# StackStorm 運用応用編

大山 裕泰 / DMM.com ラボ

2017/11/29 InternetWeek 2017

# 免責事項

- 第三者の製品・サービスについて、特定の製造者やサービス提供者につき、製品やサービスを評価するものではありません。
- 当社事例は、あくまで当社事案であり各社のシステム・サービス要件等によって、機能、パフォーマンスその他の面で該当しない場合があります。
- 本プレゼンテーションは当社構築した StackStorm に関する技術者の現時点での感想に基づいています。
- 無断複製・転載を禁じます。

# 目的

以下の運用プラクティスを共有し、ナレッジを高める

- Active Directory によるユーザ認証
- 冗長構成な StackStorm の構築
- StackStorm のアップグレード

# Active Directory によるユーザ認証

## Enterprise 版じゃなくても AD 認証はできる

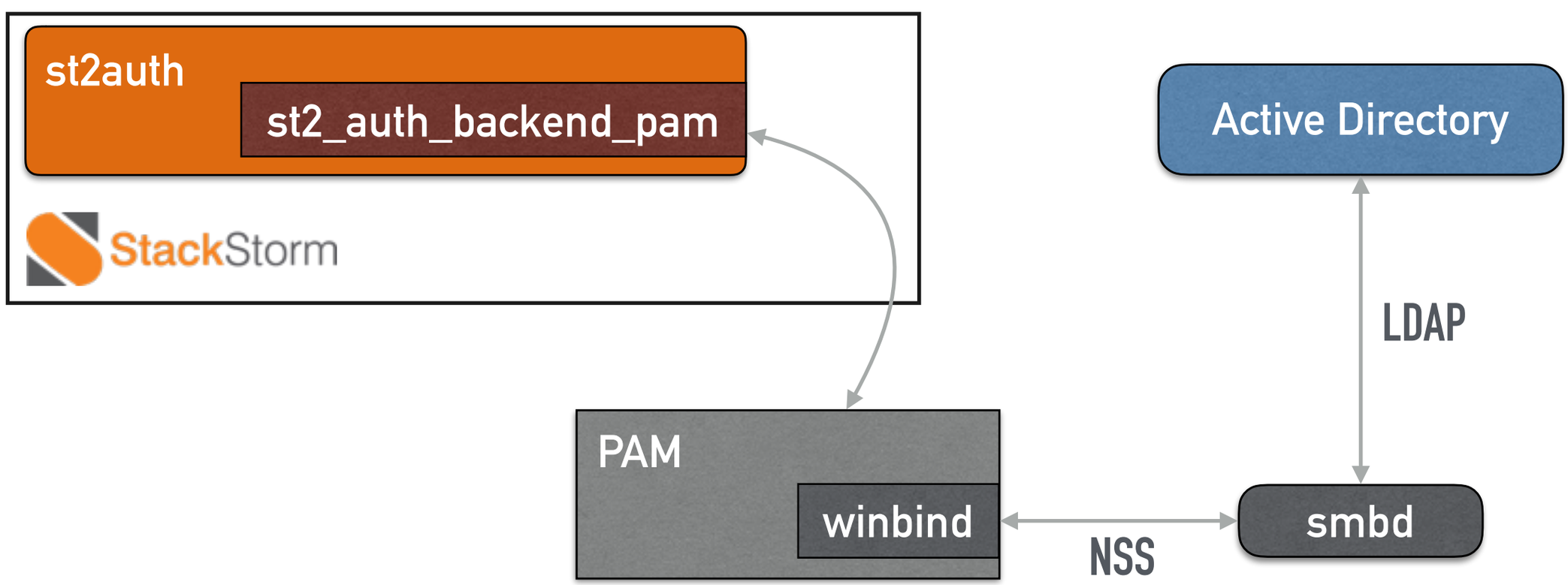
### – But at your own risk

## StackStorm で AD 認証を実現する方法

- PAM
- LDAP (by OSS)
- LDAP (by Enterprise Edition)

# 1. PAM で AD 認証を実現する方法

- st2_auth_backend_pam 認証モジュールを使用
- 設定方法詳細



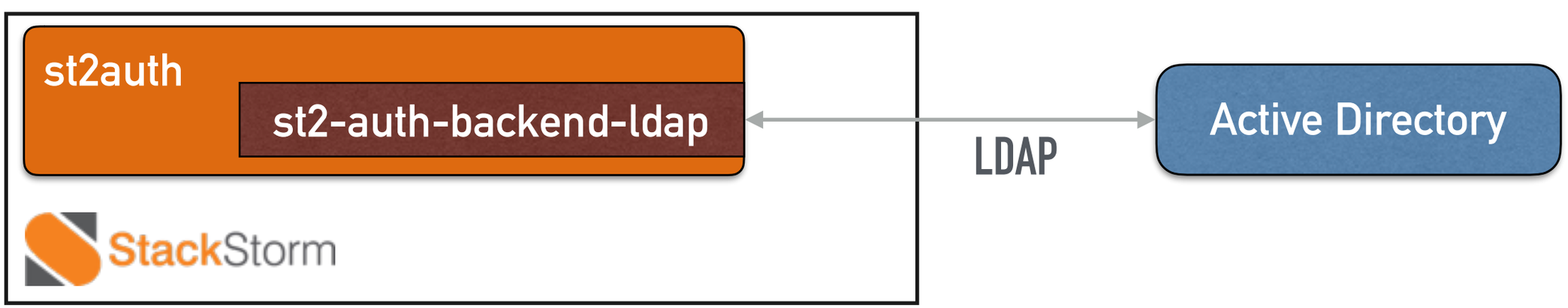

st2_auth_backend_pam 認証モジュールを使用した場合の認証フロー

# 1. PAM で AD 認証を実現する課題

- 全ての StackStorm ノードで pam_ldap 設定をしないといけない
- st2auth だけ root 権限で実行しなければならない
  (通常は一般ユーザ st2 によって起動される)

# 2. OSS で AD 認証を実現する方法

- st2_auth_backend_pam 認証モジュールを使用
- 設定方法詳細




st2-auth-backend-ldap 認証モジュールを使用した場合の認証フロー

## 2. OSS で AD 認証を実現する課題

- ベンダーの公式サポートが受けられない (at your own risk)

