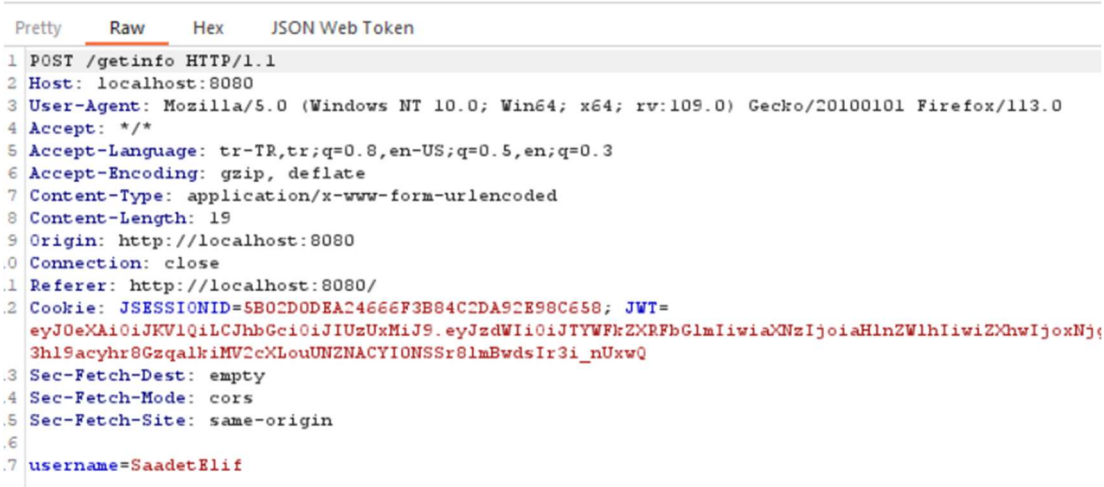
HYGEIA uygulaması SQL Injection zafiyet gerçeklemesi

HYGEIA uygulamasında iki bölümde SQL injection zafiyeti bulunmaktadır bunlardan biri kullanıcının kendi bilgilerini görebilmesi ve güncelleyebilmesi için oluşturulmuş bir alanda bulunmaktayken bir diğeri uygulamanın parola sıfırlama bölümünde bulunmaktadır. İlk örneğimizde bilgi formumuz bu şekilde görülüyor;

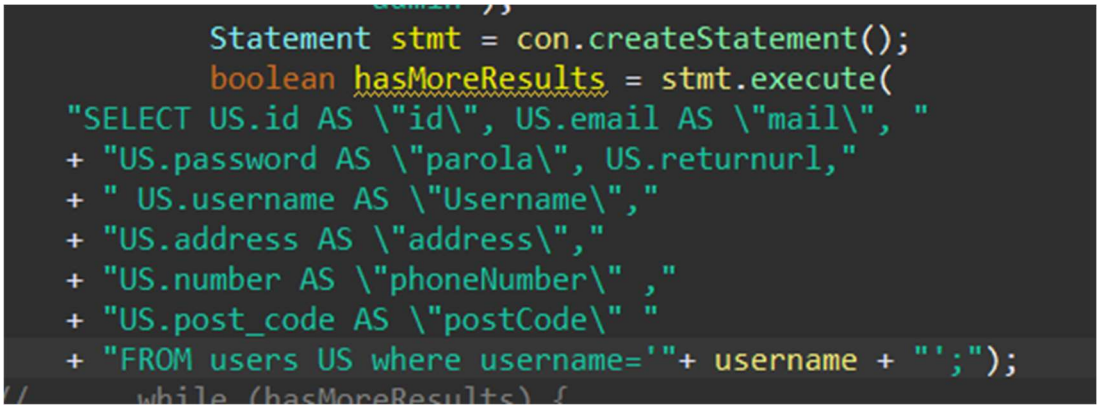
Şekil 1.1 Bilgi sayfası ekran görüntüsü

Bu bilgi formuna ulaşmak için kullanıcının siteye gönderdiği isteği inceleyelim;



Şekil 1.2 SQLi request

Bu istekte basit bir şekilde “username” parametresinde kullanıcı adının backend’e gönderildiğini görebiliriz. Şimdi, uygulamanın arka planında iletilen parametreye nasıl işlemler uyguladığını bu alana ait kod parçacığı ile birlikte görelim;



