「Nostr and ATProto」by Shreyan Jain を読む

- 「**Nostr and ATProto**」という記事が出ました。
  https://shreyanjain.net/2024/07/05/nostr-and-atproto.html
- これは、Shreyan Jain 氏が Nostr と ATProto について包括的に比較した記事です。
- すごくいい内容らしい。

- **英語！**
- **文量が多い！**

- これは、AI を活用して和訳・要約した資料です。
- 英語 + 長文が読める人は原文を読みに行こう！

"Nostr and ATProto" by Shreyan Jain

Nostr と ATProto の祖先

影響を受けた技術

- Nostr と ATProto は、既存の分散型ソーシャルネットワーキング技術、特に**Secure Scuttlebutt**の影響を受けています。

Secure Scuttlebutt（セキュア・スカトルバット）ってなに？

概要

- オフラインファーストで、意図的で、スローなコミュニケーションのために設計されたプロトコル
- 2014 年に、ドミニク・ターというニュージーランドのプログラマーによって開発

背景

- ターは、インターネット接続が限られているヨットでの生活の中で、Twitter のような常時接続を前提とした中央集権型ソーシャルメディアの限界を感じ、Secure Scuttlebutt を考案

主な特徴

- **オフラインファースト**: インターネット接続がなくても動作し、接続が回復した際に同期
- **意図的なコミュニケーション**: Constant connection を前提とした Twitter とは異なり、ゆっくりとしたペースでコミュニケーションをとる
- **Big Tech からの脱却**: 中央集権的なプラットフォームへの依存を避ける

Notr と ATProto に与えた影響

Nostr

- ピアツーピアネットワークからクライアントリレーモデルに移行。
- イベントがチェーンを形成しないように。
- Secure Scuttlebutt の概念を再考しています。

ATProto

- ATProto は、Secure Scuttlebutt の自己証明データ構造を採用して、クライアントサーバーモデルで動作するピアツーピアデータモデルを適用しています。

学びの源

- どちらのプロトコルも、Twitter、ActivityPub、Scuttlebutt などの以前の分散型ソーシャルネットワーキングの試みから学び、グローバルなソーシャルネットワーキングに適した分散型ネットワークを構築しようとしました。

データの取り扱い

- ただし、Nostr と ATProto は、データの信頼性、アイデンティティ、プライバシーへのアプローチが異なります。

Nostr と ATProto のしくみのちがい

アイデンティティ

- **Nostr**: ユーザーのアイデンティティは公開鍵と秘密鍵のペアです。このアプローチは、サーバーへの依存を完全に排除し、ユーザーに最大の自律性を与えます。
- **ATProto**: 「did:plc」と呼ばれる分散型識別子（DID）システムを採用しており、ユーザーの公開鍵と、ユーザーのデータとコンテンツをホストする Personal Data Server（PDS）へのリンクが含まれています。この設計では、ユーザーの利便性とアカウントの回復性を優先しており、PDS が重要な役割を果たします。

データ

- **Nostr**: 個別の自己完結型のデータ単位である「イベント」を中心に構築されています。各イベントはユーザーの秘密鍵で署名され、複数のリレーに分散して保存できます。この設計は、データの複製と可用性を促進し、検閲に対する回復力を高めます。
- **ATProto**: 「リポジトリ」と呼ばれる構造化された形式でユーザーデータを編成します。リポジトリには、ユーザーの投稿、コメント、いいねなどのレコードが含まれており、すべて暗号化によってリンクされ、検証されます。ユーザーのリポジトリは、主に PDS に保存されますが、自己認証型であり、他の場所に複製することもできます。

信頼

- **Nostr**: 信頼の最小化を重視しています。クライアントは、リレーから受信したイベントの整合性と信頼性を検証する責任があります。ユーザーは、クライアントを信頼する必要があります。
- **ATProto**: ユーザーは、PDS を信頼する必要があります。ただし、ATProto の自己認証型データ構造により、ユーザーは PDS の動作を検証し、必要に応じて PDS を切り替えることができます。

開発

- **Nostr**: より分散化され、ボトムアップの開発アプローチを誇っています。プロトコルは仕様のオープンセットに基づいており、開発者は自由に実験を行い、新しいクライアント、リレー、拡張機能を革新できます。
- **ATProto**: より中央集権的で慎重な開発プロセスを採用しています。コアプロトコルとリファレンス実装は、主に Bluesky チームによって管理されており、変更はより構造化された方法で導入されます。

まとめ

- Nostr は、鍵管理、データの複製、クライアント側の検証を通じて、信頼の最小化と検閲耐性を優先しています。
- ATProto は、ユーザーエクスペリエンス、データの整合性、より構造化された開発アプローチを優先しており、PDS が重要な役割を果たします。

ユーザーと開発者に与える影響

アイデンティティ

- **Nostr**: 暗号化キーペアを管理するのは、一般ユーザーにとっては技術的に困難で、エラーが発生しやすく、キーを紛失するとアカウントが回復不能になる可能性があります。
- **ATProto**: DID により、ユーザーの利便性とアカウントの回復性を優先しています。そのかわり、DIDPLC を検証し信頼する必要があります。