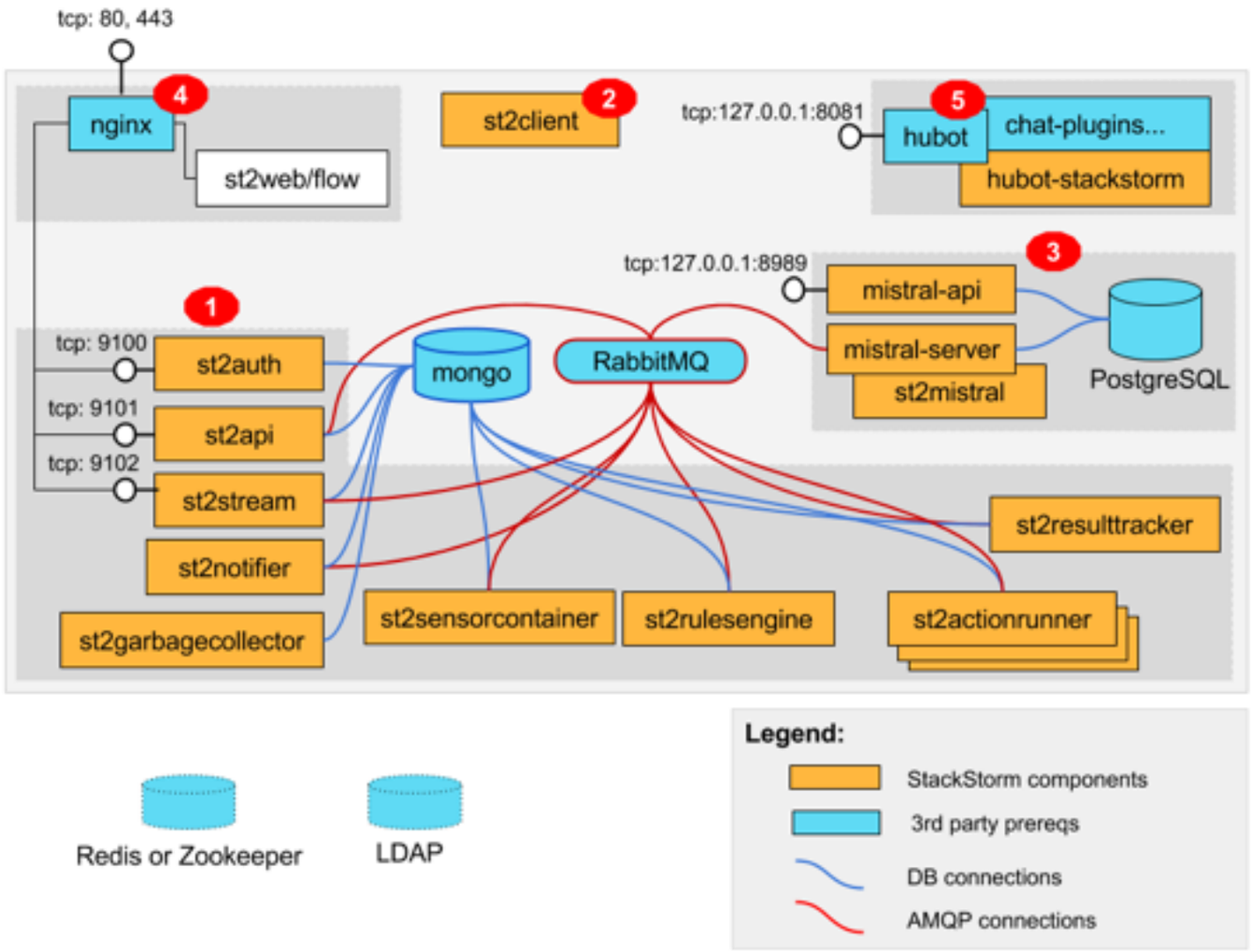
# 冗長構成な StackStorm の構築

## High Available な StackStorm サービス環境を構築

# 冗長構成な StackStorm の構築

## StackStorm の構成要素

- Stateful なコンポーネント
  - KVS (MongoDB)
  - RDB (PostgreSQL / MySQL)
  - MQ (RabbitMQ)



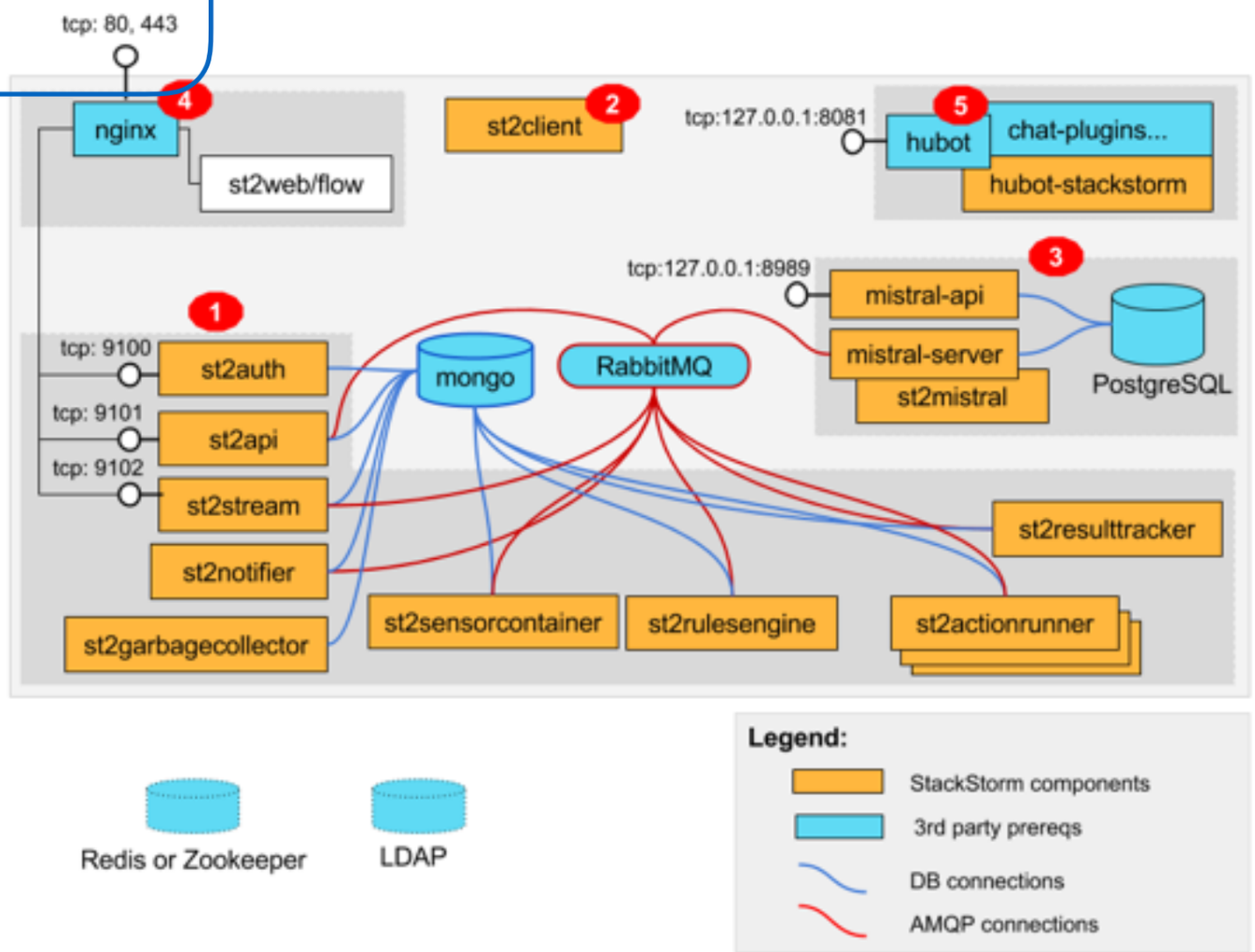

（Overview - StackStorm より転載）

# 冗長構成な StackStorm の構築

## StackStorm の構成要素

- Stateful なコンポーネント
  - KVS (MongoDB)
  - RDB (PostgreSQL / MySQL)
  - MQ (RabbitMQ)

