参考資料 - 背景

Web2とは

Web2では、ウェブアプリケーション(以降、アプリという)は単一の管理者によって管理される。現状のほとんどのアプリがこの中央集権的な方式である。Web2のアプリの例としては、私たちが普段使用するFacebook[1-1]やGoogle[1-2]などが挙げられる。Web2のアプリは以下の可能性を持つ。

1. 開発者が利用者の個人情報をみる (不透明性)
2. 得た個人情報を不正に利用する (不透明性)
3. 利用者の意図しない操作を行う (検証不可能性)
4. アプリの利用を禁止する (検閲可能性)

2はFacebookの選挙不正疑惑[2]が、3はTwitter[1-3]の内部ツールを標的としたソーシャルエンジニアリングによる影響力の強いアカウントからの意図しない投稿[3]、4はGitHub[1-4]のアクセス制限[4]が実際に起きた例として挙げられる。

曖昧さ回避のため、上記３つの性質を定義する。
不透明性
どのような情報が取得され、どのように分析され、どのように使用されているのかわからない

検証不可能性
受け取ったデータの差出人が本人か検証することができない(注[1])

検閲可能性
特定のアプリ利用者のアプリ利用を制限・禁止することができる

Web3ムーブメントとは

上記4つの可能性があるにも関わらず、ユーザーはWeb2のアプリを利用している。理由としては、利便性やアプリの開発元を信頼しているということが考えられる。Web3ムーブメントはweb2の持つ性質を踏まえ、アプリに透明性、検証可能性、検閲耐性を持たせようとする動きである。

先行研究・プロダクト

先行する研究・プロダクトとして以下の3つを紹介する。

IPFS
IPFS[5-1]は静的なファイルをP2Pネットワークで分散的に保存できるプロトコルである(ブロックチェインは使用しない)。P2Pネットワーク参加者に経済的インセンティブはなく、また静的なファイルしかホスティングできないためにインタラクティブなアプリが提供できないという問題が存在する。

Polkadot
Polkadot[5-2]は、異なるブロックチェインを繋げて大きな一つのブロックチェインを作ることで、ブ

[1] しばしば「その人の言っていることが本当かどうか」という文脈で検証可能性が語られるが、ここでの検証可能性は「本当にその人か」である。ウェブアプリに必ずしも前者の言う検証可能性が必要とはいえない。

ロックチェインの持つスケーラビリティ問題と、インセンティブ不整合[5-3]などのセキュリティ問題を解決しようとしている。

DFinity
DFinity[5-4][5-5]の位置づけとしては、動的コードの実行と、インセンティブを備えたIPFSのようなものである。プロジェクト開始初期から現在に至るまで開発設計の指針がブレているが、現状はトークンによってネットワークのインセンティブが保たれている。具体的にはDFinityのネットワークに参加しているノードに対し、DFinityの開発元が発行する通貨(トークン)で対価が支払われる。(注[2])

現状分析

Web2のなかで非中央集権にするべきアプリと、中央集権のままで十分なアプリに分類した。また、分散システムにおけるトークンの働きの概略を記述する。

中央集権のままで十分なアプリ

• 開発者への信頼が利用の絶対条件のアプリ。

非中央集権にすべきアプリ

• 中央集権のままで十分なアプリ以外のすべて。

アプリを非中央集権にすべき理由は、ユーザーが信頼できない開発者や管理者の提供するアプリを利用するという点にある。アプリの開発者はWeb2の性質(不透明性、検証不可能性、検閲可能性など)を利用することができる。提出したレポートでは、上記の非中央集権にすべきアプリのみを対象とする。

トークンのはたらき

通常のピュアP2Pネットワークのソフトウェアはネットワークのコントロールが不可能に近いことが多く、P2Pソフトウェア開発者やP2Pネットワーク参加者は金銭的な利益を得ることは難しいと考えられている。対してブロックチェインを使用したP2Pネットワークの場合、開発者およびP2Pネットワーク参加者のインセンティブは暗号通貨によって保証される。開発者は暗号通貨を(Initial Coin Offeringなどで)発行することによって法定通貨(または別の暗号通貨)を受け取り、P2Pネットワークの参加者は、発行される暗号通貨を得たいために働く。

---

[2] 情報があまり出ていないのでトークン発行方法の詳細は不明であるが、DFinityは多額の融資を受けているため、リターンを返す必要がある。プラットフォーム化を行い、プラットフォームを使用するためのトークンを何らかの方法で販売し、利益をあげる。

参考資料 - 諸考察

インセンティブ

「登場人物とインセンティブ」で示した各登場人物が持つインセンティブを満たしているかを確認する。

TP
TPはUserから、登録時または定期的にお金をもらうことで経済的インセンティブを満たす。

Dev
DevはReqからの対価を受け取ることでインセンティブを満たす。

Node
NodeはReqからの対価を受け取ることでインセンティブを満たす。

Req
ReqはNodeを通してUserに要求し、DevにUserの正当性確認を要求することでインセンティブを満たす。

User
UserはNodeにデータを挿入することによって、Nodeが利益を上げることができ、またDevは得た利益で新しいアプリを開始する可能性があり、インセンティブを満たす。