データ

- **Nostr**: イベントは自己完結型で堅牢ですが、削除や編集などのデータ操作は複雑になり、実際には不可能になる可能性があります。
- **ATProto**: リポジトリは削除や編集することができます。自己認証型であり、他の場所に複製することもできます。

信頼

- **Nostr**: 「信頼の必要性を排除する」ことを目指しており、クライアント側の検証に大きく依存しています。このアプローチは、検閲に対する耐性を向上させますが、複雑なクライアント側の検証メカニズムを必要とし、これが開発の障壁となる可能性があります。
- **ATProto**: ユーザーは依然として PDS を選択および信頼する必要があります。PDS は、署名キーの管理とデータリポジトリの整合性の維持を担当します。

開発

- **Nostr**: 最小限のプロトコル仕様と活発な開発コミュニティを特徴とする、より分散型の「バザール」スタイルの開発アプローチを採用しています。このオープンな性質により、実験と急速なイノベーションが可能になりますが、実装の断片化や標準化の問題が発生する可能性があります。
- **ATProto**: より中央集権的で慎重な開発プロセスに従い、Bluesky がプロトコルの方向性を導き、同時にコミュニティからのフィードバックも取り入れています。このアプローチにより、断片化のリスクは軽減されますが、イノベーションのペースが遅くなり、集中化の問題が発生する可能性があります。

アプリケーション

- **Nostr**: リレーとクライアントの両方の実装によるフィルタリングに依存しています。この設計により柔軟性とリレーレベルでのカスタマイズが可能になりますが、開発の複雑さが増し、特にリソースを大量に消費する操作では、クライアントのパフォーマンスが低下する可能性があります。
- **ATProto**: データをインデックス化し、クライアントに提供する集中型の「AppView」を利用しています。AppView は、クライアント側の作業負荷を軽減し、よりスムーズなユーザーエクスペリエンスを実現しますが、集中化のポイントとなり、潜在的に単一障害点となる可能性があります。

まとめ

- Nostr は、技術に精通したユーザーや開発者にとって魅力的な選択肢です。これらのユーザーや開発者は、制御、柔軟性、検閲に対する耐性を優先しています。ただし、使いやすさと標準化が課題となっています。
- ATProto は、使いやすさ、回復可能性、開発のしやすさに重点を置いています。これは、より幅広いユーザーや開発者にとって魅力的ですが、分散化と検閲に対する耐性の点で妥協が必要です。

「Zooko の三角形」と照らしてみよう

Zooko の三角形とは

- セキュア
- 人間が理解しやすい
- 分散化

という、識別子の 3 つの望ましい特性を同時に満たすことが難しいことを示す概念です。 ほとんどすべてのシステムは、これらの特性のいずれかを犠牲にせざるを得ません。

- **セキュア**: 許可されていないユーザーが ID を不正利用することを防ぐことができる。
- **人間が理解しやすい**: 人間にとって覚えやすく、使いやすい識別子であること。
- **分散化**: 特定のサーバーや機関に依存しない、分散化された方法で識別子を管理できること。

セキュリティと分散化

- 両プロトコルともセキュリティと分散化を実現しています。
- データがユーザーのものであることを暗号化によって検証可能にすることでセキュリティを確保しています。
- 特定のサーバーに恒久的に紐づかない ID を採用することで分散化を実現しています。

人間が理解しやすいか

- **Nostr**: しやすくありません。最大限の分散化を優先し、ユーザーの ID を公開鍵と非公開鍵のペアに限定し、サーバーを一切介在させません。
- **ATProto**: しやすいです。ユーザーの利便性を優先し、ユーザーの署名鍵ペアをサーバー（PDS）に置くことで、ユーザーによる鍵管理の煩雑さを解消しています。

分散型 SNS はこれからどうなっていくのか

アイデアの収束

- Nostr と ATProto はすでに互いのアイデアを借用し始めており、将来的にはさらに収束していく可能性があります。
- Nostr は、ユーザーが自分の秘密鍵をサーバーに保存できるようにする NSecBunker のアイデアを検討しています。これは、ATProto の PDS（Personal Data Server）に似ています。
- ATProto は、Nostr のフィルターモデルに似たものを採用して、開発者がアイデアを簡単に試せるようにすることを検討しています。

ブリッジング

- ブリッジングが Nostr と ATProto を含む異なる分散型ソーシャルメディアプロトコル間の相互運用性を可能にする方法として浮上しています。
- **Bridgy Fed** のようなサービスは、すでに Fediverse と ATProto をブリッジしており、将来的には Nostr へのネイティブサポートが追加される可能性があります。

結論

- いままで述べてきた違いにもかかわらず、両方のプロトコルが互いの強みから学び、将来的には**より統合された分散型ソーシャルメディアエコシステム**に向かって進化していくことができると示唆しています。