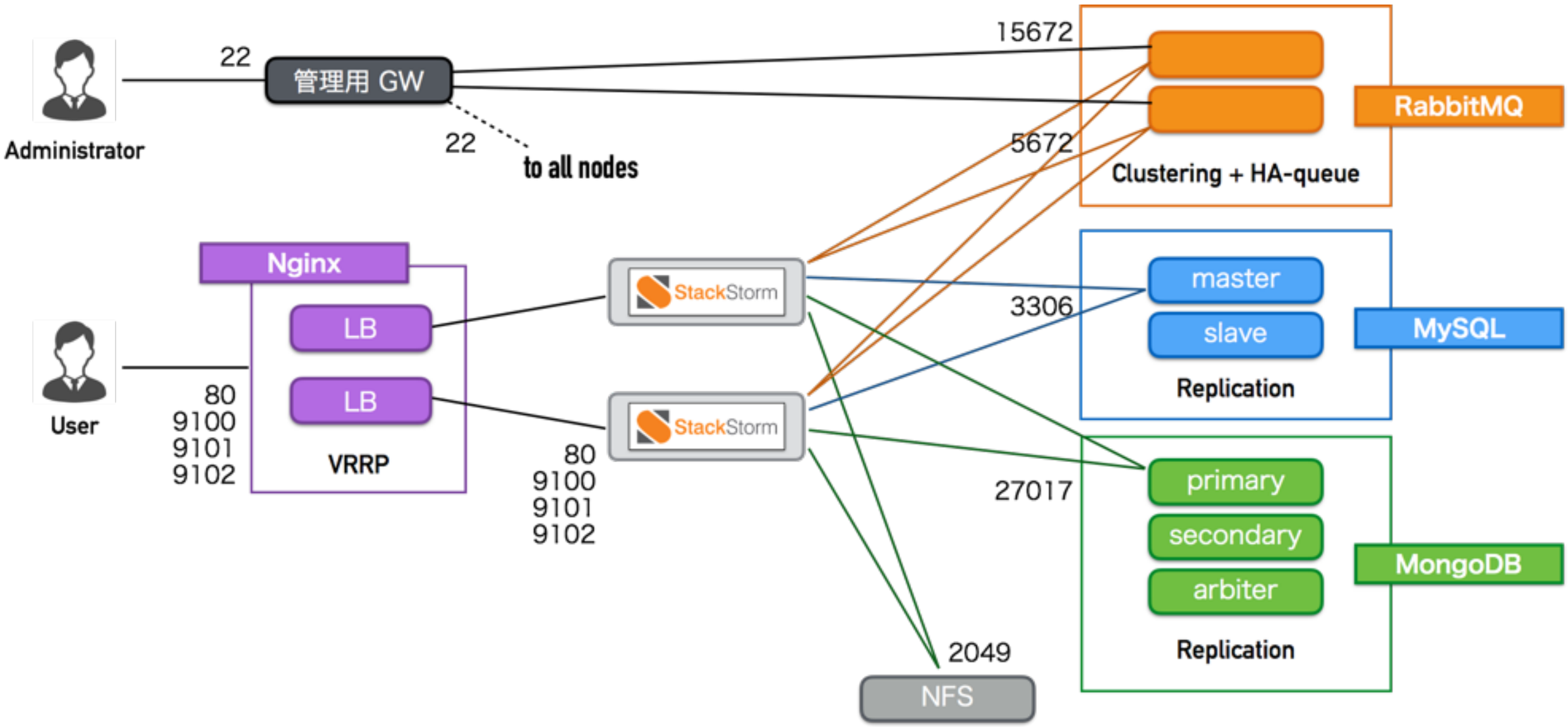
これれらを分離・冗長化させる




(Overview - StackStorm より転載)

# 冗長構成な StackStorm の構築

## 冗長構成な StackStorm 構成例




DMM.com Labo の StackStorm ノード構成

# 冗長構成な StackStorm の構築

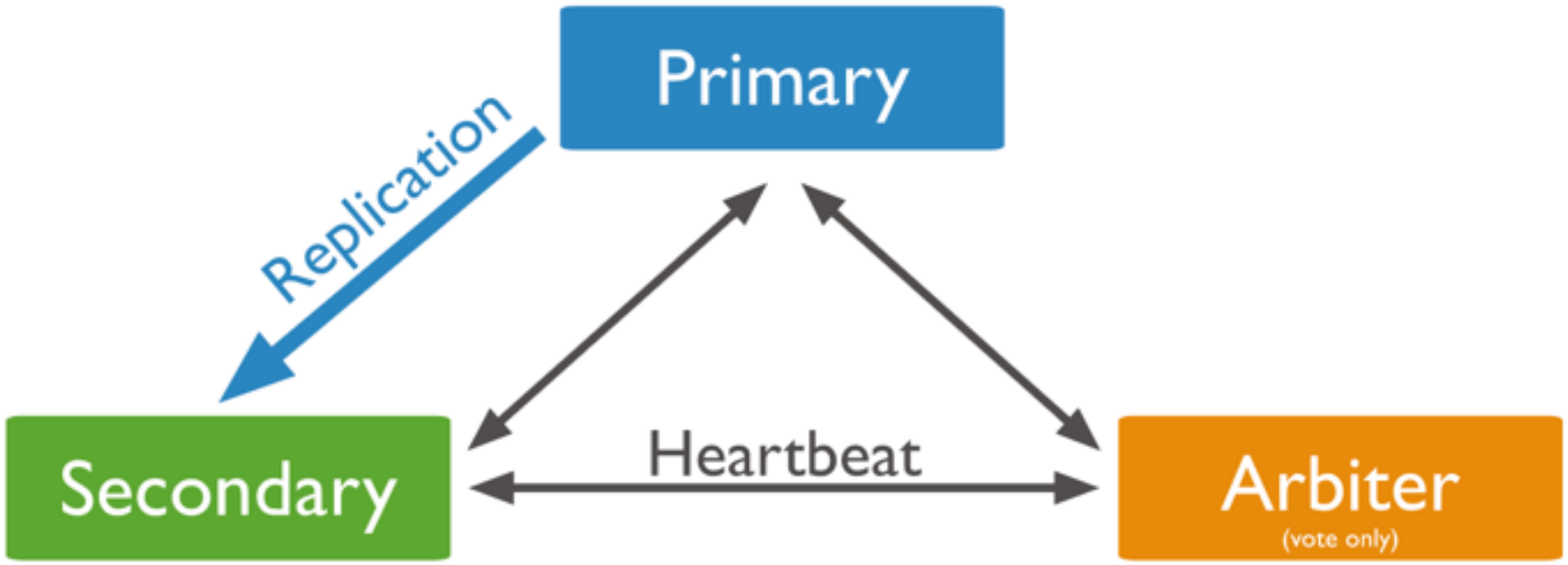
## 構築手順

1. KVS (MongoDB) の分離・冗長化
2. RDB (MySQL) の分離・冗長化
3. MQ (RabbitMQ) の分離・冗長化
4. Nginx の冗長化
5. コンテンツファイルの共有

# 冗長構成な StackStorm の構築

## 構築手順1: KVS (MongoDB) の分離・冗長化

- ３ノードでレプリケーション設定 (Active-Standby 構成)
- 設定方法詳細




(Replication - MongoDB Manual, Replication in MongoDB より転載)

