1.1.1 Tehlikeli Dosya Yükleme Mekanizması (Dangerous File Upload) (CWE-434)

Açıklık Önem Derecesi: Yüksek

Açıklığın Etkisi: Uzaktan kod çalıştırma

Açıklığın Barındıran Dosyalar/Satırlar:

Proje Dosyası/Dosya Adı	Satır Numarası

Açıklığın Açıklaması:

Uygulamalar özellik olarak kullanıcılarına dosya yüklemelerine (upload'a) izin verebilirler. Uygulamalar eğer kullanıcılarının yüklediği dosyalarda içerik ve/veya uzantı doğrulama kontrolü uygulamazlarsa bu durum saldırganlara çalıştırılabilir .asp, .php, .jsp, .jsf v.b. belirli web sunucu kodları yüklemelerine imkan verebilir. Bu ise saldırganlara doğrudan web sunucuda komut çalıştırma ortamına erişim sunabilir. Bir uygulama kısıtlanmamış bir dosya yüklemesi mekanizmasına sahipse "Tehlikeli Dosya Yükleme Mekanizması (CWE-434)" açıklığı var şeklinde ele alınır.

Bu açıklığı ifade edebilmek adına çeşitli teknolojilerde örnek kod bloklarına yer verilmiştir:

C# - Güvensiz Kod Bloğu:

```
/*
    Güvensiz Kod

    EN: File Uploading Without Checking Its Extension - ASP.NET MVC 5
    TR: Uzantı Kontrolü Olmadan Dosya Yükleme - ASP.NET MVC 5

*/

//Controller:
[HttpPost]
[ValidateAntiForgeryToken]
public bool DangerousFileUpload(HttpPostedFileBase fileUpload)
{
    string fileName, path;
    try
    {
        if (fileUpload != null && fileUpload.ContentLength > 0 &&
Request.ContentType.Contains("multipart/form-data"))
        {
            fileName = Path.GetFileName(fileUpload.FileName);

            path =
System.Web.HttpContext.Current.Server.MapPath("~/Content/" + fileName);

            fileUpload.SaveAs(path);
            return true;
        }
    }
    catch (Exception e)
    {
        //Handle Exceptions
        return false;
    }
    return false;
}

//View:
@using (Html.BeginForm("DangerousFileUpload", "Cx", FormMethod.Post, new {
enctype = "multipart/form-data" }))
{
    @Html.AntiForgeryToken()
    <input type="file" name="fileUpload">
    <input type="submit" value="Create" class="btn btn-default" />
}
```

C# - Güvenli Kod Bloğu:

```
/*
    Güvenli Kod

    EN: Uploading File With Proper Extension Check - ASP.NET MVC 5
    TR: Düzgün Uzantı Kontrolü ile Dosya Yükleme - ASP.NET MVC 5
*/

//Controller:
[HttpPost]
[ValidateAntiForgeryToken]
public bool DangerousFileUpload(HttpPostedFileBase fileUpload)
{
    string fileName, fileExt, path;

    try
    {
        if (fileUpload != null && fileUpload.ContentLength > 0 &&
Request.ContentType.Contains("multipart/form-data"))
        {
            fileName = Path.GetFileName(fileUpload.FileName);

            path = System.Web.HttpContext.Current.Server.MapPath("~/Content/"
+ fileName);

            fileExt = Path.GetExtension(path);

            if (fileExt == ".jpg" || fileExt == ".png")
            {
                fileUpload.SaveAs(path);
                return true;
            }
        }
    }
    catch (Exception e)
    {
        //Handle Exceptions
        return false;
    }

    return false;
}

// View:
// View katmanı güvensiz kod bloğundaki ile aynı. Kod uzamasın diye yazılmadı.
// ...
// ...
```

C# güvensiz kod örneğinde controller katmanı try bloğunda önce upload'lanan içeriği tutan http POST parametresi / nesnesi fileUpload gelmiş mi kontrolü, fileUpload parametresi / nesnesi 0 karakterden fazla içeriğe sahip mi kontrolü ve gelen istekte Content-Type başlığı multipart/form-data değerine sahip mi kontrolü yapılmaktadır. Eğer bu şartlar sağlanırsa sonra http POST parametresi / nesnesi fileUpload üzerinden dosyanın adı alınır, kaydedileceği path belirlenir ve dosya sunucuya kaydedilir. View katmanında http post metoduyla, multipart/form-data encytype'ı ile DangerousFileUpload controller metoduna dosya yükleme talebi yapmaya hazır form alanı arayüz olarak paylaşılır. Controller katmanında görüldüğü üzere herhangi bir kısıt konmadan dosya olduğu gibi alınıp sunucuya kaydedilmektedir. Bu durum güvensiz kabul edilir.

