どこまで守れば合格か？

リアルな脆弱性から見るWeb開発の境界線

自己紹介

- 大学生
- セキュリティ

話すこと

- どこまでセキュリティを考えるべきかの基準
- vrc-ta-hub.comにおける2件の脆弱性の提出
- 自分の管理しているサイトの脆弱性の見つけ方

責任ある開示（Responsible Disclosure）

- **このLTで話す脆弱性について**

今回紹介する脆弱性は、
すべて **開発者への報告・修正確認を経たうえで** 公開しています

責任ある開示（Responsible Disclosure）2

- 脆弱性を見つけたとき、いきなり公開するのではなく、まず報告し、修正の時間を確保してから公開する

- 今回の場合

診断→発見→報告→修正の確認→許可→公開

複数の理由で直接報告がためらわれる場合、IPAの情報セキュリティ早期警戒 パートナーシップガイドラインに基づいてIPAに報告しよう

責任ある開示（Responsible Disclosure）3

なぜ？：

未修正の脆弱性を他の人に教えると、ゼロデイ攻撃の可能性が高まる

個人開発者は企業と違ってセキュリティチームがいない → 修正に時間がかかることもある

ここでいう公開は報告元と報告先以外のすべてを指すよ！
友達とかグループチャットも
アウト！

責任ある開示（Responsible Disclosure）4

個人開発者へのお願い

連絡先やセキュリティポリシーをどこかに明記しておくと、
報告者が連絡しやすい
例：READMEに security.txt やメールアドレスを記載

ターゲットの紹介

- VRChat 技術・学術系イベント管理サービス
- VRChatで開催される技術・学術系イベントの情報を集約し、コミュニティの発展を支援するプラットフォーム

ターゲットの紹介

- 最近、大型アップデートがあった

LT申請機能やスタッフの招待、集会の切り替え……

機能が増えると脆弱性も増えるな

Privileges Requiredが下がるな

お世話になったからハックしたろ

vrc-ta-hubの変遷と脅威モデリング

- フェーズ1:招待制

ユーザー:信頼できる人
主な機能:イベントの閲覧、検索が中心
攻撃面:限定的。ユーザーが入力するポイントが限られている
→バリデーションやアクセス制御が甘くてもなんとかなる

権限無しでの脆弱性
権限のある人が悪意のない使い方をしたときに発生する脆弱性

vrc-ta-hubの変遷と脅威モデリング

- フェーズ2:一般公開

スタッフを追加する機能やアカウントを作る機能が追加された
ユーザー:不特定多数(悪意のあるユーザーを含む)
機能:LT申請、スタッフの追加など複雑な処理が追加
攻撃面:増加

