yelp

What I Wish I Knew Before Going On-call

Survey:

http://bit.ly/survey-srecon-oncall or

SCAN ME

Chie Shu

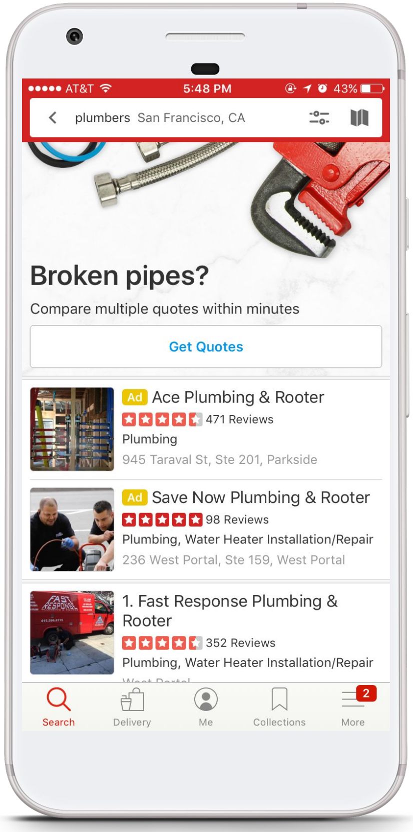
Software Engineer

chie@yelp.com

WHO WE ARE

Yelp **Local Ads**

- Connect people with great local businesses
- Advertiser billing and analytics



Broken pipes?

Compare multiple quotes within minutes

Ad Ace Plumbing & Rooter
471 Reviews
Plumbing
945 Taraval St, Ste 201, Parkside

Ad Save Now Plumbing & Rooter
98 Reviews
Plumbing, Water Heater Installation/Repair
236 West Portal, Ste 159, West Portal

1. Fast Response Plumbing & Rooter
352 Reviews
Plumbing, Water Heater Installation/Repair

Our team’s challenges

1. Financially critical systems

~90% of company revenue is from ads

2. Wears many hats 🎩

On-call + Feature + Infra

3. Owns systems with many different tech stacks

Makes being on-call more challenging

4. Majority of the team is new grad hires

Makes onboarding even more important

My story

- Joined the team as a new grad hire
- Learned how to be on-call the hard way...
- Now mentoring other engineers

STORY TIME: BEING ON-CALL

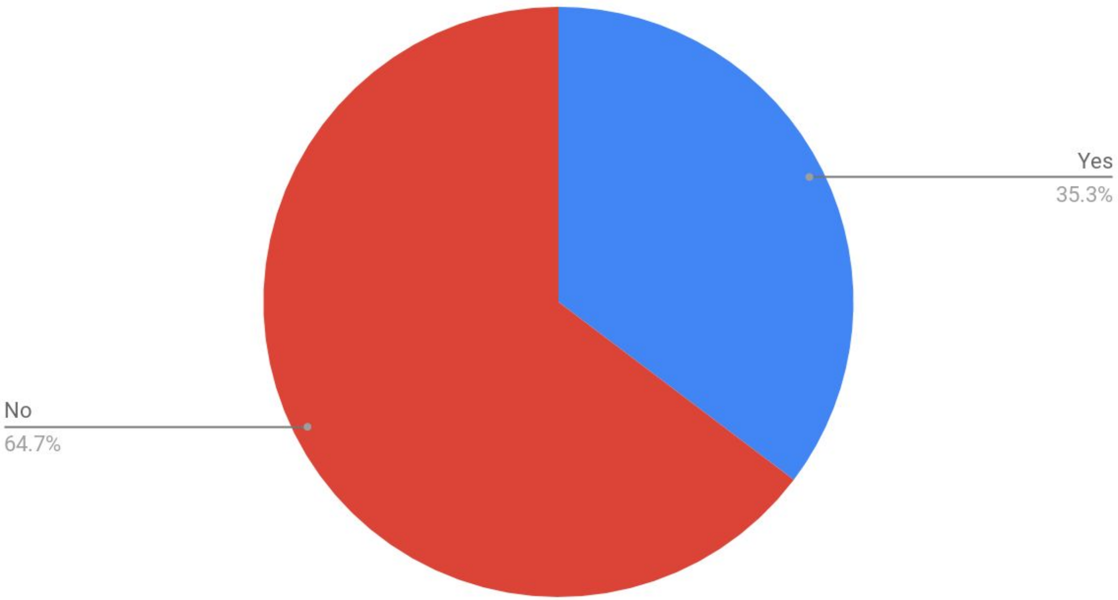
Newbie on-call struggles

- No established training process
- Decentralized + Outdated documentations
- So much financial impact/pressure!

SURVEY RESULTS

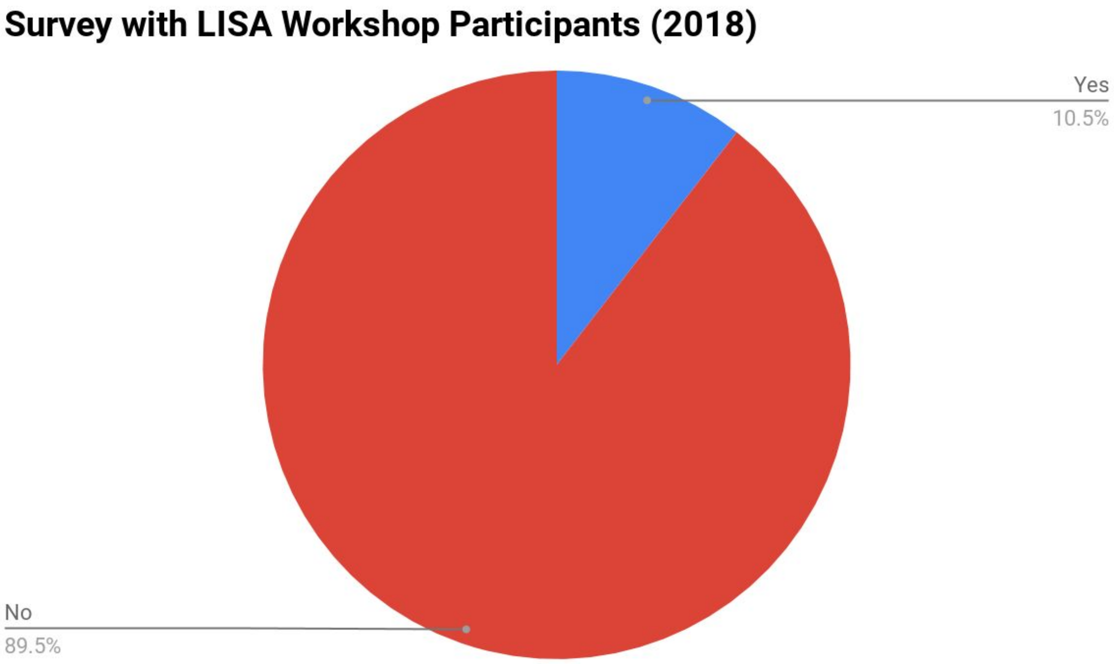
Did you feel **ready** before going on-call for the first time?

Survey within Yelp Engineering (2018)



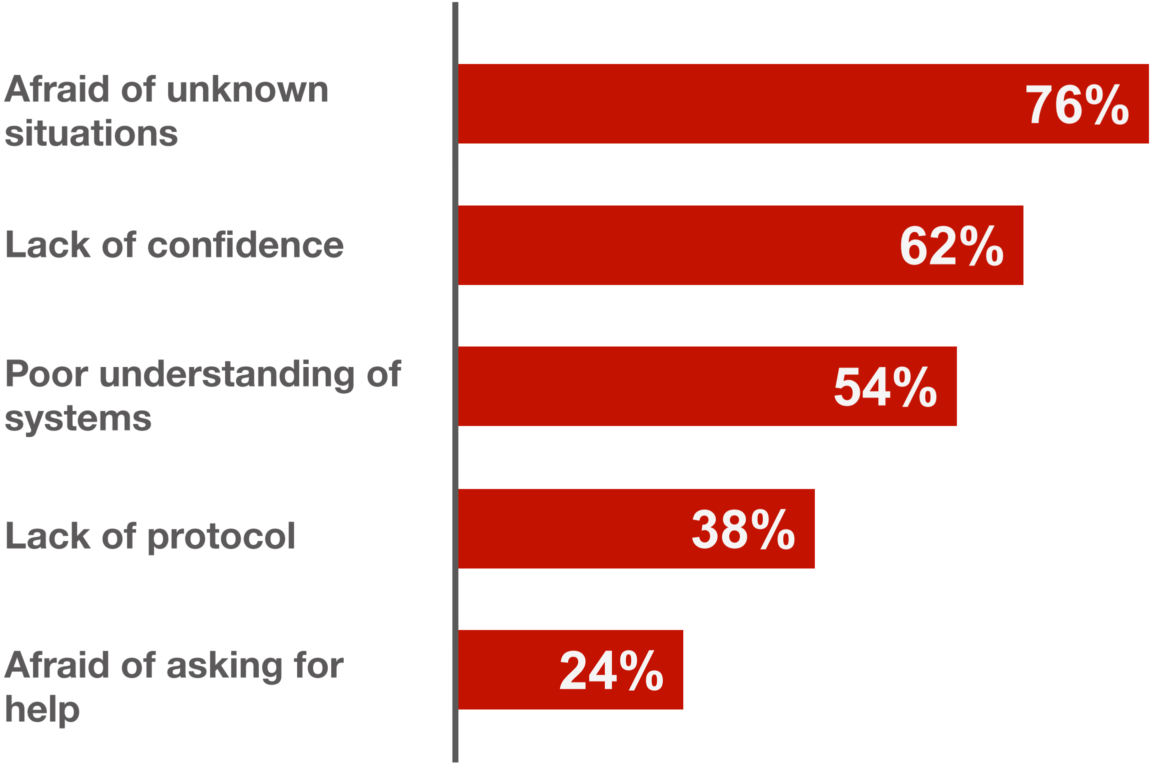
SURVEY RESULTS

Did you feel **ready** before going on-call for the first time?



SURVEY RESULTS

Why didn't you feel ready?



Survey within Yelp Engineering (2018)

Why care about good onboarding?

Win 1: Makes your team scalable!

Win 2: Improve incident response

Win 3: Teaching is the best way to learn

Win 4: Confident new oncall engineers

Workshop Goal

Build an efficient on-call onboarding system for your organization

Agenda

1. Common Myths about On-Call
2. How to Create Training Program
3. Runbook for Effective Incident Response

4 Common Myths About On-calls

Myth #1

“I need to know everything”

You are not supposed to know everything

Myth #2

“I need to solve everything by myself”

You are supposed to ask for help

Myth #3

“I need to find the root cause”

Root cause finding is a non-goal

Myth #4

“I need to make the best/long-term fix”

Mitigate the issue in the safest way!

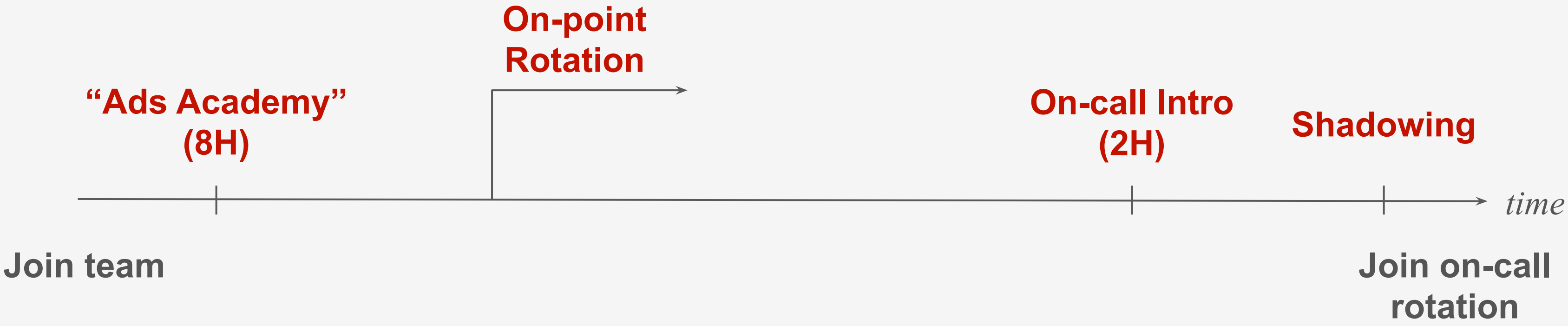
Setting the right **expectations**

1. **Reduce (unnecessary) fear**
2. **More productive + efficient on-call**

Set the right **expectations** during training!

Now onto the training program...

My On-call "Training"



What was good about my training?

- It existed
- On-point rotation
- Shadowing

What was difficult about my training?

- Information dump
- No emphasis on connections between systems
- No emphasis on investigation/debugging tools

The Goal of Training Program

Goal 1.
Be able to draw a mental **picture of your system**

Goal 2.
Understand **failure modes/alerts** for the system

Goal 3.
Know the **tools** for investigation

The Goal of Training Program

Goal 1.
Be able to draw a mental picture of your system

Goal 2.
Understand failure modes/alerts for the system

Goal 3.
Know the tools for investigation

The Goal of Training Program

Goal 1.
Be able to draw a mental picture of your system

Goal 2.
Understand failure modes/alerts for the system

Goal 3.
Know the tools for investigation

Exercise

Let’s make an on-call training program!

Exercise Agenda

Let’s make an on-call training program!

1. Make a Curriculum
2. Create Introduction
3. Cover Failure Modes
4. List Necessary Tools

What you need:

Text editor of your choice

Exercise Agenda

Let’s make an on-call training program!

1. Make a Curriculum
2. Create Introduction
3. Cover Failure Modes
4. List Necessary Tools

Exercise #1

Let’s make a curriculum!

Anti-example

Lesson	Topic
1	Everything you need to know about ads on-call (2 hours)

😭 😴

Tip: Avoid information overload

Lesson	Topic
1	On-call Expectation + Overview of Ad systems
2	Billing (Critical)
3	Ad Delivery (Critical)
4	Ad Internal Reports/Metrics (Less Critical)
5	Targeting (Less Critical)

Ask yourself a question:
Is there information-overload happening?

