2019-11-28 DNSOPS.JP BoF

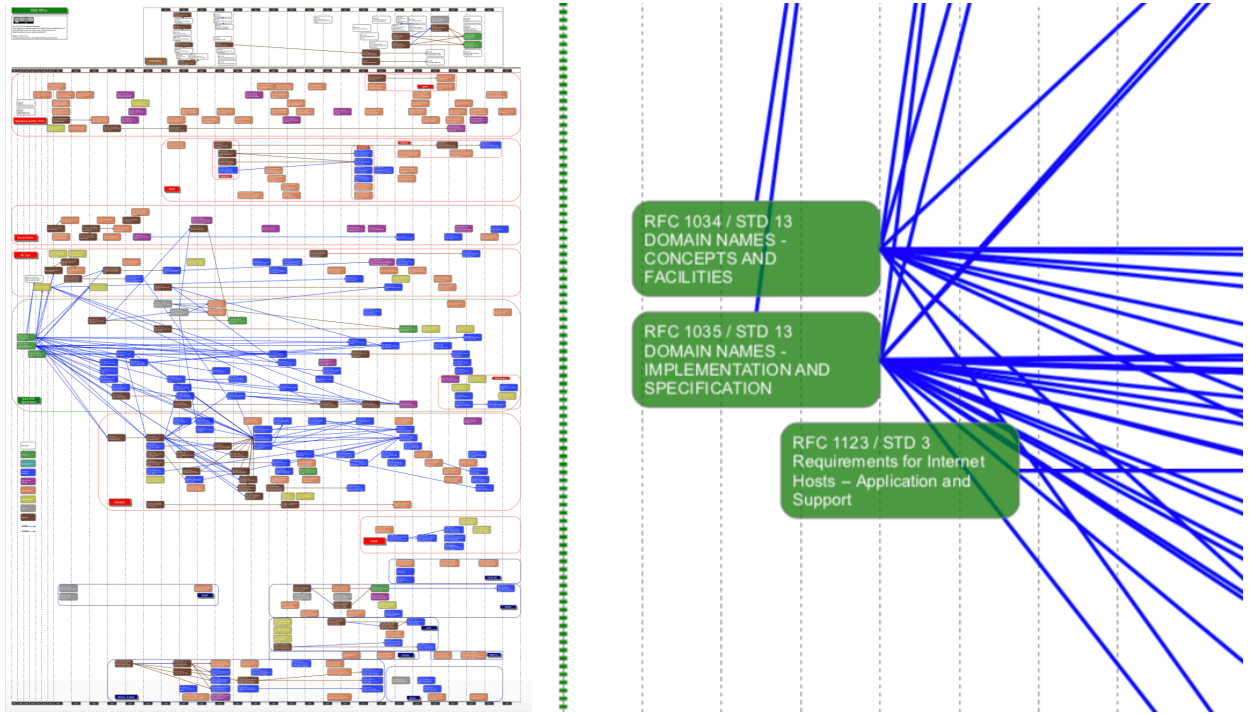
OctoDNSとGitLab CI/CDを利用した複数DNSプロバイダー構成の運用

HEARTBEATS

株式会社ハートビーツ　滝澤隆史

私は誰

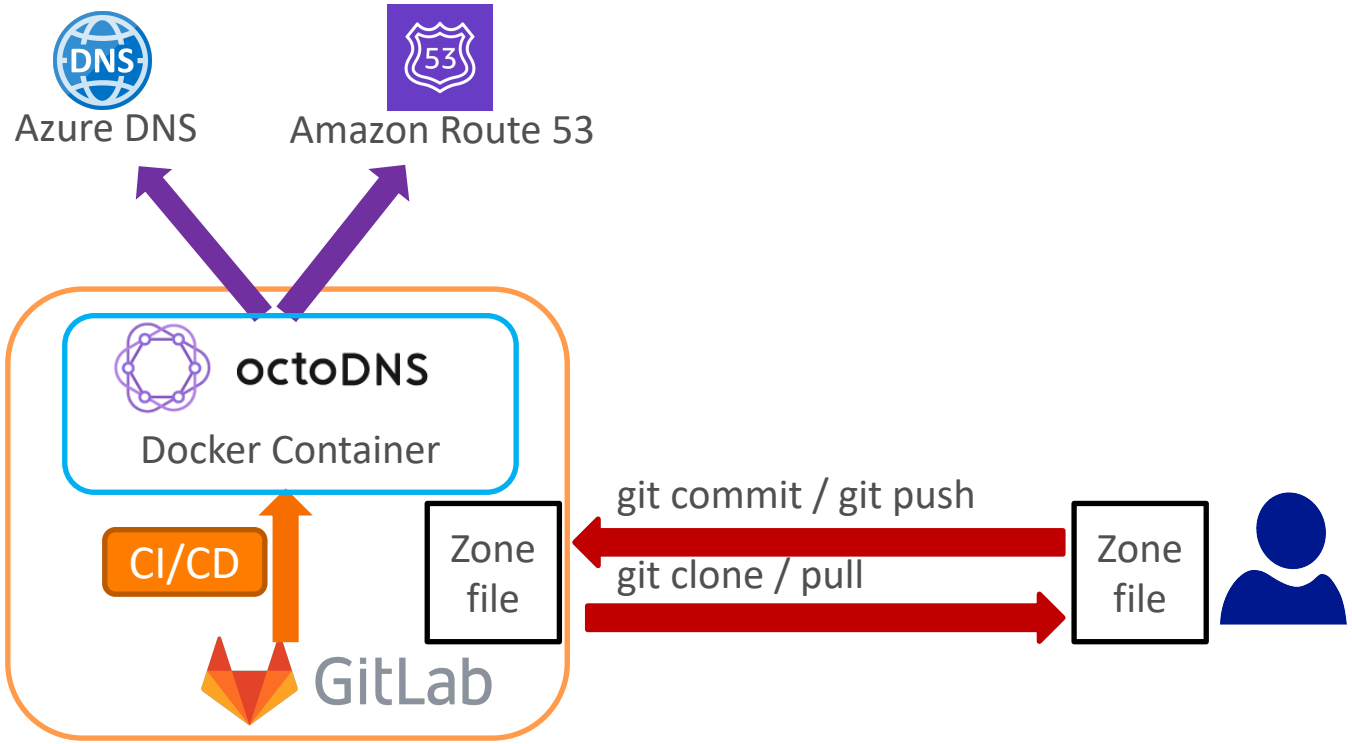
- 滝澤 隆史（たきざわ たかし）
- Twitter: @ttkzw
- DNSで遊んでいる人
  - DNS RFC系統図
  - https://emaillab.jp/dns/dns-rfc/



- 所属: 株式会社ハートビーツ
  - 事業内容
    - MSP（監視・運用代行）、受託開発（PHP + React (TypeScript)）
  - 滝澤は社内システム基盤の面倒を見ている
    - DNS権威サーバー
      - ここのお話をします

本日のお話を３行で説明すると

- DNSゾーン管理ツール OctoDNS と
- SCM（ソースコード管理）ツール GitLab のCI/CD機能を使って、
- 複数DNSプロバイダー構成を運用する事例を紹介します。



HEARTBEATS

背景（きっかけ）

課題認識

- 2016年10月のDynのDNS権威サーバーへの大規模DDoS
- 自前運用の弊社のDNS権威サーバー
  - DDoSが発生したらこの1000分の1程度でも回線飽和で使い物にならなくなるという課題認識
  - 弊社はお客様へ監視サービスを提供しており、アラートメールの送信ができなくなったり、監視システムへアクセスできなくなったりするのは事業上の大問題
  - この時点ではこの課題を社内のGitLabのIssueに起票しただけ

オープンソースのツールの認識

- 2017年春に複数DNSプロバイダー対応したOSSの管理ツールがあることを認識し、調査を実施
  - Stack Exchange社開発のDNSControl (2017-03-14 First public release)
  - GitHub社開発のOctoDNS (2017-03-14 First public release)

- この時点では、利用したいDNSプロバイダーやリソースレコードの要件には合わなかったため、検討を中断した。
  - 利用を検討していたDNSプロバイダーがNSレコードの追加に対応していなかったということもあった。これは後ほど解消した。

- この後、「課題は認識しているが、緊急性が高いわけではないため、優先度が高くならない」という典型的な状況になり、2年ほど放置されてしまった。

データセンター閉鎖

- ラックを借りているデータセンターが閉鎖することになった。
- DNS権威サーバーをどこかに移転しなければならない
- 「課題は認識しているが、緊急性が高いわけではないため、優先度が高くならない」
  - 別な意味で緊急性が高くなってしまった。

