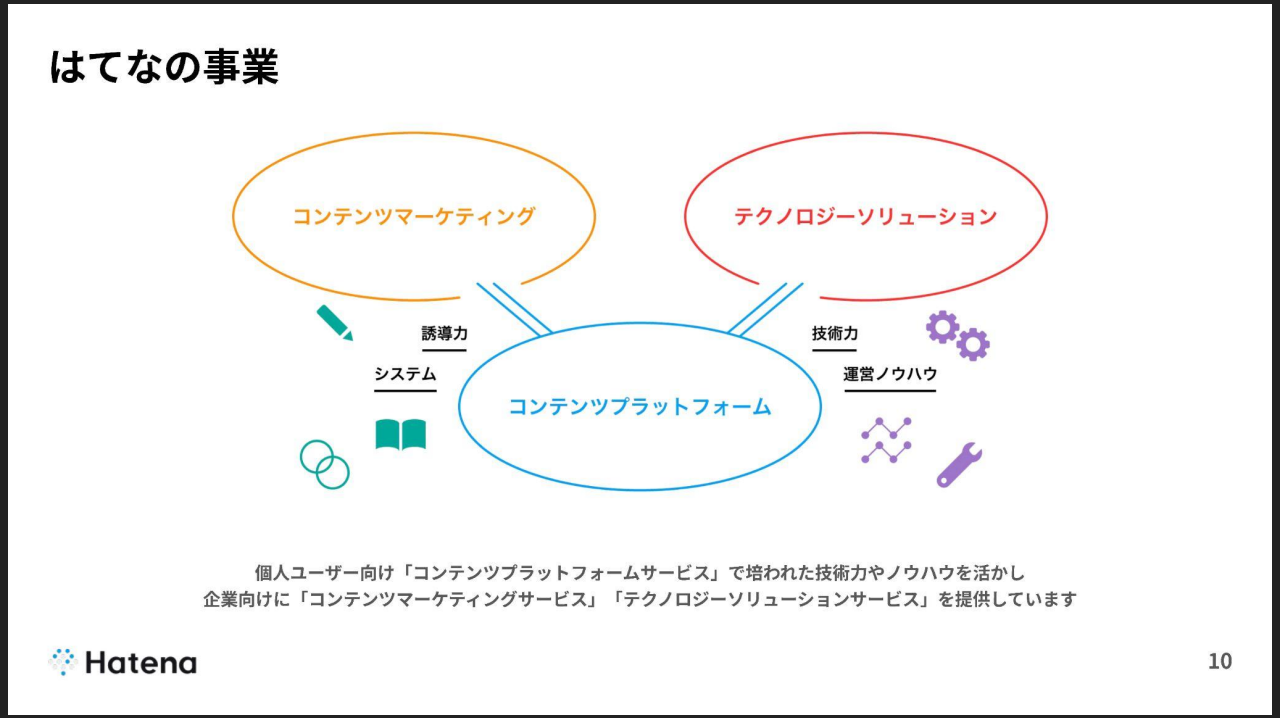
id:MysticDoll / @MysticDoll

2024/11/11 JPAAWG 7th General Meeting

# 自己紹介

- id: MysticDoll / 瀧ヶ平 充
- 株式会社はてな (2023/4〜)
  - システムプラットフォームチーム所属 Platform SRE
- 社内基盤の運用とかやり続けて7年目
  - 前職ではGHE運用やCI/CDツールの開発・運用とか
  - Screwdriver.cd Most Valuable Contributer (2021 CD Foundation Project Awards)

