WHITEPAPER

Using PubNub to Secure In-App Chat from Scams

PubNub

Introduction

In today's digital age, ensuring the security and integrity of in-app chat systems is paramount. With the rise of scams and inappropriate user behaviors, businesses need robust mechanisms to detect and block such activities to protect themselves and their end users. This whitepaper delves into the benefits of using PubNub for in-app chat, specifically focusing on its capabilities to screen for potential scams and how to enforce SHAFT rules established by the CTIA when sending SMS messages.

The Challenge

Scams and Inappropriate Behaviors in In-App Chats

Adding Chat features directly to your app provides huge benefits for user engagement. But providing the freedom to communicate also comes with risks.

A lot has been said (and built) around chat “moderation,” ensuring that users don’t say anything that would be considered “inappropriate” for the broader user base of your application. PubNub provides many flexible ways to moderate chat apps in real time, with block lists, AI, and more.

However, an even more insidious use of in-app chat is for scammers to trick other users into providing information that allows scammers to steal money and other valued assets.

This risk increases exponentially for applications in the financial sector (eBanking, crypto, portfolio management) and other industries like Real Money Gaming, NFTs, and many others.

These scams can take on any number of strategies: tricking users into “sideloading” an Android (apk) app, convincing users to share credentials, simulating chatbots to get credit card numbers, etc.

An added risk is when chat messages "fall back" to SMS or Push Notifications that appear on users' phones while in their pockets. SMS Short Code numbers have specific requirements (defined by the CTIA), and they risk getting their SMS Short Code number blocked if messages contain SHAFT-C content (Sex, Hate, Alcohol, Firearms, Tobacco, or Cannabis).

That's where PubNub's Event-Driven Development platform comes in.

Event-Driven Chat Applications

PubNub's platform makes building real-time, event-driven applications easy, including powerful in-app chat functionality. Event-driven application design is not new, but its use for large-scale social features like chat isn't always obvious, and the value is huge.

PubNub has 50+ SDKs, and its global pub/sub messaging network makes sending and receiving chat messages (or any other kind of real-time "event") easy across any device. While PubNub delivers more than 3 trillion events per month across many different use cases, chat messages are the most common use of our network.

PubNub treats all chat messages as real-time events that flow through our network, with sub-100ms delivery latencies, regardless of the number of people in a chatroom. We've even powered apps with over 25M people in the same chatroom!

The benefit of using PubNub for your In-App Chat infrastructure is that we aren't a chat-only solution. PubNub's network does more than just send and receive chat messages. The following services layer on top of our global Pub/Sub network provides limitless flexibility, especially when it comes to screening content for scams:

Functions is a service that allows you to put code *directly* into the PubNub network, screening content in real time before it's received in a chatroom. Functions provide many ways to detect malicious activity; see examples below.

Access Manager allows instant token revocation to block misbehaving users.

Events & Actions lets you detect a subset of events (e.g., suspicious messages) and stream them to your backend (or any other cloud service API) for later analysis.

Moderation Dashboard lets you review user content with an easy GUI for blocking or limiting user access. And, because the Moderation Dashboard is open source, you can modify and enhance it to add specific features you need for your organization.

PubNub Insights provides visibility into data channels, user locations, and other aggregated usage metrics that help detect unusual behaviors across your app. And, with ChatGPT analysis of the PubNub Insights metrics, it's easier than ever to get AI help detecting scammers.

Chat Components provide rich chat functionality on par with anything you could get from chat-specific vendors that don't provide the tools necessary for security, scam detection, and analysis. This includes typing indicators, threaded messages, unread message counts, multiple chat rooms, missed message catch-up, and many other features.

App Context features allow you to store metadata for users and channels (i.e., chat rooms), potentially treating users differently based on their trust level from prior interactions, other chat room metadata, etc.

These are just a few examples of the tools provided by PubNub's Event-Driven Development Platform.

Using Functions to Prevent Scams

As mentioned previously, Functions provides a scalable, affordable approach to intercepting all chat messages and running them through any bespoke logic to detect scams and other content.