プロジェクト開始

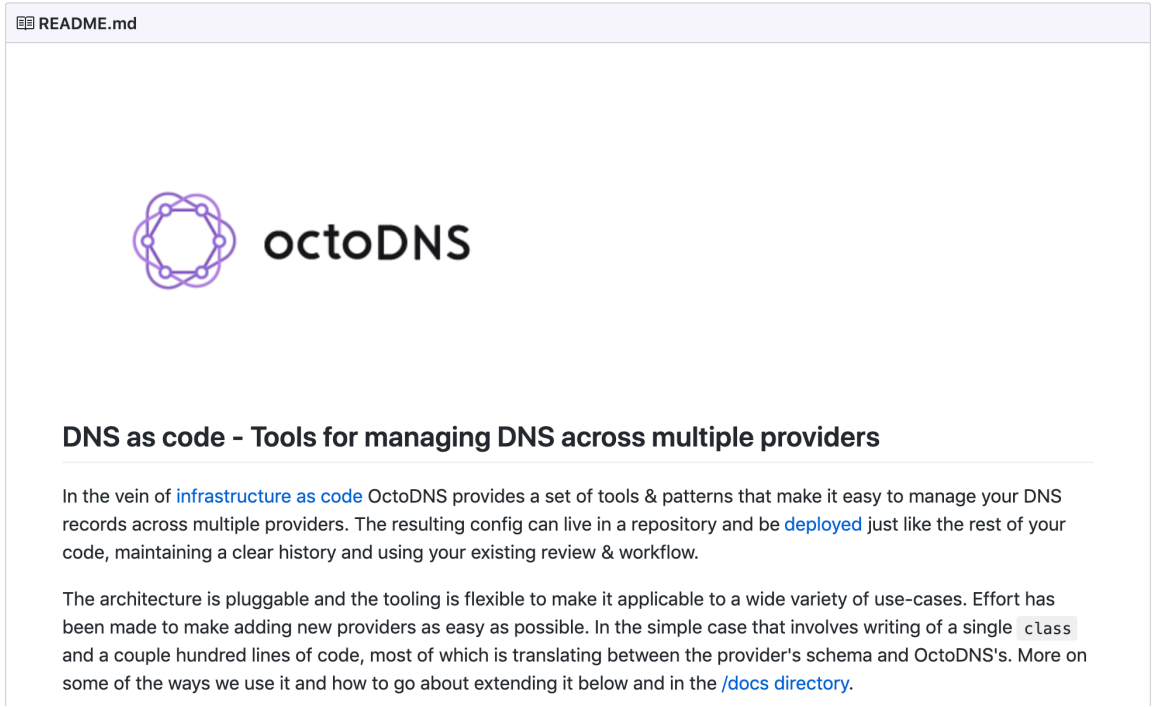
- ということで調査を再開
- 以前の調査では要件が合わなかったDNSプロバイダーとリソースレコードの件が解消されていた。
- 結果として、OctoDNSを使うことにした。
  - OctoDNSの方が設定ファイルがシンプルという理由

HEARTBEATS

OctoDNS

OctoDNSとは

- GitHub社が開発・保守しているオープンソースソフトウェア
- DNS as code - Tools for managing DNS across multiple providers
- https://github.com/github/octodns


README.md

octoDNS

DNS as code - Tools for managing DNS across multiple providers

In the vein of infrastructure as code OctoDNS provides a set of tools & patterns that make it easy to manage your DNS records across multiple providers. The resulting config can live in a repository and be deployed just like the rest of your code, maintaining a clear history and using your existing review & workflow.

The architecture is pluggable and the tooling is flexible to make it applicable to a wide variety of use-cases. Effort has been made to make adding new providers as easy as possible. In the simple case that involves writing of a single `class` and a couple hundred lines of code, most of which is translating between the provider's schema and OctoDNS's. More on some of the ways we use it and how to go about extending it below and in the /docs directory.

OctoDNSとは

- CLIツール
  - octodns-sync
    - ソースプロバイダーのゾーン情報をターゲットプロバイダーにAPIで同期する

```
$ octodns-sync --config-file=./config/production.yaml --doit
...
```

OctoDNSの設定ファイル（公式サイトのREADME.mdより）

config/production.yaml

プロバイダー定義
（認証情報含む）

```
---
providers:
  config:
    class: octodns.provider.yaml.YamlProvider
    directory: ./config
    default_ttl: 3600
    enforce_order: True
  dyn:
    class: octodns.provider.dyn.DynProvider
    customer: 1234
    username: 'username'
    password: env/DYN_PASSWORD
  route53:
    class: octodns.provider.route53.Route53Provider
    access_key_id: env/AWS_ACCESS_KEY_ID
    secret_access_key: env/AWS_SECRET_ACCESS_KEY
```

ゾーン毎の
プロバイダーの指定

```
zones:
  example.com.:
    sources:
      - config
    targets:
      - dyn
      - route53
```

config/example.com.yaml

ソース
プロバイダーの
ゾーンファイル
（YAML形式）

```
---
'':
  ttl: 60
  type: A
  values:
  - 1.2.3.4
  - 1.2.3.5
```

OctoDNSの実行例（公式サイト README.md より）

- デフォルトはDRY RUNとして動作する。

```
$ octodns-sync --config-file=./config/production.yaml
...
********************************************************************************
* example.com.
********************************************************************************
* route53 (Route53Provider)
*   Create <ARecord A 60, example.com., [u'1.2.3.4', '1.2.3.5']>
*   Summary: Creates=1, Updates=0, Deletes=0, Existing Records=0
* dyn (DynProvider)
*   Create <ARecord A 60, example.com., [u'1.2.3.4', '1.2.3.5']>
*   Summary: Creates=1, Updates=0, Deletes=0, Existing Records=0
********************************************************************************
...
```

- 実際に反映するためには --doit オプションを付ける。

```
$ octodns-sync --config-file=./config/production.yaml --doit
...
```

OctoDNSが対応しているプロバイダー

- AzureProvider
- Akamai
- CloudflareProvider
- ConstellixProvider
- DigitalOceanProvider
- DnsMadeEasyProvider
- DnsimpleProvider
- DynProvider
- EtcHostsProvider
- GoogleCloudProvider
- MythicBeastsProvider
- Ns1Provider
- OVH
- PowerDNSProvider
- RackspaceProvider
- Route53
- Selectel
- Transip
- ソースプロバイダー
  - AxfrSource
  - ZoneFileSource
  - TinyDnsFileSource
- コンフィグ
  - YamlProvider

採用した構成

- ソースプロバイダー
  - ZoneFileSource（ゾーンファイル）
  - 既存のゾーンファイルをそのまま利用でき、CLIベースの作業手順のまま利用できるため。
- ターゲットプロバイダー
  - Route53Provider（Amazon Route53）
  - AzureProvider（Azure DNS）

HEARTBEATS

GitLab CI/CD

octodns-syncをどこで実行するか

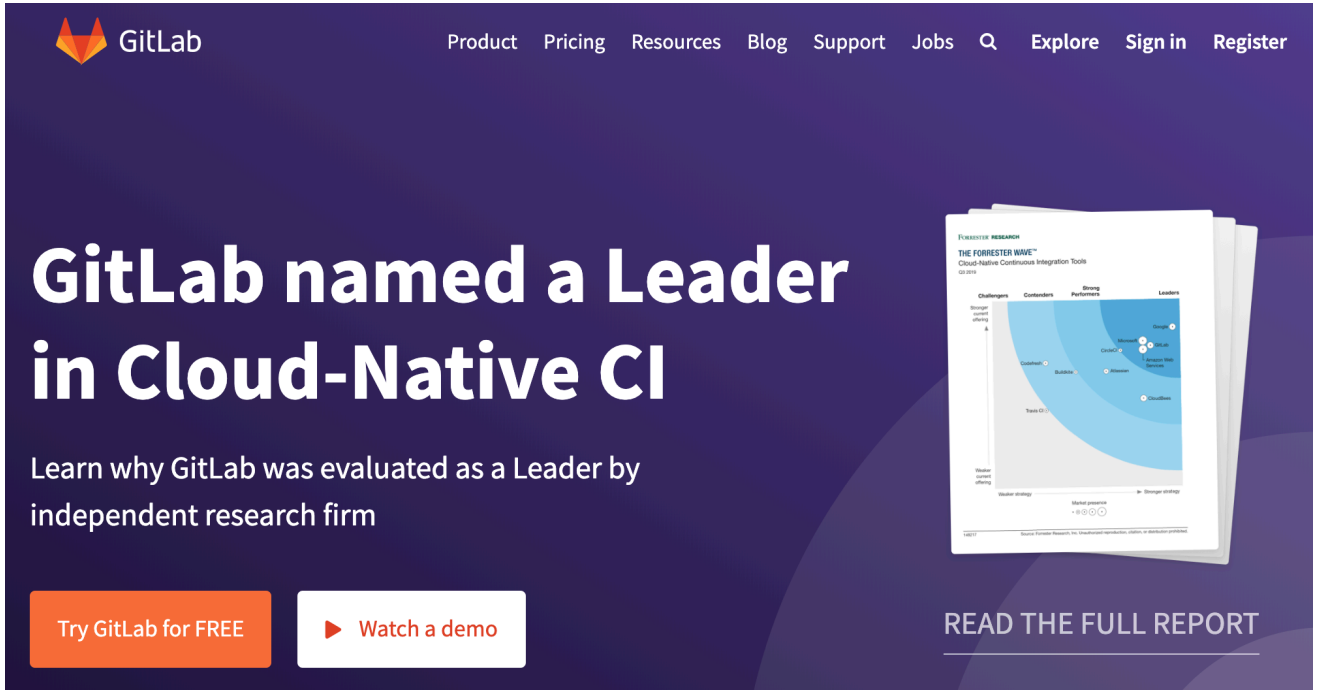
- 既存のシステム
  - DNS権威サーバーのホストにSSHログインして、Subversion管理下のゾーンファイルを直接編集していた。
    - BIND 9が動いている
    - NSDに移行する計画はあったんですけどね

octodns-syncをどこで実行するか

- 新しいシステム
  - DNS権威サーバーのホストがない！
  - ソースコード管理にGitLabを利用しているので、
    GitLab CI/CDを利用して、
    Dockerコンテナ内でoctodns-syncを実行するようにした。

