tucows/domains

Async Reseller Notifications

Key Architecture and Design Considerations

Date: Feb 12, 2024

Agenda

Jinish Bhardwaj
Sr Engineering Manager
Tucows Domains.

- Introduction
- Architecture & Design Considerations
- High Level System Architecture
- Demo
- Questions

What are they?

Async Reseller Notifications are essentially **automated HTTP POST** requests triggered by events in the TDP system, e.g. domain.created, contact.updated etc., sent to the reseller systems with a payload containing **non-sensitive** event data, enabling **one-way** communication between TDP and reseller systems

Architecture & Design Considerations

- Security
- Status Codes
- Retrying
- Throttling
- Idempotency
- Handling Failure
- Scalability
- Ordering
- Filtering
- Documentation
- User Interface
- Observability

Security

- Notifications will only be sent to **HTTPS** endpoints on the reseller systems
- All communications will be over a **secure encrypted** connection only
- At the time of subscribing to receiving notifications, the resellers will need to follow a **verification process** to verify the endpoints where they wish to receive notifications
- All notifications will be signed using a signature (**HMAC+SHA256**), ensuring that the notification cannot be tampered with
- TDP system will publish a list of **IP addresses** used for sending notifications, allowing the resellers to whitelist those addresses

✓ HTTP Status Codes

- **200** : Reseller System should respond with a HTTP Status 200 OK to indicate that they have received the notification. Upon receiving this response code, TDP system will assume that the notification was **successfully delivered**
- **3xx** : This indicates that the endpoint has a redirection setup. TDP will not post notifications to any redirected endpoints. TDP will **not retry** delivery to reseller system. Subscription will be marked as **deactivated** and reseller will be notified on **registered notification email**
- **4xx** : This would indicate an issue with the payload sent to the reseller system. TDP will **not retry** delivery to reseller system until further investigation
- **5xx** : This would indicate a transient failure with the reseller system (DNS issues, incorrect routing). TDP will **retry** delivery of this notification

↻ Retry Mechanism

- TDP will retry delivery of failed notifications **upto 3 times** with exponential backoff, only to active subscriptions
- Once 3 retries have been exhausted, an email will be sent to the **registered notification email** (email the reseller provided at the time of subscribing to receiving notifications)
- After continuous failure to deliver notifications to reseller system for upto **2 days**, TDP will mark the subscription as **Degraded**
- After continuous failure to deliver notifications to reseller system for upto **5 days**, TDP will mark the subscription as **Deactivated**
- Upon each status change to the subscription, TDP will notify the resellers by sending an email to the registered notification email

Throttling and Rate Limiting

- TDP notifications will be **rate limited** on a **per subscription** basis to **prevent abuse** and unfair usage of the notification system
- Any notifications beyond the rate limit will be **throttled** and deferred to the **following day**
- Resellers will be able to request an **increase** to the rate limit by logging into the **Reseller Notifications Management Dashboard**, which will be considered on a case by case basis

Idempotency

- Each TDP notifications will include a unique **Event Id**
- Each TDP notification will also include an **Idempotency Key**, that will be unique to every notification and will be calculated based on the hash of the notification payload
- TDP documentation will provide guidelines and best practices for the resellers to ensure that processing the same event multiple times yields the same results

☒ Handling Failures

- TDP will follow a **fixed but configurable timeout policy** when publishing notifications to the reseller systems
    - The reseller system must respond within this time limit with a HTTP 200 OK status code to indicate that the notification has been successfully received
- Any notification publishing exceeding this limit will be either marked for **retry**, or as **failed** (if maximum retries have been exhausted)
- TDP will build a resilient error handling strategy with documented error codes and descriptive messages to track failures and other key metrics in real-time

Scalability and High Availability

- TDP notification system will be built on top of RabbitMQ; the same message broker that TDP core system is built on
- Message broker will be deployed in a clustered configuration, ensuring that notifications are not lost in case of broker failure
- Notifications architecture ensures that the notification queue is not saturated and ensure a fair delivery policy by separating the retry queues from the main notification queue
- Notification system also supports Dead Lettering of notifications
- On the reseller end, TDP allows resellers the flexibility of configuring multiple endpoints with notification filtering

Ordering

- TDP will not guarantee the ordering of the notifications
- Resellers will be instructed to implement idempotency in their notification handling
  - They should observe the **Event Id** and **Idempotency Key** included in the notification to make their notification handling idempotent
- A timestamp will also be added to all notifications to allow the resellers to re-order or ignore events when necessary
  - This will also allow reseller systems to measure the processing delay for better APM reporting and monitoring

Notification Selection and Filtering

- Resellers will be able to select the notifications they wish to receive at the time of subscribing to the reseller notifications
- Later resellers will be able to manage (add, remove, view) the notifications filtering through the subscription api
- Resellers will only receive selected notifications from TDP
- Notifications are only **unique within a subscription**
- Resellers will be allowed to **register** for at most **10 subscriptions**, each allowing for all or a subset of all notifications