Lesson	Topic
1	On-call Expectation + Overview of Ad systems ← **Should be super high level**
2	Billing (Critical)
3	Ad Delivery (Critical)
4	Ad Internal Reports/Metrics (Less Critical)
5	Targeting (Less Critical)

Ask yourself a question:
Is there information-overload happening?

Lesson	Topic
1	On-call Expectation + Overview of Ad systems
2	Billing (Critical) ← **What if this is a complicated data pipeline with many alerts?**
3	Ad Delivery (Critical)
4	Ad Internal Reports/Metrics (Less Critical)
5	Targeting (Less Critical)

Ask yourself a question:
Is there information-overload happening?

Lesson	Topic
1	On-call Expectation + Overview of Ad systems
2	**Ad Analytics Pipeline (Critical)**
3	**Billing Pipeline(Critical)**
4	Ad Delivery (Critical)
5	Ad Internal Reports/Metrics (Less Critical)
6	Targeting (Less Critical)

Split it into a reasonable unit!

Lesson	Topic
1	On-call Expectation + Overview of Ad systems
2	Ad Analytics Pipeline (Critical)
3	Billing Pipeline(Critical)
4	Ad Delivery (Critical)
5	Ad Internal Reports/Metrics (Less Critical)
6	Targeting (Less Critical)

Ask yourself a question:
Does the order of the topics make sense?

Lesson	Topic
1	On-call Expectation + Overview of Ad systems
2	Ad Analytics Pipeline (Critical)
3	Billing Pipeline(Critical)
4	**Ad Delivery (Critical)**
5	Ad Internal Reports/Metrics (Less Critical)
6	Targeting (Less Critical)

Ask yourself a question:
Does the order of the topics make sense?

Lesson	Topic
1	On-call Expectation + Overview of Ad systems
2	Ad Analytics Pipeline
3	Billing Pipeline(Critical)
4	**Ad Delivery (Critical)** ← This is an upstream of #2 and #3
5	Ad Internal Reports/Metrics (Less Critical)
6	Targeting (Less Critical)

Ask yourself a question:
Does the order of the topics make sense?

Lesson	Topic
1	On-call Expectation + Overview of Ad systems
2	**Ad Delivery (Critical)**
3	Ad Analytics Pipeline (Critical)
4	Billing Pipeline (Critical)
5	Ad Internal Reports/Metrics (Less Critical)
6	Targeting (Less Critical)

Ask yourself a question:
Does the order of the topics make sense?

Exercise #1

Let’s make an on-call training curriculum!

- Come up with a list of topics
- Chunk it into a “reasonable” size
- Sort them

3 mins

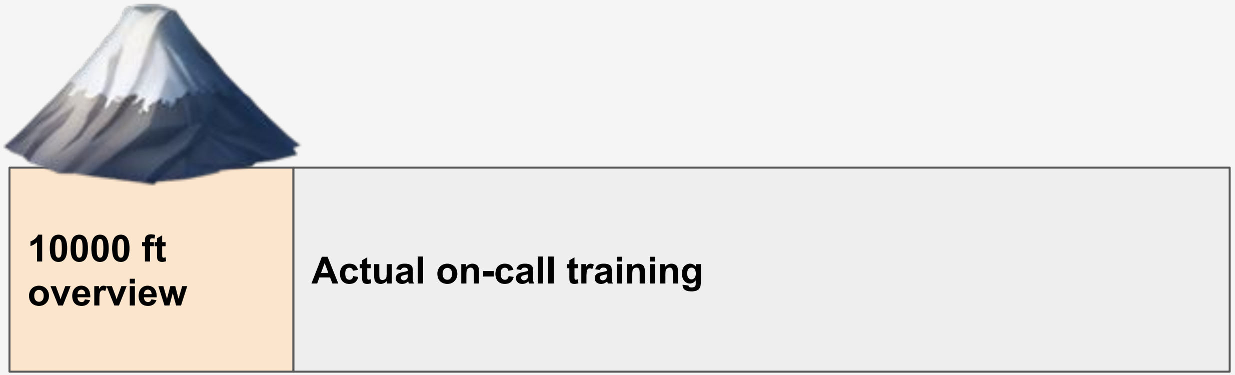
Exercise Agenda

Let’s make an on-call training program!

1. Make a Curriculum
2. Create Introduction
3. Cover Failure Modes
4. List Necessary Tools

Exercise #2

Let’s write a 10000 ft overview of the system!



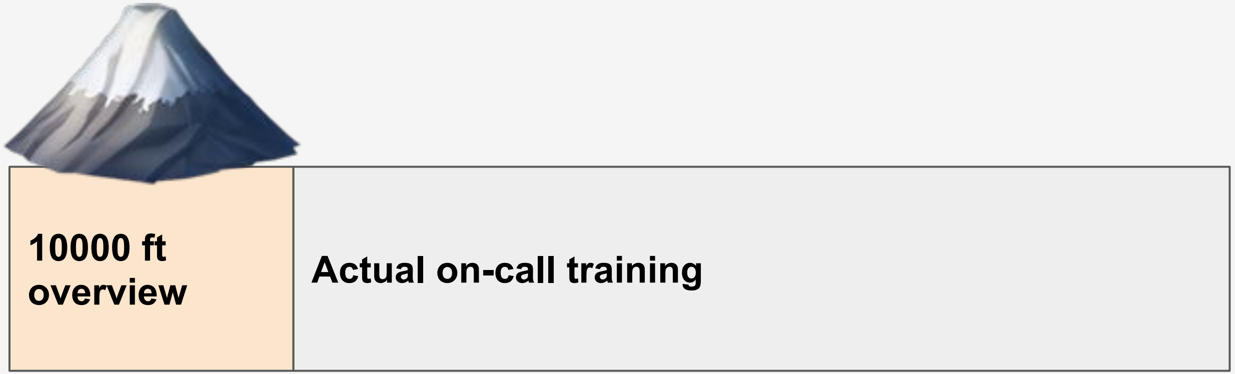
10000 ft overview

Actual on-call training

Exercise #2

Why give an overview in on-call training?

- Make sure students are on the same page
- Make failure points clearer



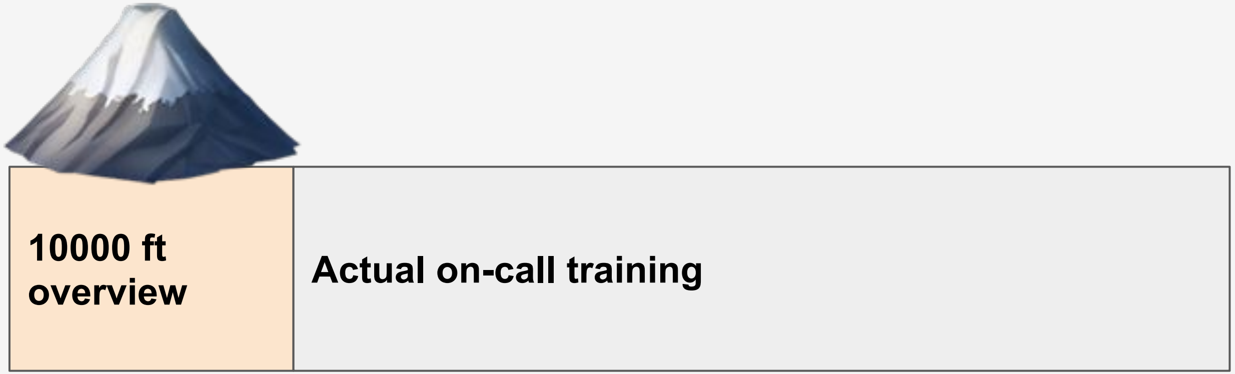
10000 ft overview

Actual on-call training

Exercise #2

What should a 10000 ft overview include?

- Simple Diagram
- Summary of the system (What it does, what depends on it etc)

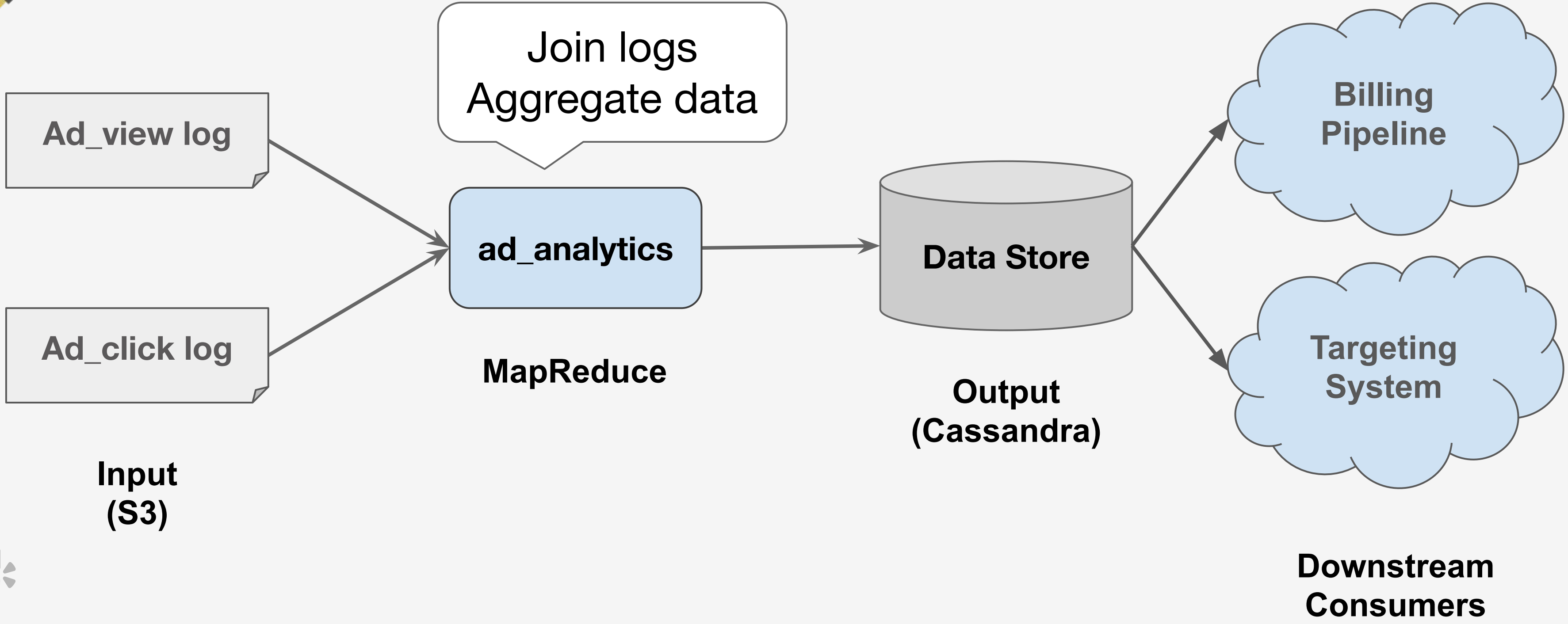


10000 ft overview

Actual on-call training

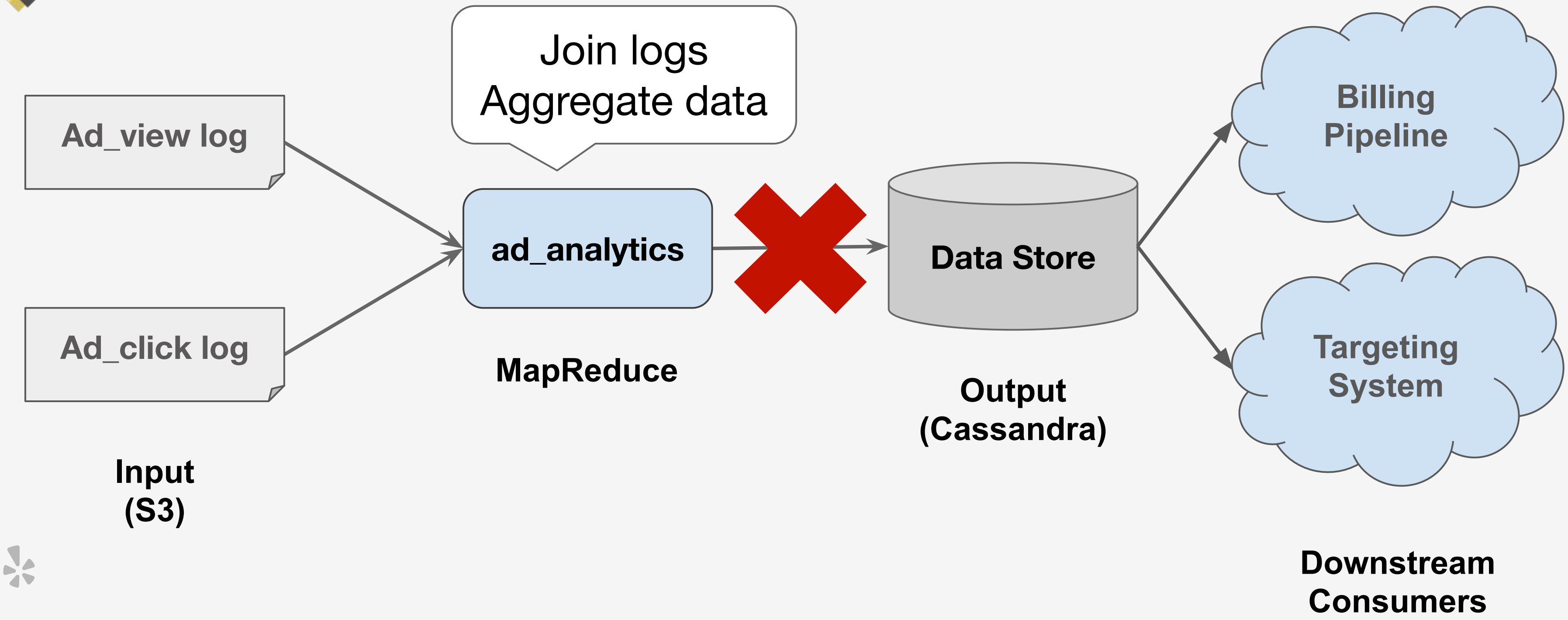
Lesson	Topic
1	What is on-call? + Overview of Ad systems
2	Ad Delivery (Critical)
3	**Ad Analytics Pipeline (Critical)**
4	Billing Pipeline (Critical)
5	Ad Internal Reports/Metrics (Less Critical)
6	Targeting (Less Critical)

