应急预案告警平台模块优化技术方案

摘要

编写目的

背景

```
Trigger MessageResolved Message

1.Trigger MessageResolved MessageJSONgo template
  go templateJSON
  ,
2.Custom WebhookMessage
  MessageJSONMessageMessage
  Message
```

任务概述

```
1.Trigger MessageResolved Message
2.MessageMessage
3.
```

目标人员

规范与约定

代码规范

大促告警平台预案告警使用规范

参考资料

告警平台webhook主体变量说明

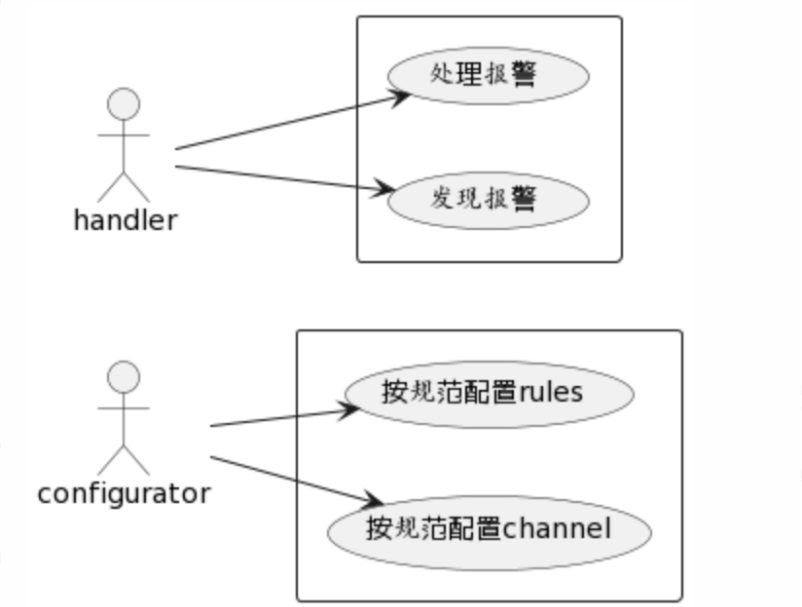
告警消息模版变量说明

系统分析设计

系统设计目标

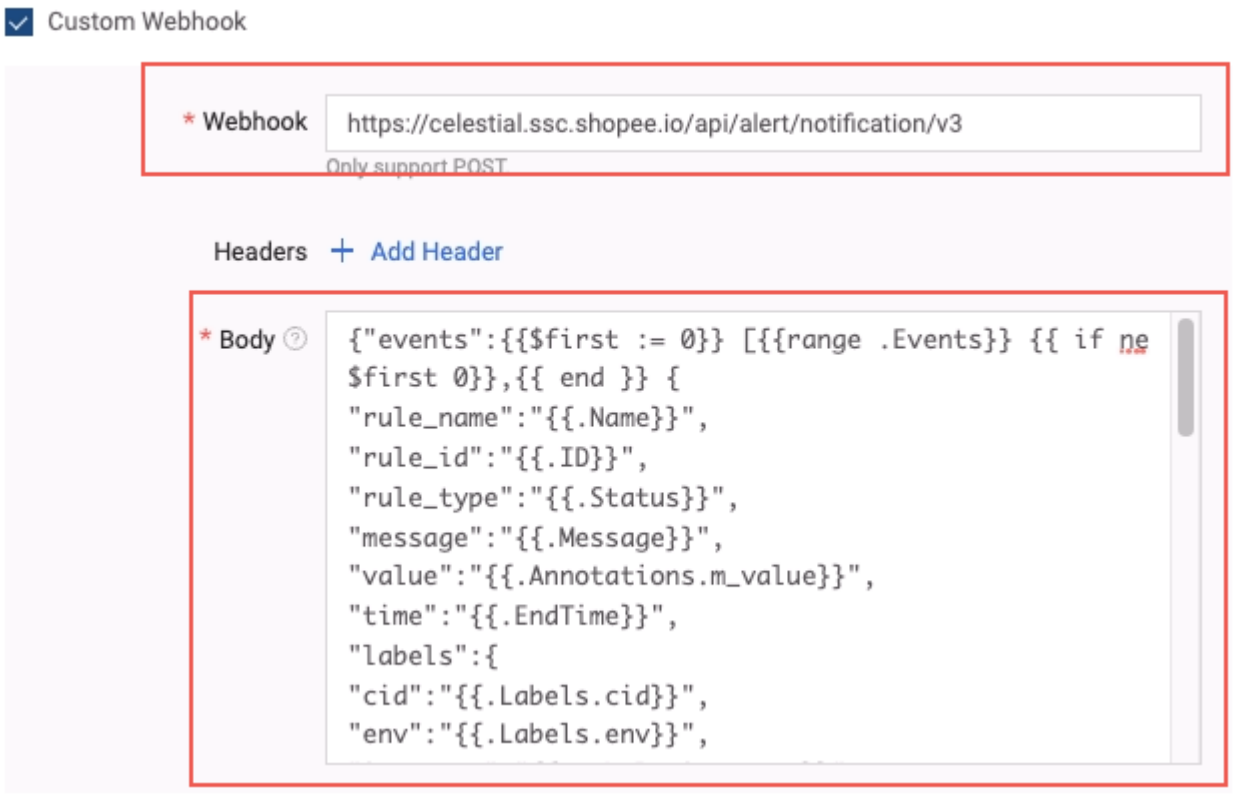
```
929
1.90%90%
2.MessageMessage
3.
4.
```