C# güvenli kod örneğinde controller katmanı try bloğunda önce yine upload'lanan içeriği tutan http POST parametresi / nesnesi fileUpload gelmiş mi kontrolü, fileUpload parametresi / nesnesi 0 karakterden fazla içeriğe sahip mi kontrolü ve gelen istekte Content-Type başlığı multipart/form-data değerine sahip mi kontrolü yapılmaktadır. Eğer bu şartlar sağlanırsa sonra http POST parametresi / nesnesi fileUpload üzerinden dosyanın adı alınır, kaydedileceği path belirlenir ve bu sefer ekstradan uzantısı da alınır. Alınan uzantı if koşulu ile kıyaslanır ve belirlenmiş uzantılardan biriyse dosya sunucu diskine kaydedilir. View katmanında ise yine http post metoduyla, multipart/form-data encytype'ı ile DangerousFileUpload controller metoduna dosya yükleme talebi yapmaya hazır form alanı arayüz olarak paylaşılır. Controller katmanında bu sefer uzantı kontrolü koşulu eklemesi nedeniyle bu dosya yükleme mekanizması güvenli kabul edilir.

PHP - Güvensiz Kod Bloğu:

```php
<?php

/*
    Güvensiz Kod

    EN: Checking a File's MIME Type Based on Data Sent by The Browser
    TR: Tarayıcıdan Gönderilen Veri Temel Alınarak Dosyanın
        MIME Türünü Kontrol Etme
*/

if ($_FILES['file_name']['type'] == "image/jpeg"){
    move_uploaded_file($_FILES['file_name']['tmp_name'] , $destination);
};
```

PHP - Güvenli Kod Bloğu:

```php
<?php

/*
    Güvenli Kod

    EN: Validating Real MIME Type of Uploaded File
    TR: Yüklenen Dosyanın Gerçek MIME Türünü Doğrulama
*/

//Verify file was uploaded via HTTP POST
if (is_uploaded_file($_FILES['file_name']['tmp_name'])) {

    //Get real MIME type
    $mime_type = mime_content_type($_FILES['file_name']['tmp_name']);

    // Generate Allowed MIME Types List
    $allowed_file_types = ['image/png', 'image/jpeg'];

    //Verify that the real MIME type is allowed
    if (! in_array($mime_type, $allowed_file_types)) {

          // File type is NOT allowed

    }

    // Set up destination of the file
    $destination = '/path/to/file/';

    // Save File into Disk
    move_uploaded_file($_FILES['file_name']['tmp_name'] , $destination);
}
```

PHP güvensiz kod örneğinde istemci tarafından gönderilen paketteki Content-Type başlığında yer alan dosya türü bilgisine göre izin verilen dosya türü kıyaslaması yapılmaktadır ve kıyaslamada koşul sağlanırsa dosya sunucu diskine kaydedilmektedir. Kötü niyetli bir kullanıcı Content-Type değerini manipüle ederek sunucuya farklı türlerde dosyaları gönderebileceği için bu durum güvensiz kabul edilir. PHP güvenli kod örneğinde ise istemci tarafından gönderilen paketteki Content-Type başlığında yer alan dosya türü bilgisine göre izin verilen dosya türü kıyaslaması yapılmaz. Bunun yerine istekle gelen dosya önce geçici olarak belleğe alınır. Ardından sunucu taraflı dosya türü belirleme fonksiyonu (mime_content_type) ile dosya türü belirlemesine tabi tutulur. Sonra belirlenen dosya türü izin verilen dosya türlerinden biri mi if koşulu kıyaslamasına tabi tutulur ve dosya bu şartı geçerse diske move_uploaded_file () ile kaydedilir. Bu durum güvenli kabul edilir. Çünkü dosya türü kontrolü istemciden gelen bilgiye göre değil de sunucu taraflı yapılmaktadır.

VisualBasic.NET - Güvensiz Kod Bloğu:

```
' Güvensiz Kod Bloğu
' ------------------------------------------------
' EN: Uploading File Without Extension Validation
' TR: Uzantı Doğrulaması Olmadan Dosya Yükleme
' ------------------------------------------------

Public Function UploadFile(fileUpload As HttpPostedFileBase)

    Dim fileName, Path As String

    Try

        If (fileUpload IsNot Nothing And
            fileUpload.ContentLength > 0 And
            Request.ContentType.Contains("multipart/form-data")) Then

            fileName = Path.GetFileName(fileUpload.FileName)
            Path = Web.HttpContext.Current.Server.MapPath("~/Content/" +
fileName)

            fileUpload.SaveAs(Path)
            Return True

        End If

    Catch

        'Handle Exceptions
        Return False
    End Try

    'If we got so far, something failed.
    Return False

End Function
```

VisualBasic.NET - Güvenli Kod Bloğu:

```
' Güvenli Kod Bloğu
' -------------------------------------------------
' EN: Validation of File Extension Before Uploading
' TR: Dosya Yüklenmeden Önce Dosya Uzantısı Doğrulaması
' -------------------------------------------------

Public Function UploadFile(fileUpload As HttpPostedFileBase)

    Dim fileName, fileExt, Path As String

    Try

        If (fileUpload IsNot Nothing And
            fileUpload.ContentLength > 0 And
            Request.ContentType.Contains("multipart/form-data")) Then

            fileName = Path.GetFileName(fileUpload.FileName)
            Path = Web.HttpContext.Current.Server.MapPath("~/Content/" +
fileName)

            ''''''''''''
            'Remediation:
            fileExt = Path.GetExtension(fileName)

             If (fileExt = ".jpg" Or fileExt = ".png") Then
                 fileUpload.SaveAs(Path)
                 Return True
             End If
            'End of Remediation.
            ''''''''''''''''''

            End If

    Catch

        'Handle Exceptions
        Return False
    End Try

    'If we got so far, something failed.
    Return False

End Function
```