Example: Ad Analytics Pipeline

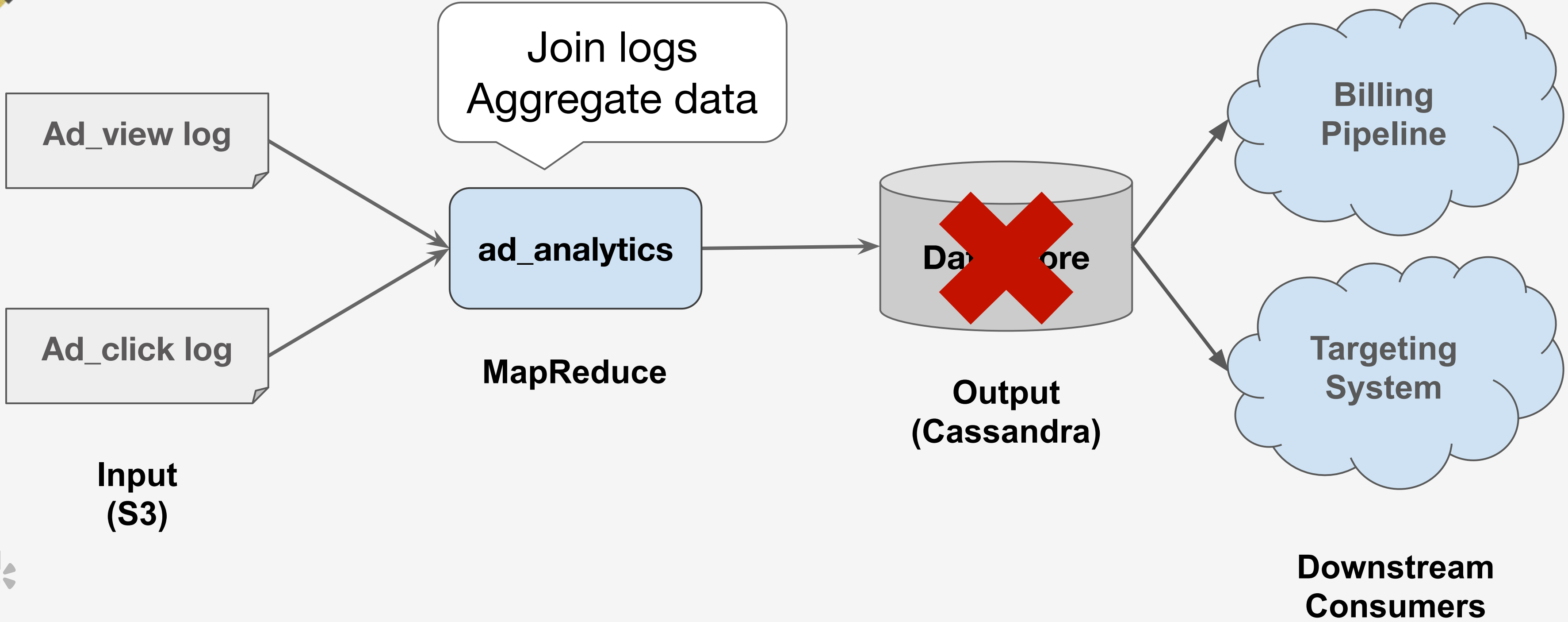


Tip: Use visual aid you can reuse

Example: Ad Analytics Pipeline



Example: Ad Analytics Pipeline



Exercise #2

Let’s write a 10000 ft overview of the system!

1. Pick one topic from the curriculum
2. Summarize the system (functionality, techstack)
3. Add a diagram

3 mins

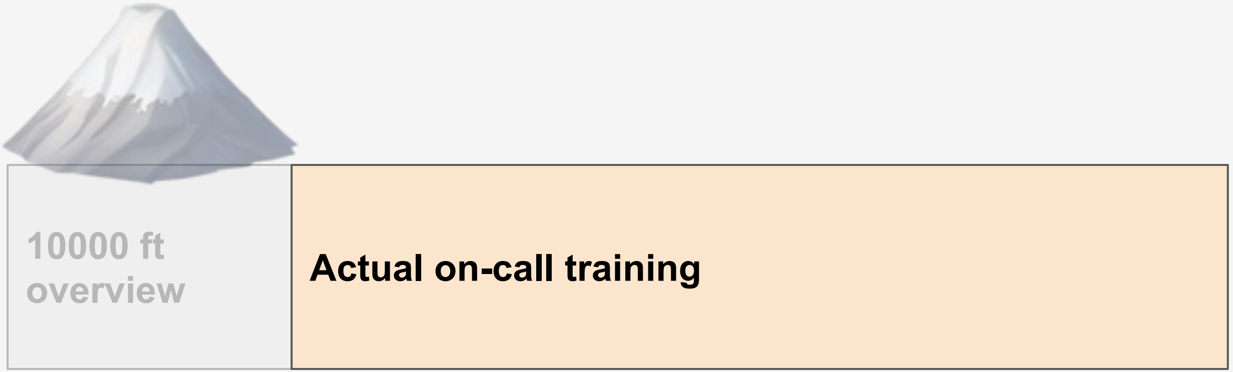
Exercise Agenda

Let’s make an on-call training program!

1. Make a Curriculum
2. Create Introduction
3. Cover Failure Modes
4. List Necessary Tools

Exercise #3

Let’s write the “actual on-call training”

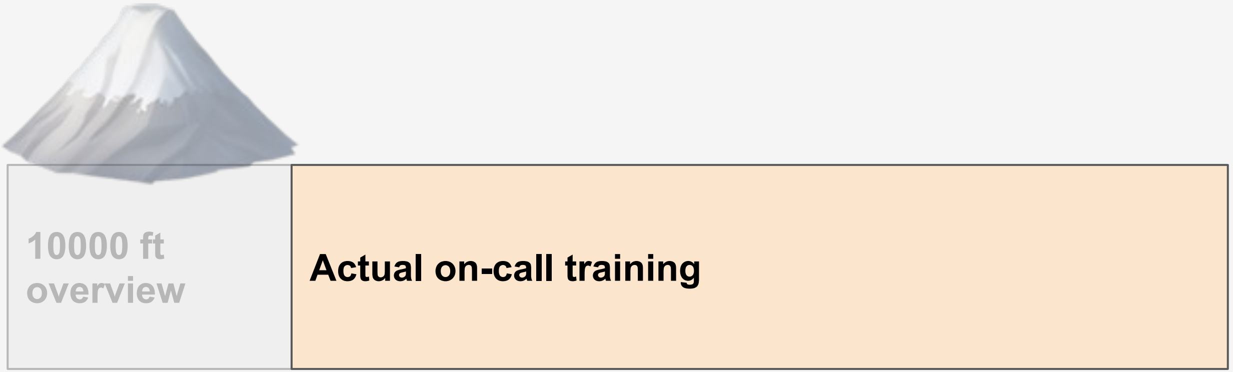


10000 ft overview

Actual on-call training

Exercise #3

Let’s write the “actual on-call training”



10000 ft overview

Actual on-call training

- Failure modes/alerts
- How to respond to them

Tip

Use Real Past Incidents

Exercise #3
Why use past incidents?

- Real examples are the best teachers!
- Opportunity to make it interactive

Example: Ad Analytics Pipeline

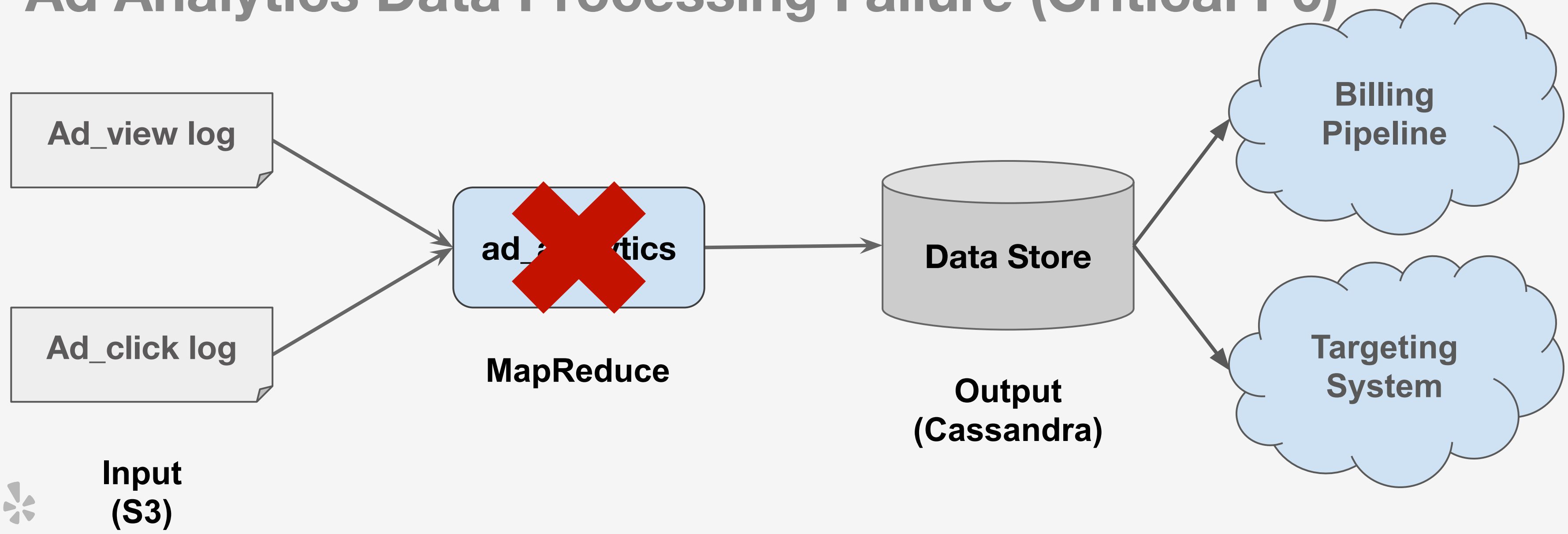
Alert:

Ad Analytics Data Processing Failure (Critical P0)

Example: Ad Analytics Pipeline

Alert:

Ad Analytics Data Processing Failure (Critical P0)



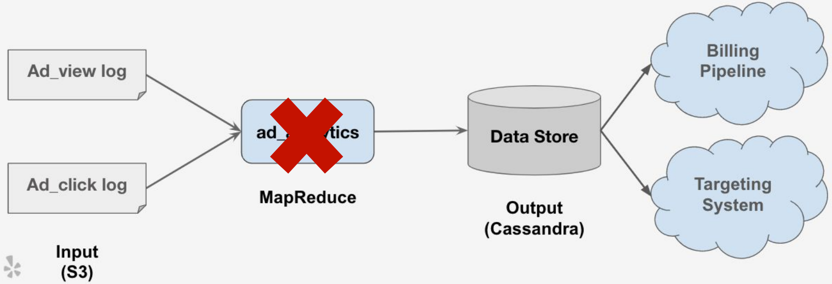
Example: Ad Analytics Pipeline

Alert:

Ad Analytics Data Processing Failure (Critical P0)

Past Incidents:

- Backward-incompatible input schema change
- MapReduce task timeouts due to bot traffic



Exercise #3

Let’s write the “actual on-call training”

- List alerts/failure modes
- Label them with priority (e.g. P0, P1)
- Map them in your 10000 ft diagram
- Find at least one past incident for each alert

3 mins

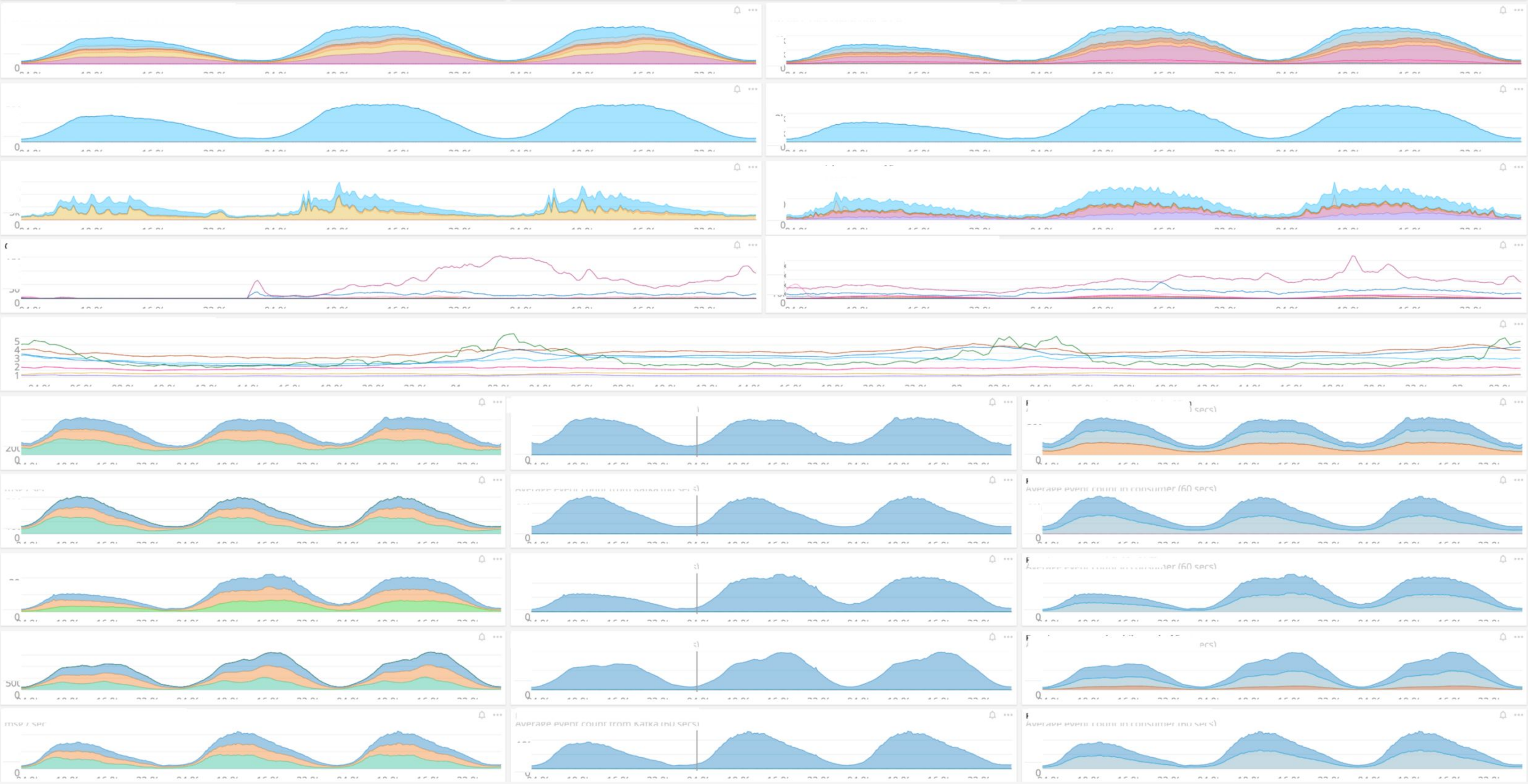
Exercise Agenda

Let’s make an on-call training program!