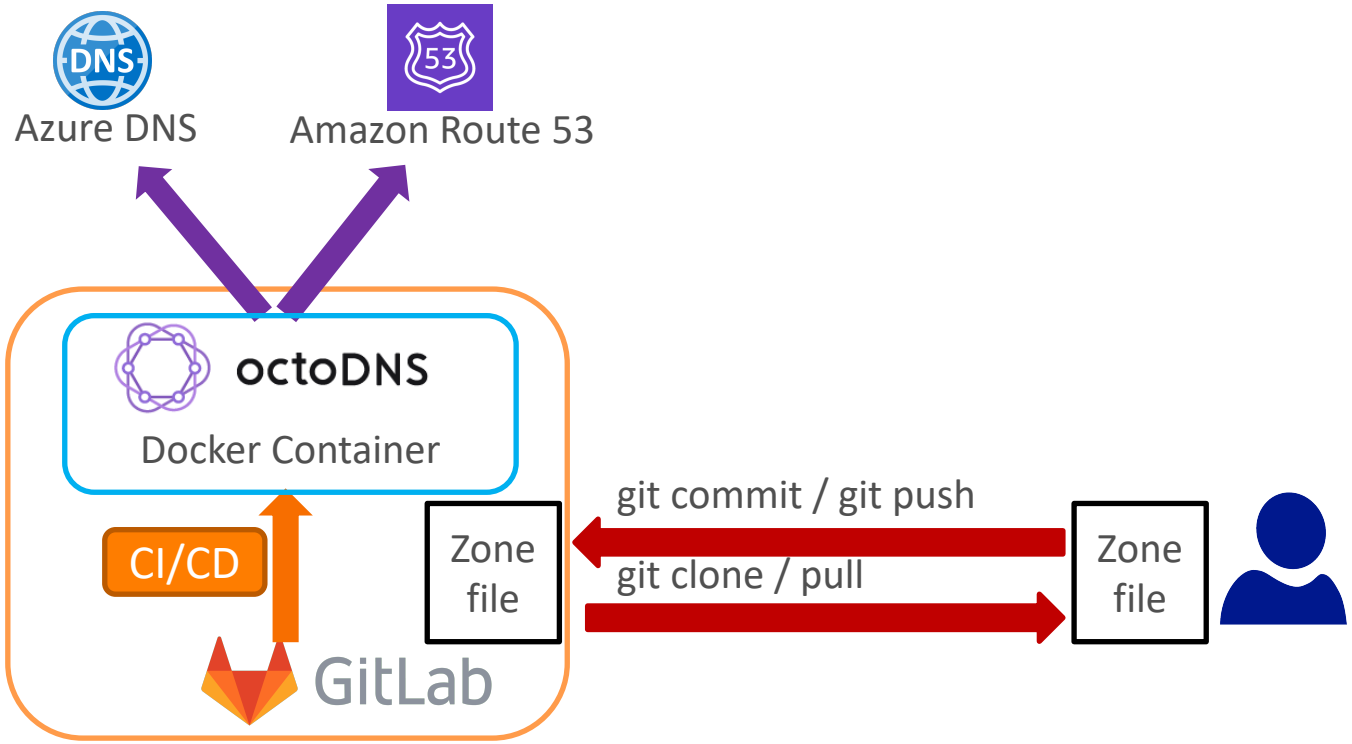
GitLabとは

- GitLab社が開発しているSCM（ソースコード管理）ツール
  - CI/CD (Continuous Integration/Continuous Delivery)機能もある
- https://about.gitlab.com/



