# **API6:2019 تخصیص جمعی**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | |  | |  | |
| **خاص API** | **قابلیت بهره‌برداری: 2** | **میزان شیوع: 2** | **قابلیت تشخیص: 2** | **پیامد فنی: 2** | **خاص کسب‌و‌کار** |
| بهره برداری از این آسیب‌پذیری غالبا نیاز به فهم منطق تجاری، روابط مابین اشیا و ساختار API از سوی مهاجم دارد. بهره برداری از مقوله تخصیص جمعی در APIها ساده تر است چرا که در مرحله طراحی، پیاده‌سازی زیرین اپلیکیشن به همراه نام ویژگی‌های اشیا افشا می‌شود و در معرض دید عموم قرار می‌گیرد. | | چارچوبهای جدید غالبا توسعه دهندگان را به استفاده از توابعی تشویق می‌کنند که بطور خودکار، ورودی‌های دریافتی از کلاینت را به متغیرهای کد و اشیاء داخلی آن پیوند می‌دهند. مهاجمین با سواستفاده از این متدلوژی می‌توانند به گونه ای اقدام به بروزرسانی یا بازنویسی ویژگی‌های اشیاء (داده) حساس نمایند که توسعه دهنده هیچگاه قصد افشای آن ویژگی‌ها را نداشته است. | | بهره برداری از این ‌‌‌آسیب‌پذیری می‌تواند منجر به افزایش سطح دسترسی، دستکاری داده، عبور از مکانیزم‌های امنیتی و ... شود. | |

**آیا API از نظر تخصیص جمعی[[1]](#footnote-1) ‌‌‌آسیب‌پذیر است؟**

اشیا در اپلیکیشن‌‌‌های مدرن می‌توانند ویژگی‌‌‌های[[2]](#footnote-2) متعددی داشته باشند. برخی از این ویژگی‌‌‌ها بایستی مستقیما توسط کلاینت قابل بروزرسانی باشند (مثلا user.first\_name یا user.address) در حالی که کلاینت نباید بتواند سایر ویژگی‌‌‌ها را دستکاری نماید (مثلا پرچم user.is\_vip).

یک تابع درAPI اگر بطور خودکار پارامترهای کلاینت را بدون لحاظ کردن حساسیت و سطح افشای[[3]](#footnote-3) ویژگی‌‌‌های آن، مستقیما تبدیل به ویژگی‌‌‌های اشیای داخلی نماید، از منظر تخصیص جمعی ‌‌‌آسیب‌پذیر خواهد بود. این ‌‌‌آسیب‌پذیری به مهاجم اجازه می‌دهد تا بتواند ویژگی‌‌‌هایی از اشیا را که نباید به آنها دسترسی داشته باشد، بروزرسانی نماید.

نمونه‌‌‌هایی از «ویژگی‌‌‌های حساس» عبارتند از:

* ویژگی‌‌‌های مرتبط با مجوزها[[4]](#footnote-4): پرچم‌‌‌هایی نظیر user.is\_admin و user.is\_vip فقط بایستی توسط ادمین‌‌‌ها تنظیم شوند.
* ویژگی‌‌‌های وابسته به فرایند[[5]](#footnote-5): user.cash فقط باید بصورت داخلی و پس از تایید پرداخت بروزرسانی شود.
* ویژگی‌‌‌های داخلی: article.created\_time فقط باید بصورت داخلی و توسط اپلیکیشن تنظیم گردد.

**API6:2019 تخصیص جمعی**

**مثال‌‌‌هایی از سناریوهای حمله**

**سناریو #1**

یک اپلیکیشن هم‌سفری[[6]](#footnote-6) به کاربر امکان ویرایش اطلاعات پایه‌ای پروفایل خود را می‌دهد. در خلال این فرایند، یک فراخوانی API به PUT /api/v1/users/me با شی مجاز JSON زیر فرستاده می‌شود:

{“user\_name”:”inons”,”age”:24}

اما درخواست GET /api/v1/users/me ویژگی اضافی credit\_balance را نیز در خود دارد:

{“user\_name”:”inons”,”age”:24,”credit\_balance”:10}

در اینجا مهاجم درخواست اول را با محتوای مخرب زیر بازارسال[[7]](#footnote-7) می‌نماید:

{“user\_name”:”attacker”,”age”:60,”credit\_balance”:99999}

**سناریو #2**

یک پلتفرم اشتراک‌گذاری ویدئو به کاربران خود اجازه دانلود محتوا با فرمت‌‌‌های مختلفی را می‌دهد. مهاجم که در حال بررسی API است، در می‌یابد که GET /api/v1/videos/{video\_id}/meta\_data یک شیء JSON با ویژگی‌‌‌های ویدئو را باز می‌گرداند. یکی از این ویژگی‌‌‌ها، “mp4\_conversion\_params”:”-v codec h264” است که نشان می‌دهد اپلیکیشن از یک دستور Shell برای تبدیل ویدئو بهره می‌برد.

همچنین مهاجم متوجه می‌شود که POST /api/v1/videos/new در برابر تخصیص جمعی ‌‌‌آسیب‌پذیر بوده و به کلاینت اجازه می‌دهد که هریک از ویژگی‌‌‌های شیءویدئو را با به صورت دلخواه تنظیم نماید. در نتیجه مهاجم مقدار مخربی را به صورت زیر در قست ویژگی ویدئو وارد می‌کند که در نتیجه آن با دانلود ویدئو با فرمت MP4 توسط مهاجم حمله تزریق دستور Shell[[8]](#footnote-8) اجرا خواهد شد.

“mp4\_conversion\_params”:”-v codec h264 && format C:/”

**چگونه از ‌‌‌آسیب‌پذیری تخصیص جمعی پیشگیری کنیم؟**

* در صورت امکان، از بکارگیری توابعی که ورودی کلاینت را بصورت خودکار تبدیل به متغیرهای کد یا اشیای داخلی اپلیکیشن می کنند خودداری شود.
* تنها ویژگی‌‌‌های ضروری که باید توسط کلاینت بروزرسانی شوند در لیست سفید[[9]](#footnote-9) قرار گیرد.

**API6:2019 تخصیص جمعی**

* از قابلیت‌‌‌های تعبیه شده در اپلیکیشن‌‌‌ها برای قراردادن ویژگی‌‌‌هایی که تغییر آنها برای کلاینت غیرمجاز است در لیست سیاه[[10]](#footnote-10) استفاده شود.
* در صورت امکان، الگوهای واضح و بدون ابهام برای داده ورودی تعریف و اعمال شود.

**مراجع**

**خارجی**

* [CWE-915: Improperly Controlled Modification of Dynamically-Determined Object Attributes](https://cwe.mitre.org/data/definitions/915.html)

1. Mass Assignment [↑](#footnote-ref-1)
2. Properties [↑](#footnote-ref-2)
3. Level of Exposure [↑](#footnote-ref-3)
4. Permission-Related [↑](#footnote-ref-4)
5. Process-Related [↑](#footnote-ref-5)
6. Ride Sharing [↑](#footnote-ref-6)
7. Replay [↑](#footnote-ref-7)
8. Shell Command Injection [↑](#footnote-ref-8)
9. Whitelist [↑](#footnote-ref-9)
10. Blacklist [↑](#footnote-ref-10)