# **API9:2019 مدیریت نادرست دارایی‌ها**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | |  | |  | |
| **خاص API** | **قابلیت بهره‌برداری: 3** | **میزان شیوع: 3** | **قابلیت تشخیص: 2** | **پیامد فنی: 2** | **خاص کسب‌و‌کار** |
| نسخه‌های قدیمی API غالبا اصلاح و بروزرسانی نشده‌اند و از آنجا که از مکانیزم‌های دفاعی نوین موجود در APIهای جدید بهره نمی‌برند، راهی آسان برای دسترسی به سیستم‌ها برای مهاجمین فراهم می‌سازند. | | مستندات قدیمی و بروزرسانی نشده، امکان یافتن و یا رفع آسیب‌پذیری‌ها را دشوار می‌سازند. همچنین نبود فهرستی از دارایی‌ها و فقدان یک استراتژی مدون برای از دور خارج کردن نسخه‌های قدیمی منجر به وجود سیستم‌های وصله یا تعمیر نشده و نهایتا نشت اطلاعات خواهد شد. امروزه با کمک مفاهیم نوینی نظیر مایکروسرویس‌ها که امکان بکارگیری اپلیکیشن‌ها بصورت مستقل را تسهیل نموده‌اند (نظیر رایانش ابری، k8s یا کوبرنیتس و ...)، یافتن APIهایی که به صورت غیرضروری در معرض دید همگان قرار دارند تبدیل به امری رایج و آسان شده است. | | مهاجم می‌تواند از طریق نسخه‌های قدیمی API که کماکان به پایگاه داده‌ی اصلی متصل هستند، به داده‌ی حساس و یا حتی سرور دسترسی یابد. | |

**آیا API از نظر مدیریت نادرست دارایی‌ها[[1]](#footnote-1) ‌آسیب‌پذیر است؟**

در صورتی که یکی ازشرایط زیر وجود داشته باشد، API ‌آسیب‌پذیر خواهد بود:

* هدف از وجود API نامشخص بوده و پاسخی برای سوال‌های زیر وجود نداشته باشد:
  + API در چه محیطی در حال اجرا است (مثلا محیط تست، توسعه، اجرا[[2]](#footnote-2) یا عملیات[[3]](#footnote-3))؟
  + چه کسانی بایستی دسترسی شبکه‌ای به API داشته باشند (همه، افراد دخیل یا شرکا)؟
  + چه نسخه‌ای از API در حال اجرا است؟
  + چه داده‌ای (نظیر PII) توسط API در حال جمع آوری و پردازش است؟
  + جریان داده به چه صورت است؟
* مستندی برای API وجود ندارد یا بروز نیست.
* برنام‌ ای برای بازنشستگی و از دور خارج شدن هریک از نسخه‌های API وجود ندارد.
* فهرست میزبان‌ها[[4]](#footnote-4) وجود ندارد یا قدیمی است.
* فهرست سرویس‌های یکپارچه[[5]](#footnote-5)، چه سرویس‌های متعلق به خود سازمان و چه سرویس‌های شرکت‌های ثالث، وجود ندارد یا قدیمی است.
* نسخه‌های قدیمی یا پیشین API بدون اصلاح و وصله شدن[[6]](#footnote-6) کماکان در حال اجرا هستند.

**API9:2019 مدیریت نادرست دارایی‌ها**

**مثال‌هایی از سناریوهای حمله**

**سناریو #1**

پس از بازطراحی یک اپلیکیشن، یک سرویس جستجوی Local وجود دارد که از یک نسخه قدیمی API (api.someservice.com/v1) به صورت محافظت نشده بهره می‌برد که در عین حال این API قدیمی به پایگاه داده کاربران دسترسی دارد. مهاجم که جدیدترین نسخه اپلیکیشن را به عنوان هدف درنظر گرفته، آدرس API (api.someservice.com/v2) را می‌یابد. جایگزینی v2 با v1 در URL سبب دسترسی مهاجم به API محافظت نشده و قدیمی می‌شود که در نتیجه‌ی آن، اطلاعات شناسایی شخصی (PII) بیش از 100 میلیون کاربر افشا گردیده است.