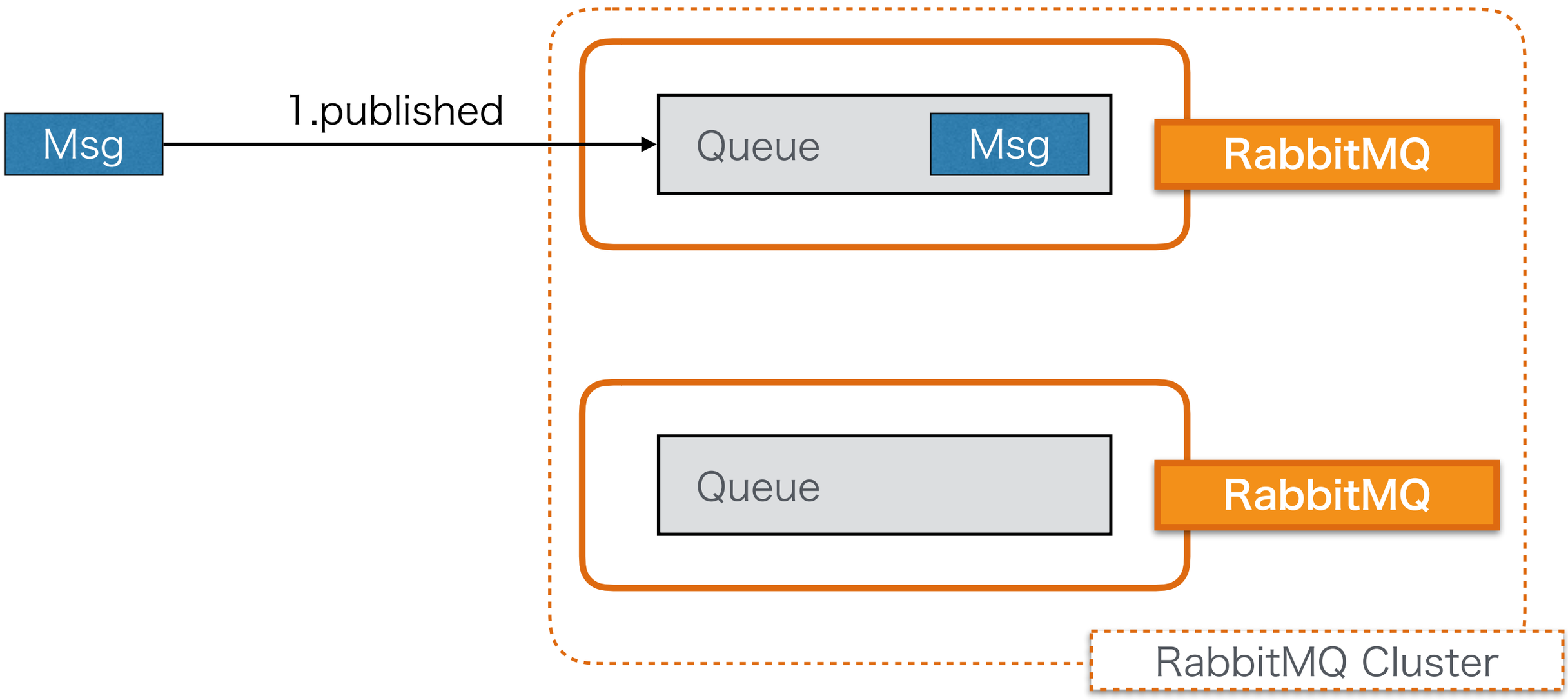
## 構築手順2: RDB (MySQL) の分離・冗長化

- ２ノードでレプリケーション設定 (Active-Standby 構成)
- 設定方法詳細

# 冗長構成な StackStorm の構築

## 構築手順3: MQ (RabbitMQ) の分離・冗長化

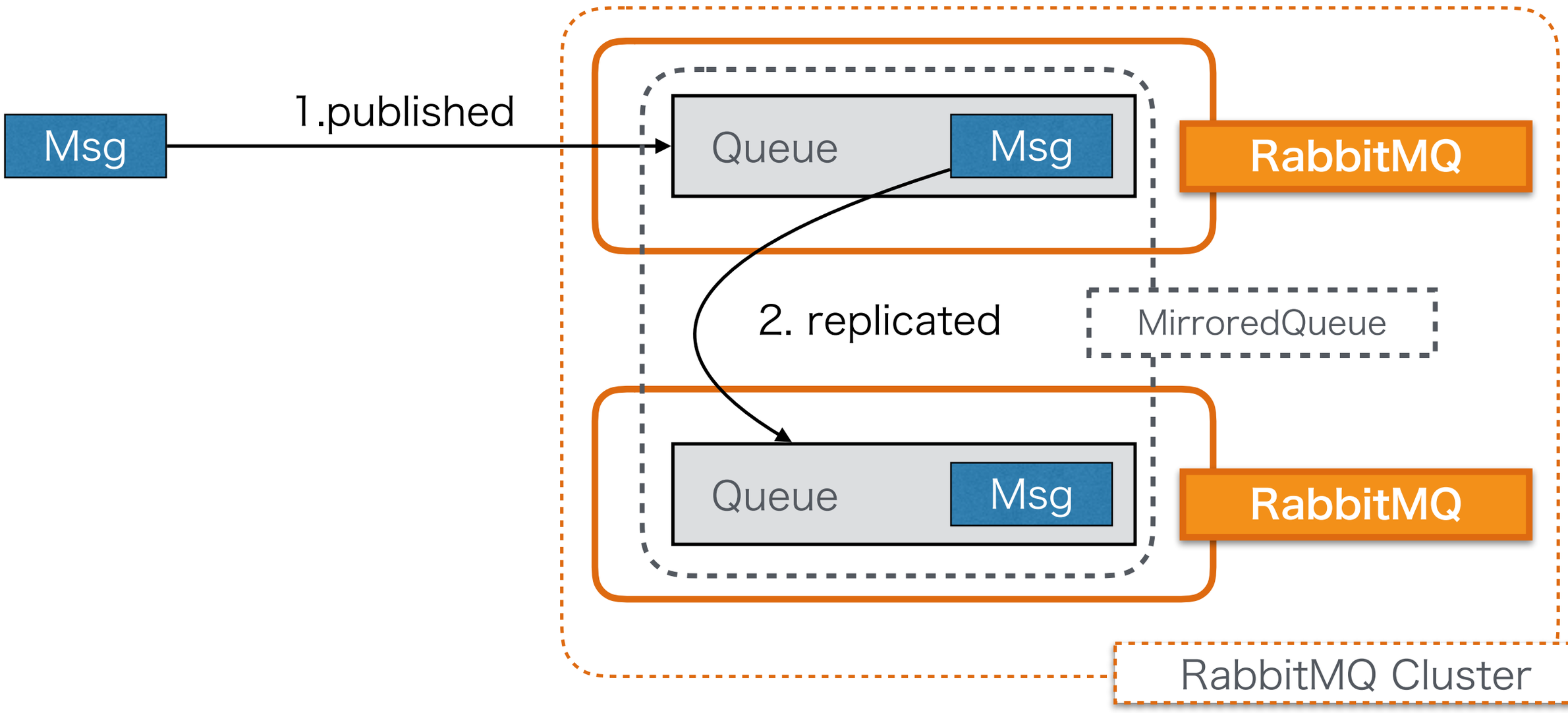
- ２ノードのクラスタで MirroredQueue 設定 (Active-Active 構成)
- 設定方法詳細