# はてなの事業 | 株式会社はてな 技術グループ紹介資料より




https://speakerdeck.com/hatena/engineers-recruitment?slide=10

# はてなの事業 | 株式会社はてな 技術グループ紹介資料より

## はてなの事業 | コンテンツプラットフォームサービス

**提供開始から10年を超える「はてなブログ」「はてなブックマーク」など、個人ユーザー向けコンテンツプラットフォームサービスを提供しています**




「はてなブログ」はユーザーの「書きたい」気持ちに応えるブログサービス。手軽に書きたい人も、しっかり書きたい人も満足できる便利な機能を備えています。シンプルかつモダンなブログサービスとして継続的な機能開発に取り組むとともに、検索流入に寄与する要件への対応や機能改善、「はてなブックマーク」との連携など、ユーザーの書いたコンテンツが読まれる仕組みも用意しています。




「はてなブックマーク」は国内最大級のソーシャルブックマークサービス。ニュースやブログのなかから気になった記事をクラウド上に保存し、ほかのユーザーと共有することができます。オンラインブックマークツールとしての利用だけでなく、時々刻々と移り変わるインターネットの人気ページや旬の話題が集まるメディアとしても多くのユーザーにご利用いただいています。

Hatena

11

https://speakerdeck.com/hatena/engineers-recruitment?slide=11

# はてなの事業 | 株式会社はてな 技術グループ紹介資料より

## はてなの事業 | テクノロジーソリューションサービス


### Mackerel 利用企業

KDDI / IDCFrontier / CyberAgent / NTT東日本 / classmethod
NTT SmartConnect / SEGA / mercari / レコチョク / SoftBank
Nintendo / BIGLOBE / docomo / SAISON CARD / GMOペパボ
nulab / アイテック阪急阪神 / カヤック / VALUE COMMERCE / DRECOM
DAC / GREE / セブン&アイ NetMedia / tv asahi mediaplex

### Mackerel（マカレル）

サーバーにおける各種ハードウェアやアプリケーションソフトウェアの性能をリアルタイムに監視することができるSaaS型サーバー監視サービスです。使いやすいUIと効率的なAPIによる総合的な監視体験と、より自動化されたインフラ基盤の構築を可能にするなど、その先進性により多くの大手企業からの採用実績を持っています。

サービスサイト（https://ja.mackerel.io/）より引用

Hatena

15

https://speakerdeck.com/hatena/engineers-recruitment?slide=15

# システムプラットフォームチームの仕事

- メール基盤
  - 今日はこれの話
- 入退職時のアカウント管理・運用
- GitHub/GitHub Enterprise管理
- AWSアカウントのControl Towerを使った管理
- etc...

# 今回話す内容

- DMARC基礎知識
- はてなのメール送信基盤の構成
- DMARCで対応すべきこと
  - 主にDKIMの話
- DMARCレポート可視化
- Postfixエラーログの監視

# GMailの新送信者ガイドライン

GMailに対して5000mail/day超を送るメール送信者に対し2024/2/1から以下の制約などが課された

- DMARCによってメールを認証すること
  - ドメインにSPF・DKIMを設定すること
- 簡単にsubscribeを解除できること
- Gmailへの送信時にTLSを利用すること
- 迷惑メール率を一定以下に維持すること

# 基礎知識

# DMARCとは

メールに関する認証結果により信頼性を判断する仕組み
管理するドメインのメールに対し以下を定義できる

- 認証失敗した際のメールの扱い
- 認証状況のレポートの送信先

DMARC自体は認証の仕組みではない

# DMARCで利用している認証方法

SPF

- 送信元IPアドレスを認証する仕組み
- 送信元ドメインに対してTXTレコードを設定する

DKIM

- メール本文・ヘッダを元にデジタル署名を作成する
- <selector>._domainkey.<domain>にTXTレコードを設定する

# SPF(RFC 7208)

受信者は以下の手順で送信元を認証する

- 送信元となるドメインを決める
  - メールのヘッダやSMTPの情報から決める
- 決定したドメインからSPFレコードを参照する
  - 対象となるドメインのTXTレコードを参照する
- 実際の接続元IPと参照したIPアドレス範囲を比較する