イベントを操作できるアカウントでの悪意のある操作が
現実的な脅威に

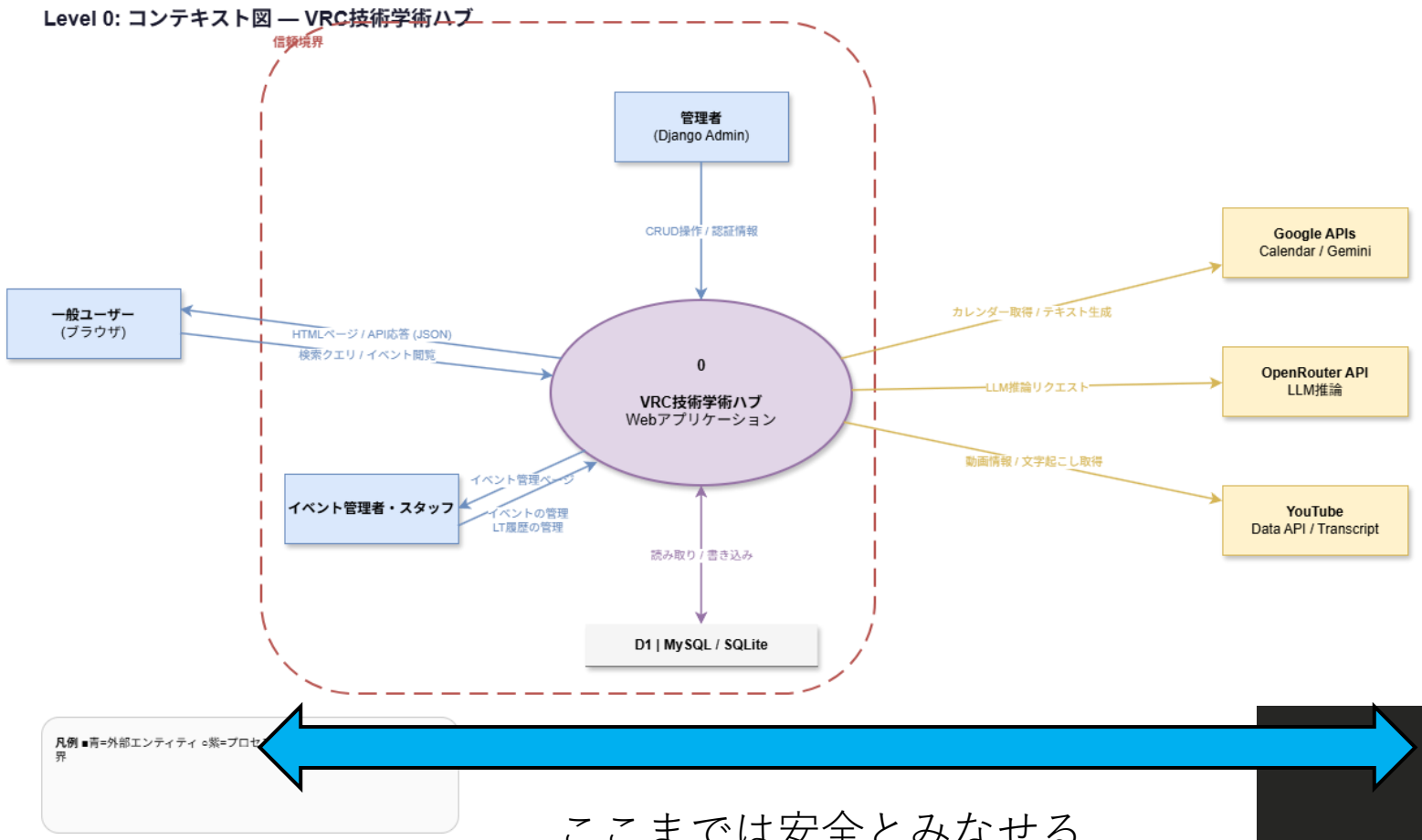
vrc-ta-hubの変遷と脅威モデリング

- フェーズ2:一般公開

スタッフを追加する機能やアカウントを作る機能が追加された
ユーザー:不特定多数(悪意のあるユーザーを含む)
機能:LT申請、スタッフの追加など複雑な処理が追加
攻撃面:増加