VisualBasic.NET güvensiz kod örneğinde önce upload'lanan içeriği tutan http POST parametresi / nesnesi fileUpload gelmiş mi kontrolü, fileUpload parametresi / nesnesi 0

karakterden fazla içeriğe sahip mi kontrolü ve gelen istekte Content-Type başlığı multipart/form-data değerine sahip mi kontrolü yapılmaktadır. Eğer bu şartlar sağlanırsa http POST parametresi / nesnesi fileUpload üzerinden dosyanın adı alınır, kaydedileceği path belirlenir ve dosya sunucuya kaydedilir. Görüldüğü üzere herhangi bir kısıt konmadan dosya olduğu gibi alınıp sunucuya kaydedilmektedir. Bu durum güvensiz kabul edilir.

VisualBasic.NET güvenli kod örneğinde önce yine upload'lanan içeriği tutan http POST parametresi / nesnesi fileUpload gelmiş mi kontrolü, fileUpload parametresi / nesnesi 0 karakterden fazla içeriğe sahip mi kontrolü ve gelen istekte Content-Type başlığı multipart/form-data değerine sahip mi kontrolü yapılmaktadır. Eğer bu şartlar sağlanırsa http POST parametresi / nesnesi fileUpload üzerinden dosyanın adı alınır, kaydedileceği path belirlenir ve bu sefer ekstradan uzantısı da alınır. Alınan uzantı if koşulu ile kıyaslanır ve belirlenmiş uzantılardan biriyse dosya sunucu diskine kaydedilir. Bu sefer uzantı kontrolü koşulu eklemesi nedeniyle bu dosya yükleme mekanizması güvenli kabul edilir.

Kurum uygulamada "Tehlikeli Dosya Yükleme Mekanizması (CWE-434)" açıklığı olduğu tespit edilmiştir:

:::::BULGU:::::

Açıklığın Önlemi:

Bu açıklığın giderilmesi noktasında tavsiyeler şu şekildedir:

- **Depolama:** Eğer mümkünse upload'lanan (yüklenen) dosyaların sunucu diskine kaydedilmesinde sakınılmalıdır. Bunun yerine upload'lanan dosyalar harici depolama ünitelerinden örneğin veritabanına veya DMS'e (Document Management System'a) depolanmalıdır.

- **Konum:** Kullanıcı dosyaları web sitenin dizini dışında, doğrudan web tarayıcıyı ile gidilemeyen ve izole olan bir dizinde depolanmalıdır. Gerektiğinde uygulama kaynak kodu içerisinde yüklenmekte olan dosya açık bir şekilde okunmalıdır ve içeriği kullanıcıya uygun bir şekilde iletilmelidir.

- **Yürütmeme Güvencesi:** Kullanıcı tarafından yüklenen dosyalar web sunucusu / uygulama sunucusu tarafından örneğin konfigürasyon dosyası veya dizin ile çalıştırılamaz olarak açık bir şekilde işaretlenmelidir.

- **Doğrulama:** Daima tüm kullanıcı gidileri doğrulamadan geçirilmelidir. Özellikle dosya uzantısı ve/veya MIME türüne göre izinli dosya türleri izin listesi uygulanmalıdır. Rastgele dosyalara (örn; .ASP, .PHP, .JSP, .EXE, .HTML, … v.b.lerine) izin

verilmemelidir. Örneğin yalnızca resimlere (.GIF, .JPG, .PNG, .BMP,…) izin verilmelidir. Bu izin listesi iş gereksinimine göre temel alınmalıdır. Eğer iş gereksinimi medya içeriğinde olmayan dosyaların da (örn; dökümanların, çalıştırılabilir dosyaların veya kodların da) izin listesine dahil edilmesini gerektiriyorsa bu dosyalarda zararlı içerik var mı diye tarama yapacak ayrı bir güvenlik katmanı oluşturulmalıdır.

- **Yeniden İsimlendirme:** Diske dosyayı kaydetmeden önce açık bir şekilde dosya türüne göre öntanımlı bir uzantı adı ile kayıt zorlamasında bulunulmalıdır. Bu işlem dosya ismine hardcoded (elle) bir uzantı eklenmesi ile veya yapılandırılmış izinli dosya türleri listesindeki uzantılardan birinin eklenmesi (concatenation) ile yapılabilir.

- **İzinler:** Derinlikli defans (defense-in-depth) gereği uygulama sunucusunu minimal işletim sistemi hak ayrıcalıkları ve kısıtlı sistem hesabında çalışacak şekilde konfigüre etmek gerekir.

Referanslar:

1. https://en.wikipedia.org/wiki/Document_management_system
2. http://cwe.mitre.org/data/definitions/434.html