RabbitMQ の Cluster 設定

## 構築手順3: MQ (RabbitMQ) の分離・冗長化

- ２ノードのクラスタで MirroredQueue 設定 (Active-Active 構成)
- 設定方法詳細




RabbitMQ の Cluster 設定 + Mirrored Queue

## 構築手順4: Nginx の冗長化

- ２ノードで VRRP を設定 (Active-Standby 構成)
- 設定方法詳細

## 構築手順5: コンテンツファイルの共有

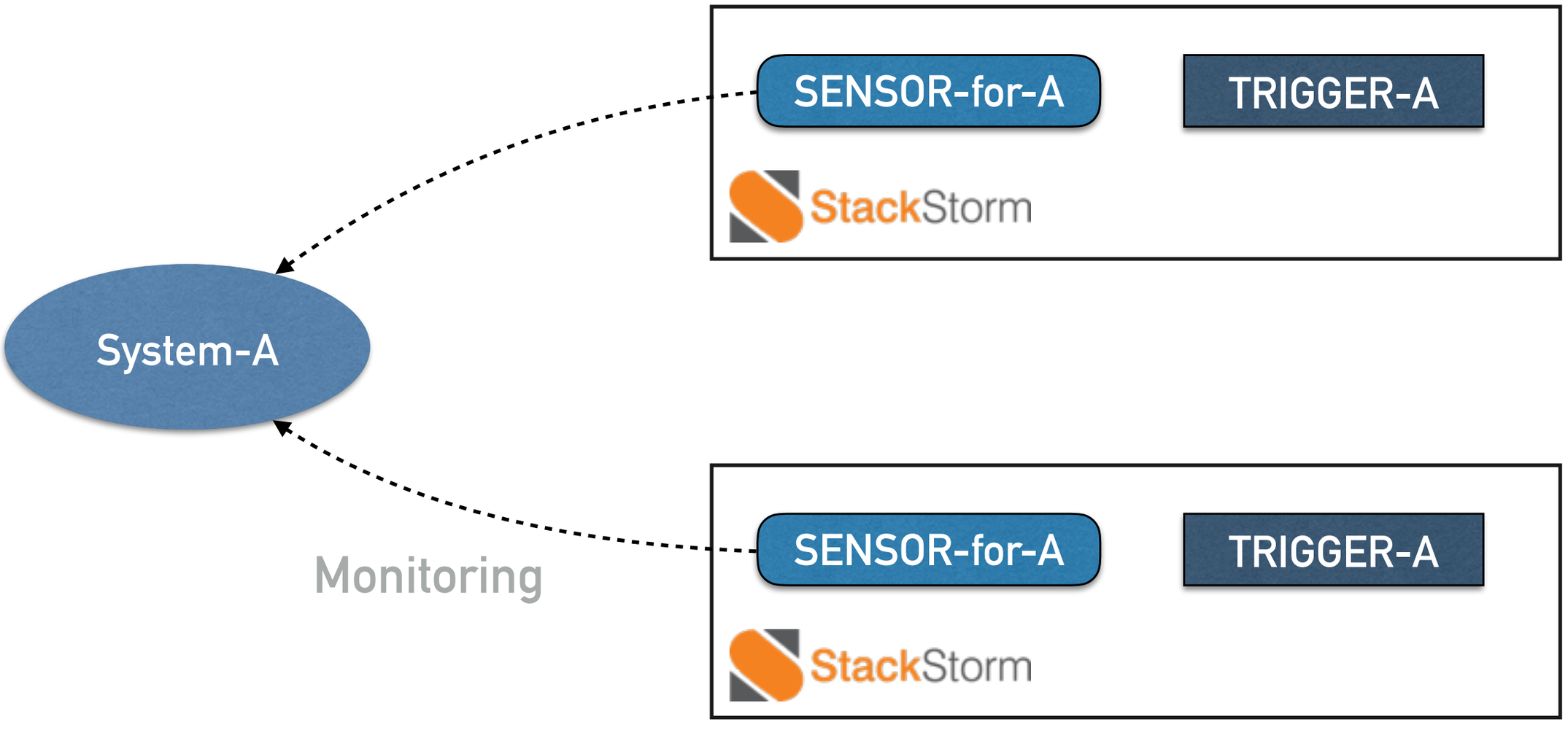
- 以下のファイル群を全 StackStorm ノードで共有
  - /opt/stackstorm/packs
  - /opt/stackstorm/configs
  - /opt/stackstorm/virtualenvs
- 設定例 (NFS設定)