セキュリティ

鍵や宛先のすり替え

Devはソフトウェアを配布する際、DevキーとIPアドレスを含めるが、配布する際に何者かがDevキーとIPアドレスを書き換え、収益のみを横取りする可能性が考えられる。この問題への対策として、最初は信頼できるNodeへソフトウェアを配布し、大多数がDevを認知した段階で不特定多数にソフトウェアを公開するといった方法が挙げられる。しかし結局はUserやNodeの自由意志によって、Devではない者にID発行要求を行うことが可能であるため、UserやNodeのインセンティブ次第である。正当なDevに対するID発行要求を行うインセンティブとして、既存のアプリを改善したり新しいアプリへの期待などが挙げられる。つまりDevは登場人物から感謝や応援、期待をされなければ、他の者に利益を横取りされる。

Devへの信頼性度合い

DevはUserに対するTPの保証をReqに販売しているため、Devへの信頼が高くなるにつれ、ReqはDevにUserの正当性確認を行わなくなる可能性がある。

Devの可用性

Devがオフラインになった場合、新規Userがサービスを使用できなくなり、また、ReqはUserを確認できなくなるため、アプリのインセンティブが崩壊する。インセンティブ崩壊の可能性を低下させるための方法として、共同開発者や出資者のIPアドレス及び公開鍵もソフトウェアに内蔵するなどして予備のDevを確保することでDevの可用性が上がり、それに伴いアプリの可用性が向上するといった対策が

ある。DevはReqから対価を受け取れるため、予備のDevが存在しない場合に誰がDevになるかといった争いが起きることが予想される。争いの結果として、アプリに複数のDevが存在することとなる可能性がある。

Devによるバックドア設置

DevはNodeやUserのソフトウェアにバックドアの設置や意図しないコードを組み込む可能性がある。実装段階での対策として、それらのソフトウェアをDSLで記述することが挙げられる。そうすることで、DevからDSLの開発者へ信頼を(DSL開発者への信頼は必須であるが)移行することができる。

TPからのトラッキング

TPはUserの個人情報を知っているため、その個人情報とアプリ内IDを紐付ける可能性がある。

アプリ内ユーザーID発行フローで、UserはTPに対してアプリ内ユーザーIDとは別のIDを伝えている。そのため、TPによるトラッキングの可能性はTPとDevが共謀してトラッキングを行う場合を除いて排除される。TPとDevが共謀してトラッキングを行う場合、アプリ内ユーザーIDと個人情報は紐付けられる。

悪意を持つNode

以下ではNodeについて考えられる場合や攻撃について考える。

データの改竄など
NodeはUserから受け取ったデータを改竄することはできない。パブリックなデータには署名が必須であり、プライベートなデータには暗号化がかかっているためである。しかし、データを受け取ったにも関わらず、受け取っていないように振舞うことは可能である。そのためにUserはランダムに選択した複数のNodeを同様に使用する。

Devの開発したソフトウェアではない場合
NodeはDevが開発したソフトウェアをホストしていないのにもかかわらずDevが開発したサービスのように振る舞う可能性がある。Devのソフトウェアと同じ機能を提供していれば問題はない。ただしDevが開発したソフトウェアではない場合、ソフトウェア内部で何を行なっているかが不透明となる。

シビルアタック
上記ではUserがランダムに選択した複数のNodeを使用することでデータの損失は防がれると書いたが、そのNodeがすべて単一の管理元であった場合、簡単にNodeの悪意が通るため、データは損失する。この問題の対策として、Nodeが複数のTPにユーザ登録を行い、自身を保証してもらうことなどが挙げられるが、TPの権力が強まるという欠点が存在する。

悪意を持つReq

ReqはDevに対してUserの正当性の確認を行い、その正当性を他者に転売する可能性が考えられる。DevはTPから受け取った秘密鍵で自由に署名を作成することができるため、署名にReqの個人情報を含めることが可能である。転売の際には自身の個人情報も含めて転売することになるため、Reqによる転売は一定数防ぐことができる。この対策は、ReqがDevに個人情報を与え、Devは個人情報を検証する必要があるが、”保証の転売”には有効な対策であると考える。

ReqがNodeへの対価の支払いを怠った場合、Nodeはいつでも Reqとの関係を切ることができ、また Node間でP2Pネットワークが構築されている場合に、その評判を落とすことができる。ただし、Node が不当に評判を落とす可能性もあるため、評判を判断する際には、評判を落としているNodeへのある程度の信頼が必要である。

迷惑なUser

Userがすべきこととして Reqが要求したデータのNodeへの挿入があった。NodeはUserが提供する Req の要求に対する反応のデータによって利益が生まれるため、Userがそのデータの挿入を拒否した場合、そのUserに対して、Nodeは自身の利用を拒否する可能性が考えられる。

また、NodeとUserが手を組んで Reqからの要求反応を過剰に行う可能性がある。ReqはNodeから受け取ったUserのデータを Req自身で設計した判定アルゴリズムを使用するなどして解析することで、一定数は防ぐことができると考える。

課題

インセンティブ面

”鍵や宛先のすり替え”や”Devへの信頼性度合い”で記述したように、Devのインセンティブをシステムとして完全に保証することができていない。

セキュリティ面

“TPからのトラッキング”で記述したように、アプリの一部分に不透明性が残る。