用例分析



本次升级涉及用例更改主要在配置方包括：
1.在告警平台配置rule时不需要在Trigger Message和Resolved Message中聚合太多配置，只需要配置一个唯一标识告警信息的key就好了，格式：{plan_seq}#PROM|{xxx} 例如：LFS-02#PROM|LFS CHECKOUTGRPC BatchLogisticsServiceableV2 > 200ms

2.channel配置默认抓取大部分数据并进行JSON化传递给大促系统，大促系统会对数据进行解析和组装，如下图：



为了识别监控事件，channel中一些fied是必须配置的，详情如下表：

channel_field_name	type
rule_id	required
rule_name	required
rule_type	required
plan_seq	required
time	required
value	required

为了兼容之前的监控配置，channel中预留的一些filed与监控采集到的labels中数据是一一对应的，这些filed可以根据实际监控项按需配置，也可以直接使用下面的配置模版，对应关系如下表：

channel_field_name	collect_data_name	type
cid	labels.cid	general

env	labels.env	general
instance	labels.instance	general
addr	labels.addr	general
service	labels.service	general
application	labels.application	general
proxy	labels.proxy	general
cluster_name	labels.cluster_name	mysql
cluster_role	labels.cluster_role	mysql
target_instance	labels.target_instance	mysql
db_name	labels.db_name	mysql
db_role	labels.db_role	mysql
alias	labels.alias	codis
cmd	labels.cmd	codis
pod_name	labels.pod_name	container
mon_pod	labels.mon_pod	container
mon_env	labels.mon_env	container
mon_cid	labels.mon_cid	container
mon_mod	labels.mon_mod	container
mon_proj	labels.mon_proj	container
pod	labels.pod	container
namespace	labels.namespace	container
topic	labels.topic	mq
consumergroup	labels.consumergroup	mq
vhost	labels.vhost	mq
cluster	labels.cluster	mq
queue	labels.queue	mq
operation	labels.operation	others
url	labels.url	others
code	labels.code	others

ps:目前大促系统只适配了以上这些字段

channel参考模版

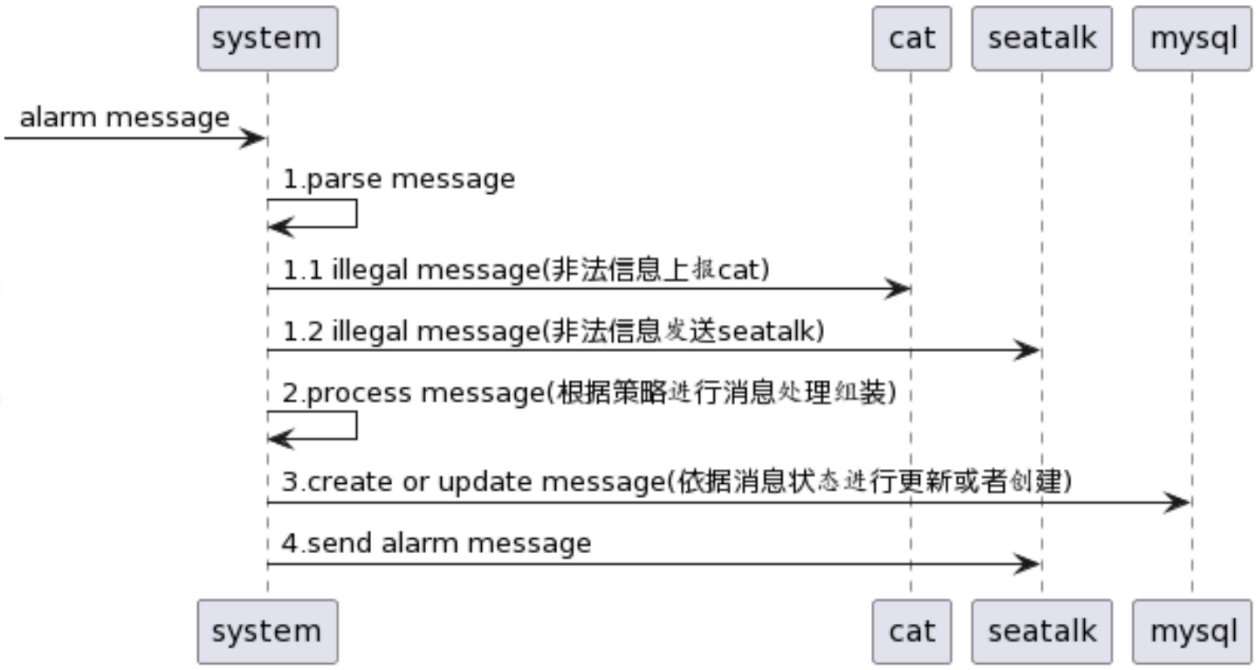
配置数据模版：

```
{"events":{{$first := 0}} [{{range .Events}} {{ if ne $first 0}},{{ end }} {
"rule_name":"{{.Name}}",
"rule_id":"{{.ID}}",
"rule_type":"{{.Status}}",
"message":"{{.Message}}",
"value":"{{.Annotations.m_value}}",
"time":"{{.EndTime}}",
"labels":{
"cid":"{{.Labels.cid}}",
"env":"{{.Labels.env}}",
"instance":"{{.Labels.instance}}",
"addr":"{{.Labels.addr}}",
"service":"{{.Labels.service }}",
"application":"{{.Labels.application}}",
"proxy":"{{.Labels.proxy}}",
"cluster_name":"{{.Labels.cluster_name}}",
"cluster_role":"{{.Labels.cluster_role}}",
"target_instance":"{{.Labels.target_instance}}",
"db_name":"{{.Labels.db_name}}",
"db_role":"{{.Labels.db_role}}",
"alias":"{{.Labels.alias}}",
"cmd":"{{.Labels.cmd}}",
"pod_name":"{{.Labels.pod_name}}",
"mon_pod":"{{.Labels.mon_pod}}",
"mon_env":"{{.Labels.mon_env}}",
"mon_cid":"{{.Labels.mon_cid}}",
"mon_mod":"{{.Labels.mon_mod}}",
"mon_proj":"{{.Labels.mon_proj}}",
"pod":"{{.Labels.pod}}",
"namespace":"{{.Labels.namespace}}",
"topic":"{{.Labels.topic}}",
"consumergroup":"{{.Labels.consumergroup}}",
"vhost":"{{.Labels.vhost}}",
"cluster":"{{.Labels.cluster}}",
"queue":"{{.Labels.queue}}",
"operation":"{{.Labels.operation}}",
"url":"{{.Labels.url}}",
"code":"{{.Labels.code}}"}{{ $first = 1 }} {{end}}]}
```

