عنوان: **یافتن اکشن متدهای به اشتباه کش شده در ASP.NET MVC**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۶/۱۱ ۲۳:۴۵
آدرس: www.dotnettips.info
برچسب‌ها: MVC, Security

مرسوم است برای کش کردن خروجی یک اکشن متد در ASP.NET MVC از ویژگی OutputCache استفاده شود. نکته‌ی مهمی که در مورد نحوه پیاده سازی آن وجود دارد، استفاده از OutputCacheModule استاندارد ASP.NET است. در این حالت پس از فراخوانی ابتدایی اکشن متد و کش شدن محتوای حاصل از آن، در دفعه‌ی بعد فراخوانی این آدرس خاص، اصلا چرخه کاری یک کنترلر روی نداده و تمام مسایل توسط OutputCacheModule به صورت مستقل و پیش از رسیدن آن به کنترلر، مدیریت می‌شوند.
خوب، تا اینجا ممکن است مشکلی به نظر نرسد و هدف از کش کردن اطلاعات یک اکشن متد نیز همین مورد است. اما اگر این اکشن متد کش شده، به اشتباه در یک کنترلر مزین شده با ویژگی Authorize قرار گیرد، چه خواهد شد؟ مثلا این کنترلر امن، برای ارائه فایل‌ها یا حتی نمایش قسمتی از صفحه یا کل صفحه، از کش استفاده کرده است. در بار اول دریافت فایل، بدیهی است که تمام مسایل اعتبارسنجی باید مطابق طول عمر یک کنترلر روی دهند. اما در بار دوم فراخوانی آدرس دریافت صفحه یا فایل، اصلا کار به فراخوانی کنترلر نمی‌رسد. به عبارتی کلیه کاربران سایت (اعم از لاگین شده، نشده، دارای دسترسی مشاهده صفحه یا آدرس امن و یا بدون دسترسی)، به این محتوای خاص بدون مشکلی دسترسی خواهند داشت (فقط کافی است که از آدرس نهایی به نحوی مطلع شوند).

سؤال: چگونه می‌توان کلیه اکشن متدهای یک پروژه ASP.NET MVC را که دارای ویژگی OutputCache در یک کنترلر امن هستند، یافت؟

```
using System;
using System.Linq;
using System.Reflection;
// Add a ref. to \Program Files\Microsoft ASP.NET\ASP.NET MVC 4\Assemblies\System.Web.Mvc.dll
using System.Web.Mvc;
// Add a ref. to System.Web
using System.Web.UI;

namespace FindOutputCaches
{
    class Program
    {
        static void Main(string[] args)
        {
            var path = @"D:\site\bin\Web.dll";
            var asmTarget = Assembly.LoadFrom(path);

            checkSecuredControllers(asmTarget);

            Console.WriteLine("Press a key...");
            Console.Read();
        }

        private static void checkSecuredControllers(Assembly asmTarget)
        {
            // یافتن کلیه کنترلرهایی که فیلتر اوتورایز دارند
            var securedControllers = asmTarget.GetTypes()
                                              .Where(type => typeof(IController).IsAssignableFrom(type)
&&
                                                             Attribute.IsDefined(type,
typeof(AuthorizeAttribute)) &&
                                                             !type.Name.StartsWith("T4MVC"))
                                              .ToList();

            foreach (var controller in securedControllers)
            {
                // یافتن کلیه اکشن متدهای کنترلر جاری
                var actionMethods = controller.GetMethods(BindingFlags.Public | BindingFlags.Instance |
BindingFlags.DeclaredOnly)
                                              .Where(method =>
typeof(ActionResult).IsAssignableFrom(method.ReturnType))
                                              .ToList();

                foreach (var method in actionMethods)
                {
                    // یافتن متدهایی که دارای آوت پوت کش هستند
```

```
                    var attributes = method.GetCustomAttributes(typeof(OutputCacheAttribute), true);
                    if (attributes == null || !attributes.Any())
                        continue;

                    var outputCache = (OutputCacheAttribute)attributes[0]; // AllowMultiple = false
                    if (outputCache.Location == OutputCacheLocation.None)
                        continue; //سبب عدم کش شدن شده است؛ مثلا برای کارهای ای‌جکسی

                    Console.WriteLine("Detected incorrect usage of OutputCache in:\n {0}-->{1}",
                                      controller.FullName, method.Name);
                }
            }
        }
    }
}
```

کدهای کامل این بررسی را در اینجا ملاحظه می‌کنید.
ابتدا مسیر اسمبلی کامپایل شده پروژه ASP.NET MVC که حاوی کنترلرهای برنامه است، باید مشخص گردد.
سپس در این اسمبلی کلیه نوع‌های تعریف شده، یافت گردیده و آن‌هایی که پیاده سازی کننده IController هستند (یعنی کلاس‌های کنترلر واقعی برنامه) و همچنین دارای ویژگی AuthorizeAttribute نیز می‌باشند، جدا خواهند شد.
در ادامه، در هر کنترلر امن یافت شده، متدهایی را بررسی خواهیم کرد که دارای خروجی از نوع ActionResult باشند (فقط اکشن متدها مدنظر هستند). اگر این اکشن متد یافت شده دارای ویژگی OutputCacheAttribute بود و همچنین Location آن به None تنظیم نشده بود ... یعنی مشکل امنیتی وجود دارد که باید برطرف شود.

البته برای تکمیل این مطلب باید دو حالت زیر هم پیاده سازی و بررسی شوند:
- کلیه Viewهای برنامه بررسی شوند. اگر در View خاصی که متعلق است به یک کنترلر یا حتی اکشن متد امن، ارجاعی به اکشن متدی کش شده در کنترلری دیگر وجود داشت، این مورد هم یک باگ امنیتی است.
- کلیه کنترلرهای عمومی که دارای اکشن متدی امن هستند نیز باید جهت یافتن OutputCache بررسی شوند.