1. Make a Curriculum
2. Create Introduction
3. Cover Failure Modes
4. List Necessary Tools

Exercise #4

Let’s teach necessary tools and know-hows


0
50
5
4
3
2
1
200
) secs)
Average event count from Kafka (60 secs)
Average event count in consumer (60 secs)
500
mse / sec

Example

How to read a service SignalFx dashboard

Example

How to read a service SignalFx dashboard

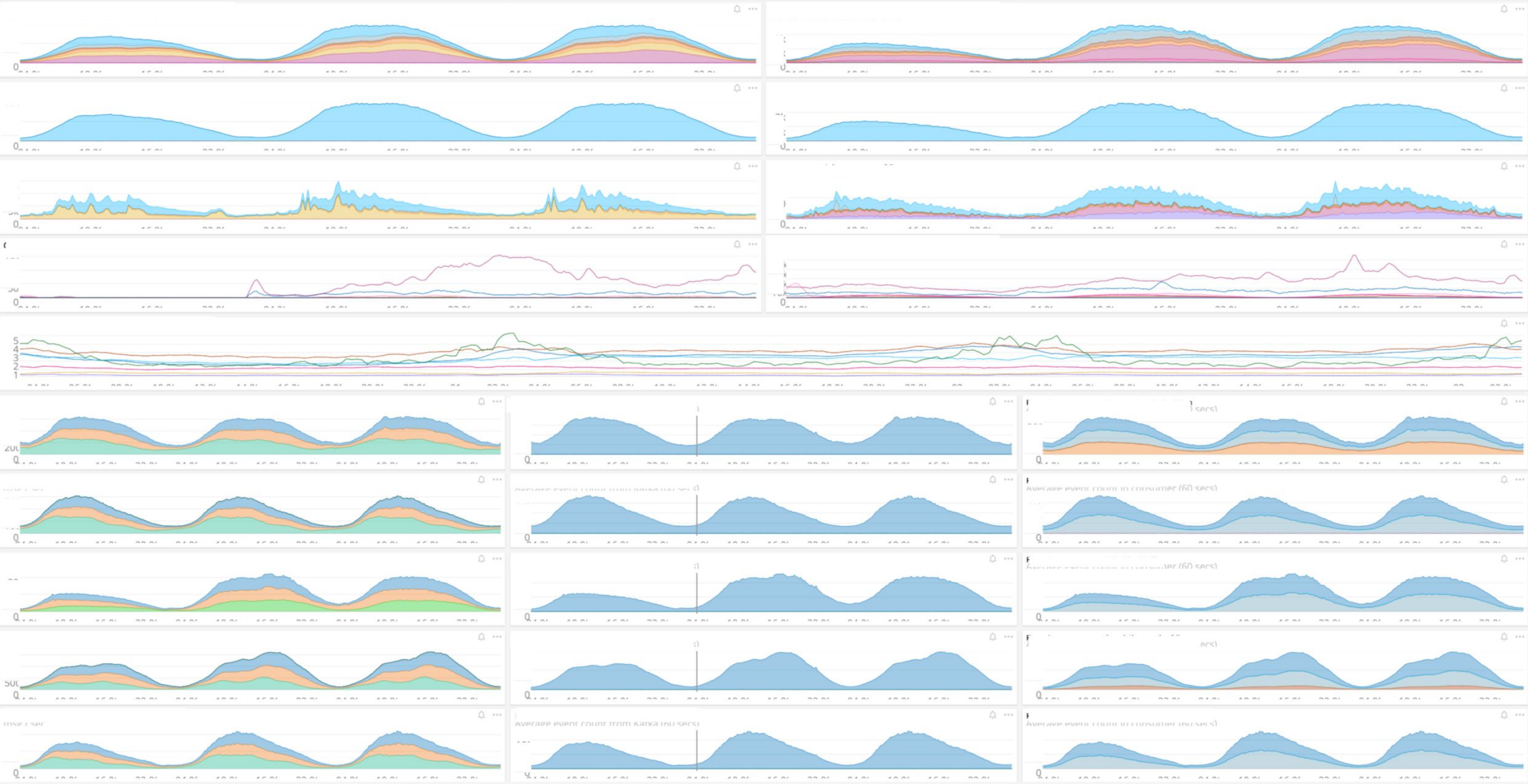
(This should ideally be in runbook)

Tip: Let students apply knowledge ASAP

Example

How to read a service SignalFx dashboard

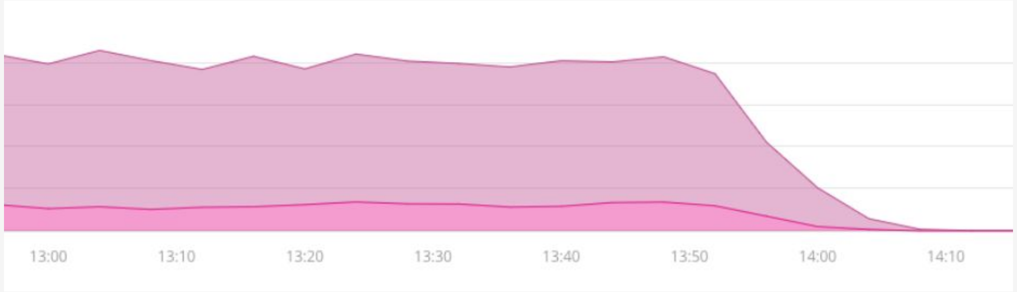
1. Explain
2. Show a dashboard screenshot from a past incident
3. Let students debug + ask questions


0
5
4
3
2
1
200
500
50
Average event count from Kafka (60 secs)
Average event count in consumer (60 secs)
Average event count in consumer (60 secs)
Average event count from Kafka (60 secs)
Average event count in consumer (60 secs)

Example

How to read a service SignalFx dashboard

1. Explain what it is
2. Show dashboard screenshots from a past incident
3. Let students debug and ask questions



Exercise #4

Let’s teach necessary tools and know-hows

1. List tools and know-hows
   (Based on your answers from Exercise #3)
2. Make it interactive

3 mins

Congratulations!

You have a (partially complete) on-call training program!

Tips (Recap)

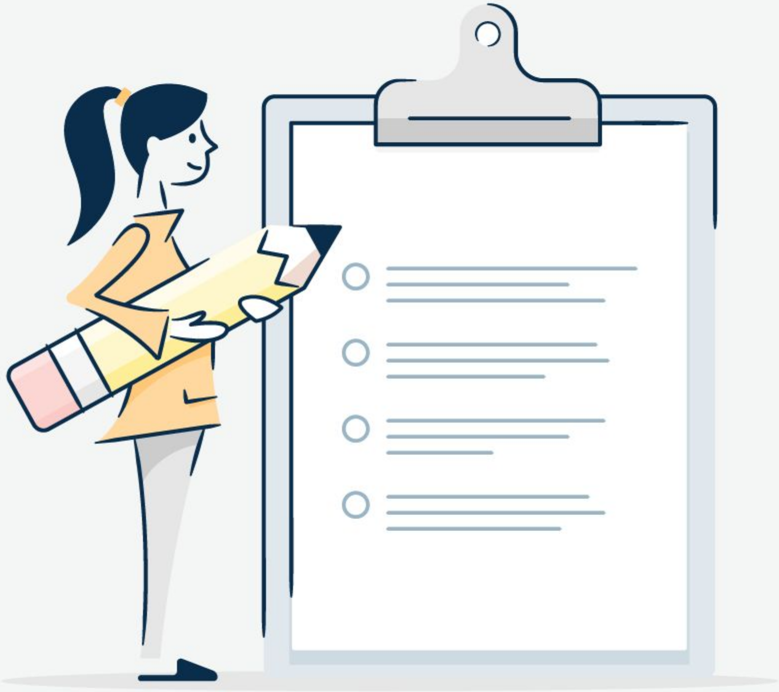
Avoid information overload

Use visual aid you can reuse

Use real past incidents

Let students apply knowledge ASAP

Beyond Training



Knowledge sharing

On-call handoff meeting

Show and tell how recent incidents were resolved

Postmortem

Learning from the past incidents

Wargame

Practice incident response in a safe environment

Wargames

- Incident simulation game
- Game master
  - Reproduce/narrate the incident
  - Ask questions and give hints
- On-call Player(s)
  - Investigate and mitigate the incident

3 steps to start a wargame

Step 1:
Pick a scenario

Pick a scenario

Real past incidents

- Low cost to prepare with

Imaginary incidents

- Brainstorm what could happen and how to handle

Pick a scenario

- **Interactive**
  - Actually break things (in a safe environment)

- **Static**
  - Use dashboard screenshots/logs/code snippet

Step 2:

Make a wargame template

Example

Wargame template

Incident Setup

Instruction on how to trigger a batch failure incident

- ❏ Reserve stage env <runbook link>
- ❏ Prepare bad source code
- ❏ Prepare dashboard link <link>
- ❏ Cmd to run batch in the env
  - ❏ `python ./mybatch.py --config config.yaml`
- ❏ Wait for the batch to crash

Example

Wargame template

Player roles

- ❏ Investigators --- <names>
- ❏ Communicator --<name>
- ❏ Commander -- <name>

Player checklist

- ❏ Get relevant permissions
- ❏ Join external wifi/set up VPN
- ❏ Use wargame-only communication tools
  - ❏ channel #wargame
  - ❏ email alias wargame@
  - ❏ JIRA project WARGAME

Example

Wargame template

Hints

- ❏ What does runbook say?
- ❏ Was there any exceptions in batch log?
- ❏ Were there any recent code changes?
- ❏ Does dashboard show any abnormality?

Step 3:
Run the game

Tips for running the game

Invite Audience

- More people can benefit from one session

Ask questions

- Have players explain why they took certain actions
- Give hints by asking questions

Take notes

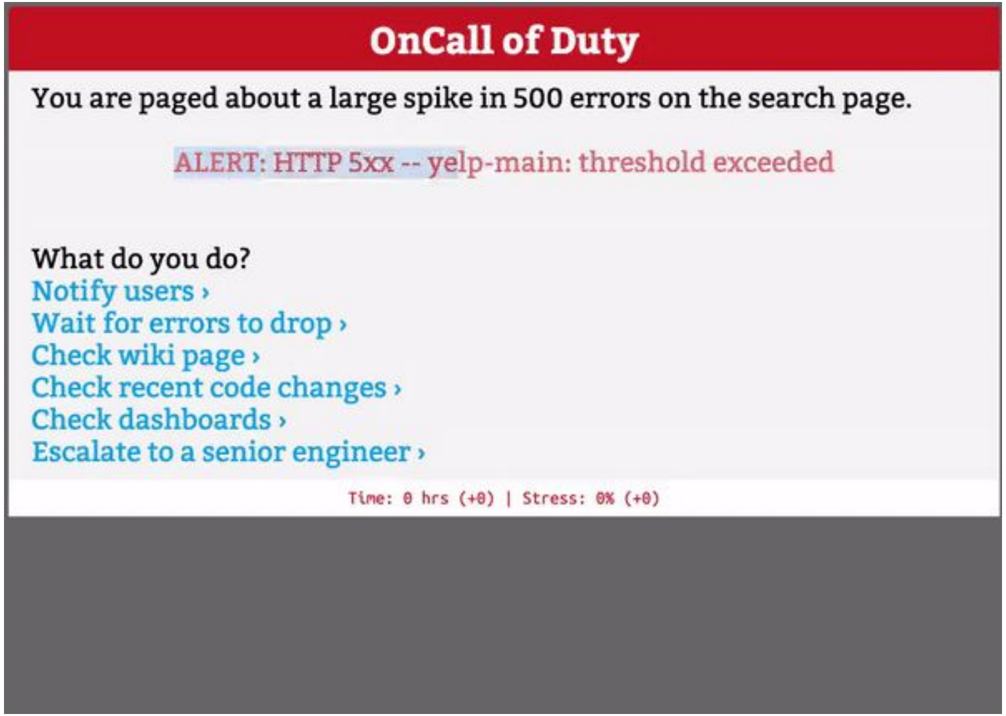
- Unclear or outdated runbook/alerts to fix
- Improvement future wargames

Wargames

http://bit.ly/on-call-game

Use tools to build your game

on-call simulation text adventure game using **Twine**



OnCall of Duty

You are paged about a large spike in 500 errors on the search page.

ALERT: HTTP 5xx -- yelp-main: threshold exceeded

What do you do?