# DKIM(RFC 6376)

DKIMでは以下の方法でメールの送信者を認証する

送信者はメール本文・ヘッダの内容を元に

- 電子署名を作成
- DKIM-Signatureヘッダに記録
  - 署名に使用した秘密鍵に対応するセレクタ、署名に利用したヘッダなどを記録

受信者はメールのDKIM-Signatureヘッダの内容から

- 対応する公開鍵をDNSレコードから取得
  - <selector>._domainkey.<domain> のTXTレコードを参照する
- 署名を検証

# DMARCで必要な対応

# DMARCの対応でやるべきこと

自社ドメインで送信しているメール送信元を把握する

- はてなの場合は以下で送信している
  - 独自システム
  - SendGrid
  - HubSpot
  - Salesforce
  - Zendesk
  - Gmail(Google Workspace)など

# DMARCの対応でやるべきこと

SPF:

- 把握した送信元IPをSPFレコードに全て追加する
  - SaaSの場合は各社のサポートページ等に設定内容がある
  - 独自システムの場合はシステムの利用しているIGWを追加する

DKIM:

- 把握した送信元でに対応するDKIMレコードを追加する
  - SaaSの場合、設定ページなどから設定項目を取得する
  - 独自システムの場合はMTA毎に設定方法が異なる

# DMARCの対応でやるべきこと

SPF/DKIMの設定を確認するために以下を実施する

- ポリシーとレポート先をDMARCレコードに登録
- レポートを確認し、未知の送信元を把握する

# DMARCの対応でやるべきこと

まとめると

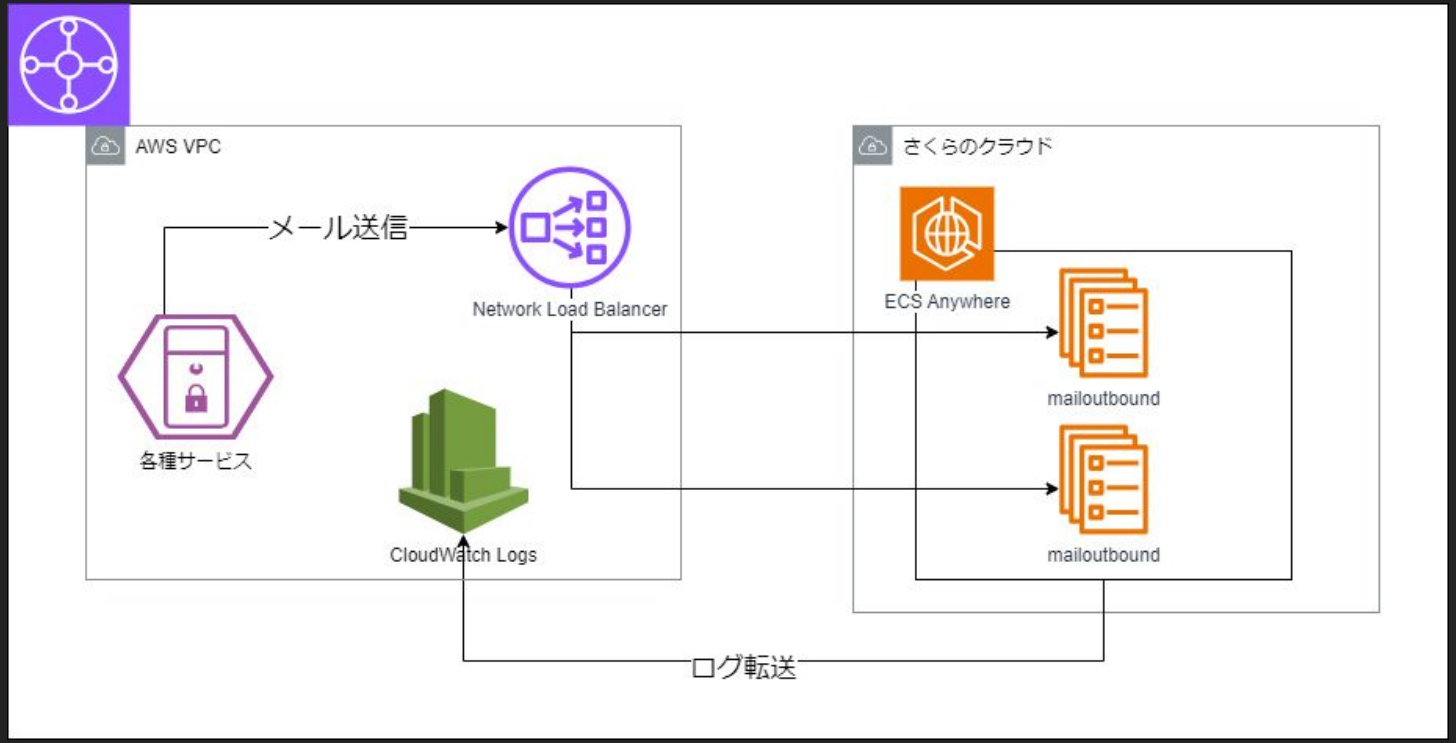
- 送信元を把握
- SPFを設定
- DKIMを設定
- DMARCレコードを設定して、漏れを潰していく

