# Alerting

## Monitoring

We currently have several monitoring systems:

* ProdMon
* Cloudwatch
* Sentry.io
* Lambdas

These all emit ‘failures’ - either explicit errors or failure states based on exceeding thresholds.

Of these, the most important is prodmon; it tests if we can currently create and publish content.

### ProdMon

Actually uses the front end. Therefore the most truthful indicator of serious front end problems.

However, does not test the end result (a published front). Also does not test the front tool itself.

### Cloudwatch

Cloudwatch is the simplest attempt to monitor. Most alerts seem to be configured to send email to Pagerduty, which is adequate, if basic. All alerts should belong to the stack they monitor, and ideally be created using Cloudformation.

### Sentry.io

Used to spot problems at the client (javascript/browser). Results are not usually alert on, but are used for analysis after the fact if a problem is prioritised.

## Lambdas

Used for routing of success/fail messaging between services. There is some confusion because the alerting process is not necessarily in the same account as the process sending the success/fail.

In analysis of the front-press-lambda, there is little attempt to handle ‘self clearing’ errors.

## General observations

There is little attempt to include information about what to *do* with an alert. A run book link and/or a link to the service itself would be huge improvement.

For example, if a front fails the press process, a link to the front complete with the error would be useful.

## Notification

We can elect to have these conditions notified. There are effectively three levels of notification:

* Immediate push (eg to phone)
* Update a dashboard and/or push during business hours
* Log only (eg email)

Our intention is to turn off all notification which is not explicitly actionable.

We therefore require, as a first step, each alert to have a link embedded in it which describes the action(s) which can be taken.

Any alert which does not have an action, cannot be actionable, and will be silenced.

### Rota Strategy

Alerts will be sent to a rota, which will be maintained outside of this process.

#### Janus

24/7 Team currently have more permissions during their ‘on’ shift.  
Janus should be considered the source of truth for responsibility.

#### PagerDuty

See https://gupage.pagerduty.com

As the deliverer of alerts, PagerDuty needs to be aware of the current holder of the office.

However, we do not wish to duplicate the maintenance of the rota, or write a shim to copy the rota to PD from Janus.  
Instead we will require any person going ‘off’ shift to update PagerDuty to mark the next holder as ‘on’ shift.

## Current Notification Strategy

Cloudwatch alerts place a notification on an SNS topic, which sends email to a specific email integration address within PD.

## Possible Future Strategy

#### Anghammerad

Currently, Anghammerad routes messages from SQS to email or GChat, according to the tags on the message.

CloudWatch can already notify SQS. A small piece of work would make Anghammerad aware of (1) Cloudwatch as an input format and (2) PagerDuty as an endpoint.

Cloudwatch alerts could then be built into each stack, and automatically routed to either PagerDuty or GChat depending on App/Stack/Stage.

However, Anghammerad requires a message, which would need to be enriched onto the cloudwatch message at some stage.

## Responders

Some alerts are either immediately actionable or too important to ignore. Others may fix themselves (eg network glitching).

The appropriate responder should be selected for each alert.

### 24/7

Alerts which represent an immediate loss of functionality in the production of user facing content should always go to 24/7.

### CAT Rota

Alerts which represent a degradation of functionality (but not loss) may go to CAT rota. For example a failure to publish could be sent to CAT, but a continued failure should go to 24/7.

## Suggested actions.

### Improving noise

* Document available actions for any alert, whether on or off, in an agreed location where it can be deep-linked in the alert.

AND

* Give blanket permission to turn off any alerts permanently if it is not immediately actionable.

### Improving coverage

* Expand ProdMon to cover the Fronts tool.

### Improving management

(Some of these may already be satisfactory)

* Use Pagerduty AND GChat for notifications.
* Do not use PagerDuty for rota management, but require the person going ‘off-rota’ to hand over to the person going ‘on rota’.
* Provide an easy way for a support person to find the rota.