As messages flow through PubNub as individual events, they can be screened and filtered in numerous ways. This is also where business logic can sit that routes messages to Push Notifications or SMS as a fallback. Thus, you can use a single solution to screen and filter both in-app chat messages and those that flow externally to the users' devices.

Using Functions for scammer detection can be done in various ways, all providing different levels of protection and value. And because Functions is anchored with *your* code, it's always easy to improve and change.

A Simple "hard-coded" example

Some people simply create a "block-list" of substrings they wish to flag. In the example below, the customer sends chat messages via PubNub when the user is online but also falls back to SMS when the user is offline due to the Presence feature.

Presence enables the detection of offline users, so when a recipient is offline, companies can use Functions to determine if the message should be sent via SMS instead. This ensures that important messages reach users even when they are not actively using the app.

In the example, the customer has written a simple Function to screen all chat messages for SHAFT-banned content, whether or not they are going to be sent via SMS:

```
// Function to enforce SHAFT rules
export default (request) => {

     const messageContent = request.message.content;

     // Example: Using K/V store to look up SHAFT rules

     const shaftRules = kvstore.get('SHAFT_RULES');

     if (shaftRules.includes(messageContent)) {
          // Republish the message on the SCAM_ALERT channel
          pubnub.publish({
                  channel: 'SCAM_ALERT',
                  message: request.message
           });

           // Abort the original message

           return request.abort("SHAFT rule violation detected");
     }

     return request.ok();
};
```

The example above uses the PubNub K/V Store to retain a list of flagged terms to screen for. Suppose a pattern match for a keyword is found. In that case, the message is not sent into the chatroom but instead is re-published to a channel called "SCAM_ALERT." Using the *Events & Actions* service, you can funnel all messages on this channel to an endpoint of your choice for later (or real-time analysis).

This code could easily be modified to revoke the offending user's access token to prevent further application usage.

Using AI Models for Scam Detection

While starting with a "block-list" approach is a quick way to get started, a more powerful detection scheme is to send some or all chat messages through an AI model for detection. Rather than providing a single model, PubNub makes it easy to incorporate any AI scam model, and many new ones are being developed each week. For example, HuggingFace.co has many scam detectors worth checking out.

As messages flow through PubNub as individual events, they can be screened and filtered in numerous ways. This is also where business logic can sit that routes messages to Push Notifications or SMS as a fallback. Thus, you can use a single solution to screen and filter both in-app chat messages and those that flow externally to the users' devices.

Running these models is easy using Google Cloud, AWS, Azure, and many other services. Calling these models from Functions is as easy as the earlier “block-list” example:

```
// Function for scammer detection

export default (request) => {
    const messageContent = request.message.content;

    // Make an API call to the hosted AI model for scammer detection
    const xhr = require("xhr");
    const http_options = {
        "method": "POST",
        "headers": {
            "Content-Type": "application/json"
        },
        "body": {
            "message": messageContent
        }
    };


    return xhr.fetch('YOUR_AI_MODEL_ENDPOINT_URL',
        http_options).then((response) => {
        const likelihood = parseFloat(response.body.likelihood);
        // If the likelihood of being a scammer is greater than 0.7, flag
the message
        if (likelihood > 0.7) {
            // Republish the message on the SCAM_ALERT channel
            pubnub.publish({
                channel: 'SCAM_ALERT',
                message: request.message
            });
            // Abort the original message
            return request.abort("Scammer detection rule violation
detected");
        }

        return request.ok();
    }).catch((error) => {
        console.error("Error calling the AI model:", error);
        return request.ok();
        // Continue processing the message even if there's an error
    });
};
```

The previous pseudo-code example makes an external API call to a hosted AI model, scoring each chat message for scam content. Anything > 0.7 (i.e., 70% likelihood) gets flagged and handled appropriately.

Using AI Models Cost Effectively at Scale

While examples like the one above make for a powerful demo, there are economic considerations, especially with the cost of running 100% of chat messages through an AI model. Besides, you probably know which users are more “trusted” and thus don’t need as much screening with them.