# 冗長構成な StackStorm の注意点

## Trigger 重複起動問題と対応策の紹介

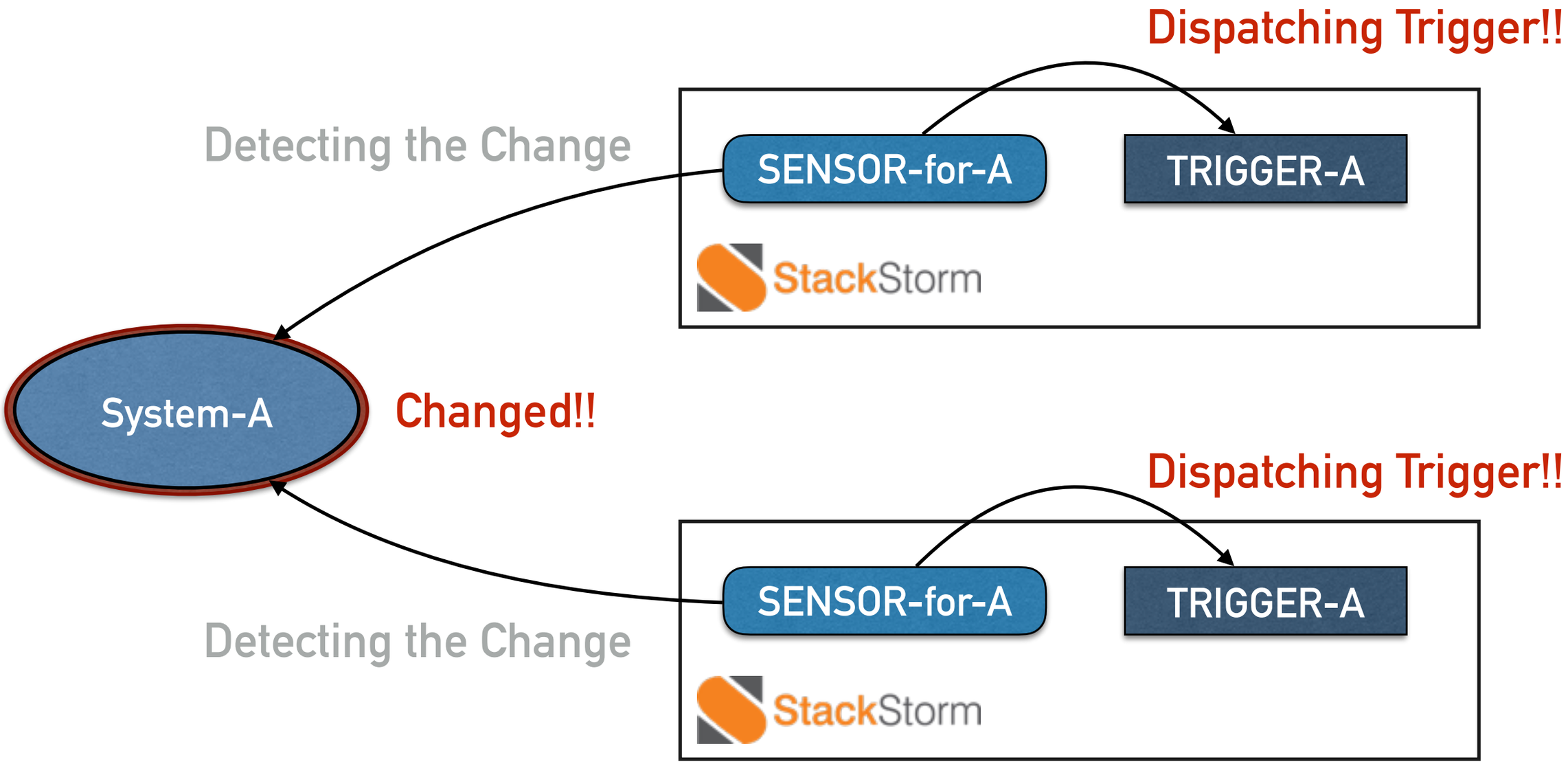
# Trigger の重複起動問題

- 1 イベントに対して複数の検知が発生しうる



# Trigger の重複起動問題

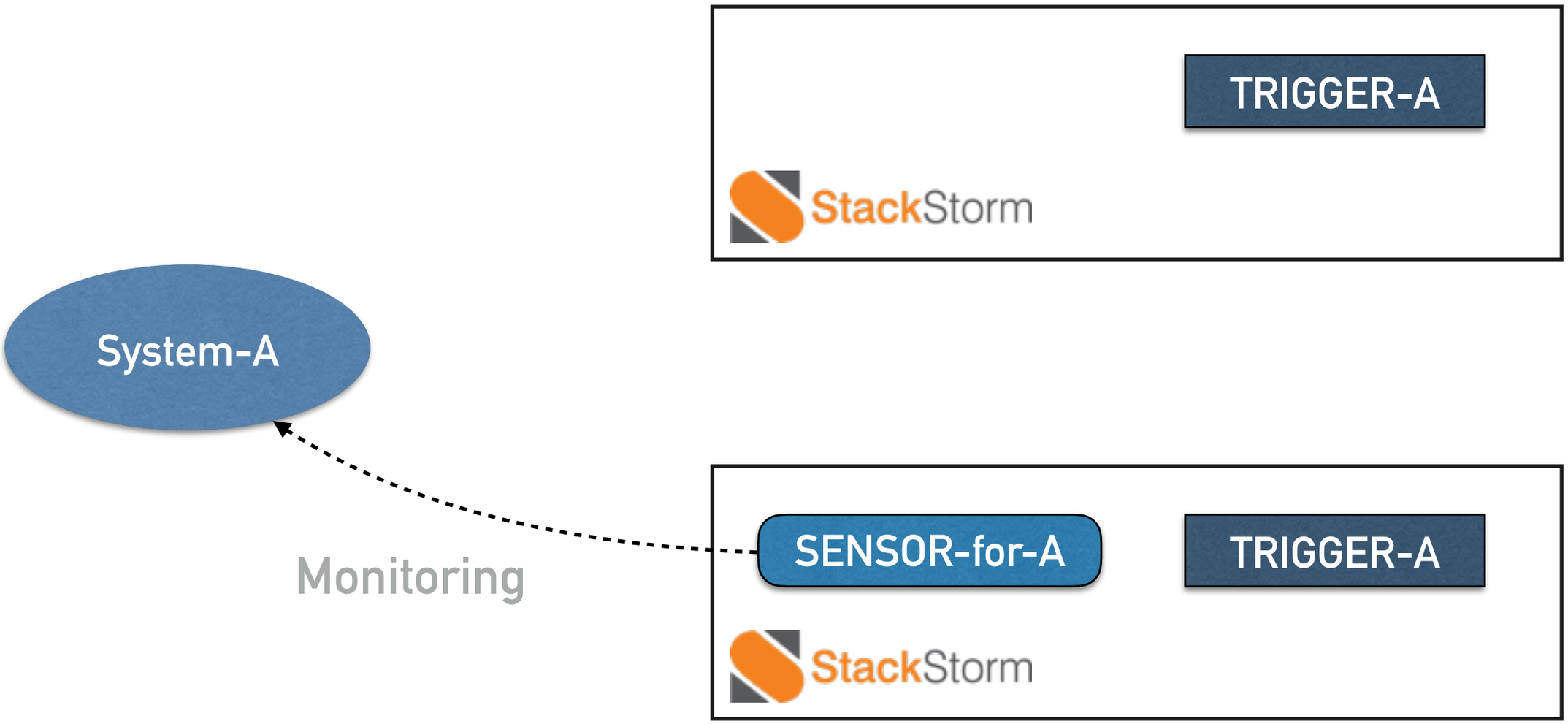
- １イベントに対して複数の検知が発生しうる



# 冗長構成な StackStorm の運用の注意点

## Trigger の重複起動問題の対応 “Partitioning Sensor”

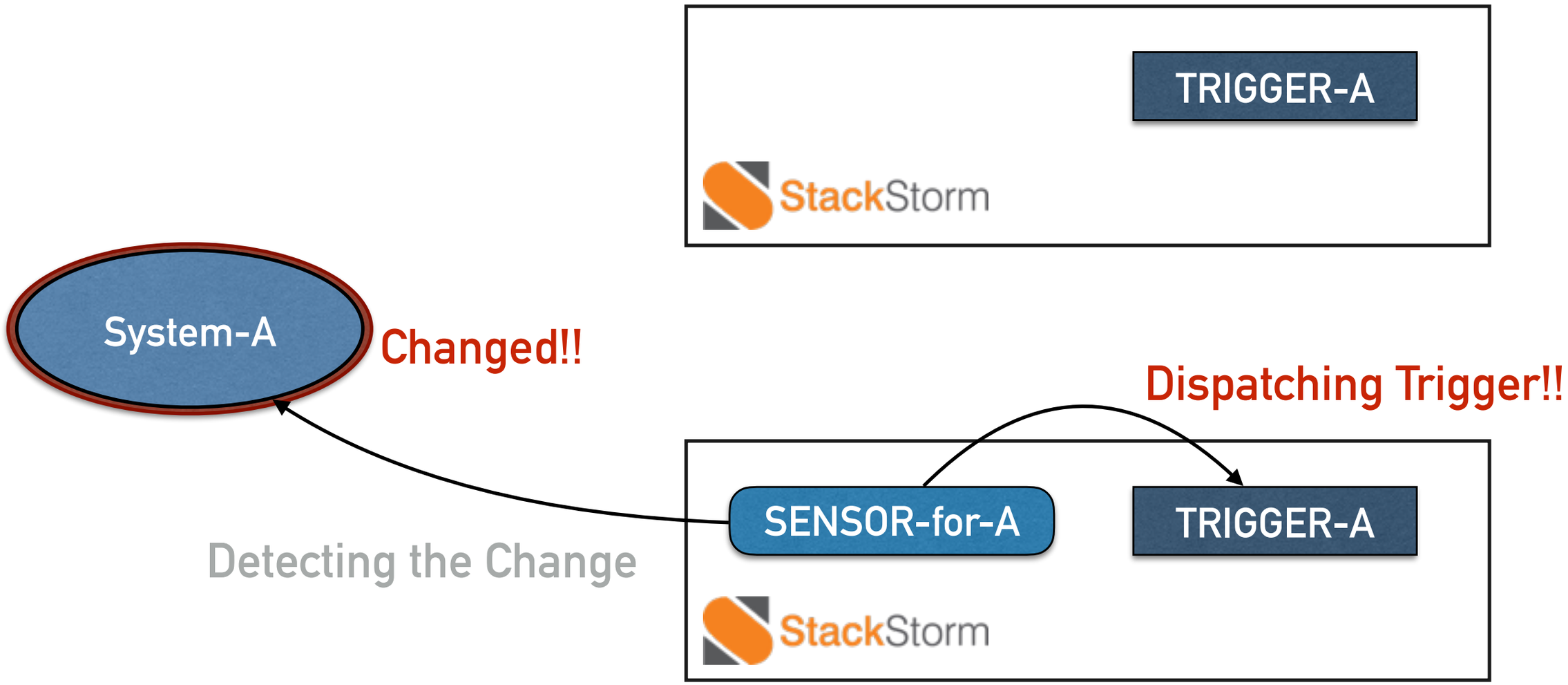
- Sensor を起動するノードを設定により選択する



# 冗長構成な StackStorm の運用の注意点

## Trigger の重複起動問題の対応 “Partitioning Sensor”

- Sensor を起動するノードを設定により選択する



# 冗長構成な StackStorm の運用の注意点

## “Partitioning Sensor” の設定方法

- データストアにノードとセンサプロセスのマッピングを記述

or

- ファイルにノードとセンサプロセスのマッピングを記述

or

- 設定ファイルに起動するセンサの Hash Range を記述

# 冗長構成な StackStorm の運用の注意点

## “Partitioning Sensor” の設定方法 1

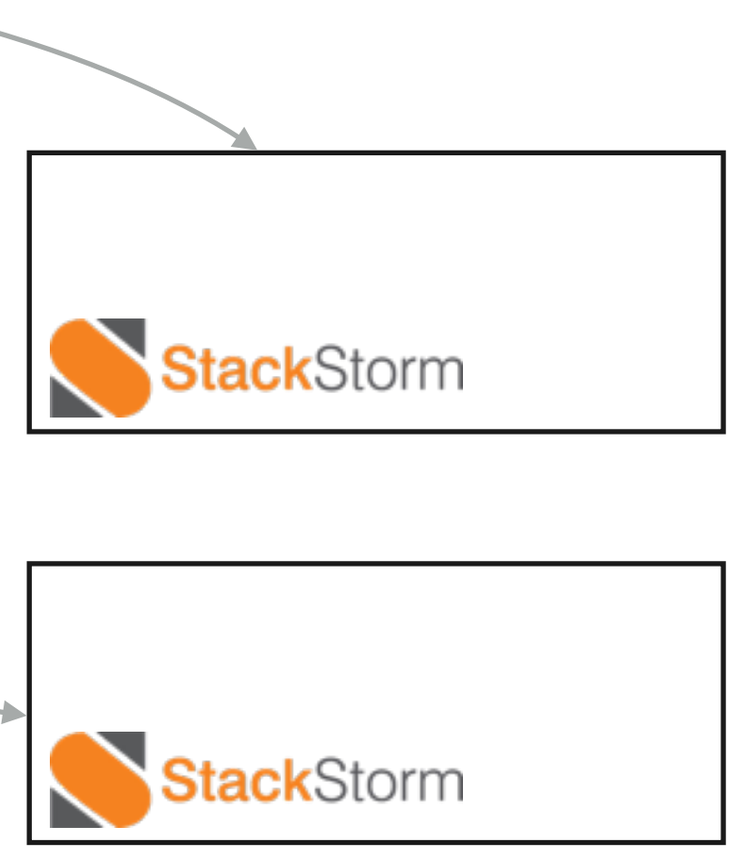
- データストアにノードとセンサプロセスのマッピングを記述

**/etc/st2/st2.conf**

```
[sensorcontainer]
...
sensor_node_name = node1
partition_provider = name:kvstore
```

**/etc/st2/st2.conf**

```