Notify users ›
Wait for errors to drop ›
Check wiki page ›
Check recent code changes ›
Check dashboards ›
Escalate to a senior engineer ›

Time: 0 hrs (+0) | Stress: 0% (+0)

Break (5 mins)

Check out our Twine game:

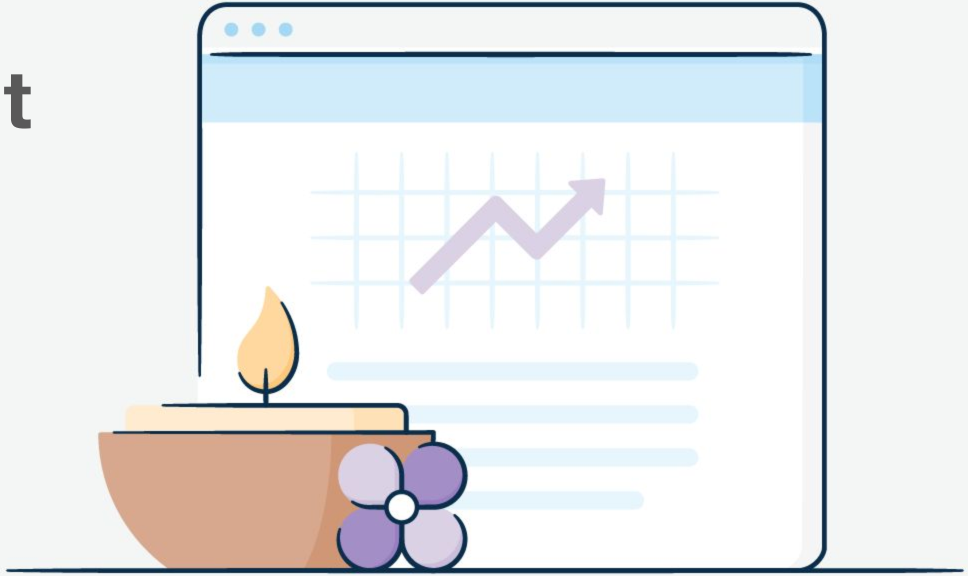
http://bit.ly/oncall-game or

SCAN ME

Optional Materials:

http://bit.ly/srecon19-oncall

Runbooks for Effective Incident Response



SURVEY RESULTS

Why didn't you feel ready?

40% Didn’t feel ready due to lack of protocol

70% Reviewed the team’s runbooks before going on-call

“Update and improve documentation and runbooks”

“More documentation”

“Better documentation”

“Clear protocol of pages we can get and how to handle them”

“Runbooks should be obvious to find and execute. At 3 AM you need dummy-proof instructions.”

Survey within Yelp Engineering (2018)

Why care about good runbooks?

Win 1: Make incident response efficient

Win 2: Require less in-depth knowledge

Win 3: Reduce nervousness

What is a runbook?

- Step-by-step instructions on incident response

What is a runbook? (Common mistake)

- **Deep dive on how the system works**
    - “Everything you can ever know about the service!”

What is a runbook? (Common mistake)

Deep dive on how the system works

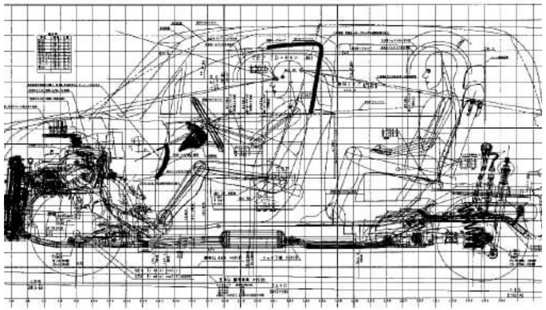
- “Everything you can ever know about the service!”

Incident: Dead car battery

How to jump start a car

≠

How a car works



What is a runbook? (Common mistake)

- **Deep dive on how the system works**
  - “Everything you can ever know about the service!”

** Still an important documentation to have.
But it should NOT live with runbooks

Two types of runbook

Technical runbook

Step-by-step instruction on investigation and mitigation

- Impact assessment
- Mitigation
- Disaster recovery

Non-technical Runbook

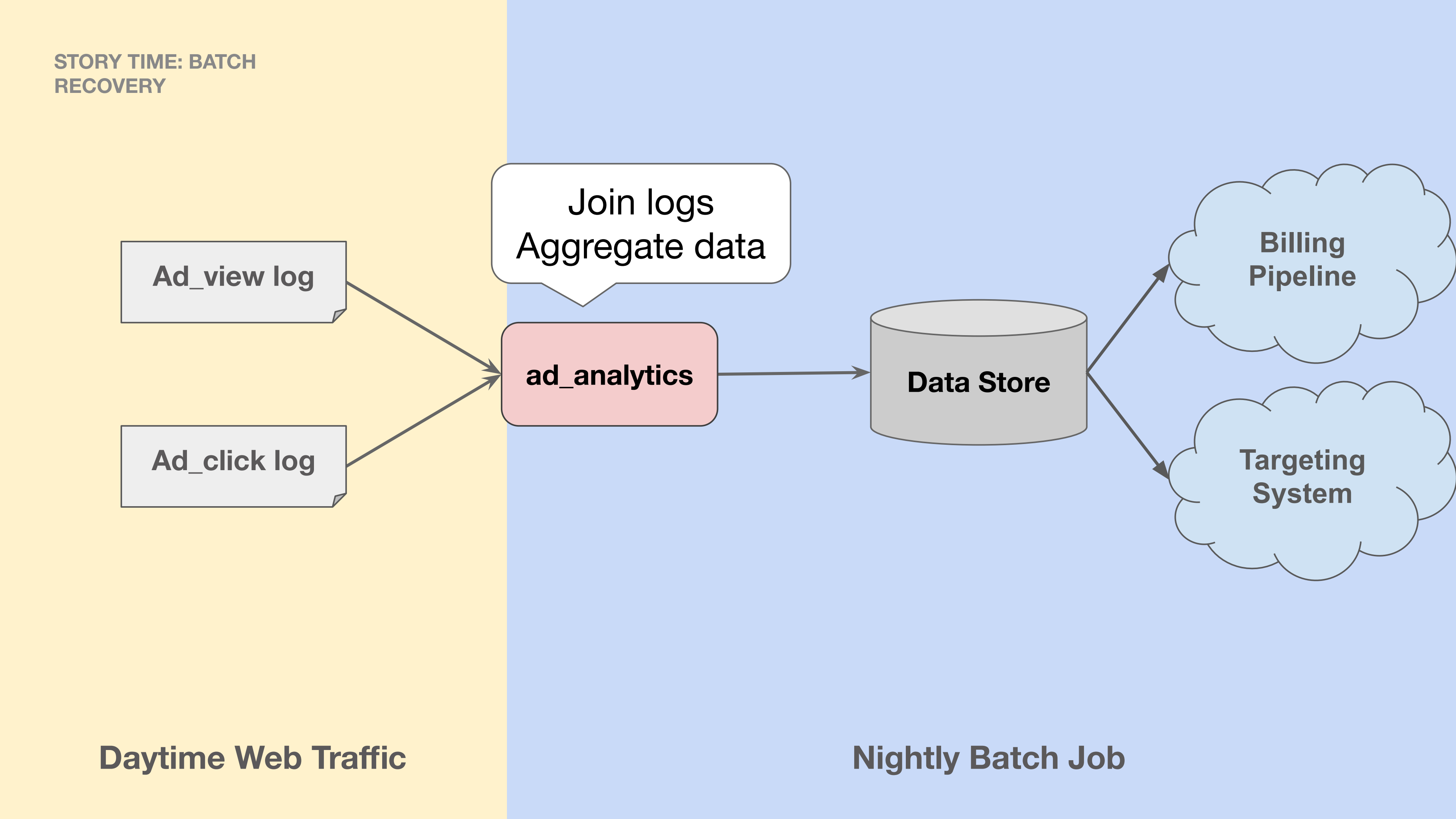
Step-by-step instruction for human process

- Roles (e.g. Investigator, communicator)
- Communication process
- Escalation policy

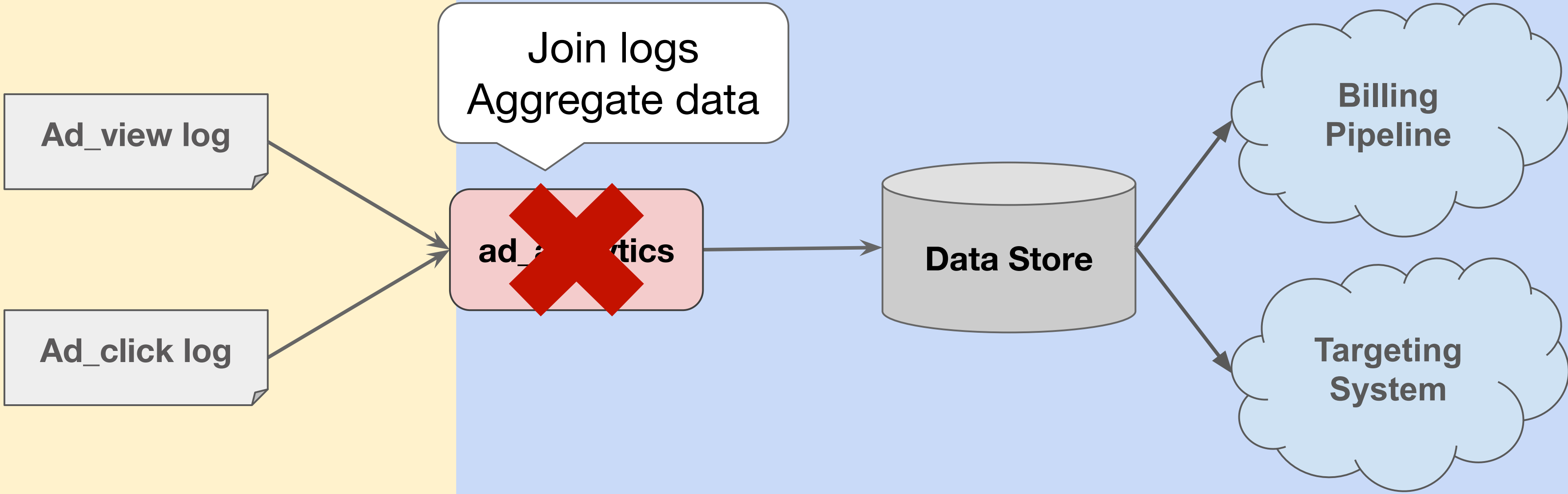
Example
Symptoms of a bad runbook

http://bit.ly/srecon19-oncall

STORY TIME: BATCH RECOVERY



STORY TIME: BATCH RECOVERY



Join logs
Aggregate data

Ad_view log

Ad_click log

ad_a ytics

Data Store

Billing Pipeline

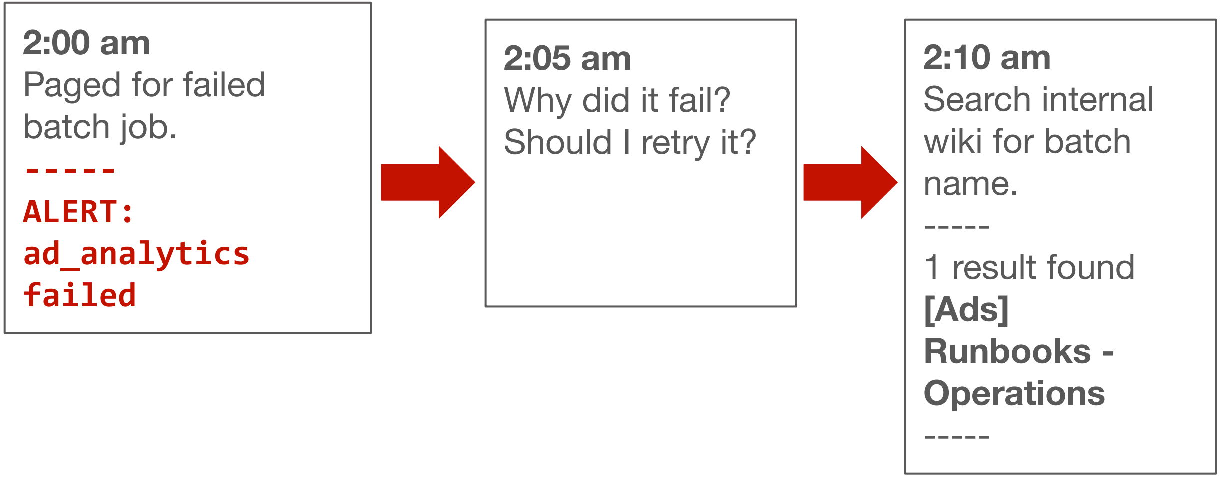
Targeting System

Daytime Web Traffic

Nightly Batch Job

STORY TIME: BATCH RECOVERY

What made this runbook **difficult** to use?



2:00 am
Paged for failed batch job.

```
-----
ALERT:
ad_analytics
failed
```

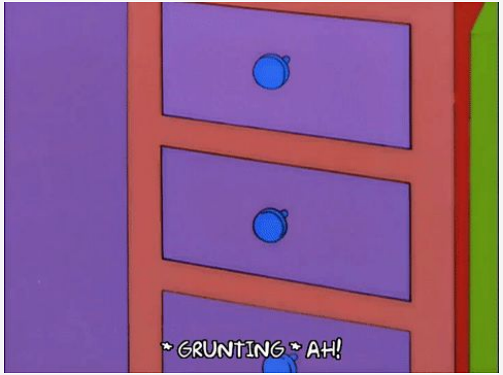
2:05 am
Why did it fail?
Should I retry it?