イベントを操作できるアカウントでの悪意のある操作が
現実的な脅威に

vrc-ta-hubの変遷と脅威モデリング



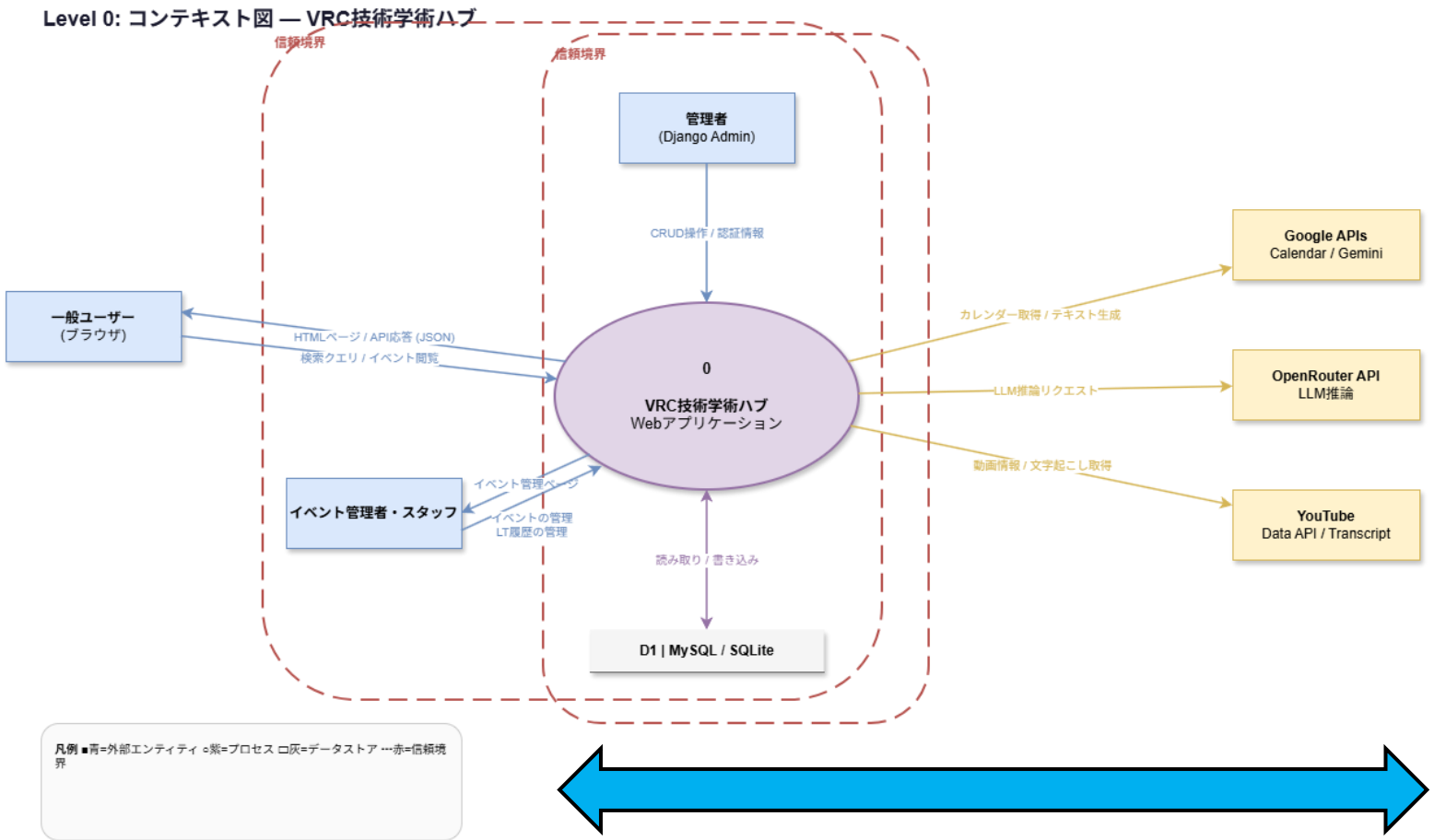
ここまでは安全とみなせる

ドローイオとかつかってきれいにDFDかける？機能中心で

Check frontend-design skill for any relevant guidance >

draw.ioのXML形式でDFDを作成します。機能中心で、データフローを明確にしたものを描きます。

vrc-ta-hubの変遷と脅威モデリング



安全とみなせるラインが後退

vrc-ta-hubの変遷と脅威モデリング

- 誰が、何を入力できるのか棚卸するのは大事
- githubをAIに投げれば図示してくれる(頭の中にあるはずなので、ハルシネーション対策はおそらく不要)
- 今回の場合だと、ユーザーが外側に置かれ、機能が増えたことがすぐにわかる、重点的に調べることができるようになった！
- 多層防御の観点から信頼境界の中は無対策でいいとは言えないが、現実、優先順位やリソースの振り分けの参考にすることができる

