1.1.1 Yorumlar Yoluyla Bilgi İfşası (Information Exposure through Comments) (CWE-615)

Açıklık Önem Derecesi: Düşük

Açıklığın Etkisi: Bilgi ifşası

Açıklığın Barındıran Dosyalar/Satırlar:

Proje Dosyası/Dosya Adı	Satır Numarası

Açıklığın Açıklaması:

Uygulama geliştiricileri ürünlerini geliştirirken kaynak kodlarına yorum satırları girebilirler. Bu sayede binlerce, milyonlarca yazılan kod satırlarına bir süre sonra geri döndüklerinde "Ben burada ne yapmışım?" sorusuna karşılık koda tekrardan hızlıca adapte olabilirler. Fakat uygulama kaynak kodlarına yorum satırları girme alışkanlığı saklı tutulması gereken verilerin yanlışlıkla ifşa edilmesi riskini doğurabilir.

Uygulama sunucusundaki veya uygulama istemcisindeki kaynak kodların yorum satırlarında yer alan hassas veriler saldırganların uygulama sunucusuna sızması sonrası veya uygulama istemcisine tersine mühendislik yapması sonrası elde edilebilir. Saldırgan bundan hareketle sızma girişiminde mevcut şartlarda daha ileriye gidemeyecekken yorum satırlarındaki hassas verileri kullanarak daha da öteye gidebilir. Defense-in-depth (DiD), yani derinlikli defans prensibi gereği kaynak kodların yorum satırlarında parola, API anahtarı, hata ayıklama (debugging) bilgi satırları, uygulamada yer alması düşünülen, ancak daha sonra iptal edilen eski kod satırları v.b hassas bilgiler içermemelidir. Uygulama kaynak kodlarındaki yorum satırlarında bu tarz hassas veri yer aldığında “Yorumlar Yoluyla Bilgi İfşası” (CWE-615) açıklığı olarak işaretlenirler.

Bu açıklığı somutlaştırmak için çeşitli dillerde “Yorumlar Yoluyla Bilgi İfşası” açıklıklı kod blokları verilmiştir:

Java Güvensiz Kod Bloğu: (Yorumda Admin Hesap Bilgileriyle Controller Login Action’ı)

```
<html>
<head><title>JSP</title></head>
<body>

  <%
    // ## Admin User ##
    // user = admin
    // password = dgh436yuhc

    byte[] password = request.getParameter("password").getBytes();

    if (password != null){

      validatePassword(password);

    }

    Arrays.fill(password, '\0');
  %>
</body>
</html>
```

Javascript Güvensiz Kod Bloğu:

```
function login() {
    // send credentials to server instead of reading from database
    // constring = "Initial Catalog=mytest;User Id=sa;Pwd=mypass;";
    var creds = "username=" + txtUsername.text + "&password=" +
txtPassword.text;
    var resp = sendToServer("/login", creds);
    return (resp == "success");
}
```

C# Güvensiz Kod Bloğu: (Yorum Satırlarında Hesap Bilgileriyle Razor Login Form'u)

```
[HttpPost]
[AllowAnonymous]
[ValidateAntiForgeryToken]
public async Task<ActionResult> Login(LoginViewModel model, string returnUrl)
{
    // ## ADMIN USER  ##
    // Username: admin@example.com
    // Password: Pa$$w0rd

    if (!ModelState.IsValid)
    {
        return View(model);
    }

    var result = await SignInManager.PasswordSignInAsync(model.Email,
model.Password, model.RememberMe, shouldLockout: true);

    switch (result)
    {
        case SignInStatus.Success:
            return RedirectToLocal(returnUrl);
        case SignInStatus.LockedOut:
            return View("Lockout");
        case SignInStatus.RequiresVerification:
            return RedirectToAction("SendCode", new { ReturnUrl = returnUrl,
RememberMe = model.RememberMe });
        case SignInStatus.Failure:
        default:
            ModelState.AddModelError("", "Invalid login attempt.");
            return View(model);
    }
}
```

Kurum uygulamada yorum satırlarında hassas verilere yer verildiği tespit edilmiştir:

::::::::BULGU:::::::

Açıklığın Önlemi:

Uygulama kaynak kodlarında yorum satırlarında hassas verileri depolamayın. Var olanları ise ivedilikle kaldırın.

Referanslar:

1. https://cwe.mitre.org/data/definitions/615.html
2.