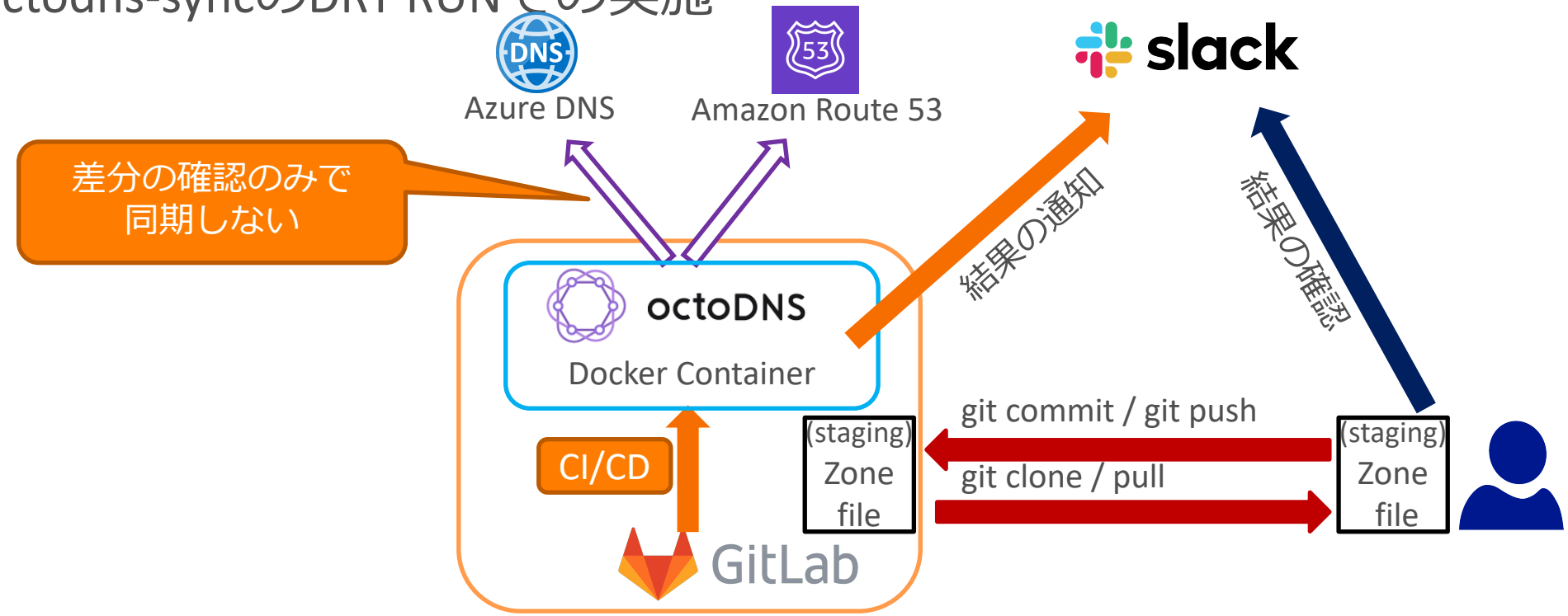
基本構成

- GitLab CI/CDにより、
- OctoDNSをインストールしたDockerコンテナを起動して、
- octodns-syncを実行するようにする。



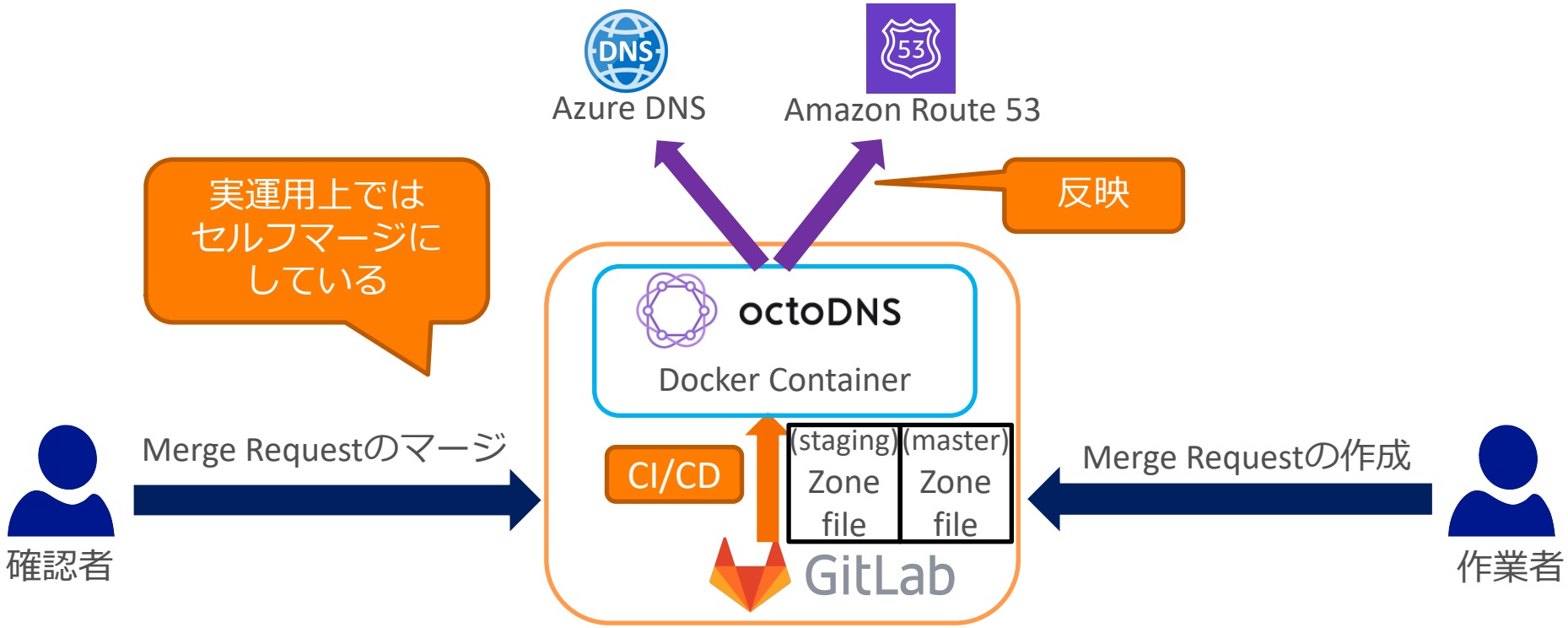
テスト

- 何も確認せずにDNSプロバイダーのゾーンに反映するのは危険
- stagingブランチを用意して、テストを行う
  - ゾーンファイルの構文チェック（named-checkzone）
  - octodns-syncのDRY RUNでの実施



Merge Requestの作成とマージ

- 作業者はstagingブランチからmasterブランチへのMerge Requestを作成する。
- 確認者はMerge Requestの内容を確認してマージする。
- GitLab CI/CDによりoctodns-syncを--doitオプションを付けて実行し、DNSプロバイダーのゾーンに反映する。



