Soft Failures, Hard Goals
Accelerating Payments At Scale During the Pandemic

Kyle Guichard
VP Engineering, Data Platform & Technology Operations
@kylesj

Venus So
Sr. Director Engineering, Payments & Risk
Overview of Challenges
Initial Impact of COVID-19 Pandemic

The use of online platforms for back-office automation increased load on system

Switched immediately to 24/7 remote operations:

- Payment Operations
- Site Operations
- Data Center Operations

Increased reliance on our own internal cloud vendors
Batch Processing Payments

Pay

Get Paid

Approve

Invoice

Sync

ACH

Check

Instant

Card

International
Challenges

Can't slow down to deliver new features and innovate

Saw a notable increase in incidents

Customer-facing issues

Tribal knowledge vs. documentation & clear procedures

Straining of third-parties
Why focus on batch processing?

More than $96 billion moved annually through our platform.
Every $ of revenue goes through the Payment Engine.
Several incidents were non-code related.
Emerging Payments and new technologies add complexity to the system.
Scale - Need to onboard more engineers successfully.
Extreme Ownership Culture
Extreme Ownership

1. Own Everything
2. Create a Blame-Free Culture
3. Shift the Team Mindset
4. Do What It Takes
5. Break Down Barriers
Leaders Practice Extreme Ownership

“No bad teams, only bad leaders” – Jocko Willink and Leif Babin

- Set a target
- Focus the team
- Remove obstacles
- **Believe**
- **Commit**
- Change mindsets
- Empower the team
Building Resilience
Resilience as a project

- **Driver**: Designate a champion
- **Mission**: Explain why is it important
- **Principles**: Set guidelines, philosophy
- **Strategy**: Align on priorities
- **Timelines**: Timebox, then iterate
- **Target**: Hard Goals, RAISE the BAR
Payments Resilience Strategy

**Ops Efficiency - Automate Tasks**
Move repeated manual tasks to automated jobs

**Alerts & Metrics - Be Proactive**
Track both success and failures via dashboards and push notifications

**Antifragile Design - Be Defensive**
From "assume this will pass" to "assume this will fail"

**Reconciliation - Know Sooner**
We should know within 24 hours of recon mismatch.

**Education - Onboard Smarter**
Training in Payments (our code, design principles, jobs etc) for new members

**Test Automation - Catch Bugs Sooner**
80% Payments QA automation

**Target:** 40 days without customer impact
**Timeline:** Mar - May (90 days)
<table>
<thead>
<tr>
<th>Workstream</th>
<th>Deliverables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Catalog</td>
<td>1. Full Catalog of all (~70 daily) payment files and severities</td>
</tr>
<tr>
<td>Systematic Alerts</td>
<td>1. Enabled &gt;2x paging process alerts in prod</td>
</tr>
<tr>
<td></td>
<td>2. 100% coverage of critical positive log-based alerts</td>
</tr>
<tr>
<td></td>
<td>3. Self serve Operations Tool to add job alerts (v1)</td>
</tr>
<tr>
<td>Dashboards</td>
<td>1. Enhanced existing dashboard</td>
</tr>
<tr>
<td></td>
<td>2. Created additional operational dashboards</td>
</tr>
<tr>
<td></td>
<td>a. 60 reporting and analytics charts</td>
</tr>
<tr>
<td></td>
<td>b. 20 log-based charts</td>
</tr>
<tr>
<td>Process/Collaboration</td>
<td>1. Game Day</td>
</tr>
<tr>
<td></td>
<td>2. Training on alert handling</td>
</tr>
</tbody>
</table>
Cross-Functional Game Days

**Why** - If a scenario happened, would we know how to remediate?

**Who** - Cross-functional engagement is key

**What** - Documentation of process/playbooks

**Where** - in Production

**When** - Exercise crisis response without an actual crisis
Built new extendable payments automation framework
Conducted Unit test training sessions
Improved Unit Test Process - Unit test required for check-in
Enhanced Developer Tools - Gitlab migration
## Antifragile Design - Assume Failure

<table>
<thead>
<tr>
<th>Design principle</th>
<th>Old way</th>
<th>New way</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allow for &quot;soft failures&quot; (partial success) in critical batch processing</td>
<td>All records are rolled back. “all or nothing”</td>
<td>Partial Success allowed with minor intervention</td>
</tr>
<tr>
<td>Prevent human error</td>
<td>Scramble to create the exact query during an incident</td>
<td>Create script templates for each likely scenario</td>
</tr>
<tr>
<td>Focus on performance critical path</td>
<td>Combine primary and secondary processes in one job</td>
<td>Secondary tasks separated in their own jobs in non-blocking asynchronous queue</td>
</tr>
<tr>
<td>Reduce &quot;central&quot; choke-points like database</td>
<td>Pray and wait till the DB spike passes</td>
<td>Balance load across different time windows. Commit smaller batches.</td>
</tr>
</tbody>
</table>
Accomplishments

**Ops efficiency**
Operations Tool enhancements and self-serve alerts management

**Defence**
Partial success for batches DB scaling, and batch replay

**Education**
Revamped Payments Wiki including payments101 and new HLD guidelines

**Automation**
75% QA automation in card and domestic payments

**Reconciliation**
System and Cash Recon tools.

**Alerts**
Positive and negative alerts. Operational dashboards
Goal exceeded!

Customer Incidents

# Customer Incidents

Jan | Feb | Mar | Apr | May | June | July | August | Sept | Oct

152 Incident Free Days!
Third-Party Management & Business Process
How we approach RCAs

- Have a standard template
- Complete within three days of an incident
- Understand the timeline
- Categorize the remediation items:
  - Process
  - Monitoring
  - Code
- Ask the "Five Whys"
- Capture metrics
  - Not just SLI/SLO/SLAs, don't forget the MTTD/MTTR/MTBF

"The cost of failure is education" - Devin Carraway
Operational Readiness

**Common** dashboarding tools: TechOps, Payment Ops, Engineering

**Systematic Alerts**: Everyone can see, but clear ownership on failure for "calling the ball"

**Alert Catalog**: Complex systems require documentation

**Standard Operating Procedures (SOPs)**: Define early and centrally manage

**Process/Collaboration**: Over-communicate, hop on video if no immediate clarity on resolution
Business Continuity Plan (BCP)

Clear understanding of Disaster Recovery plan (people and technology)
Establish another kind of "Game Day" - annual table-top exercises
Emergency Response Team (ERT) - identify stakeholders
Crisis Communication Plan - call trees and notification patterns
Ask for help - look externally for best practices, especially if new
Business Impact Analysis (BIA) - including alternative providers
  - First-Party - What if X breaks?
  - Third-Party - What if Y goes down?
Vendor Management

Even if an issue is caused by a vendor, it is still your issue

Identify clear ownership of relationship

Add to Business Impact Analysis and consider redundant providers

Share BIA with executive team

Security reviews (initial and ongoing)

Financial review

Document integration
Next Steps & Capability Model

What would it take to build an extreme ownership culture?
Are deployments fully automated?
Code checkin only permitted if code coverage improves?
Automation testing completes in a timely manner?
Do engineers have time to address tech debt (target 20%)?
Do you practice Continuous Integration?
Is code review mandatory and efficiently delivered?
Following modern branching model process?
If a critical fix required, can you get it to prod safely in <1 hour?
Do you have "Top 3" metrics per team (ops & engineering)?
Thank You
We’re Hiring!