**سناریو #2**

یک شبکه اجتماعی از مکانیزم محدودسازی نرخ ارسال درخواست[[7]](#footnote-7) برای جلوگیری از انجام حملات Brute Force توسط مهاجمین جهت حدس توکن‌های تغییر گذرواژه بهره می‌برد. این مکانیزم نه به عنوان بخشی از کد API، بلکه به عنوان مولفه ای مابین کلاینت و API اصلی (در www.socialnetwork.com) ‌پیاده‌سازی شده است. مهاجم یک نسخه بتا از میزبان API (www.mbasic.beta.socialnetwork.com) می‌یابد که از API یکسانی بهره می‌برد و رویه تغییر گذرواژه یکسانی دارد با این تفاوت که در آن هیچ مکانیزمی جهت محدودسازی نرخ درخواست تعبیه نشده است؛ در نتیحه مهاجم قادر خواهد بود که گذرواژه هر یک از کاربران را طی یک عملیات Brute Force ساده با حدس زدن یک توکن 6 رقمی تغییر دهد.

**چگونه از ‌آسیب‌پذیری مدیریت نادرست دارایی‌ها پیشگیری کنیم؟**

* فهرستی از تمامی میزبان‌های API تهیه شده و جنبه‌های مهم هرکدام با تمرکز بر محیط API (محیط تست، توسعه، اجرا یا عملیات)، افراد مجاز به دسترسی شبکه‌ای به میزبان (همه، افراد دخیل یا شرکا) و نسخه API مستند شود.
* فهرستی از سرویس‌های یکپارچه تهیه شده و جنبه‌های مهم این سرویس‌ها نظیر نقش آنها، داده‌ی مبادله شده (جریان داده) و میزان حساسیت آنها مستند شود.
* تمامی جنبه‌های API نظیر نحوه احراز هویت، خطاها، ریدایرکت‌ها، محدودسازی نرخ درخواست، خط مشی‌های اشتراک گذاری منابع متقابل (CORS) و نقاط پایانی یا توابع (Endpointها) شامل پارامترها، درخواست‌ها و پاسخ‌ها مستند شوند.
* با بکارگیری و انطباق با استانداردهای باز، فرایند تولید مستند بطور خودکار انجام شده و این فرایند در CI/CD Pipeline تعبیه گردد.

**API9:2019 مدیریت نادرست دارایی‌ها**

* مستندات API در اختیار افرادی که مجاز به دسترسی به API هستند قرار گیرد.
* از مکانیزم‌های محافظتی خارجی از جمله فایروال‌های امنیت API برای محافظت از تمامی نسخه‌های در معرض دید API (نه فقط نسخه فعلی) استفاده گردد.
* از استفاده همزمان نسخه‌های عملیاتی شده[[8]](#footnote-8) و عملیاتی نشده[[9]](#footnote-9) API اجتناب شود. اگر این همزمانی اجتناب ناپذیر است، برای نسخه‌های عملیاتی نشده API نیز باید همان حفاظت‌های امنیتی نسخه‌های عملیاتی شده برقرار باشد.
* هنگامی که در نسخه‌های جدیدتر API بهبودهای امنیتی اعمال می‌شود، بایستی فرایند تحلیل ریسک نیز صورت پذیرد تا بتوان تصمیمات لازم در خصوص اقدامات جبرانی برای رفع مشکلات امنیتی نسخه‌های قدیمی‌تر را اتخاذ نمود. بعنوان نمونه، آیا می‌توان بدون تحت‌الشعاع قراردادن انطباق‌پذیری[[10]](#footnote-10) API بهبودهای امنیتی را در نسخه‌های قدیمی نیز وارد نمود یا اینکه بایستی تمامی نسخه‌های قدیمی به سرعت از دسترس خارج شده و تمامی کلاینت‌های مجبور به استفاده از آخرین نسخه شوند؟

**مراجع**

**خارجی**

* [CWE-1059: Incomplete Documentation](https://cwe.mitre.org/data/definitions/1059.html)
* [OpenAPI Initiative](https://www.openapis.org/)

1. Improper Asset Management [↑](#footnote-ref-1)
2. Stage [↑](#footnote-ref-2)
3. Production [↑](#footnote-ref-3)
4. Host Inventory [↑](#footnote-ref-4)
5. Integrated Service Inventory [↑](#footnote-ref-5)
6. Patch [↑](#footnote-ref-6)
7. Rate-Limiting [↑](#footnote-ref-7)
8. Production [↑](#footnote-ref-8)
9. Non-Production [↑](#footnote-ref-9)
10. Compatibility [↑](#footnote-ref-10)