Prototype Pollution入門

謝罪

サンプルWebアプリケーションを作ってくるつもりでしたが
そんな余裕ありませんでした
(でも最後に実演はあります)

JavaScriptのオブジェクト

- いわゆる辞書とかHashMapのような機能

- メンバの値として関数を持つ事が出来るのでクラスのような機能を有する
  - JSでもクラスが定義出来るようになっているが、実体はオブジェクト
  - 実はJSでは数値のようなプリミティブなもの以外は全部オブジェクト

- 他の言語との大きな違いとして「プロトタイプベース」という方式でインスタンスが作成されることが挙げられる

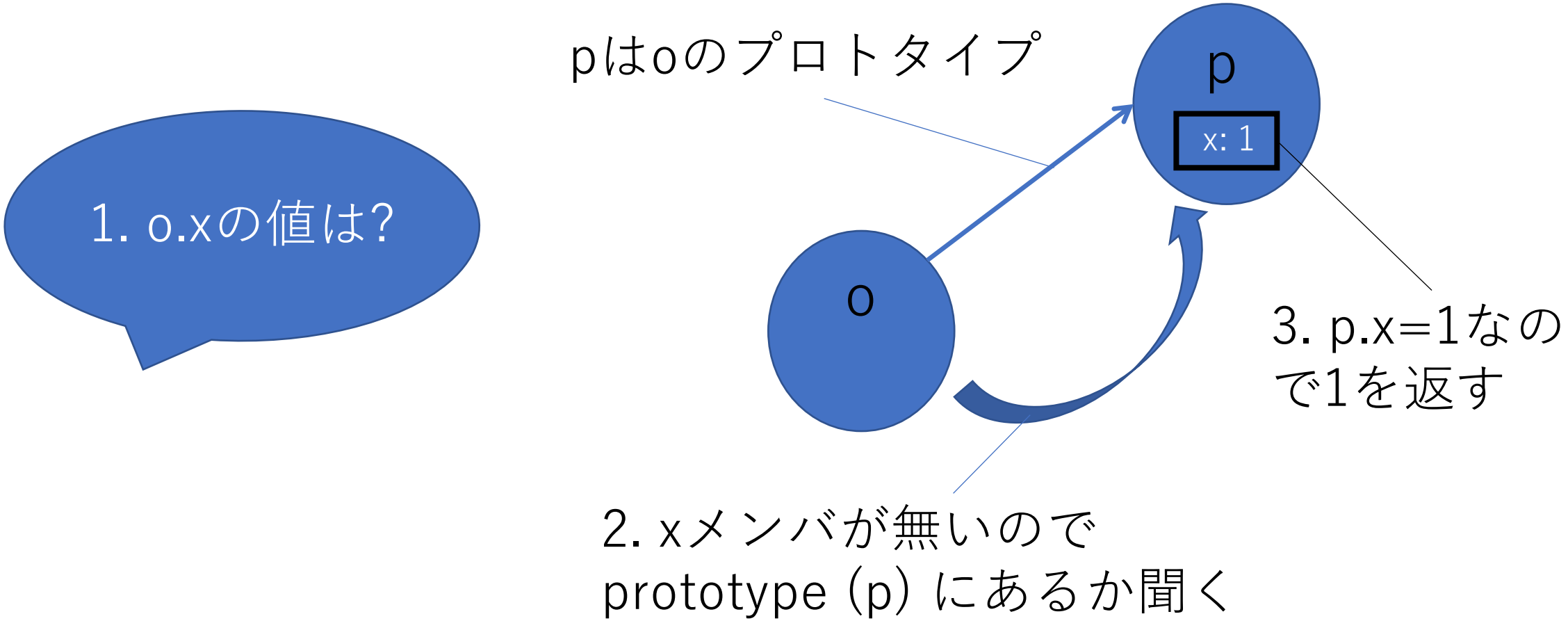
クラスベースとプロトタイプベース

- クラスベースの言語ではクラス定義を元にインスタンスを作成
- プロトタイプベースの言語(JS)では、既に存在するインスタンス(これをプロトタイプと呼ぶ)を元にインスタンスを作成
- どんなオブジェクトもプロトタイプを辿っていくと`Object.prototype`というオブジェクトへ辿り着く
- プロトタイプには`__proto__`というメンバでアクセス出来る
  - `x["__proto__"]`かx.__proto__でアクセス(前者が特に問題)