This is when Event-Driven development makes things easy, especially with PubNub. Here are some great examples of cost reduction when operating AI models at scale:

- **App Context** allows you to keep arbitrary information about each user. This is a great place to build a *trust score* with power users whose scam likelihood is much lower. You can bypass expensive AI detection and use a simple block-list for these high-trust users.
- Alternatively, you can use the trust score to change the “scam likelihood” threshold for more trusted users, thus minimizing false positives for the users you trust the most.
- With **Functions** and **Events & Actions,** there are several ways to continually tune your event flow behavior to improve performance and lower costs as you start recognizing user behavior patterns.

Training AI Models over Time

As scammers get more sophisticated, your chosen scam detection AI model must also be improved. Historically, the few teams that built a detection model didn’t have a good way to keep the model up to date. With PubNub and event-driven architectures, this now becomes easy.

When messages are flagged, and scammers are detected (through programmatic means or human screening), these events flow through PubNub and are picked up with a filter via Events & Actions. These events get queued and delivered to any endpoint you want, thus providing a stream of training data directly into your AI model. Updating your AI model in production is as simple as swapping the hosted model and/or providing a new endpoint for the function to call.

Beyond Functions: More Powerful Event-Driven Services

PubNub Functions is only one vector for detecting, alerting, and blocking scammers. When building chat apps on PubNub, the following built-in API Services also become indispensable:

Access Manager

PubNub's Authorization layer is token-based and easily integrates into your Authentication system. Since these tokens are fine-grain with variable TTL (time-to-live) settings, you can easily block a user from accessing specific chat rooms, or all of them, with a simple API call…or even have Functions automatically block the user when a scam is detected—no server-side code is needed in your software stack.

Events & Actions

PubNub's E&A service allows you to easily create filters (using a simple point-and-click no-code interface) that can find events and route them to your backend database or file store for future analysis. So, messages flagged by Functions can easily be routed to other data channels, detected by E&A, and exported to your back-end asynchronously without slowing down the customer experience. This will allow you to further analyze suspected scammers without writing any code.

Moderation Dashboard

PubNub provides an open-source moderation dashboard that provides myriad ways for a chat administrator to screen and moderate content across any chatroom. The Moderation Dashboard supports both a word list and pre-built integrations into multiple third-party moderation vendors.

However, because the Moderation Dashboard is open source, you can easily modify it, fork it, and add further end-points for new kinds of screening without building up a complete UI from scratch. It's an infinitely extensible, pre-built framework that can be extended to meet the needs of your business.

Chat Components

Chat components provide rich chat functionality on par with anything you could get from chat-specific vendors that don't provide the tools necessary for security, scam detection, and analysis. This includes typing indicators, threaded messages, unread message counts, multiple chat rooms, missed message catch-up, and many other features.

Beyond Functions: More Powerful Event-Driven Services

App Context

PubNub's powerful App Context APIs allow you to store any meta-data about your users. It's a great place to build a "trust index" based on prior customer chat messages. As end-users use the system more, and as you screen their content, you can easily increase a "trust index" score, saving money and increasing performance by only occasionally screening messages from "high trust" users while screening new and untrusted users more frequently. This is simply one approach you can use; App Context provides a hugely flexible approach to creating a bespoke scam detection and moderation solution without spending a lot of time, effort, or cost.

Conclusion

Using PubNub's Event-Driven Platform to power your in-app chat solution offers a powerful and efficient way for companies to detect and block scams and inappropriate user behaviors. By leveraging Functions and our other API services, companies implement real-time scam detection features, adapt quickly to new scamming techniques, and scale their detection capabilities as their user base grows.

Integrating AI services further enhances scam detection by leveraging advanced machine learning algorithms and staying up-to-date with the latest scamming techniques. With PubNub, companies can provide users with a safer in-app chat experience while saving time and reducing infrastructure setup and maintenance costs.

Let PubNub help you protect your organization from the damage caused by scammers targeting in-app chat users. Contact us today to discuss your project, or sign up for a free trial to get up to 200 MAUs or 1M monthly transactions for free.