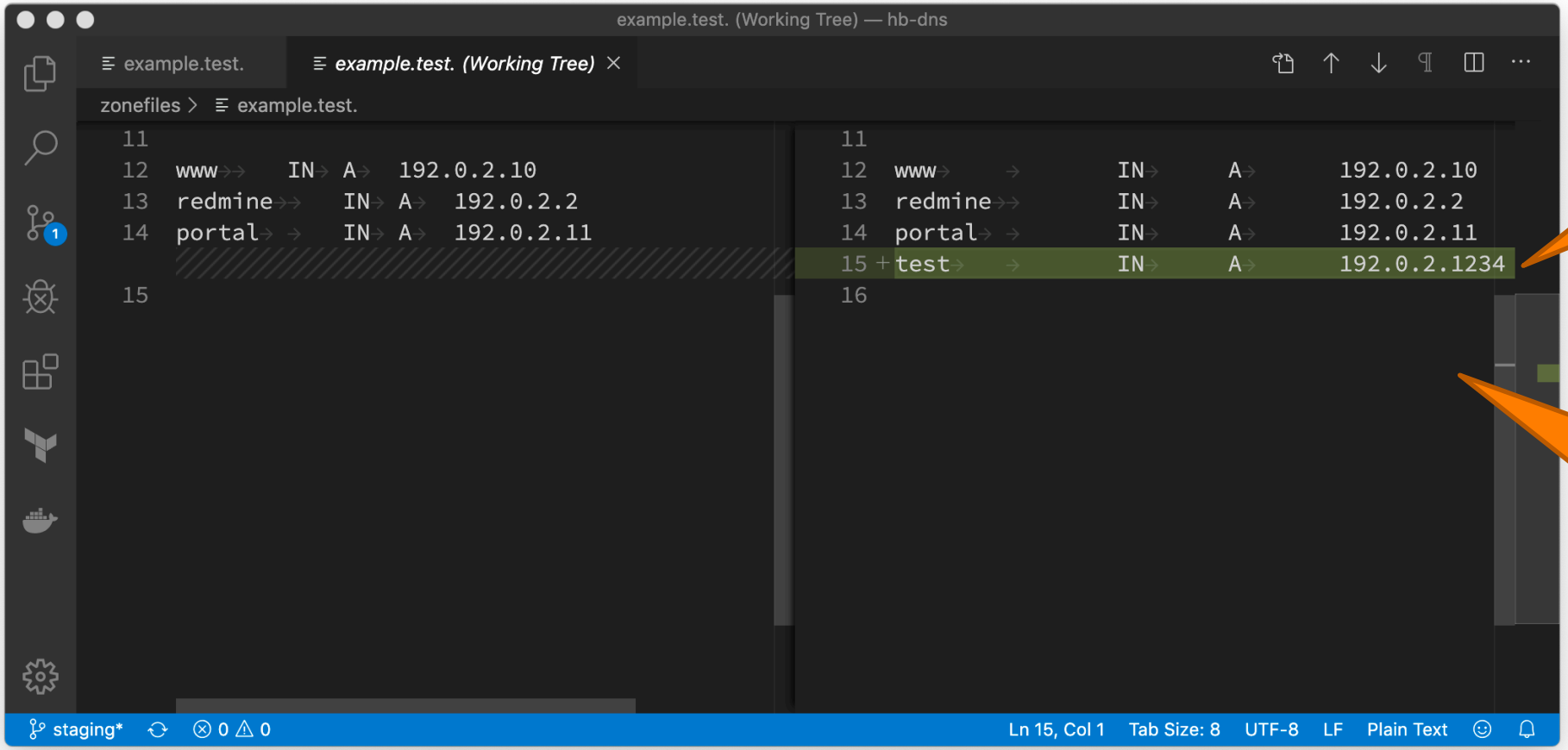
実行例の紹介

実行例の紹介

- 失敗する実行例
  - 構文ミス
- 成功する実行例

失敗する実行例

- stagingブランチからgit pullして、ゾーンファイルを編集し、差分を確認し、git commit & git pushする。



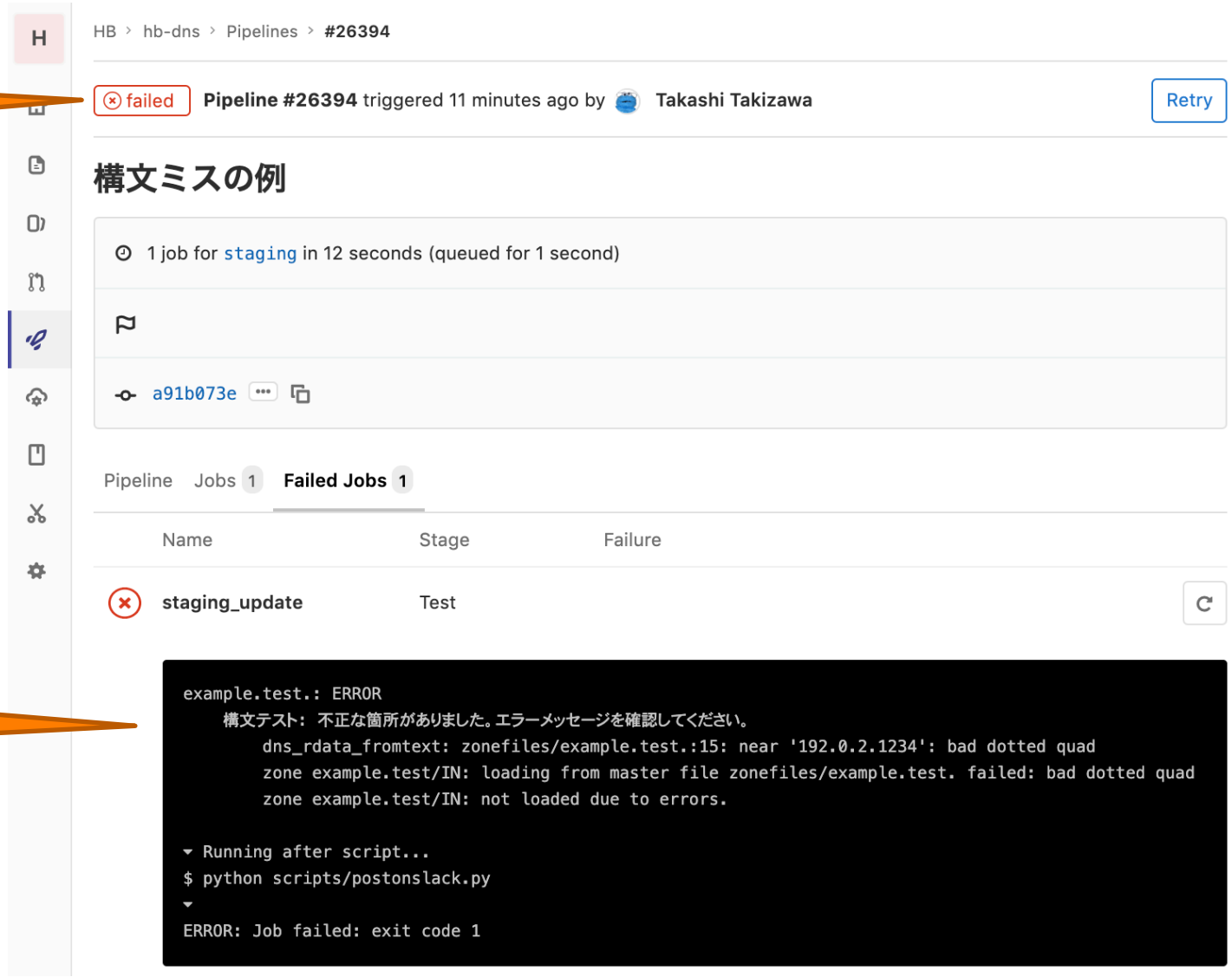
差分の確認
（構文ミス）

実際の手順はvimで行うけど、VSCodeでgit pushまでやってみた例

GitLab CI/CDのジョブの確認画面（失敗する実行例）

ステータス

ジョブの出力



Slackへのジョブ失敗の通知

Slackへの通知

GitLabの画面を開かなくてもすむようにジョブの出力内容を通知

GitLab APP 8:10 PM

@takizawa

シンタックスチェックでエラーを検知しました。修正してください。

```
================================================
ゾーンファイルのテスト
================================================
[redacted]: OK
[redacted]: OK
[redacted]: OK
[redacted]: OK
[redacted]: OK
example.test.: ERROR
    構文テスト: 不正な箇所がありました。エラーメッセージを確認してください。
        dns_rdata_fromtext: zonefiles/example.test.:15: near '192.0.2.1234': bad
dotted quad
        zone example.test/IN: loading from master file zonefiles/example.test.
failed: bad dotted quad
        zone example.test/IN: not loaded due to errors.
```

GitLabの
Slack Integrationによる通知

gitlab APP 8:10 PM

Takashi Takizawa (takizawa)

