# **API1:2019 مجوزدهی نادرست در سطح اشیاء**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | |  | |  | |
| **خاص API** | **قابلیت بهره‌برداری: 3** | **میزان شیوع: 3** | **قابلیت تشخیص: 2** | **پیامد فنی: 3** | **خاص کسب‌و‌کار** |
| مهاجمین می‌توانند از نقاط و توابع ‌آسیب‌پذیر (از منظر مجوزدهی نادرست در سطح اشیا) با دستکاری شناسه شیء[[1]](#footnote-1) ارسالی درون درخواست سوءاستفاده و بهره برداری نمایند. این امر می‌تواند منجر به دسترسی غیرمجاز به داده حساس شود. دسترسی غیرمجاز به داده حساس، مساله‌ای رایج در اپلیکیشن‌های مبتنی بر API است چرا که مولفه سرور غالبا به طور کامل وضعیت کلاینت را رهگیری نمی‌کند و در عوض برای تصمیم گیری درباره دسترسی کلاینت به اشیاء از پارامترهایی نظیر شناسه شی که از سوی خود کلاینت ارسال می‌شوند، تکیه دارند. | | این حمله رایج ترین ‌آسیب‌پذیری APIها بوده و بیشترین پیامدها را نیز در پی دارد. مکانیزم‌های مجوزدهی و کنترل دسترسی در اپلیکیشن‌های مدرن، پیچیده و گسترده هستند. حتی اگر اپلیکیشن زیرساخت مناسب را برای کنترل‌های مجوزدهی ‌‌‌‌پیاده‌سازی نماید، ممکن است توسعه دهندگان پیش از دسترسی به اشیا حساس، استفاده از این کنترل‌ها را فراموش نمایند. تشخیص نقایص مربوط به کنترل دسترسی از طریق تست‌های ایستا یا پویا به صورت خودکار غالبا امکان پذیر نیست. | | دسترسی غیرمجاز می‌تواند منجر به افشای اطلاعات به طرف‌های غیرمجاز، از دست رفتن داده یا دستکاری آن شود. همچنین دسترسی غیرمجاز به اشیا می‌تواند سبب تحت کنترل گرفتن کامل حساب کاربری توسط مهاجم گردد. | |

**آیا API از نظر** **مجوزدهی نادرست در سطح اشیاء[[2]](#footnote-2) آسیب‌پذیر است؟**

مجوزدهی در سطح اشیا مکانیزمی برای کنترل دسترسی است که غالبا در سطح کد ‌‌‌‌پیاده‌سازی شده و دسترسی کاربر به اشیایی که بایستی به آنها دسترسی داشته باشد را تضمین می‌نماید.

هر تابعی در API که یک شناسه شی دریافت نموده و نوعی عملیات بر روی آن شی انجام می‌دهد، بایستی کنترل‌های مجوزدهی در سطح اشیا را بکار گیرد. این کنترل‌ها باید دسترسی کاربرِ واردشده[[3]](#footnote-3) به انجام عمل درخواستی بر روی شی درخواستی را اعتبارسنجی نمایند.

وجود ایراد و نقصان در این مکانیزم منجر به افشای اطلاعات غیرمجاز، تغییر یا از بین رفتن تمامی داده خواهد شد.

**API1:2019 مجوزدهی نادرست در سطح اشیاء**

**مثال‌هایی از سناریوهای حمله**

**سناریو #1**

یک پلتفرم تجارت الکترونیک، برای فروشگاه‌های آنلاین نمودارهای سود فروشگاه‌های میزبانی شده را در قالب یک لیست چندصفحه‌ای ارائه می‌دهد. مهاجم با بررسی درخواست‌های مرورگر، توابعی از API که نقش منبع داده برای نمودارهای مذبور را دارند و الگوی آنها به صورت /shops/{shopName}/revenue\_data.json می‌باشد را شناسایی می‌کند. با استفاده از یک تابع دیگر API، مهاجم می‌تواند لیست نام کلیه فروشگاه‌های میزبانی شده را استخراج نماید. همچنین مهاجم با استفاده از یک اسکریپت ساده و جایگزین کردن {shopName} در URL خواهد توانست به داده‌ی فروش هزاران فروشگاه دسترسی یابد.

**سناریو #2**

با پایش ترافیک شبکه‌ی یک گجت پوشیدنی[[4]](#footnote-4) درخواست HTTP PATCH زیر توجه مهاجم را به وجود سرآیند HTTP سفارشی X-User-Id: 54796 جلب می‌نماید. با جایگزین کردن مقدار X-User-Id با 54795، مهاجم پاسخ HTTP موفقیت آمیز گرفته و قادر به تغییر اطلاعات حساب سایر کاربران خواهد بود.

**چگونه از آسیب‌پذیری مجوزدهی نادرست در سطح اشیاء پیشگیری کنیم؟**

* بکارگیری یک مکانیزم مجوزدهی که بر خط مشی و سلسله مراتب کاربری تمرکز دارد.
* استفاده از یک مکانیزم مجوزدهی برای بررسی اینکه آیا کاربر واردشده مجوز لازم برای انجام عملیات درخواستی بر روی رکورد در تمامی توابعی که از کلاینت، ورودی می‌گیرند تا به رکورد مذبور در پایگاه داده دسترسی داشته باشند را دارا است یا خیر؟
* ارجحیت استفاده از مقادیر تصادفی و غیرقابل پیش بینی بعنوان GUID[[5]](#footnote-5) برای شناسه رکوردها.
* طراحی آزمونهایی برای ارزیابی صحت عملکرد مکانیزم‌های مجوزدهی.

**مراجع**

**خارجی**

* [CWE-284: Improper Access Control](https://cwe.mitre.org/data/definitions/284.html)
* [CWE-285: Improper Authorization](https://cwe.mitre.org/data/definitions/285.html)
* [CWE-639: Authorization Bypass Through User-Controlled Key](https://cwe.mitre.org/data/definitions/639.html)

1. Object ID [↑](#footnote-ref-1)
2. Broken Object Level Authorization [↑](#footnote-ref-2)
3. Logged-in User [↑](#footnote-ref-3)
4. Wearable Device [↑](#footnote-ref-4)
5. Globally Unique Identifier [↑](#footnote-ref-5)