[sensorcontainer]
...
sensor_node_name = node2
partition_provider = name:kvstore
```

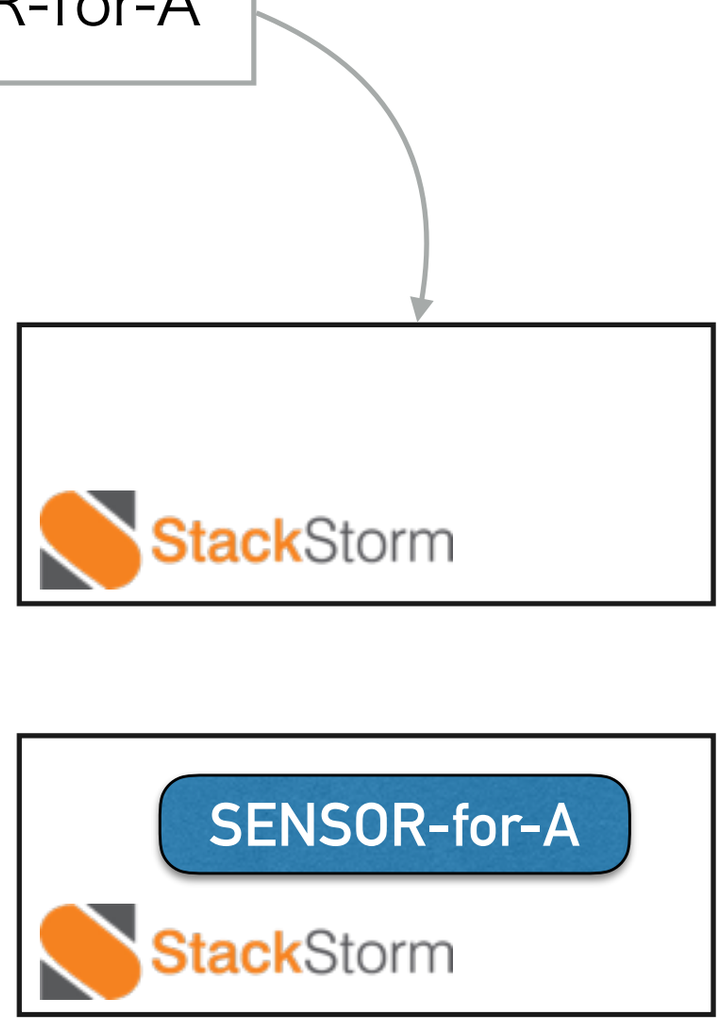


# “Partitioning Sensor” の設定方法 1

- データストアにノードとセンサプロセスのマッピングを記述

shell

```
$ st2 key set node2.sensor_partition "pack.SENSOR-for-A"
```



# 冗長構成な StackStorm の運用の注意点

## “Partitioning Sensor” の設定方法 1

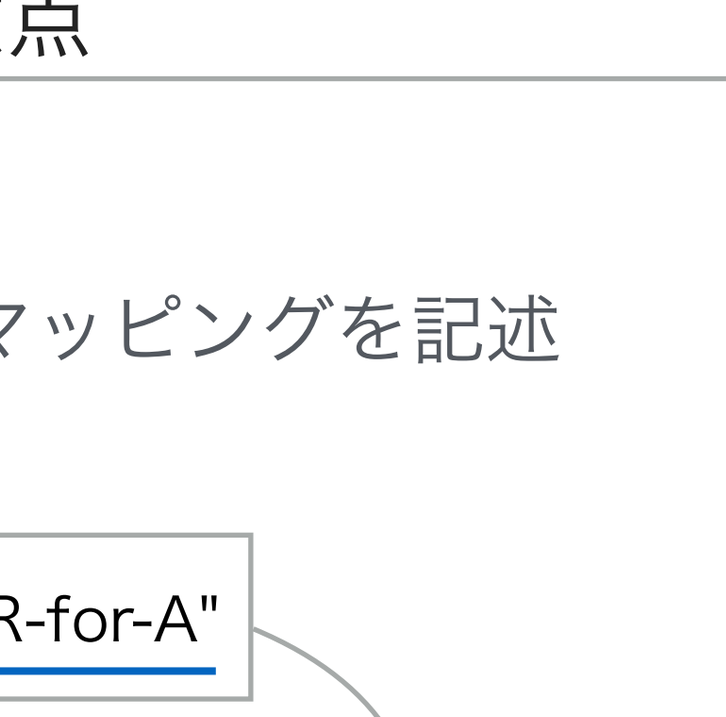
- データストアにノードとセンサプロセスのマッピングを記述

shell

```
$ st2 key set node2.sensor_partition "pack.SENSOR-for-A"
```

ノード名

センサ名



# 冗長構成な StackStorm の運用の注意点

## “Partitioning Sensor” の設定方法２

- ファイルにノードとセンサプロセスのマッピングを記述

**/etc/st2/st2.conf**

```
[sensorcontainer]
...
sensor_node_name = node1
partition_provider =
    name:file, partition_file:/etc/st2/partition_file.yaml
```

**/etc/st2/st2.conf**

```
[sensorcontainer]
...
sensor_node_name = node2
partition_provider =
    name:file, partition_file:/etc/st2/partition_file.yaml
```