事例1:格納型XSS

vrc-ta-hubにおけるファイルアップロード制限の回避とiframeインジェクションの連鎖による格納型XSS

対象:https://vrc-ta-hub.com/

/event/detail/*のエンドポイントはstored xssに脆弱です。
vrc-ta-hub アプリケーションに格納型クロスサイトスクリプティング（XSS）の脆弱性が存在します。「ファイルアップロードのバリデーション回避（Content-Typeの偽装）」と「不適切なHTMLサニタイズ設定（iframeの許可）」を組み合わせることで、認証済みの攻撃者が、PDFを装った悪意のあるHTMLファイルをアップロードし、イベントページに埋め込むことが可能になります。これにより、他のユーザーのセッションコンテキストで任意のJavaScriptが実行され、アカウントの乗っ取り（ATO）、セッションハイジャック、データの流出などの被害が発生する恐れがあります。

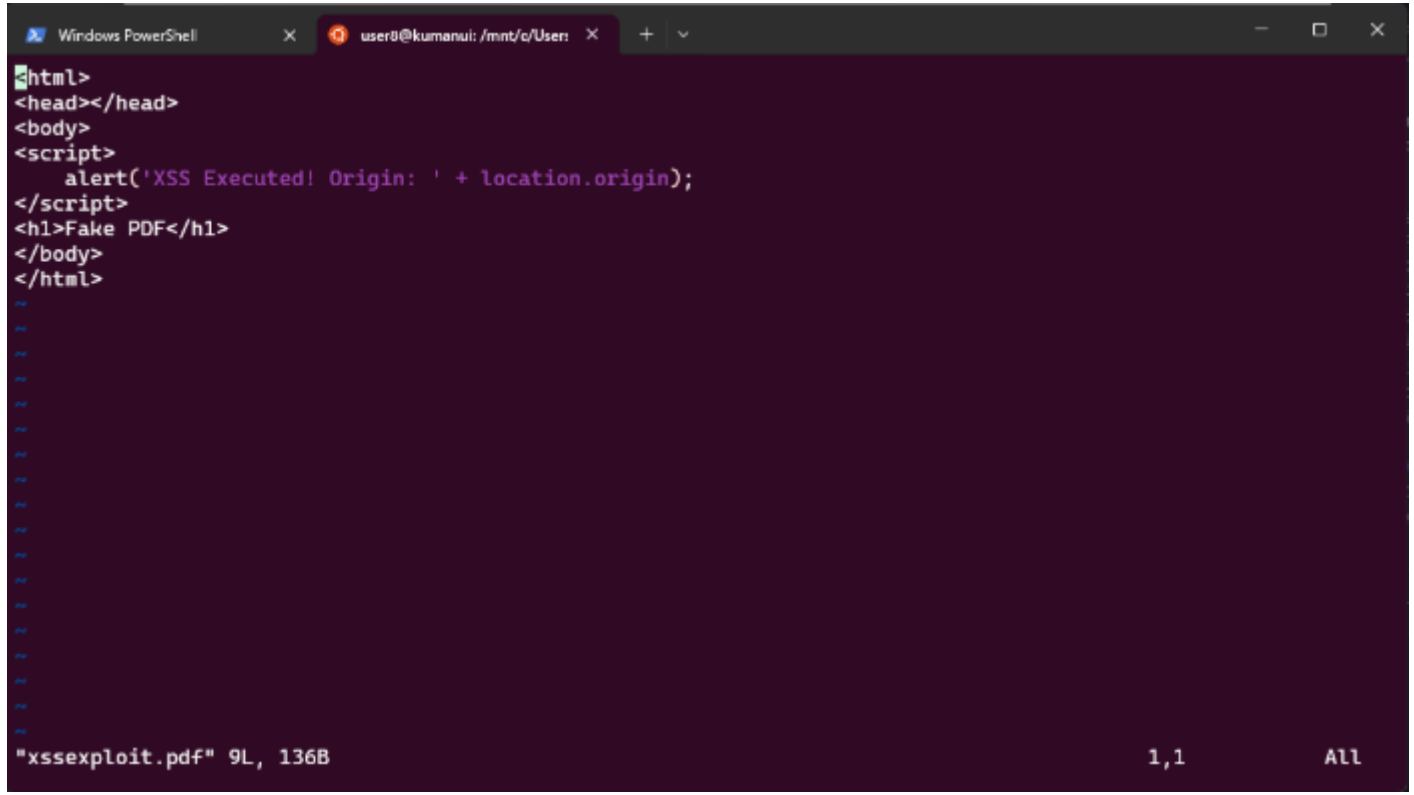
・脆弱性の種類：格納型クロスサイトスクリプティング（XSS）
・CWE ID：CWE-79（XSS）、CWE-434（危険なタイプのファイルの無制限なアップロード）
・CVSS v3.1 スコア：8.7（重要/High）
・攻撃ベクトル：CVSS:3.1/AV:N/AC:L/PR:L/UI:R/S:C/C:H/I:H/A:N