# はてなのメール送信基盤

# はてなのメール送信システム

- ECS Anywhereで管理・運用
  - さくらのクラウド上のコンテナ実行基盤を利用
- サービスからはNLBを経由してSMTPでアクセス
  - さくらのクラウドとAWS VPCはTGWで疎通可能となっている
  - コンテナからNLBのtarget groupへの登録の話は今回は割愛
- ECS TaskとしてPostfixが稼動

# はてなのメール送信システム



# はてなのメール送信システム

ECS Anywhereを利用する理由は主にIPレピュテーション

- AWSから払い出されるIPアドレスの信用問題
  - 一部のサービスはAWSのIPアドレスをbot判定している
  - 割り当てられたIPが過去に悪用されていた可能性がある
- 送信数の規模的にSaaSに置き換えが難しい

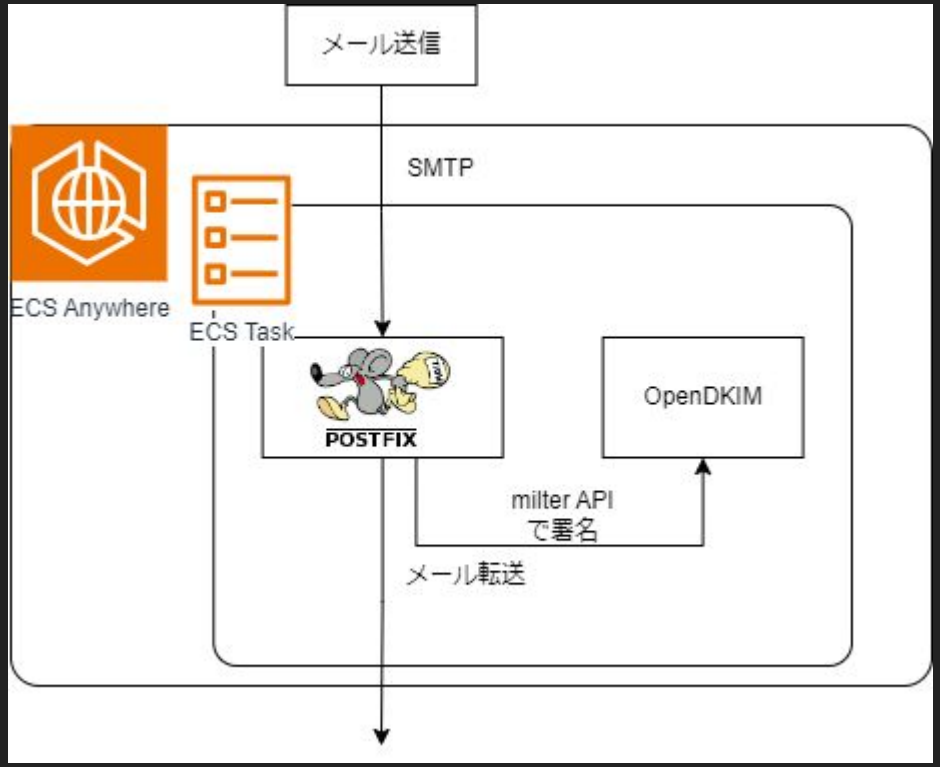
# DKIMの対応

# OpenDKIM

PostfixでのDKIM対応にはOpenDKIMを利用する

- PostfixとはmilterAPIを利用して連携する
- milter APIでの通信は以下どちらかで行う
  - Unix Domain Socket
  - TCP Socket
- 今回はsidecarとしてOpenDKIMコンテナを起動した
  - 以下の点を考慮しsidecarとして起動することにした
    - Postfixと同居させる場合コンテナ起動スクリプトが煩雑化する
    - 個別のタスクとLBを立てる場合アクセス制御を考える必要がある

# OpenDKIM sidecarの構成




# OpenDKIM sidecar

ECS Anywhereでは以下の点に注意が必要

- プロセス間でTCP通信する際はlinksの設定が必要
  - Fargateならこの問題はない
  - 厳密にはポートマッピングをすれば通信可能ではある
    - 外部のMTAから署名されたくないため今回はlinks設定を使用
  - linksを使う場合ネットワークモードがbridgeである必要がある

# OpenDKIMの設定

- 送信ドメインのDNSにDKIMのレコードを追加しておく
- opendkim.confに以下を設定する
  - 署名すべきドメイン (Domain)
  - 使う鍵ファイルのパス (KeyFile)
  - ListenするSocketの設定 (Socket)
  - Postfixへの送信元IPレンジ (InternalHosts)