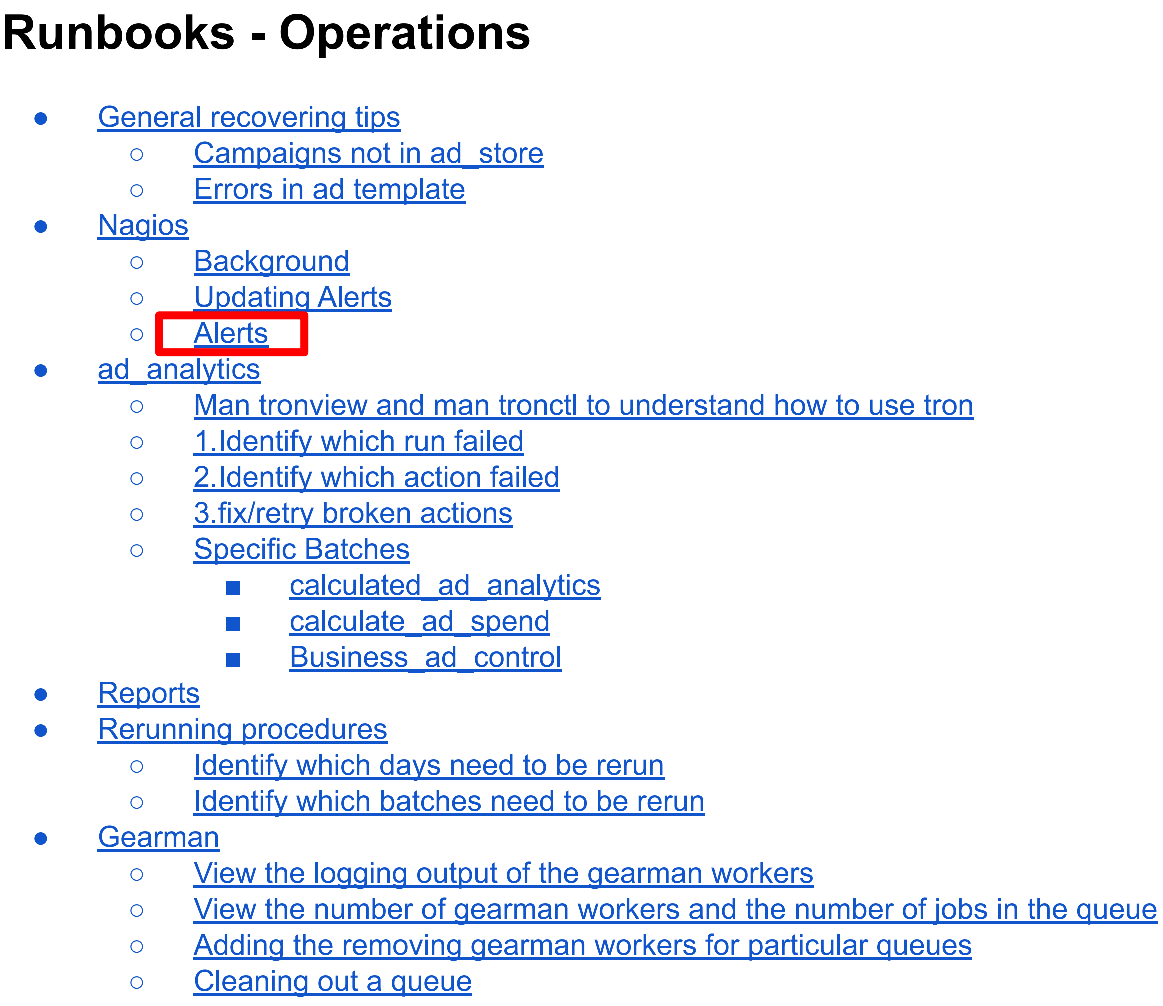
2:10 am
Search internal wiki for batch name.

-----

1 result found
[Ads] Runbooks - Operations

What made this runbook **difficult** to use?

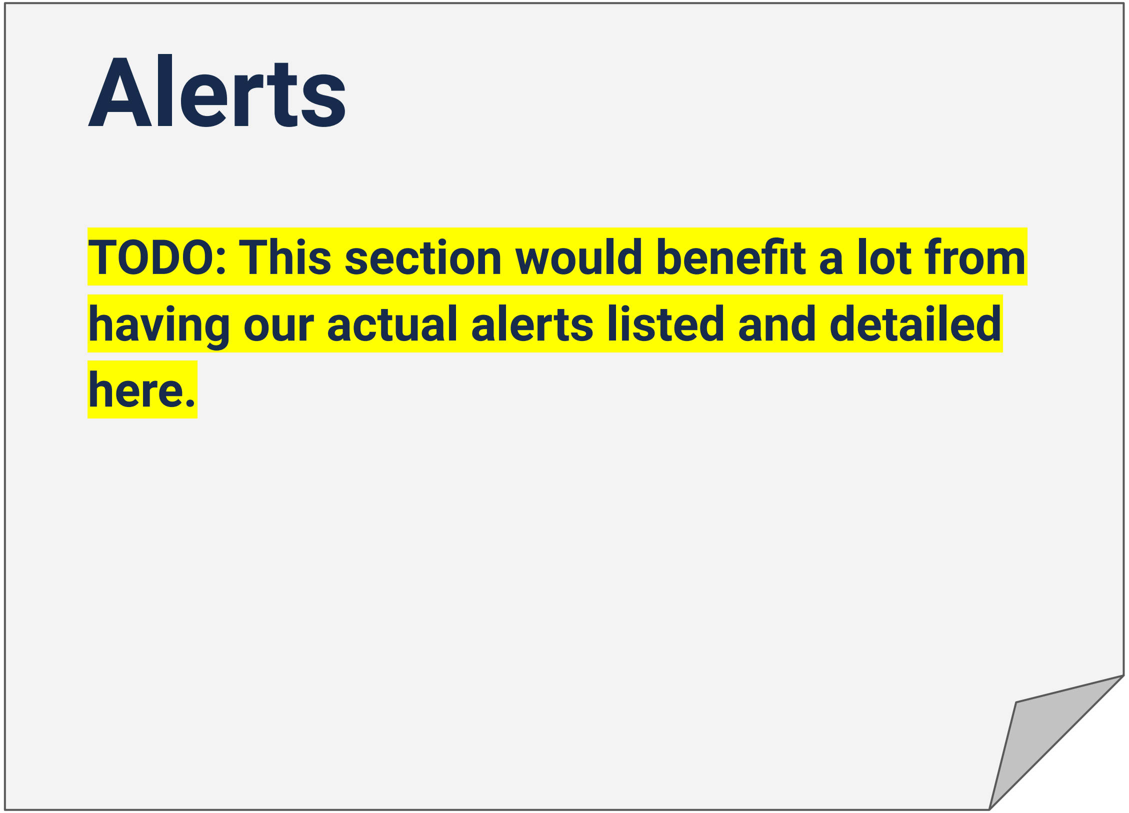




Runbooks - Operations

- General recovering tips
  - Campaigns not in ad_store
  - Errors in ad template
- Nagios
  - Background
  - Updating Alerts
  - Alerts
- ad_analytics
  - Man tronview and man tronctl to understand how to use tron
  - 1.Identify which run failed
  - 2.Identify which action failed
  - 3.fix/retry broken actions
  - Specific Batches
    - calculated_ad_analytics
    - calculate_ad_spend
    - Business_ad_control
- Reports
- Rerunning procedures
  - Identify which days need to be rerun
  - Identify which batches need to be rerun
- Gearman
  - View the logging output of the gearman workers
  - View the number of gearman workers and the number of jobs in the queue
  - Adding the removing gearman workers for particular queues
  - Cleaning out a queue

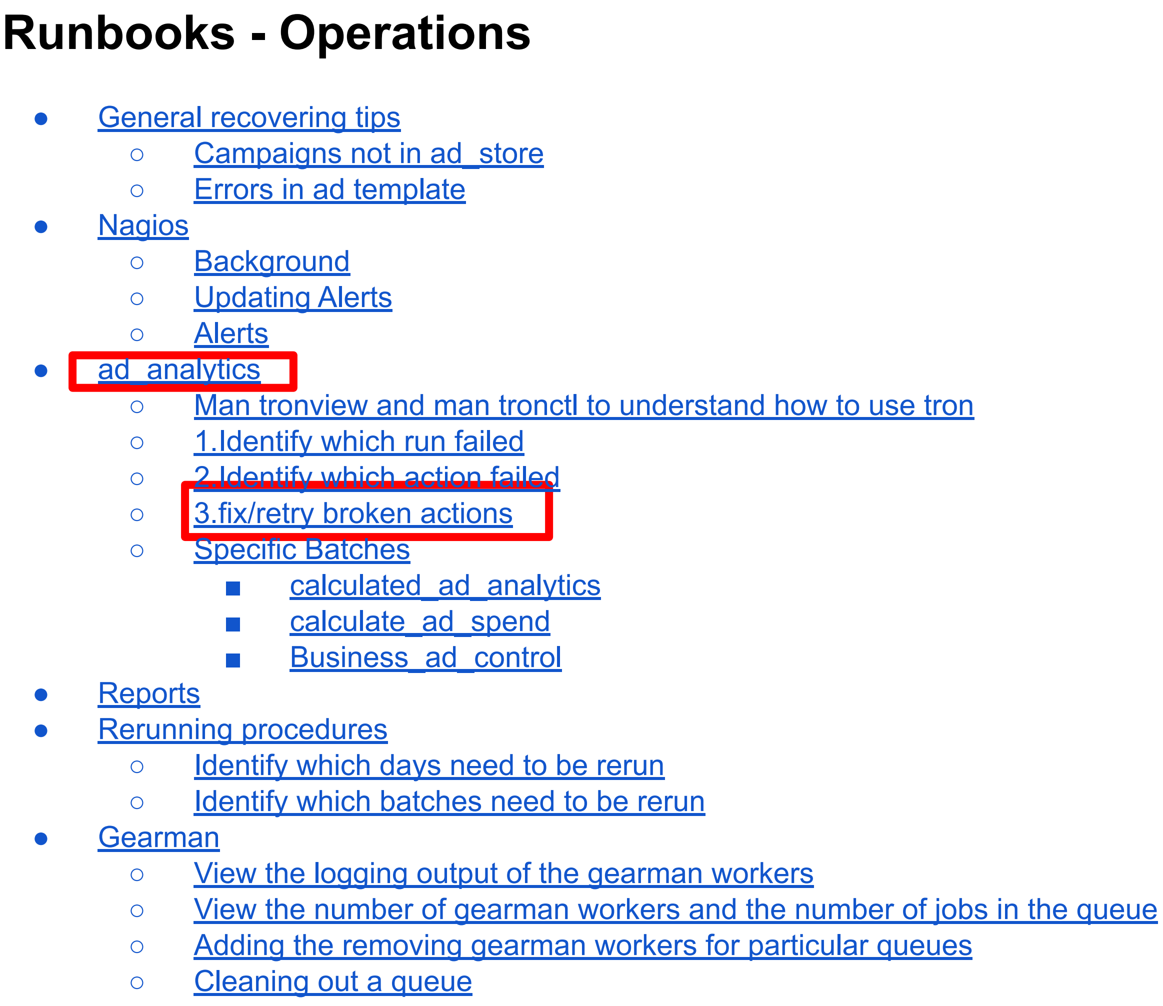
What made this runbook **difficult** to use?



Alerts

TODO: This section would benefit a lot from having our actual alerts listed and detailed here.

What made this runbook **difficult** to use?

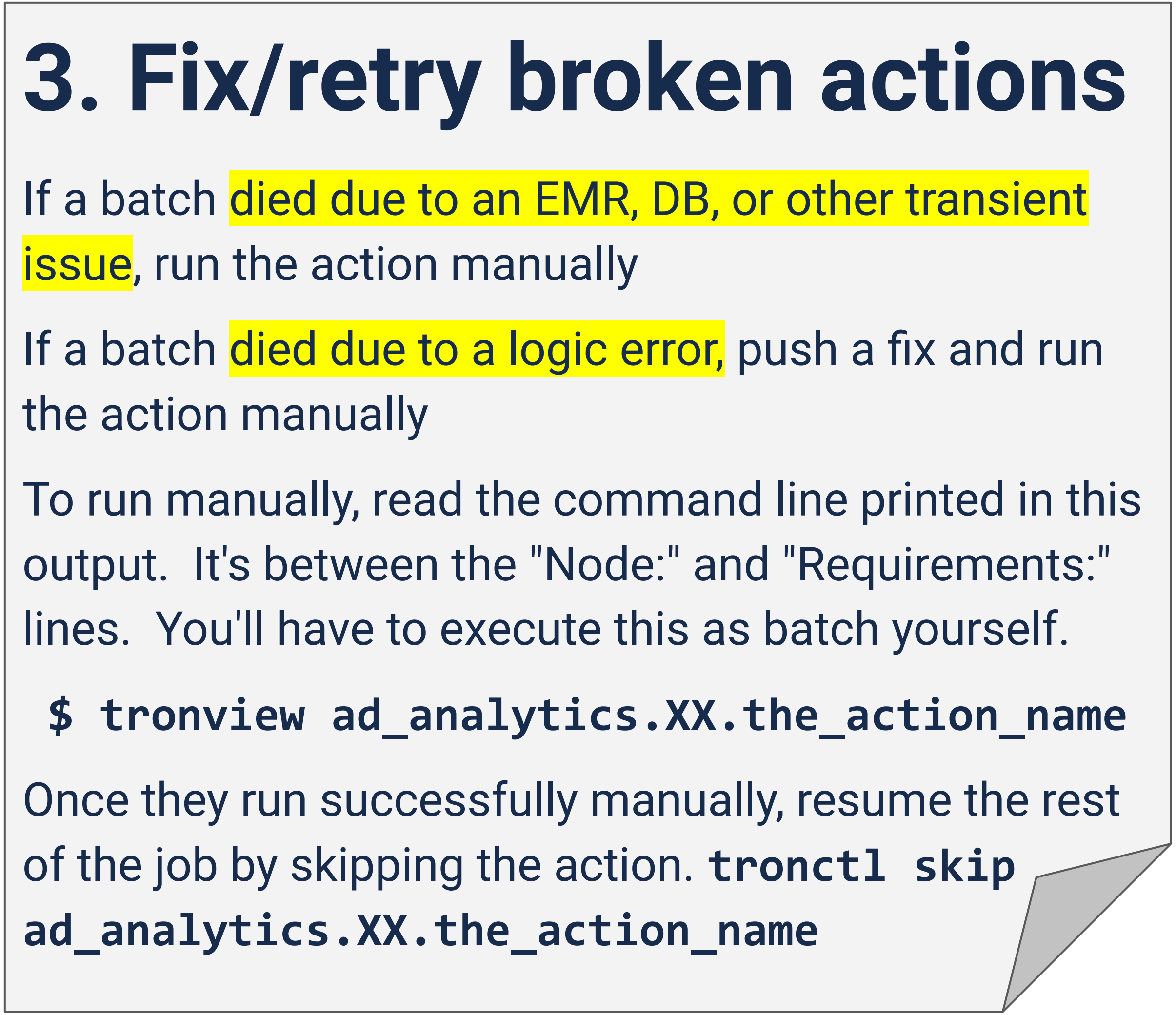


Runbooks - Operations

- General recovering tips
  - Campaigns not in ad_store
  - Errors in ad template
- Nagios
  - Background
  - Updating Alerts
  - Alerts
- ad_analytics
  - Man tronview and man tronctl to understand how to use tron
  - 1.Identify which run failed
  - 2.Identify which action failed
  - 3.fix/retry broken actions
  - Specific Batches
    - calculated_ad_analytics
    - calculate_ad_spend
    - Business_ad_control
- Reports
- Rerunning procedures
  - Identify which days need to be rerun
  - Identify which batches need to be rerun
- Gearman
  - View the logging output of the gearman workers
  - View the number of gearman workers and the number of jobs in the queue
  - Adding the removing gearman workers for particular queues
  - Cleaning out a queue

RUNBOOK EXAMPLES

What made this runbook **difficult** to use?