・影響を受けるコンポーネント：

- app/event/libs.py：convert_markdown 関数（bleachの不適切な設定）
- app/event/models.py：validate_pdf_file 関数（拡張子のみの脆弱な検証）
- app/event/forms.py：EventDetailForm

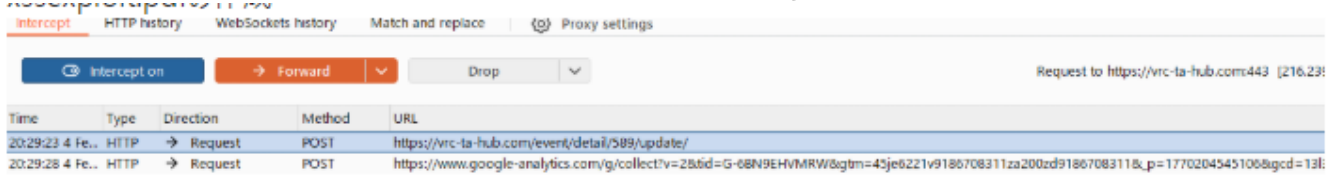
事例1:格納型XSS

- xssexploit.pdf



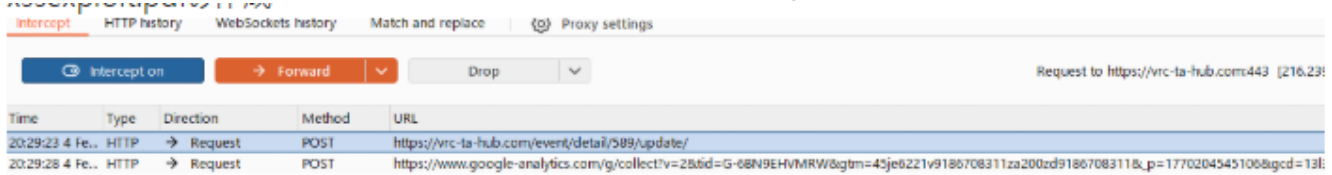
事例1:格納型XSS

- mimetypeをapplication/pdfからtext/htmlに変更することでiframeで読み込んだ時にhtmlとして読み込まれる。



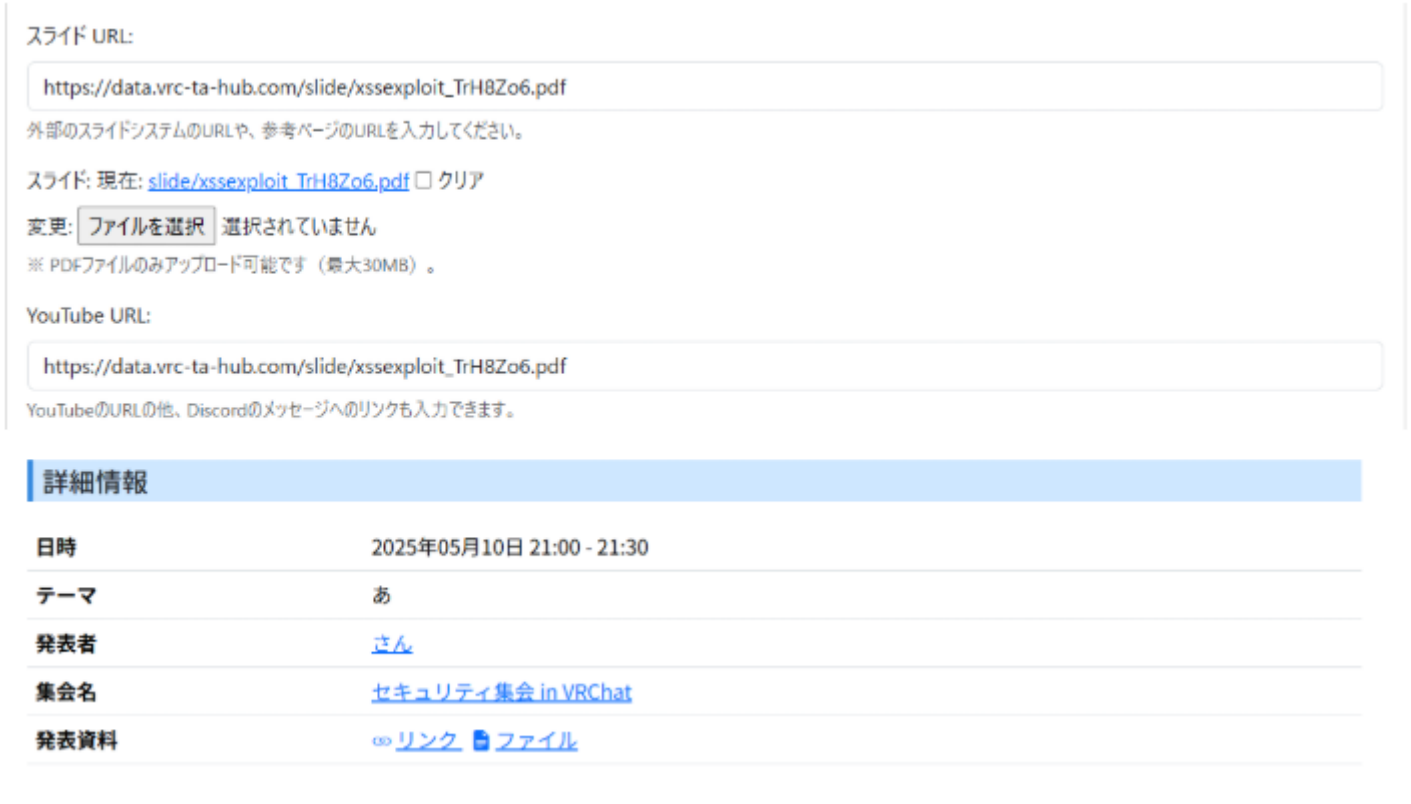
事例1:格納型XSS

- mimetypeをapplication/pdfからtext/htmlに変更することでiframeで読み込んだ時にhtmlとして読み込まれる。



事例1:格納型XSS

- 発表資料をクリックさせる



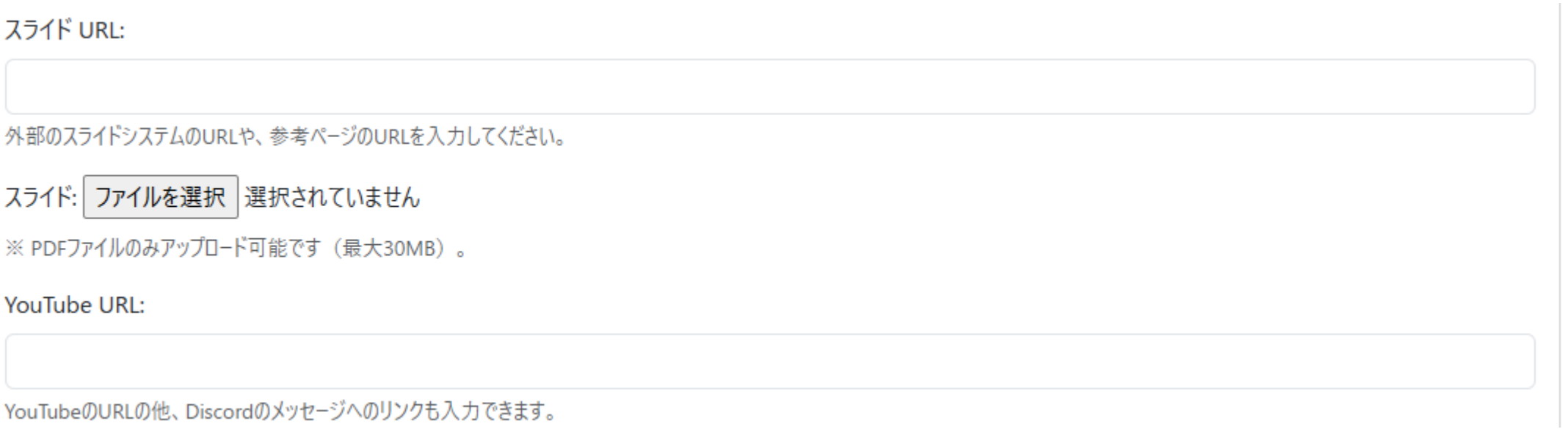
事例1:格納型XSS

- 発火する

事例1:格納型XSS

- ところで……

スライドとYoutubeを挿入できるらしい



事例1:格納型XSS

- ところで……

スライドとYoutubeを挿入できるらしい
→XSSのページを挿入すれば？

事例1:格納型XSS