- Postfixの main.cf で non_smtpd_milters にopendkimのSocketを指定

これらの設定はdocker-compose等でメール送受信環境を作って検証できる

- 仮にDNSを設定していなくてもDKIM署名ができていることは確認できる
  - その場合意味の無い署名ではあるが、DKIM署名の設定の確認には便利

# バウンスメールのDKIM署名

バウンスメールはデフォルトではmilterが適用されない

→ main.cf で internal_mail_filter_classes を設定

- bounceを指定する
- 迷惑メール率を下げるために考慮する必要がある
  - 不特定多数に送るシステムではバウンスメールが大量発生しうる
    - 受信側からレポートが送信されないことが保証できれば必要ない
  - はてなではバウンスメールをAmazon SESに送っている
    - Amazon SESからのDMARCレポートで発覚した

# DMARCレポートの可視化

# レポート可視化の構成

parsedmarc + OpenSearch + Grafanaで構築

dmarc-visualizer (https://github.com/debricked/dmarc-visualizer)
として公開されているdocker-composeなどを参考に構築

parsedmarc:

- DMARCレポートの統計情報を取るためのツール
- imapやS3などでレポート取得し、OpenSearchに送る

# レポート可視化の様子


› Guide (1 panel)
⌄ DMARC Summary
SPF Alignment
true
false
DKIM Alignment
true
false
DMARC Passage
true
false
SPF Results Over Time
100000
80000
60000
40000
20000
0
03/13
03/14
03/15
03/16
03/17
03/18
03/19
unknown
temperror
softfail
permerror
pass
none
neutral
hardfail
fail
DKIM Results Over Time
100000
80000
60000
40000
20000
0
03/13
03/14
03/15
03/16
03/17
03/18
03/19
unknown
temperror
permerror
pass
fail