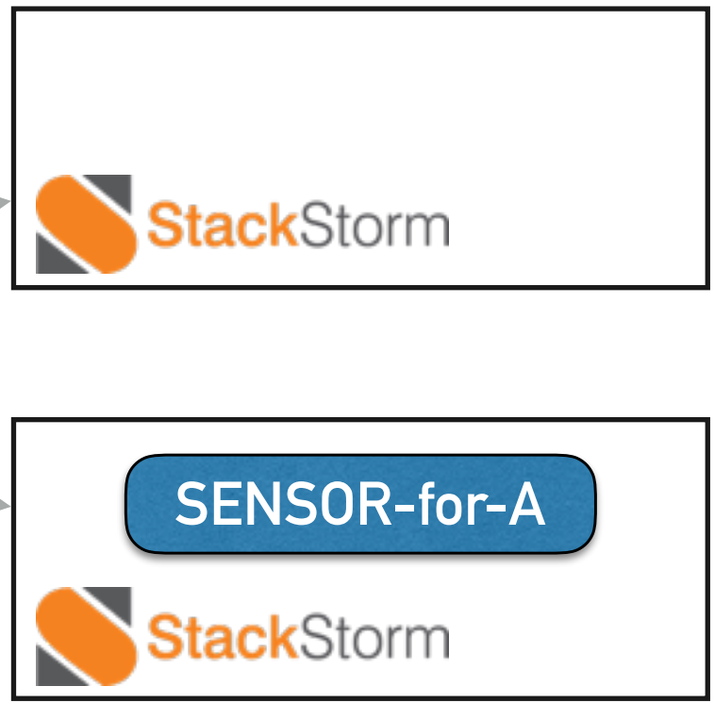

# 冗長構成な StackStorm の運用の注意点

## “Partitioning Sensor” の設定方法２

- ファイルにノードとセンサプロセスのマッピングを記述

/etc/st2/partition_file.yaml

```
---
node1:
  - pack.SENSOR-for-A
```



## “Partitioning Sensor” の設定方法３

- 設定ファイルに起動するセンサの Hash Range (0 ~ 2^32) を記述

Sensor に設定される Hash 値

- e.g. “pack.Sensor-for-A” -> 2658109679

# 冗長構成な StackStorm の運用の注意点

## “Partitioning Sensor” の設定方法３

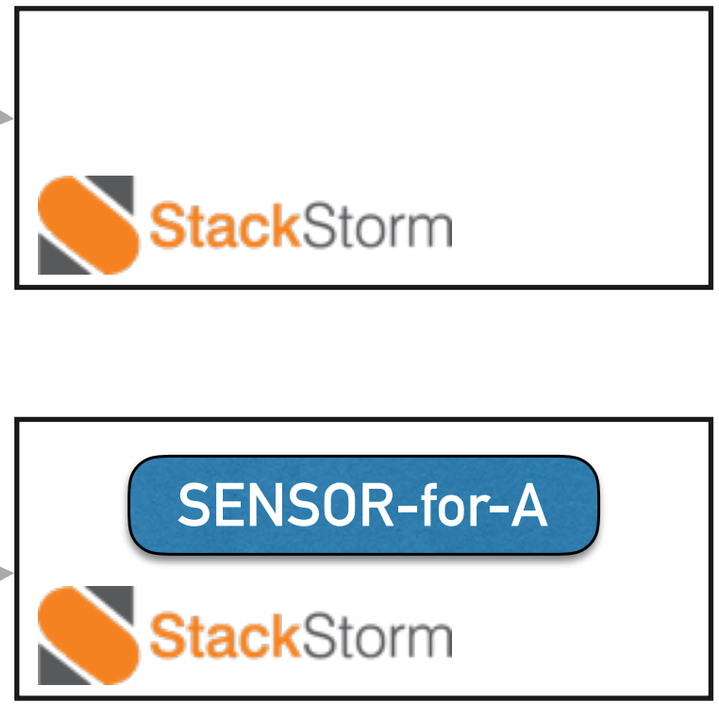
- 設定ファイルに起動するセンサの Hash Range (0 ~ 2^32) を記述

**/etc/st2/st2.conf**

```
[sensorcontainer]
...
sensor_node_name = node1
partition_provider =
   name:hash, hash_ranges:0..2147483648
```

**/etc/st2/st2.conf**

```
[sensorcontainer]
...
sensor_node_name = node2
partition_provider =
   name:hash, hash_ranges:2147483648..MAX
```



# StackStorm のアップグレード

短いメンテ時間で新旧バージョンを切り替える

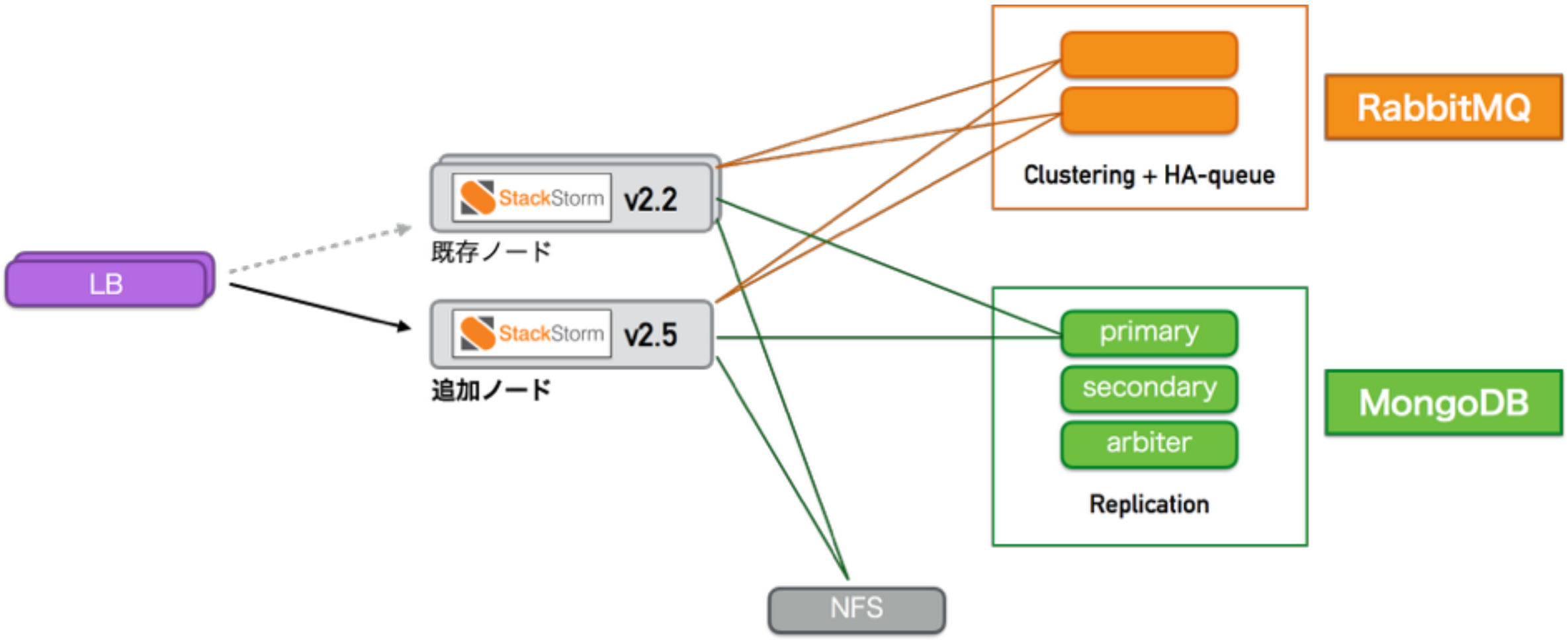
# StackStorm のアップグレード

## 実施要件

- バージョンに関わらず手順を極力同じにする
  - “v2.4~>v2.5” でも “v2.1~>v2.5” でも同様の手順で実施できる
- メンテナンス時間は極力短く (数分以内) する
- 問題が生じた場合には直ちに切り戻せる

## 実施概要

- 移行後の StackStorm ノードを構築し、LB を切り替える




“v2.2~>v2.5” へのアップグレードの実施概要

## 実施手順

1. 新ノードの構築
2. RabbitMQ クラスタの設定変更
3. StackStorm 設定ファイルの修正
4. Nginx (LoadBalancer) 設定ファイルの修正 (ノード追加・修正)

## 実施手順１：新ノード構築

- package からインストール：詳細 (Installation - documentation)

# 実施手順２：RabbitMQ クラスタの設定変更

- 新バージョン用 vhost ”st2_v2_5” を追加

**shell @RabbitMQ_Node**

```
$ sudo rabbitmqctl add_vhost /st2_v2_5
$ sudo rabbitmqctl set_permissions -p /st2_v2_5 mquser ".*" ".*" ".*"
```

# 実施手順３：StackStorm 設定ファイルの修正

- /etc/st2/st2.conf を修正

  - RabbitMQ ノード設定の変更
    (localhost ~> RabbitMQクラスタ)

  - データストアノード設定の変更
    (host　　: localhost ~> リモートの MongoDB ノード)
    (database: st2 ~> st2_v2_5)

- 設定変更内容 (diff)

## 実施手順４：Nginx 設定ファイルの修正

- /etc/nginx/conf.d/st2.conf を修正
  - 以下に対するリクエストの転送先を変更
    (st2web, st2api, st2auth, st2stream)
- オリジナルの設定内容
- 設定変更内容 (diff)

# さいごに

# まとめ

## 以下の運用プラクティスについて共有

1. st2-auth-backend-ldap による AD 認証
   - OSS を利用して AD 認証を実現する方法を紹介

2. 冗長構成な StackStorm 環境の構築
   - SPOF を排した High Available なサービス環境を構築する方法を紹介

3. StackStorm のアップグレード
   - バージョンに依存しないアップグレード方法を紹介