- LTのページで発火！より引っかかる人の増加が期待できる

事例1:格納型XSS

- LTのページで発火！より引っかかる人の増加が期待できる

事例2:データ不適切公開

- XSSのような派手な脆弱性ではない

LT申請却下後におけるデータの不適切な公開

1. 脆弱性の概要

VRC技術学術系Hubのイベント管理システムにおいて、運営者が却下したはずのLT（ライトニングトーク）申請内容が、依然としてイベント公開ページのスケジュール一覧に表示され続ける不備を確認しました。これにより、本来非公開であるべき情報が第三者に閲覧される状態になっています。

2. 脆弱性の詳細情報

- **脆弱性の種類**: 不適切なアクセスコントロール / 情報漏えい
- **CWE ID**: CWE-213（不適切なポリシーによる機密情報の露出）、CWE-284（不適切なアクセス制御）
- **CVSS v3.1 スコア**: 4.3（中 / Medium）
- **攻撃ベクトル**: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N

3. 影響を受けるコンポーネント

- フロントエンドのスケジュール表示コンポーネント（イベント一覧ページ）
- バックエンドビュー: `app.event.views.EventListView`

事例2:データ不適切公開

入力時

```python
# VRChatterの生活リズムに合わせて朝4時を日付の境界とする
today = get_vrchat_today()
queryset = queryset.filter(date__gte=today).select_related('community').prefetch_related(
    'details').order_by('date', 'start_time')
```

status='approved’
みたいなフィルタリングの
付け忘れ

表示時

```
{% for detail in event.details.all %}
    <div class="mb-2">
        <span class="text-warning">★</span>
        <a href="...">{{ detail.theme }}</a><br>
        <small class="text-success">by {{ detail.speaker }}さん</small>
    </div>
{% endfor %}
```

事例2:データ不適切公開

- 却下・待機された状態のLTが表示されてしまう

AIは検知できたの？

- claude Opus 4.6を使用

- stored XSS→部分的に検知
- (pdfの偽装アップロード、iframeにおける任意ファイル読み込みは検知できたが、それらを統合してより脅威度を上げることができなかった)
- 情報漏洩→今回は検知できなかった
- 動的テスト中に発見したものを報告

AIは検知できたの？

- claude Opus 4.6を使用

- stored XSS→部分的に検知

(pdfの偽装アップロード、iframeにおける任意ファイル読み込みは検知できたが、それらを統合してより脅威度を上げることができなかった)