# Postfixエラーログ監視

# ログからのエラー判別

Postfixのログには以下が含まれている、これらでエラーが判別可能

- SMTP Reply Code
  - 3桁の数値 210とか
  - RFC5321で定義
- Delivery Status Notification(以下DSN)
  - RFC3464で定義
  - 判別に使うのは status field で 「.」 区切りの3つの数値の組
    - 5.7.26 みたいなやつ
    - RFC3463で定義
    - RFC3463で書かれているコード以外は IANA.org に一覧がある (RFC 5248)
      - https://www.iana.org/assignments/smtp-enhanced-status-codes/smtp-enhanced-status-codes.xhtml

詳しくは: Postfixの口グ監視で注意すべきSMTPのステータス仕様について - Hatena Developer Blog

# Postfixのログ形式

CloudWatch Logs Insightsで調べてみると

- SMTP Reply Codeは接続先MTAからのメッセージ中
  - said: 550 ....のような形
- DSNは以下2パターン、どちらかしかないこともある
  - dsn=2.0.0 のような形式のパラメータ
  - 接続先MTAが返すメッセージ中の 「said: 550-4.7.1 ...」など

# 実際のログ形式

Feb 05 04:54:08 mailoutbound-fb postfix/smtp[1285887]: 38E2AC2616: to=<****@gmail.com>, relay=alt1.gmail-smtp-in.l.google.com[142.250.141.26]:25, delay=974719, delays=974661/55/2.3/0.22, dsn=4.2.2, status=deferred (host alt1.gmail-smtp-in.l.google.com[142.250.141.26] said: 452-4.2.2 ...(省略))

赤字の部分がエラー判別に必要

# ログ監視実装当時の記録

つらそう

## Postfixログのここがすごい！

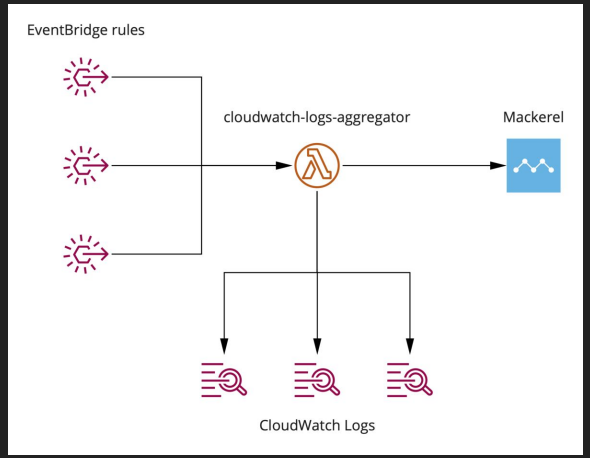
| dsn | dsn_in_message |
| --- | --- |
| 4.3.2 | 4.3.2 |
| 5.0.0 | |
| | 4.3.2 |
| 5.0.0 | |
| 5.0.0 | |
| | 4.3.2 |
| | 4.3.2 |

エラーメッセージの中にDSNが入っているがdsnパラメータが無いケース、その逆のケースなどがあり、地獄

# Postfixのログメトリクス監視

cloudwatch-logs-aggregatorを使って監視

- CloudWatch Logsのログをメトリック化し、Mackerelに送信するツール




(引用元: https://mackerel.io/ja/blog/entry/cloudwatch-logs-aggregator)