存在しないメンバへのアクセス

- `x`というメンバを持つオブジェクト`p`を用意
- `o = Object.create(p)`として新たに`o`を作る
- `o.x`は存在しないが`o.prototype.x`があればこれが使われる
  - 同様にプロトタイプを辿っていきどこかで存在したらそれが使われる
- よってプロトタイプチェーンの根にある`Object.prototype`に生えているメンバにはアクセスできる

プロトタイプを利用したメンバアクセス



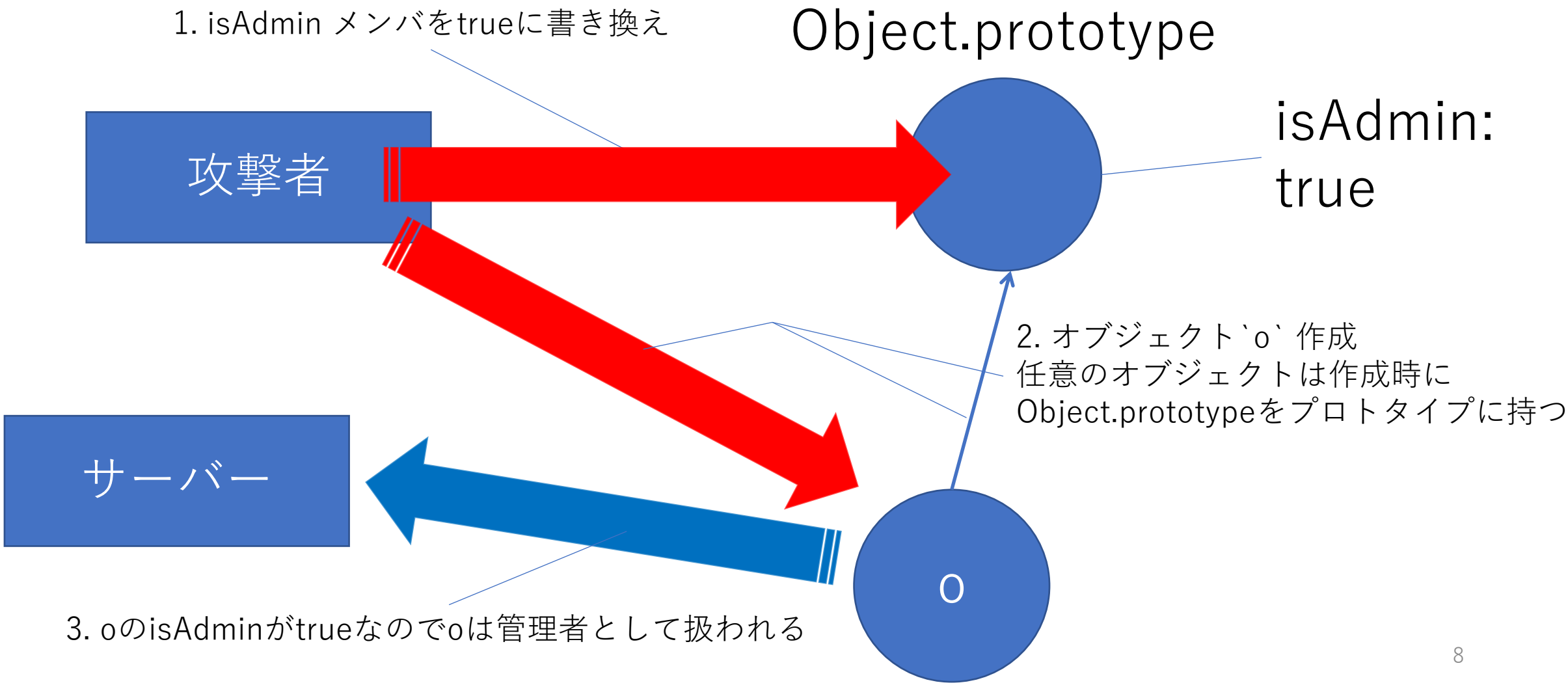
Prototype Pollution

- もし`Object.prototype`を書き換える事が出来たら何が起こる?
  → (これをプロトタイプとして持つ)任意のオブジェクトに対して存在しないメンバへの参照が出来てしまう

- アプリケーションでは初期化時など「そのメンバが存在しない」事を前提に進んでいることがある
  - 参照時に`undefined`だったら初期値を付与してあげる等
  - ヌル外し演算子: `??`や三項演算子: `cond ? T : F`がJSにある

- ここでの制御を奪う事が出来るのでは?