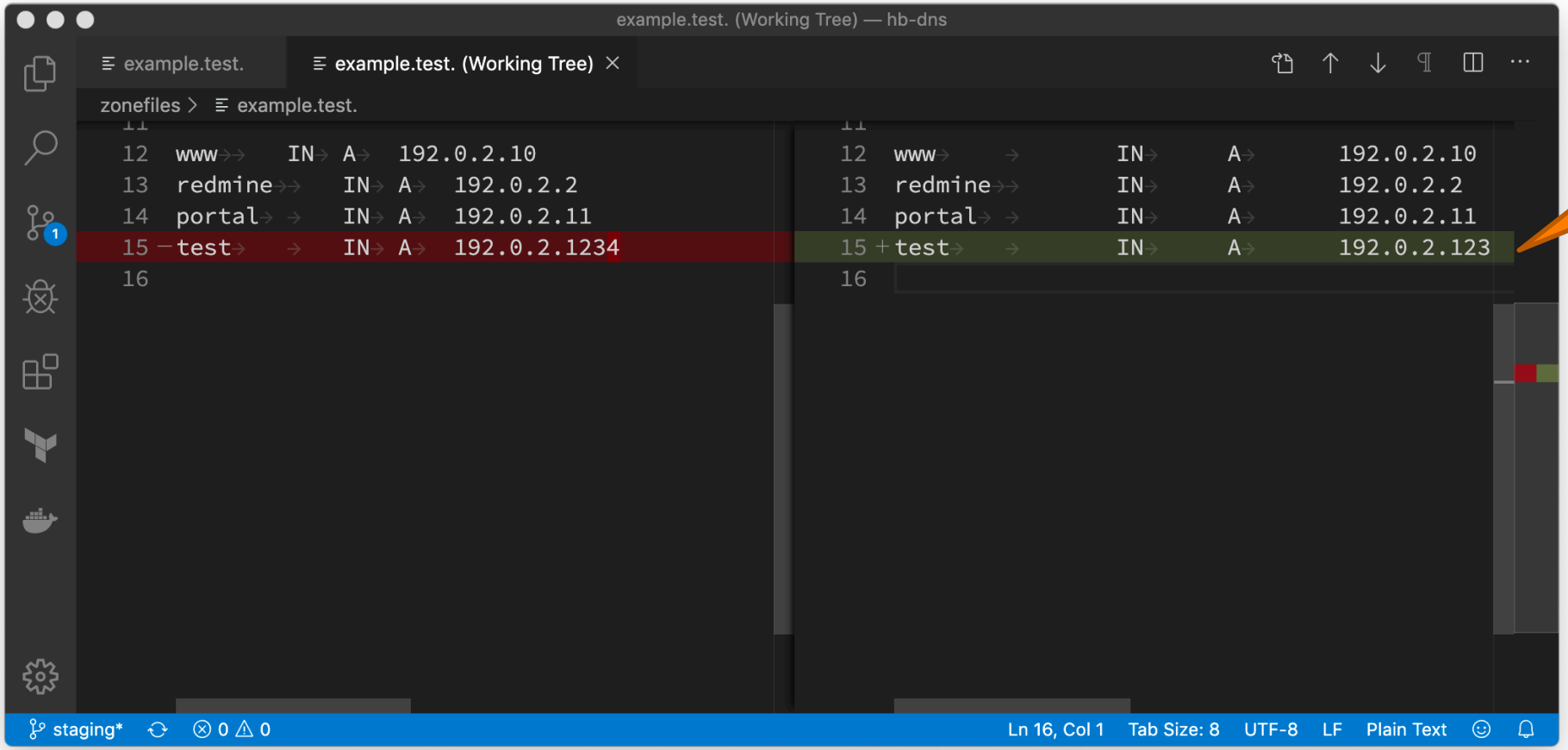
Pipeline #26394 has failed in 00:12

Branch	Commit
staging	構文ミスの例
Failed stage	**Failed job**
test	staging_update

hb-dns | Today at 8:10 PM

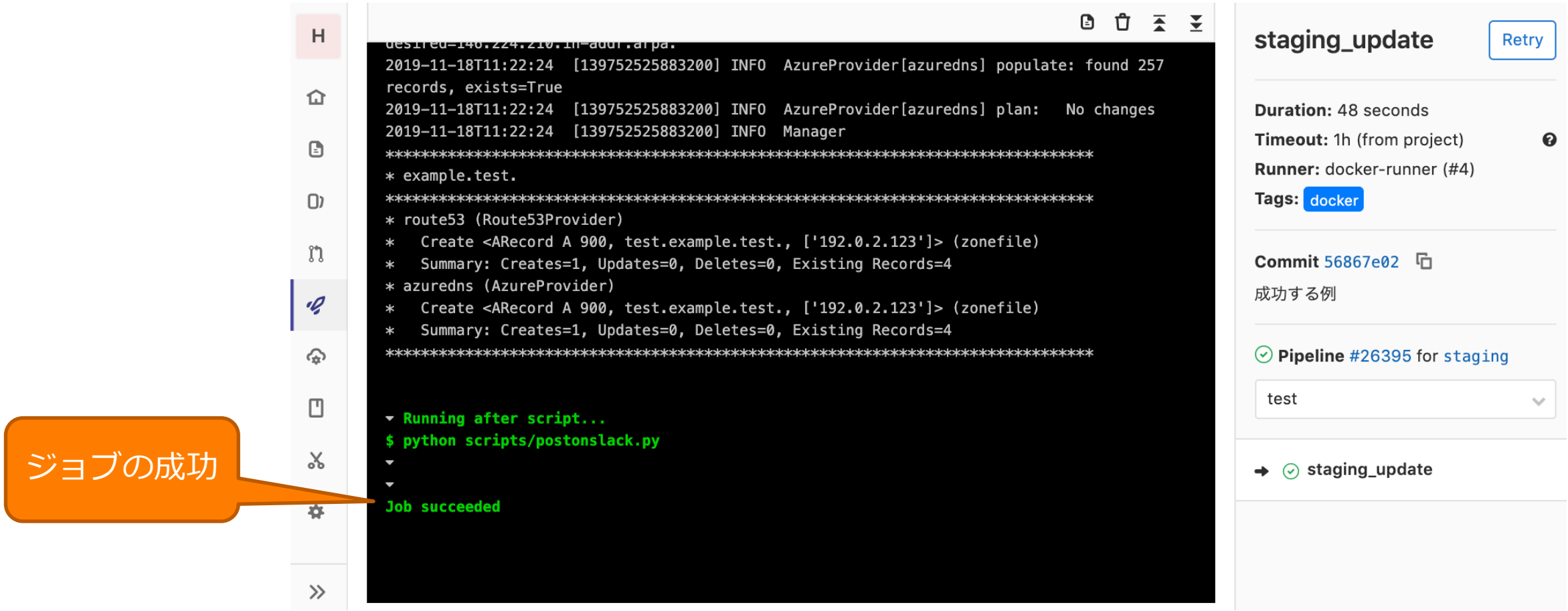
成功する実行例

- stagingブランチからgit pullして、ゾーンファイルを編集し、差分を確認し、git commit & git pushする。



差分の確認
（正しい）

GitLab CI/CDのジョブの確認画面（成功する実行例）



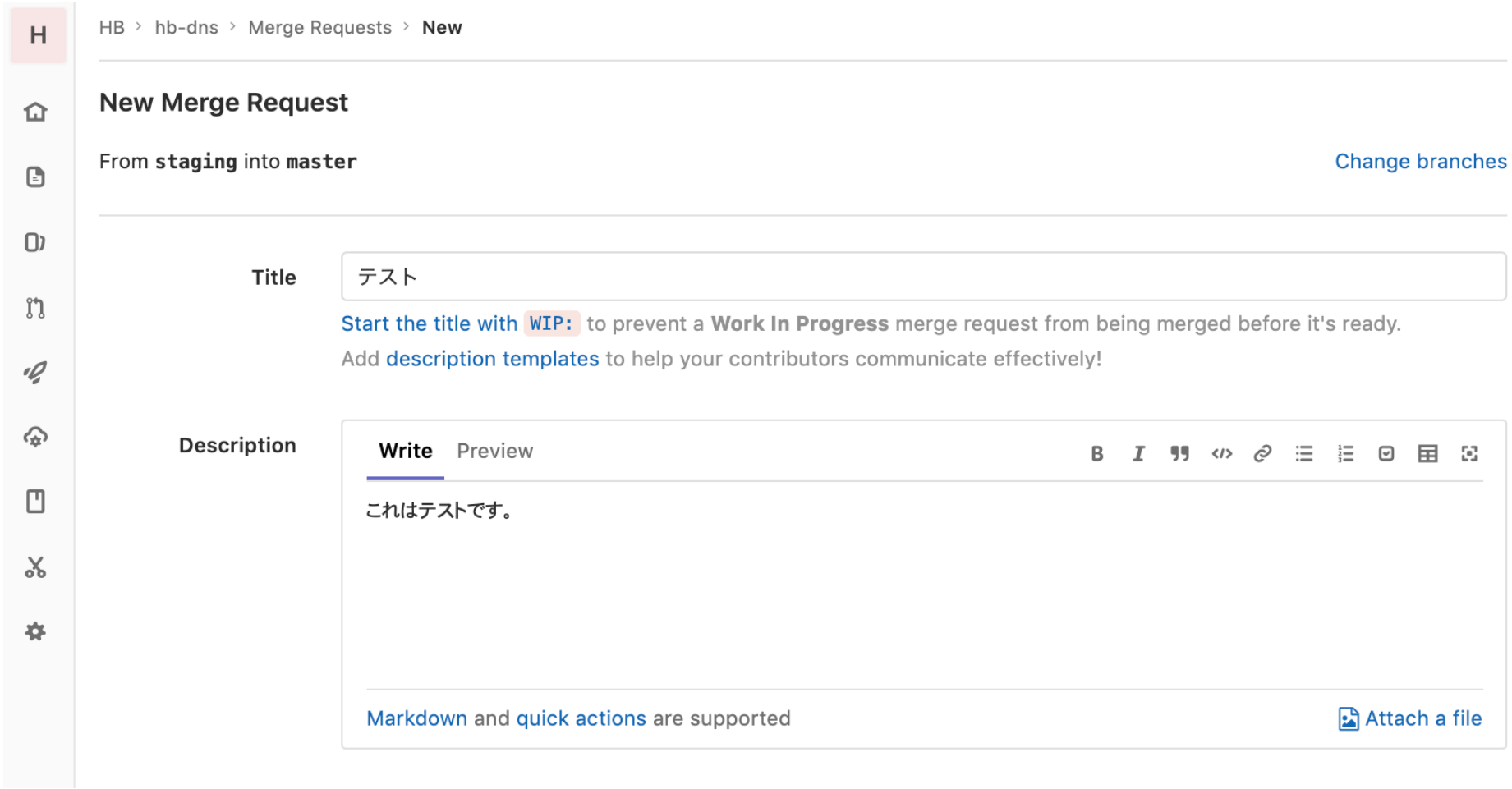
Slackへのジョブ成功の通知

Slackへの通知

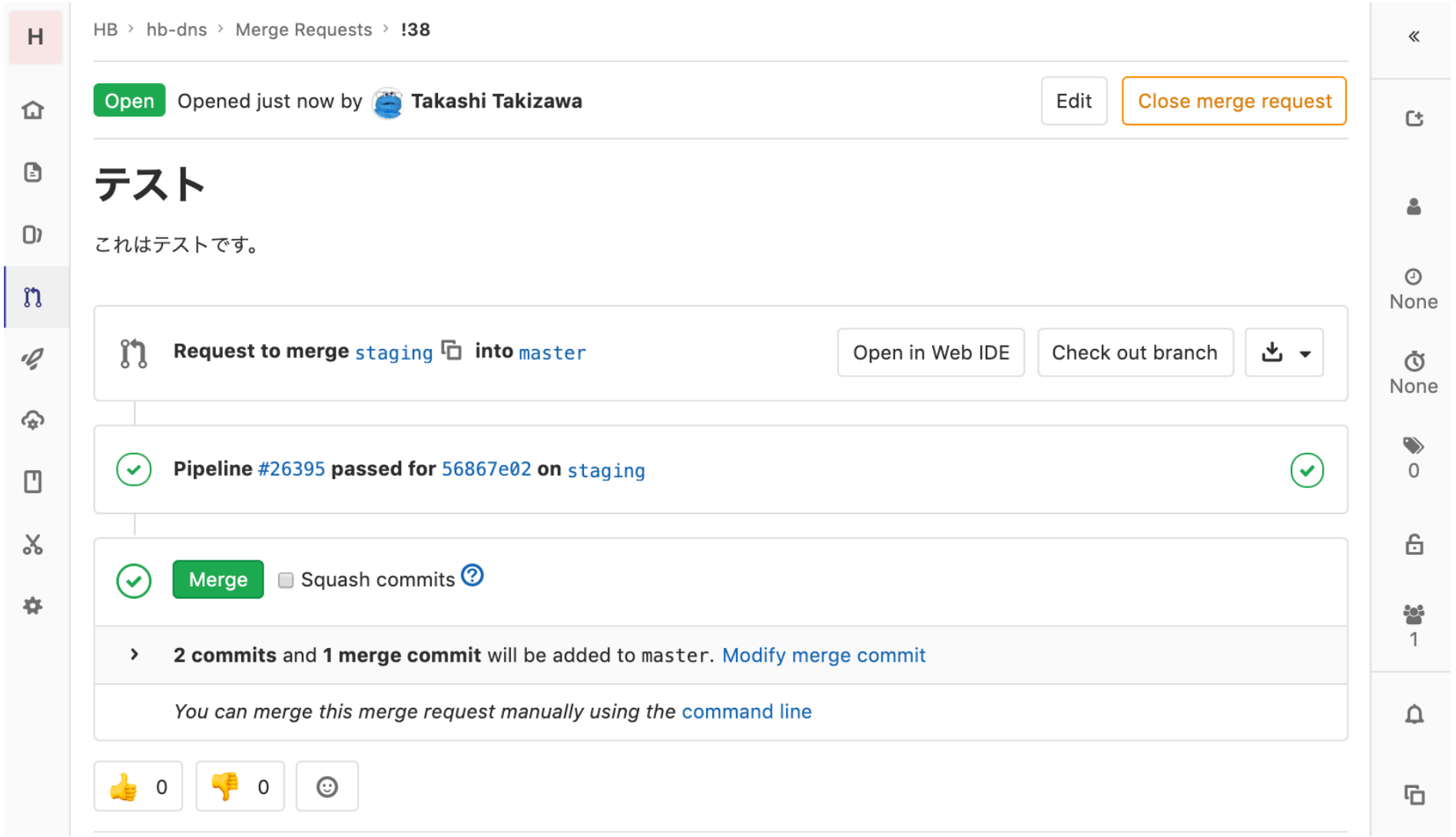
GitLabの画面を開かなくてもすむようにジョブの出力内容を通知

GitLabの
Slack Integration
による通知

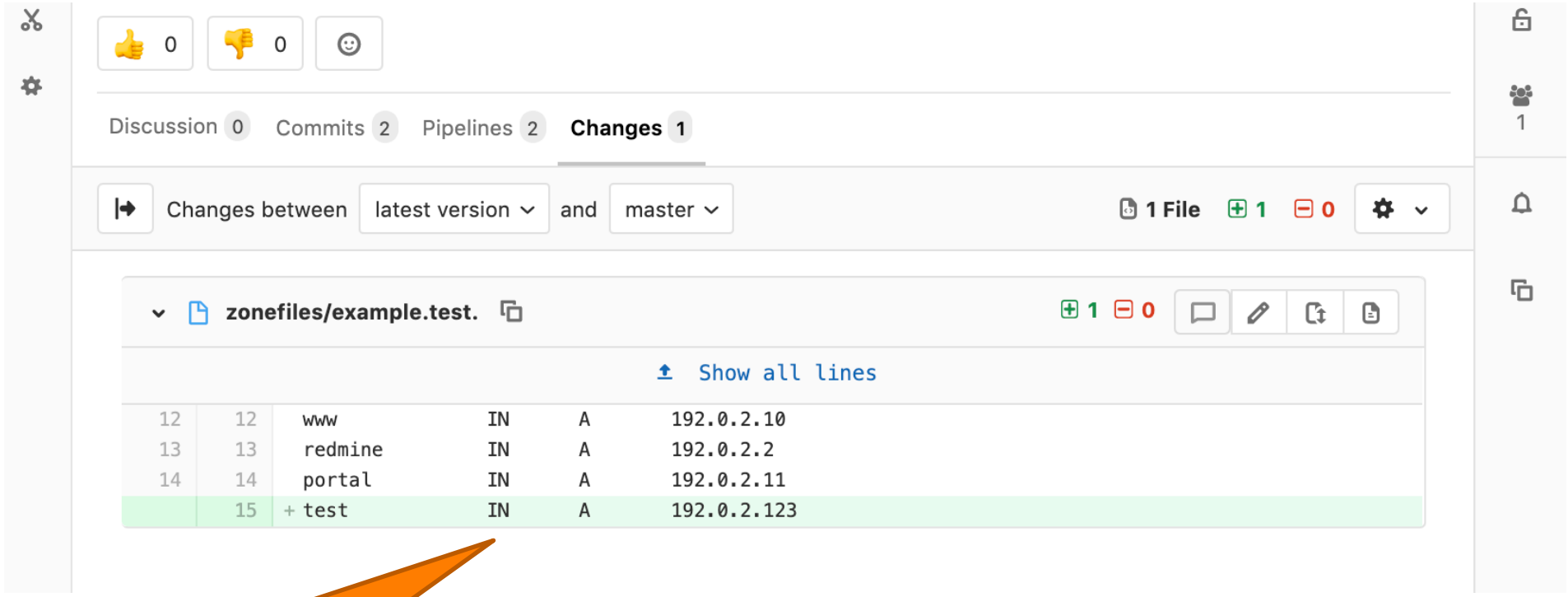