- 情報漏洩→今回は検知できなかった

動的テスト中に発見したものを報告

どのAIがいい？

AIモデル	どのくらい検知できるか
GPT	
Gemini	
Claude	

どのAIがいい？

AIモデル	どのくらい検知できるか
GPT	GPTができる分だけ
Gemini	Geminiができる分だけ
Claude	Claudeができる分だけ

どのAIがいい？

AIモデル	どのくらい検知できるか
GPT	GPTができる分だけ
Gemini	Geminiができる分だけ
Claude	Claudeができる分だけ

個人的にはコーディング能力と一致。その時々で優秀なのを使えばいいと思う

GPT系だけはセキュリティに関するポリシーが厳しくなる噂があるので注意

聞いたことある？

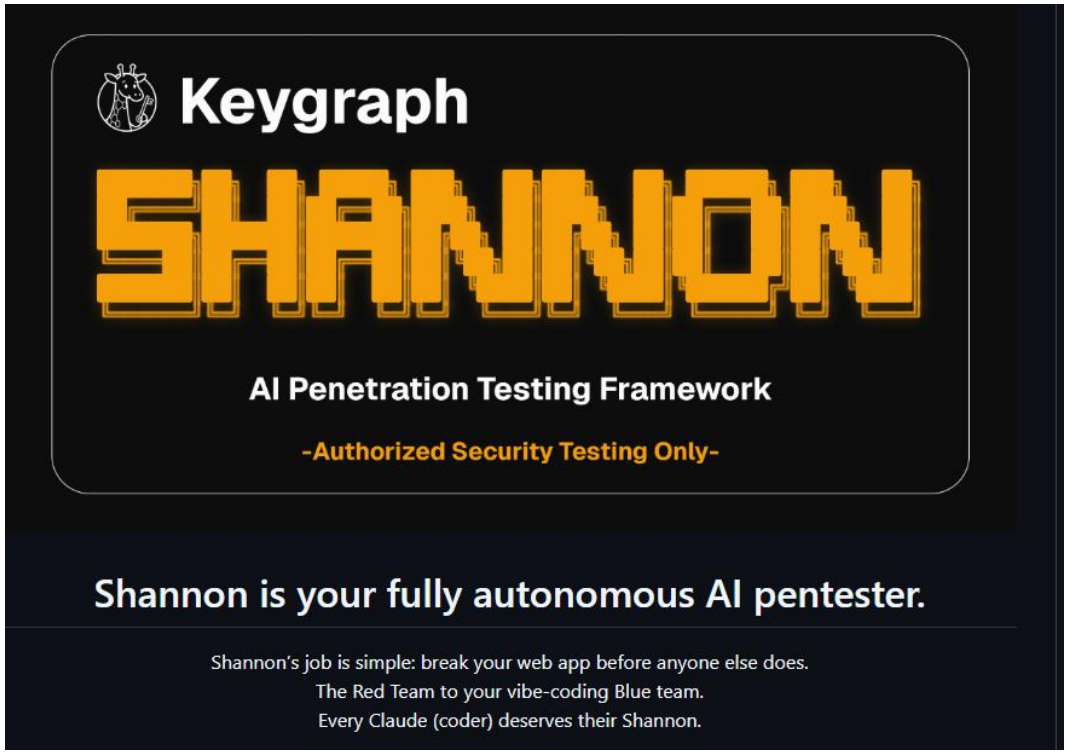
- 絶対に堅牢なシステムはない
- かけた時間やモデルの性能がものをいうと思う
- 公開前に「脆弱性ある？」と聞くだけでもかなりましになる

開発をする際
AIに「セキュリティに配慮していますか」
と尋ねることで
開発者の意識も変わっていく

AIツールの今後

- 脆弱性診断に特化したAIツールも登場してきた

一回50とか60$かかるけど
優秀だよ



AIツールの今後

- 静的解析のAIツールは今後も多く登場するであろう

「Claude Code Security」が登場、コード内の脆弱性をスキャンして修正提案もしてくれる