Object.prototypeのメンバ書き換えの影響



どうやって書き換えるか? (よくある例)

- `obj[key1][key2] = value`のようになっている時に`key1`と`key2`が制御出来れば次のようにすればよい
  - key1 = “__proto__”
  - key2 = お好きな文字列 (ここではmsgとする)

- これで`Object.prototype.msg`に`value`を入れる事が出来る
  - もしついでに`value`がある程度制御出来れば更に攻撃可能性が広がる

- 問題: こんなことが出来るのか? 出来たとしてそれからは?

Exploitに関する前提

- クライアントが送信する値は任意に指定出来る
  - HTMLのselect要素内のoption要素は「UI」のためでしかない
  - そもそもHTML(やブラウザ)を介さずともリクエストは送信可能
  - 送られてきた値はサーバー側でしっかりチェックしよう

- 自分のバグを踏み台にして他の強力なライブラリを利用される
  - 他のライブラリで初期値がundefinedな事を前提にしていると危険
  - (前回扱った)ROPも、バイナリ中のコード辺を利用された
  - バグが1つあるだけで(攻撃の)可能性は無限大

ネストするオブジェクトでの例

- `students`というオブジェクトが各学系`area`に対して学生番号をキー、名前を値とする配列を持つとする
  - `students = {IS: [...], KS: [...], MS: [...]}`

- `students[area][number] = name`という実装をする
  - `area`を選ばせて`number`を数値で入力し`name`を送信
  - HTMLでareaに入る部分を`"IS", "KS", "MS"`だけにしていても送信リクエストでは`"__proto__"`に変更可能
  - numberもHTMLでは数値としていても、文字列で送信可能
  - `students["__proto__"]["key"] = "polluted"`という処理が起こる
  - これは`students.__proto__.key = "polluted"`と等価

汚染箇所を発見してからの例

- node.jsで動いているサーバーなら、コマンドを実行する関数を呼んでいる他の実装やライブラリを狙える
  - もしそこでデフォルトのコマンドやオプションを最初に設定する、といった処理があれば事前にそこに仕込める可能性がある

- テンプレートエンジンのキャッシュを正常な登録をする前に好きな値で書き換えてsecretな変数の窃取
  - 複雑な構文解析を要するテンプレートエンジンならそのASTが構築される前に書き換えることでRCEに持っていく事も可能

CTFでの出題例

- DownUnderCTF 2021 – zap
  - `zip`コマンドの制限されたオプションに引数を渡せる実装
  - Prototype Pollutionで制約を突破し、オプションに`-T –TT`を仕込む
  - このオプションの引数はコマンドとして実行されるのでRCEになる

- CakeCTF 2022 – Panda Memo
  - `console.table()`におけるPrototype Pollutionを利用して非想定の配列の要素を作り出し、要素の有無で権限を確認してるザル認証を突破
  - 認証後にもPrototype Pollutionが出来るのでテンプレートエンジンのキャッシュに非公開情報の変数を開示するテンプレートを仕込む

参考文献

- Object のプロトタイプ - ウェブ開発を学ぶ | MDN
- プロトタイプベース | TypeScript入門『サバイバルTypeScript』
- Prototype Pollution - Satoooon1024 CTF Wiki
  - この資料作成中に大幅加筆されてこの資料の意味が「紹介」だけに…
- AST Injection, Prototype Pollution to RCE

実演1

愚直にプロトタイプを書き換えてみる
(面白くないので時間無かったら飛ばします)

実演2

CVE-2022-21824: `console.table()`におけるPrototype Pollution

宣伝 (時間があったら)

SECCON CTF 2022

SECCON CTF 2022

- JNSA主催の情報セキュリティイベントに伴って開催されるCTF
  - 2022年11月12日(土)~13日(日)の24時間に渡ってオンライン開催
  - 成績上位チームは来年2月に開催されるオンサイト本戦に出場可能

- 歴史が長く、使用インフラや開催委員会の規模は国内最大級

- 昨年に引き続き、問題作成メンバーとしてCTF運営に参加しているので興味のある方は是非参加してください
  - 他の問題作成メンバーは国内現役上位チームのメンバー