GitLab CI/CD: Merge Requestの作成



GitLab CI/CD: Merge Requestの確認とマージ

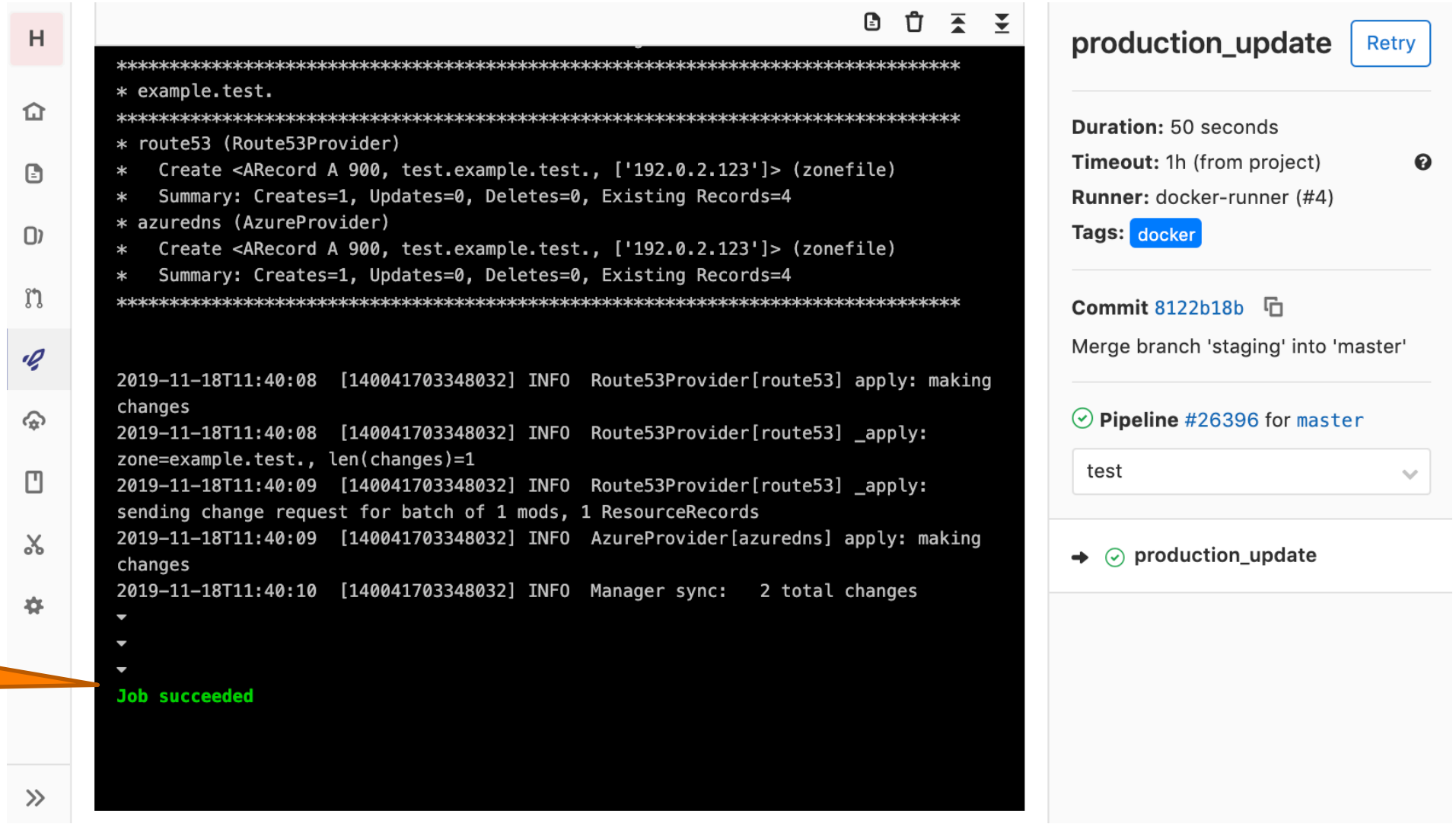


GitLab CI/CD: Merge Requestの確認とマージ



変更内容を最終確認し、
問題なければMergeボタン
をクリックする。

GitLab CI/CDのジョブの確認画面（同期の実施）



ジョブの成功

SlackへのmasterブランチのジョブのGitLabの実行結果の通知

GitLabの
Slack Integration
による通知

gitlab APP 8:36 PM

Takashi Takizawa (takizawa) opened !38 テスト in HB/hb-dns

Takashi Takizawa (takizawa) merged !38 テスト in HB/hb-dns

Takashi Takizawa (takizawa)

Pipeline #26396 has passed in 00:50

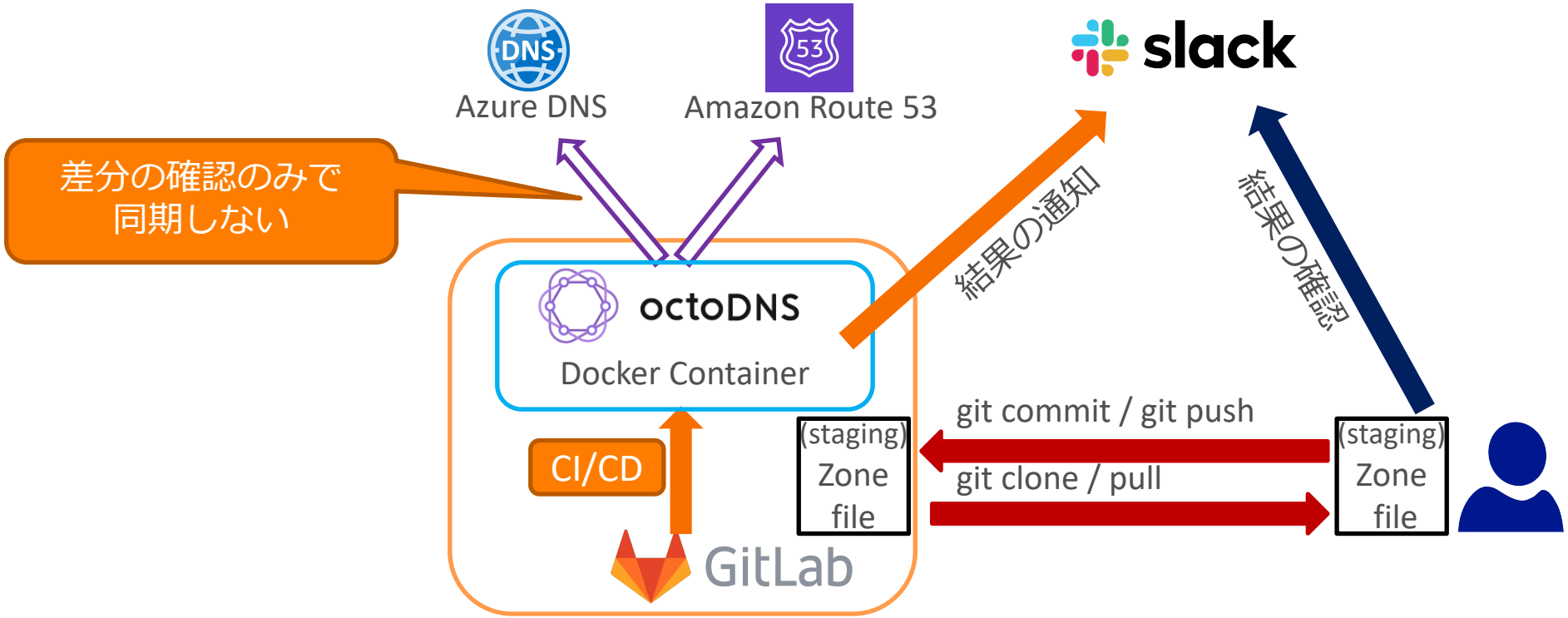
Branch	Commit
master	Merge branch 'staging' into 'master'