3. Fix/retry broken actions

If a batch died due to an EMR, DB, or other transient issue, run the action manually

If a batch died due to a logic error, push a fix and run the action manually

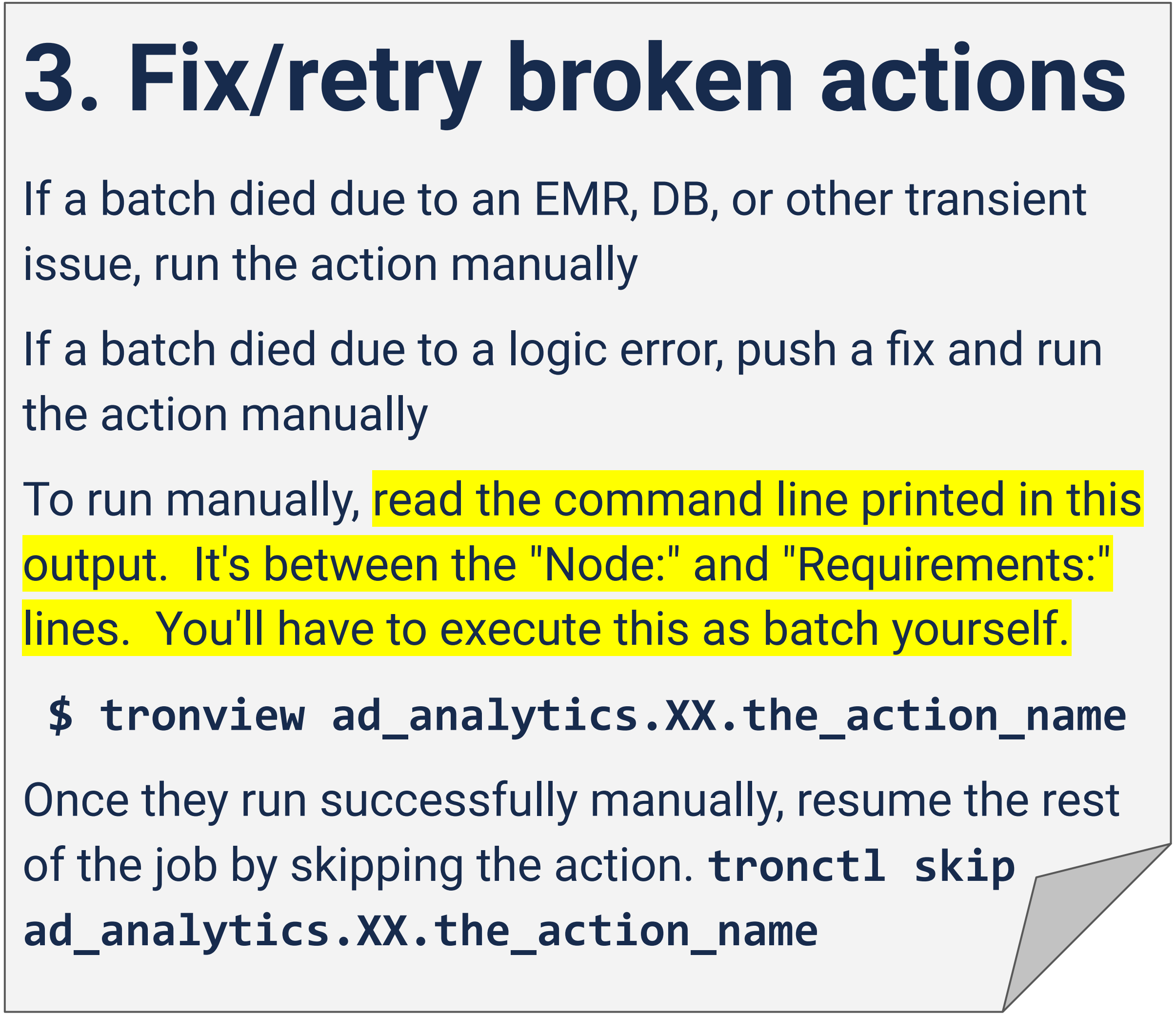
To run manually, read the command line printed in this output. It's between the "Node:" and "Requirements:" lines. You'll have to execute this as batch yourself.

```
 $ tronview ad_analytics.XX.the_action_name
```

Once they run successfully manually, resume the rest of the job by skipping the action. `tronctl skip ad_analytics.XX.the_action_name`

RUNBOOK EXAMPLES

What made this runbook **difficult** to use?



3. Fix/retry broken actions

If a batch died due to an EMR, DB, or other transient issue, run the action manually

If a batch died due to a logic error, push a fix and run the action manually

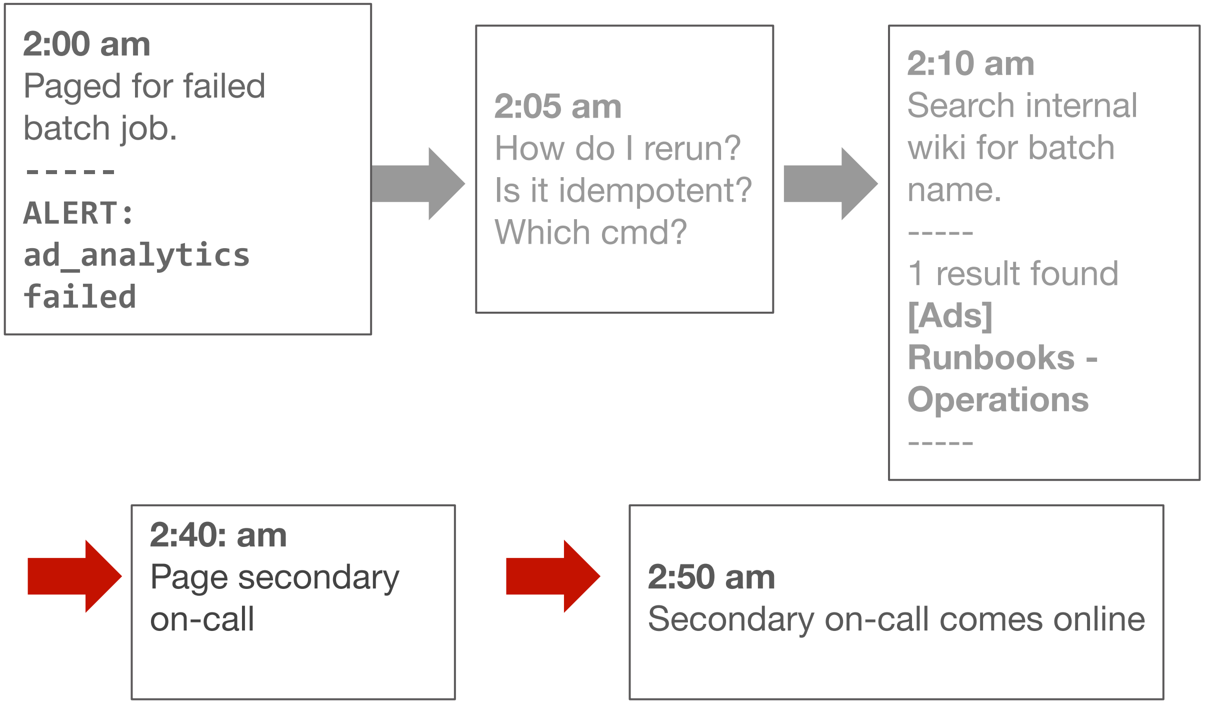
To run manually, read the command line printed in this output. It's between the "Node:" and "Requirements:" lines. You'll have to execute this as batch yourself.

```
 $ tronview ad_analytics.XX.the_action_name
```

Once they run successfully manually, resume the rest of the job by skipping the action. `tronctl skip ad_analytics.XX.the_action_name`

STORY TIME: BATCH RECOVERY

What made this runbook difficult to use?



2:00 am
Paged for failed batch job.

```
-----
ALERT:
ad_analytics
failed
```

2:05 am
How do I rerun?
Is it idempotent?
Which cmd?

2:10 am
Search internal wiki for batch name.
-----
1 result found
[Ads] Runbooks - Operations

2:40: am
Page secondary on-call

2:50 am
Secondary on-call comes online

Me: “Where can I find the rerun command?”

Secondary: “You can try looking for that in the wiki”

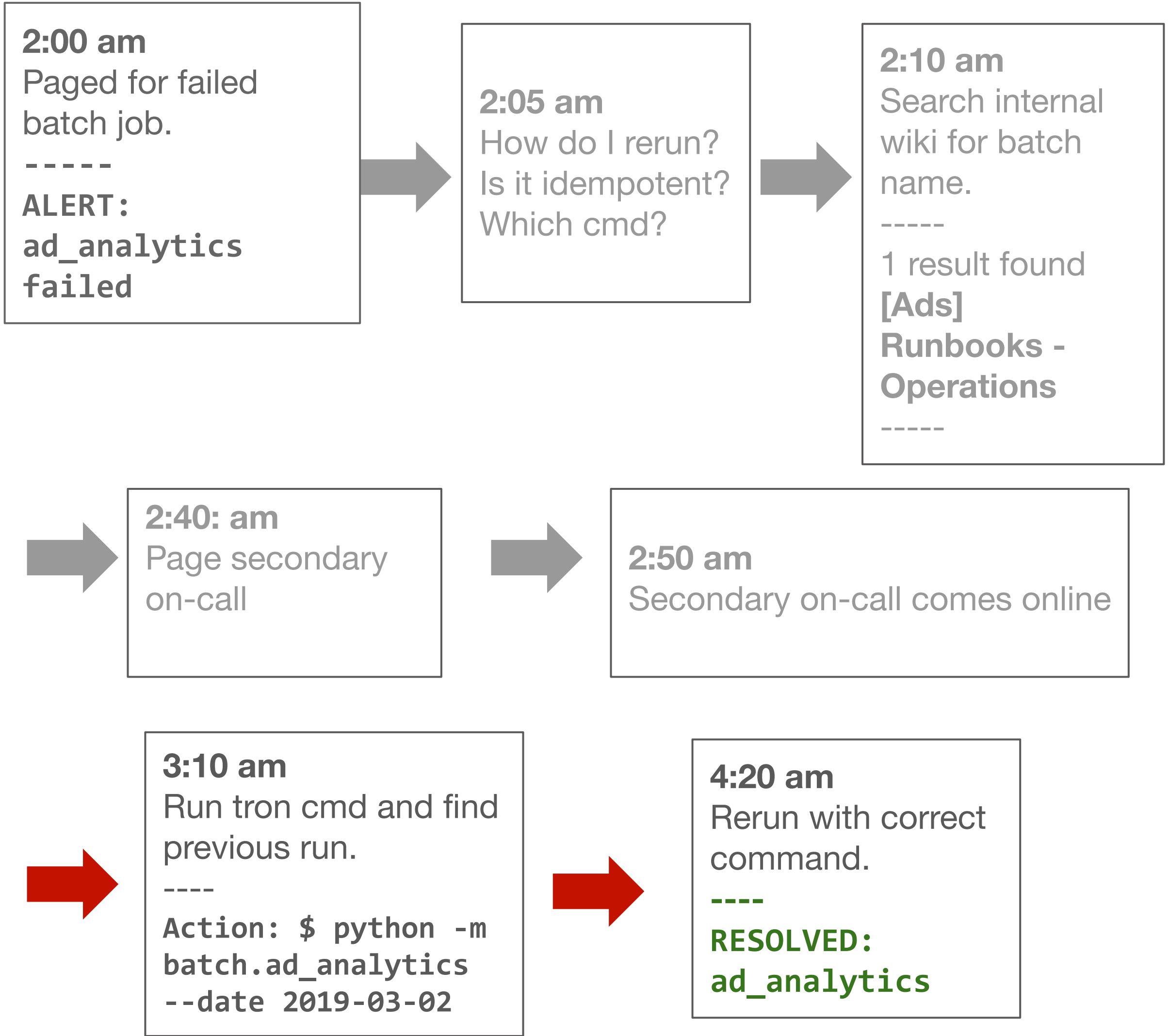
Me: “I just checked, but it’s not very clear.”

Secondary: “Or maybe it’s in the Google Docs repo. Oh, and I’ve got some notes in my home directory, and I think I saw some emails about that a while ago.”

Me: 😭

STORY TIME: BATCH RECOVERY

What made this runbook difficult to use?



2:00 am
Paged for failed batch job.

```
-----
ALERT:
ad_analytics
failed
```

2:05 am
How do I rerun?
Is it idempotent?
Which cmd?

2:10 am
Search internal wiki for batch name.
-----
1 result found
[Ads] Runbooks - Operations

2:40: am
Page secondary on-call

2:50 am
Secondary on-call comes online

3:10 am
Run tron cmd and find previous run.

```
----
Action: $ python -m
batch.ad_analytics
--date 2019-03-02
```

4:20 am
Rerun with correct command.

```
----
RESOLVED:
ad_analytics
```

STORY TIME: BATCH RECOVERY

What made this runbook difficult to use?