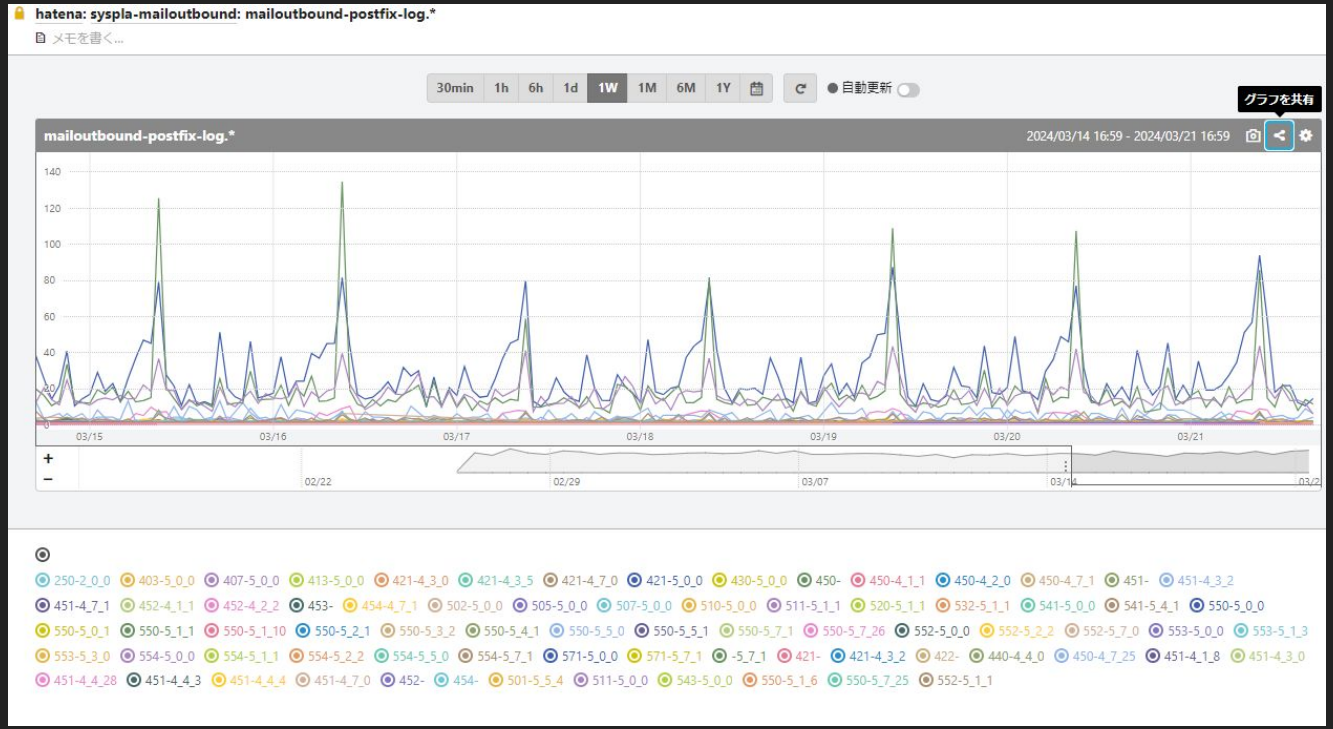
# cloudwatch-logs-aggregatorの設定

ログ集計用のクエリを作成

注意点としては以下

- DSNはパラメータ・メッセージのどちらかにしかないことがある
  - →coalesce でどちらかを取る
- メッセージ中のDSNの区切りは「-」「 」(空白)のいずれか
  - →正規表現で頑張る...
- SMTP Reply Codeとの組で監視する
  - Reply CodeとDSNをconcatしてハイフンで繋ぐ
- メトリクス名の「.」は別の字に置換したほうが良い
  - → replace関数で「.」を「_」に置換
  - そのままだとメトリクスが同じグラフに入ってくれなくて見辛い