hb-dns | Today at 8:40 PM

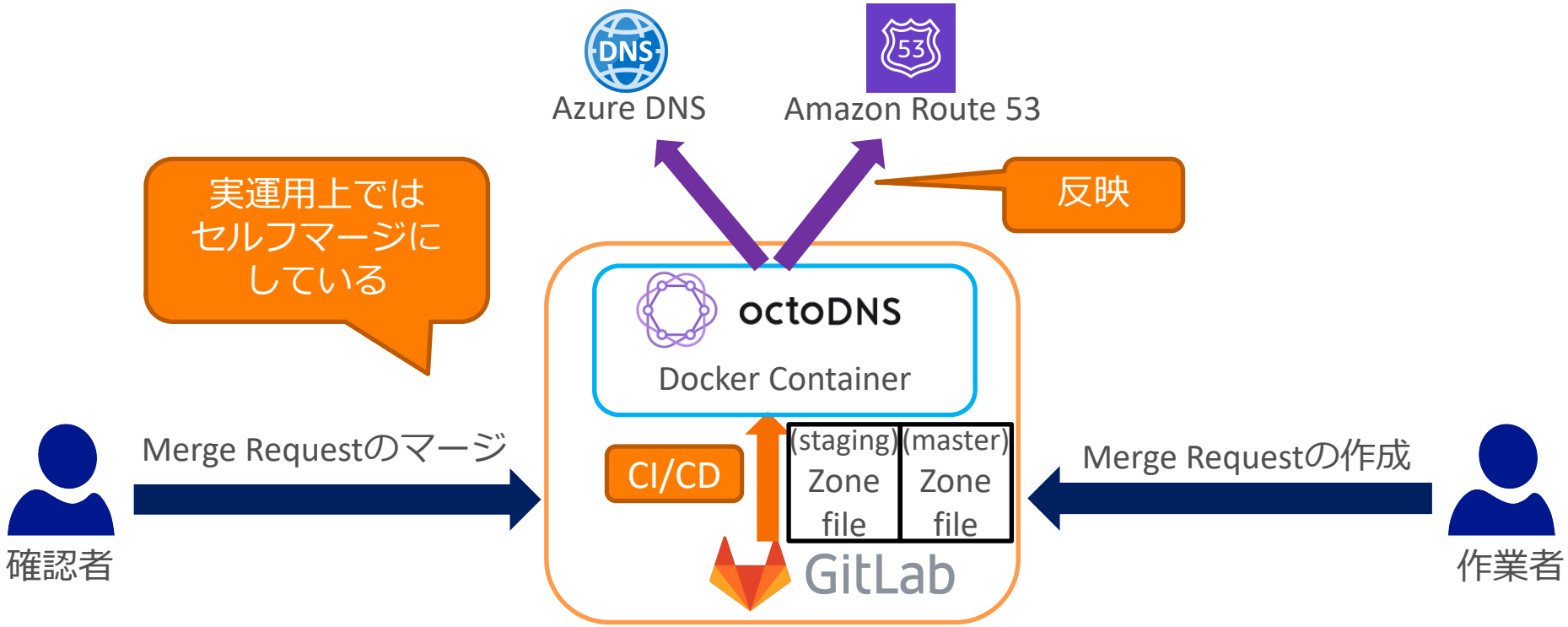
実行例は以上！

- 実際はdigで確認もするけど、例としては省略。

振り返り: stagingブランチ

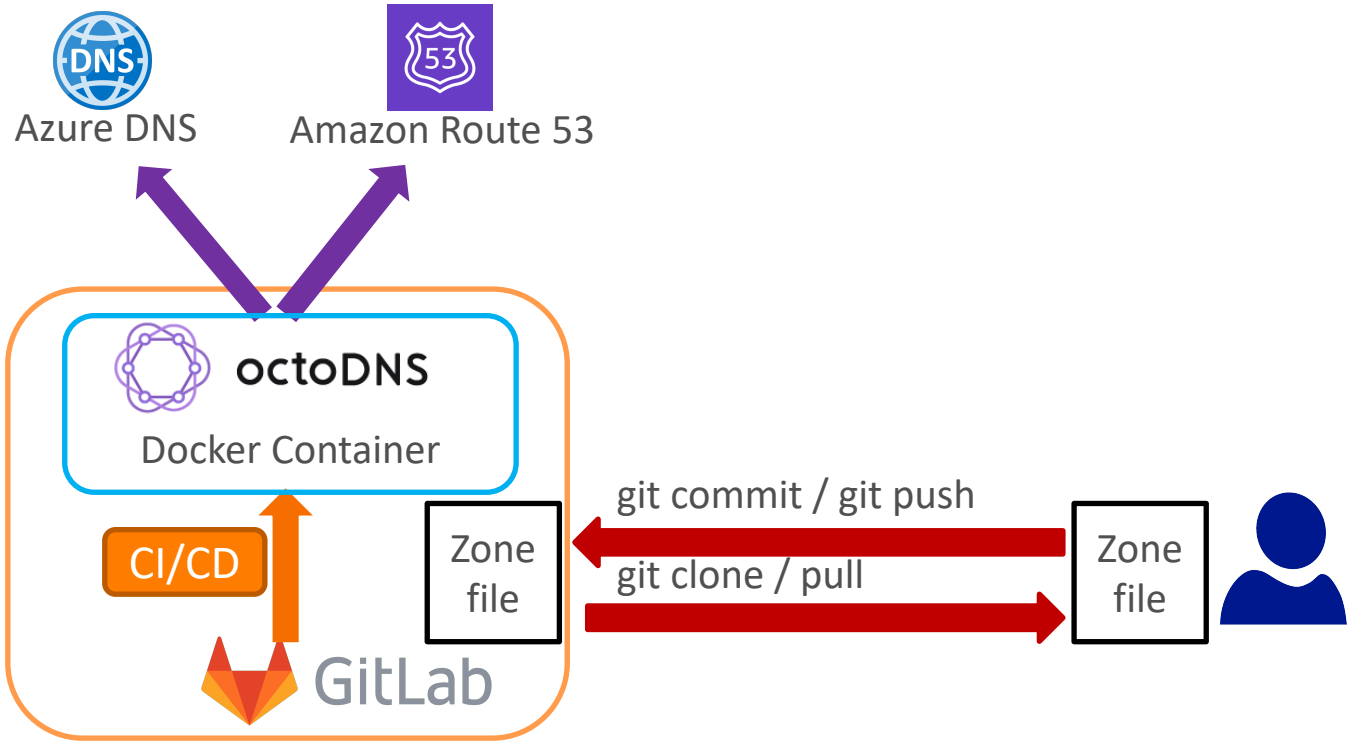


振り返り: masterブランチ



まとめ

- DNSゾーン管理ツール OctoDNS と
- SCM（ソースコード管理）ツール GitLab のCI/CD機能を使って、
- 複数DNSプロバイダー構成を運用する事例を紹介しました



おまけ: OctoDNSの知見

OctoDNSの知見

- OctoDNS v0.9.8時点での情報
- リソースレコード(RR)
  - SOAレコードとNSレコードは同期されない。
  - プロバイダーにより対応しているRRタイプは異なる。
- 安全機能 (octodns.provider.plan.Planクラスのraise_if_unsafe()で定義)
  - 10個以上のRRが存在するときに
    - 全体の30%以上のRRが更新される場合は中断する
    - 全体の30%以上のRRが削除される場合は中断する
  - --force オプションを付けるとこの安全機能を無視する

OctoDNSの知見

- プロバイダー毎の設定方法
  - 各プロバイダーのPythonモジュールのDocstringとして記述されている。

```
>>> import octodns.provider.route53
>>> help(octodns.provider.route53.Route53Provider)
Help on class Route53Provider in module octodns.provider.route53:

class Route53Provider(octodns.provider.base.BaseProvider)
 |  Route53Provider(id, access_key_id=None, secret_access_key=None, max_changes=1000,
client_max_attempts=None, session_token=None, *args, **kwargs)
 |
 |  AWS Route53 Provider
 |
 |  route53:
 |      class: octodns.provider.route53.Route53Provider
 |      # The AWS access key id
 |      access_key_id:
 |      # The AWS secret access key
 |      secret_access_key:
 |      # The AWS session token (optional)
 |      # Only needed if using temporary security credentials
 |      session_token:
```

OctoDNSの知見

- ZoneFileSource
  - SOAレコードは必須（同期されないけど）
  - ファイル名は絶対ドメイン名にする。
    - ./zonefiles/example.test.

```
class ZoneFileSource(AxfrBaseSource):
    '''
    Bind compatible zone file source

    zonefile:
        class: octodns.source.axfr.ZoneFileSource
        # The directory holding the zone files
        # Filenames should match zone name (eg. example.com.)
        directory: ./zonefiles
```