对应格式数据：

{"events":[{
"rule_name":"test",
"rule_id":"1234",
"rule_type":"firing",
"message":"LPS-03#PROM|LFS CHECKOUTGRPC BatchLogisticsServiceableV2 > 200ms",
"value":"1",
"time":"",
"labels":{
 "cid":"sg",
 "env":"test",
 "instance":"",
 "addr":"",
 "service":"",
 "application":"",
 "proxy":"",
 "cluster_name":"",
 "cluster_role":"",
 "target_instance":"",
 "db_name":"",
 "db_role":"",
 "alias":"",
 "cmd":"",
 "pod_name":"",
 "mon_pod":"",
 "mon_env":"",
 "mon_cid":"",
 "mon_mod":"",
 "mon_proj":"",
 "pod":"",
 "namespace":"",
 "topic":"",
 "consumergroup":"",
 "vhost":"",
 "cluster":"",
 "queue":"",
 "operation":"",
 "url":"",
 "code":""}
}]
}

核心业务规则



1.直接获取解析alarm message，相应结构如下：

```
PromAlertReq struct {
   Events []Event `json:"events"`
}
Event struct {
   RuleName string            `json:"rule_name"`
   RuleId   string            `json:"rule_id"`
   RuleType string            `json:"rule_type"`
   Message  string            `json:"message"`
   Value    string            `json:"value"`
   Time     string            `json:"time"`
   Labels   map[string]string `json:"labels"`
}
```

ps:如果消息非法，捕获发送到seatalk进行排查并且上报到cat上（5分钟内的重复异常信息不会发送）

2.默认策略是针对一条message每次都会获取最新的event来进行处理和组装等操作，组装结构如下：

```
AlertEntry struct {
   AlertType    string  `json:"alert_type"`
   AlertTime    string  `json:"alert_time"`
   AlertName    string  `json:"alert_name"`
   AlertContent string  `json:"alert_content"`
   PrePlanSeq   string  `json:"pre_plan_seq"`
   Key          string  `json:"key"`
   Value        float32 `json:"value"`
   ViewDetail   string  `json:"view_detail"`
   Env          string  `json:"env"`
   Message      string  `json:"message"`
   Cid          string  `json:"cid"`
   TargetType   string  `json:"target_type"`
}
```

alertentry_fied_name	promalert_fied_name	description
AlertType	RuleType	告警类型：error，resolved
AlertTime	Time	告警时间
AlertName	Message	由用户按规范配置

AlertContent	/	/
PrePlanSeq	Message	从Message中解析出
Key	Message	从Message中解析出
Value	Value	/
ViewDetail	RuleId	由RuleId+固定前缀生成
Env	Labels	从Labels中获取
Message	Labels	由Labels中的k-v按照固定顺序组装而成（剔除掉Env,Cid）
Cid	Labels	从Labels中获取
TargetType	/	/

非功能特性设计

可靠性

```
1.Trigger MessageResolved Message
2.Message
3.
```

可扩展性

```
1.template
2.Event
```