- Hard to find
- No organization/scattered information
- Incomplete and outdated
- Unclear instructions

What makes
a **good** technical runbook?

Tips for writing good technical runbooks

- Directly link alert to runbook
- Single source of truth
- Minimize incomplete or outdated sections
- Include commands and screenshots

Alert Name	<exact alert name>
Description	<1 sentence description>
Stakeholder impact	<1 sentence impact>
Mitigation steps	1. Try restarting: <command> 2. Monitor dashboards. 3. Inspect logs to diagnose issue: <link or See steps below> If things do not recover, follow Escalation steps.
Escalation steps	Contact <team>. Massive ingestion delays should be communicated to <upstream and downstream teams>.
Related services	<upstream and downstream dependencies>
Dashboards	<links>
Related links	<other docs or related runbooks>

http://bit.ly/srecon19-oncall

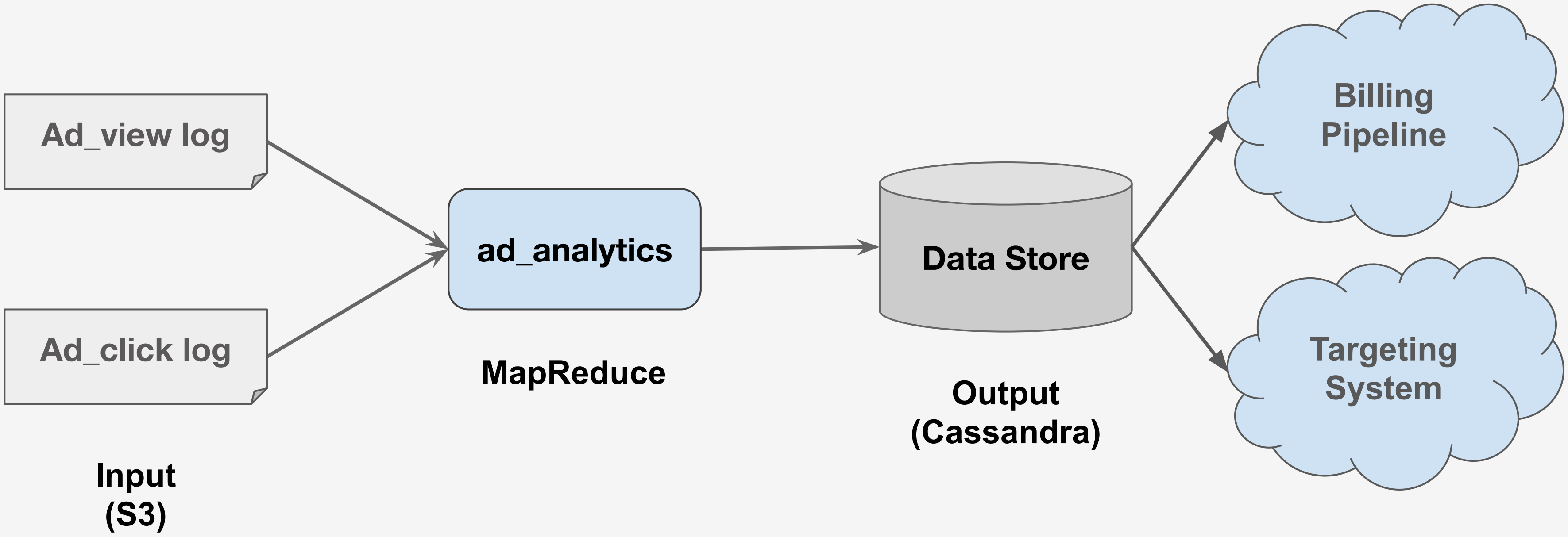
Exercise

Let’s make your own runbook!

1. List all alerts
2. Fill out details
3. Make it easy to find

Step 1:
List all alerts

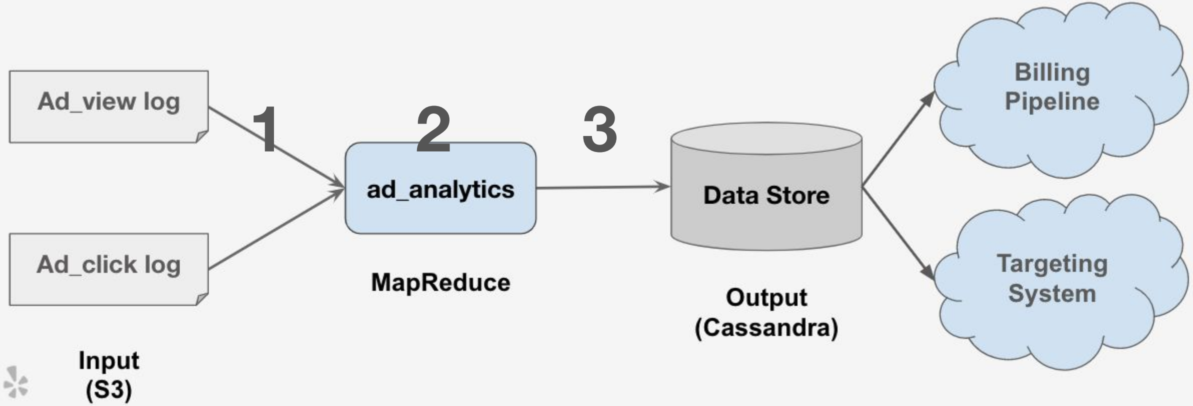
Example: Ad Analytics Pipeline



Example: Ad Analytics Pipeline

Alerts:

1. Ad Analytics Upstream Data Delay
2. Ad Analytics Data Processing Batch Failure
3. Ad Analytics Cassandra Connector Error



Step 1:
List all alerts

⏰ **2 mins**

Step 2:
Fill out details

Tips for writing good technical runbooks

- Directly link alert to runbook
- Single source of truth
- Include commands and screenshots
- Minimize incomplete or outdated sections

http://bit.ly/srecon19-oncall

Alert Name	<Alert name>
Description	<One-sentence alert description>
Stakeholder impact	<One-sentence impact description>
Mitigation steps	1. Try: <command> 2. Monitor dashboards. 3. Inspect logs to diagnose issue: <link or See detailed steps below> If things do not recover, follow Escalation steps.
Escalation steps	Contact <team>. Major ingestion delays should be communicated to <upstream and downstream teams>.
Related services	<upstream and downstream dependencies>
Dashboards	<links>
Related links	<other docs or related runbooks>

http://bit.ly/srecon19-oncall

2 mins

Alert Name	<Alert name>
Description	<One-sentence alert description>
Stakeholder impact	<One-sentence impact description>
Mitigation steps	1. Try: <command> 2. Monitor dashboards. 3. Inspect logs to diagnose issue: <link or See detailed steps below> If things do not recover, follow Escalation steps.
Escalation steps	Contact <team>. Major ingestion delays should be communicated to <upstream and downstream teams>.
Related services	<upstream and downstream dependencies>
Dashboards	<links>
Related links	<other docs or related runbooks>

Step 3:
Make it easy to find

“You can try looking for that in the wiki, or maybe it’s in the Google Docs repo. Oh, and I’ve got some notes in my home directory, and I think I saw some emails about that a while ago”

😭

Step 3:
Make it easy to find

Centralized “home”

Easily searchable

Include relevant keywords or #tag

Inverted Pyramid

Most important/critical things first

Step 3: Make it easy to find

Centralized “home”

Easily searchable

Include relevant keywords or #tag

Inverted Pyramid

Most important/critical things first

2 mins

A good runbook is easy to find

Make alerts rich

Put actual commands and/or runbook link in the alert

Centralized “home”

Make runbooks searchable

Step 3:
Make it easy to find

2 mins

RUNBOOK USAGE

Beyond runbooks

- **Good for common cases**
- **What about unexpected situations?**
  - Provide tools to help in decision-making
  - Pattern match with past incidents
- **Automate as much as possible**

Example
Non-Technical Runbook

Non-technical runbook

http://bit.ly/on-call-srecon19

Incident Response Checklist

This document is for Ads incident first responders. First assess, escalate until the appropriate team is established, and take on the appropriate role.

Assess

Escalate

Communicate

Investigate and Fix

Clean Up

Non-technical
runbook

Incident Response Checklist

Assess

For example: errors served, % clients impacted, or financial loss to the business.

If it takes more than a few minutes to assess, assume it is very bad and move on to escalation.

- ❑ What is the business-facing impact?
- ❑ What is the consumer-facing impact?

Dashboards to consult:

- ❑ SignalFx - error percentages, latencies
- ❑ Splunk - log lines

Non-technical
runbook

Incident Response Checklist

Escalate

Outages run longer and with worse outcomes when tackled alone. It's better to escalate a false alarm than fail to escalate a serious issue.

Page the following as appropriate:

- ❏ Secondary on-call
- ❏ Manager
- ❏ Database Reliability Team (#dba)
- ❏ AWS Support Liaison

Non-technical
runbook

http://bit.ly/on-call-srecon19

Incident Response Checklist

Communicate

- ❏ Create a ticket in the ADS project with a brief description of the issue.
    - ❏ Add secondary and manager as watchers
- ❏ Consolidate triage communications to #ads-incident.
- ❏ Send email to ads-incident@ to liaise with financial stakeholders and downstream consumers of data: email templates.

Non-technical
runbook

Incident Response Checklist

Investigate and Fix

❏ Ads Runbooks List

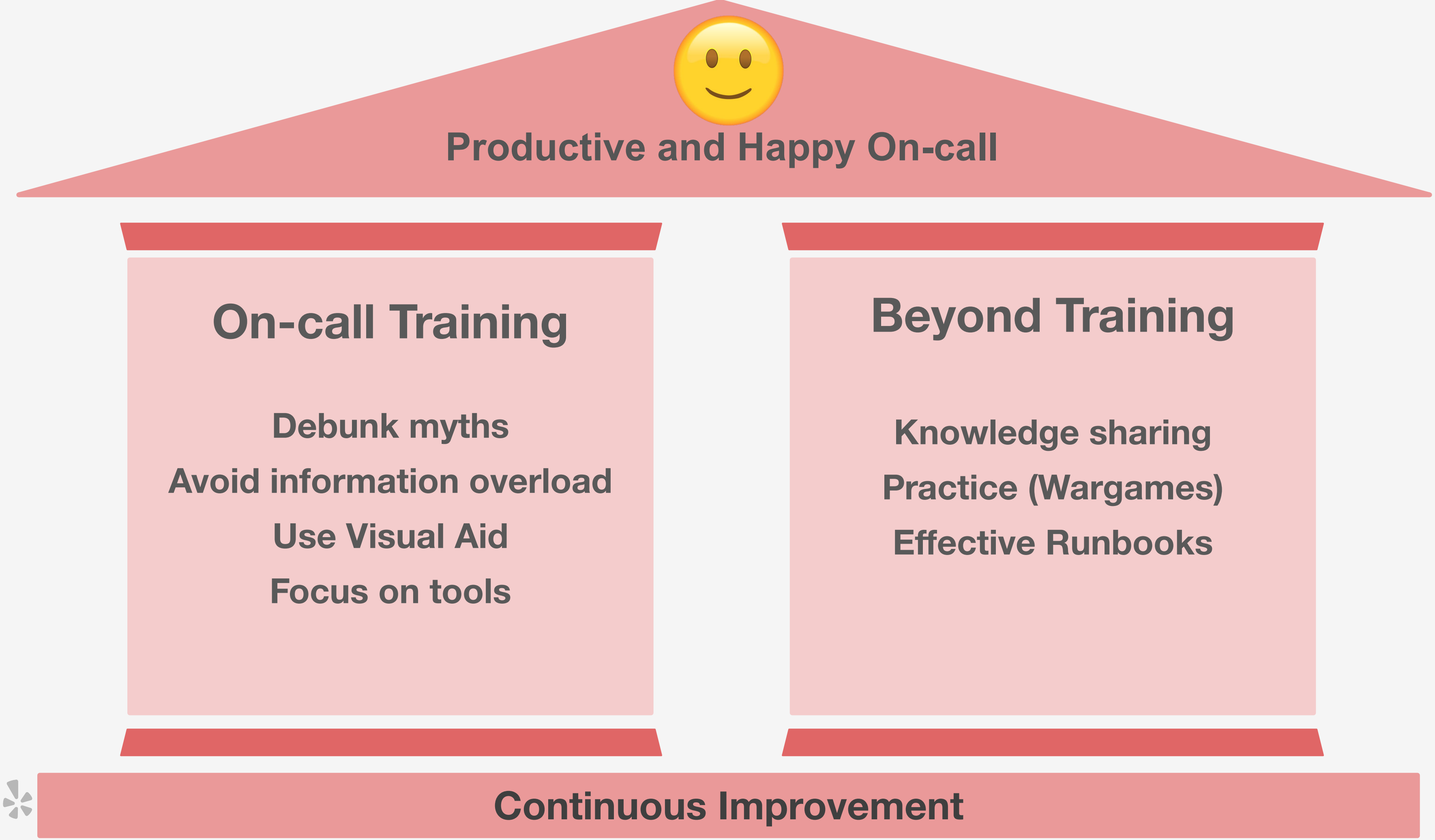
Clean Up

❏ Send all-clear email to ads-incident@

❏ File follow-up ticket for postmortem and set yourself as the assignee

Effective runbooks

- Clear instructions
- Easy to find and search
- Automate as much as possible



Productive and Happy On-call

On-call Training

Debunk myths

Avoid information overload

Use Visual Aid

Focus on tools

Beyond Training

Knowledge sharing

Practice (Wargames)

Effective Runbooks

Continuous Improvement

REFERENCES

Training materials

http://bit.ly/srecon19-oncall

Thank you!

Training Materials can be found here!

http://bit.ly/srecon19-oncall

REFERENCES

Additional Resources

Training new on-calls

- Accelerating SREs to On-Call and Beyond
- From Zero to Hero: Recommended Practices for Training your Ever-Evolving SRE Teams

Runbooks

- 7 Deadly Sins of Documentation
- Do Docs Better: Practical Tips

Postmortems/wargames

- Postmortem culture: learning from failure
- The on-call simulator: Building an interactive game for teaching incident response!