# 完成したグラフ


hatena: syspla-mailoutbound: mailoutbound-postfix-log.*
メモを書く...
30min 1h 6h 1d 1W 1M 6M 1Y
自動更新
グラフを共有
mailoutbound-postfix-log.*
2024/03/14 16:59 - 2024/03/21 16:59
140
120
100
80
60
40
20
0
03/15
03/16
03/17
03/18
03/19
03/20
03/21
02/22
02/29
03/07
250-2_0_0 403-5_0_0 407-5_0_0 413-5_0_0 421-4_3_0 421-4_3_5 421-4_7_0 421-5_0_0 430-5_0_0 450- 450-4_1_1 450-4_2_0 450-4_7_1 451- 451-4_3_2
451-4_7_1 452-4_1_1 452-4_2_2 453- 454-4_7_1 502-5_0_0 505-5_0_0 507-5_0_0 510-5_0_0 511-5_1_1 520-5_1_1 532-5_1_1 541-5_0_0 541-5_4_1 550-5_0_0
550-5_0_1 550-5_1_1 550-5_1_10 550-5_2_1 550-5_3_2 550-5_4_1 550-5_5_0 550-5_5_1 550-5_7_1 550-5_7_26 552-5_0_0 552-5_2_2 552-5_7_0 553-5_0_0 553-5_1_3
553-5_3_0 554-5_0_0 554-5_1_1 554-5_2_2 554-5_5_0 554-5_7_1 571-5_0_0 571-5_7_1 -5_7_1 421- 421-4_3_2 422- 440-4_4_0 450-4_7_25 451-4_1_8 451-4_3_0
451-4_4_28 451-4_4_3 451-4_4_4 451-4_7_0 452- 454- 501-5_5_4 511-5_0_0 543-5_0_0 550-5_1_6 550-5_7_25 552-5_1_1

# ログメトリクス活用例

- 社内から送信されるbatchの通知メールの不備の検知
  - メール基盤のTCP resetがある時期から増えていることに気付く
  - グラフを確認しその時期増えているものを確認
    - 550-5.7.1 エラーが増えている
  - CloudWatch Logs Insight で当該Status/DSNのものを確認
    - 社内向けのメールがgmailから拒否されていることが分かる
      - メールフォーマットのRFC違反によるもの
    - batch通知メールだったので当該チームに連絡
      - メールクライアントの設定等を見直してもらった

# ログメトリクス活用

- 能動的にメール送信時の不備を検知するのは難しい
  - 特定のReply Code/DSN毎にアラートするしかない
  - DSNは割と解釈の幅があり、狼少年アラートを作りかねない
- 5xx系で重要なもののみ検知するのが良さそう
  - GmailのDMARC関連エラーとか
  - Reply Code/DSN以外のメトリクスと組み合わせて活用したい
    - バウンスメール数を見て、細かい原因はDSNで確認するなど

# アラート設定

# Gmailから来るメールエラー

DMARC/DKIM/SPFの検査に失敗した場合

以下のSMTPエラーを返す

- 550-5.7.26
  - このエラーには3パターンある
  - メッセージからDMARC/DKIM/SPFのどのエラーか判別できる

# アラート設定

ひとまず今回のGmailのポリシー対応では

- 550-5.7.26 の数をアラート対象とした
  - 現状ではとりあえず1件でも起きたら発火させている
    - →今後の状況を見て調整していく
    - 設定後半年以上経ったものの今のところ無風
- アラートが発生したらとりあえずログを見に行く
  - DMARC/DKIM/SPFのいずれのエラーなのかを把握するため

# アラート設定の今後

- Gmail以外のDMARC関連エラーに対応したい
  - 米Yahoo!等も厳しくなるので対応したい
- アラート対応方法を固めたい
  - DNSと設定と秘密鍵を確認するくらいしかやることがない
  - 設定が正しければ起こらないはずだが...
- DMARC以外のエラーへの対応
  - 5xx系で致命的なものがあるなら検知したい

# まとめ

# まとめ

- DMARC対応には以下が必要
  - メール送信元の把握
  - SPF/DKIM 認証の設定など
    - 独自システムでDKIMを設定するためにはOpenDKIMを使う
  - DMARCのレコード設定・レポート送信先の作成
- レポートから設定漏れを潰していく必要がある
  - 可視化はparsedmarc + OpenSearch + grafana での構築が簡単
  - DMARCを可視化してくれるSaaSに頼るのも良い

# まとめ

- メールのログ監視は難しい
  - 独自基盤でやる場合は避けられない
  - 完璧にやるなら各メールサービス毎にエラーの対応の把握が必要
- DMARC対応はちゃんとやろう！
  - Gmailを使っている顧客に重要なメールが届かないと大変
  - 今後ガイドライン適用範囲が変更される可能性もあるため...