Documentation

- All notification types will be documented within TDP documentation website including:
    - Notification Types and Names
    - Notification Attributes Definitions
    - Example payloads for all Notifications (with all variants)
- Documentation will also contain details on how to manually validate the notification signature (in case the reseller does not wish to use the code library provided by TDP)
- Complete documentation for developers to integrate with TDP Async Reseller Notifications platform (with code samples in C#, Golang and Python)

User Interface

- Resellers will have access to a **Notifications Management Dashboard** that will allow them to:
  - **Manage** Subscriptions like activate, deactivate, add and remove existing subscriptions
  - View **current rate limit quotas** and **request increase**
  - View delivered and failed in the **last 7 days**
  - View **scheduled** notifications (not yet delivered because of throttling)
  - **Manually resend** already delivered notifications
  - View and update the **secret** used for signing the payloads
  - View and manage all existing endpoints

Observability

- TDP will manage the following key Metrics (some of these metrics will also be available on the UI Dashboard for resellers to see):
  - Delivery Metrics (Success rate, Failure rate, Delivery Latency, Retry rate)
  - Queue Metrics (Queue Length, Queue Lag)
  - Notification Payload Size
  - Resource Utilization Metrics (Workers, CPU, Memory, Network Usage)
  - 4xx Error Codes
- Notifications will follow same Tracing and Monitoring guidelines as followed by TDP

Wait, there’s more

Code Library

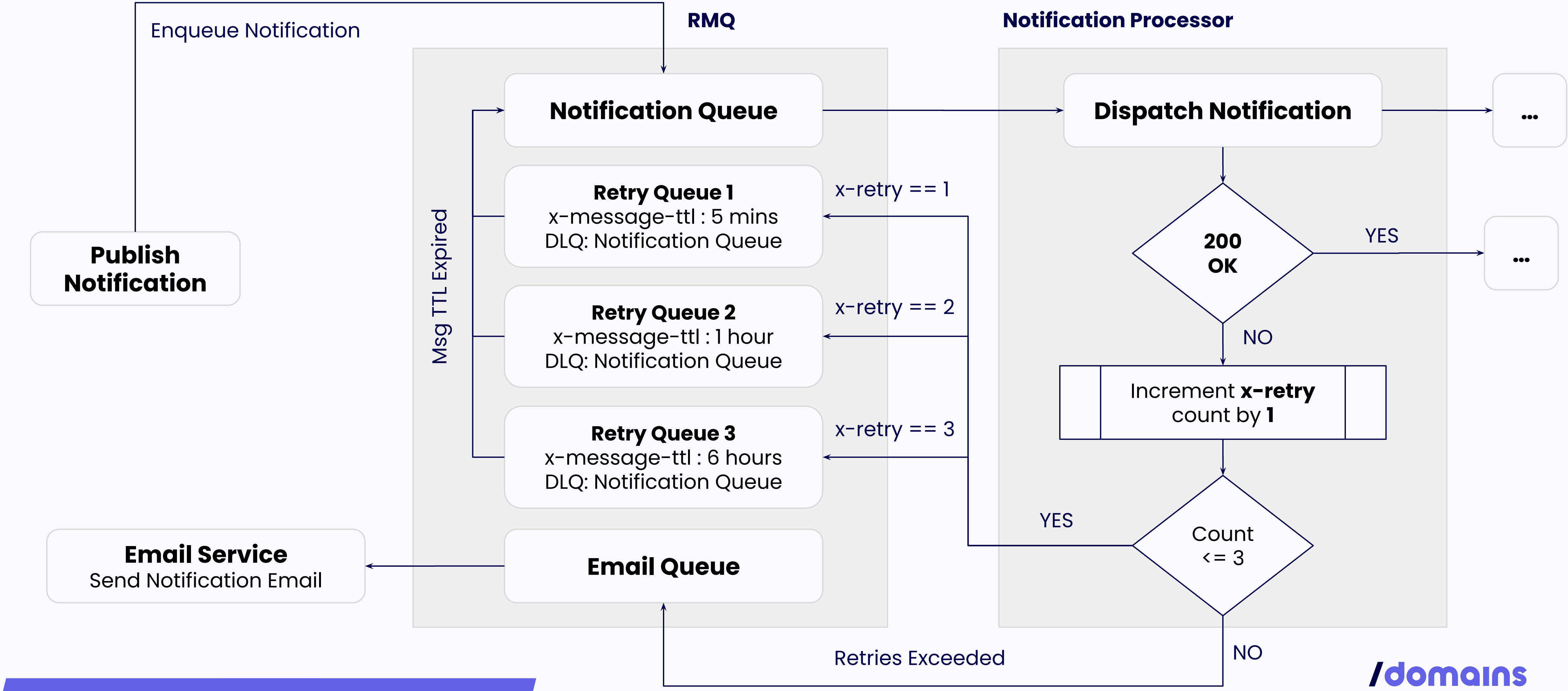
- Official TDP library to verify the notification signature
- Available in 3 programming languages
  - Golang
  - Python
  - C# .Net Core

Testing

- TDP CLI to enable integration testing for resellers on their local environments

/domains

High Level System Diagram



/domains

DEMO

/domains

Questions?

/domains

Feedback

Join at slido.com #Domains

/domains

Thank you!

tucows/domains