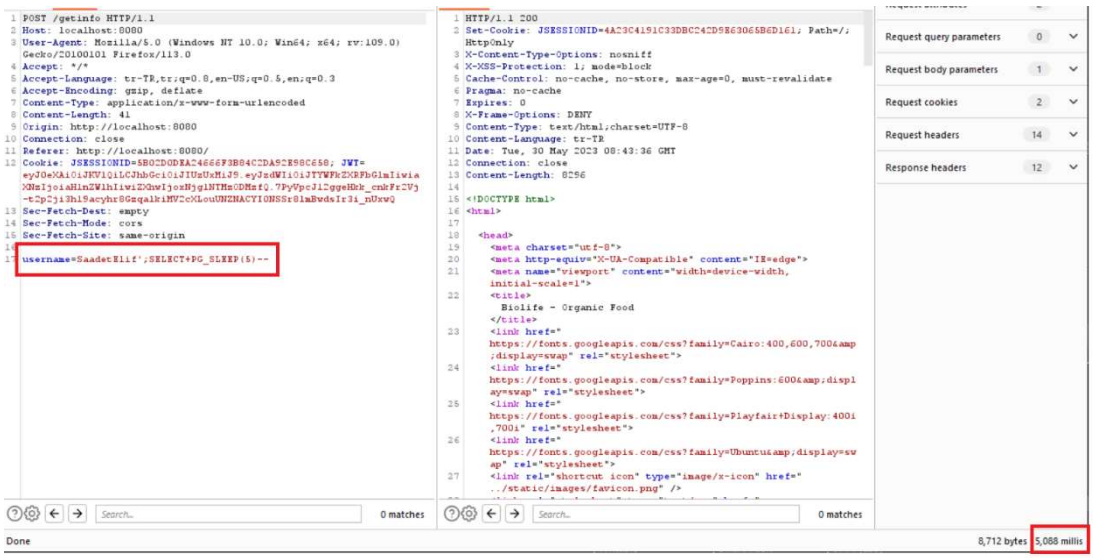
Şekil 1.3 zafiyetli kod parçacığı

Yukarıda verilen kodu incelersek sorguda “users” adlı tablodan parametrede gönderilen kullanıcı adına ait email, adres, telefon numarası gibi bir çok bilginin veri tabanından istendiğini görebiliriz. Bu sorguda ana sorun “username” parametresinin sorguya hiç bir filtreleme önlemi alnımadan direk sorgu içine yazılmasından kaynaklanmaktadır bu durumda sorgu içine kullanıcı proxy ile araya girerek verilen kullanıcı adına verdiği girdiyi *SaadetElif';SELECT PG_SLEEP(5)—*haline çevirirse arka planda gerçekleşecek olan sorgunun son hali aşağıdaki gibi olur;

SELECT US.id AS "id", US.email AS "mail", US.password AS "parola", US.returnurl, US.username AS "Username",US.address AS "address",US.number AS "phoneNumber" ,US.post_code AS "postCode" FROM users US where username='**SaadetElif';SELECT PG_SLEEP(5)--**

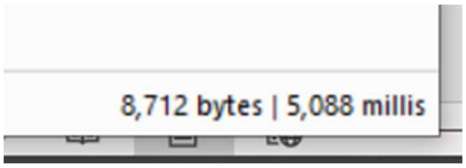
Bu sorgunun ilk bölümünde SELECT US.id AS "id", US.email AS "mail", US.password AS "parola", US.returnurl, US.username AS "Username",US.address AS "address",US.number AS "phoneNumber" ,US.post_code AS "postCode" FROM users US where username='**SaadetElif';**

Uygulama normalde de yapacağı gibi SaadetElif kullanıcısının verilerini kullanıcıya geri yansıtacaktır ama saldırganın SQLi zafiyetinden yararlandığı ikinci alanda ise `SELECT PG_SLEEP(5)--` sorgusu çalışacak ve veritabanının cevabını beş saniye geciktirecek ve burada SQLi zafiyeti olduğunu kanıtlayacaktır.



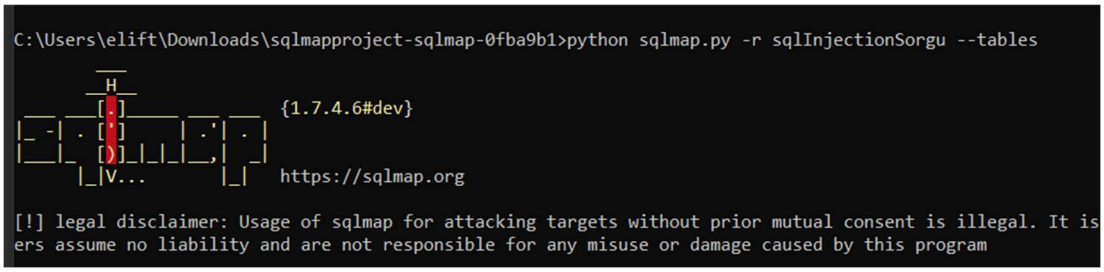
Şekil 1.4 SQLi isteği

Şekil 1.5 SQLi isteği yakından

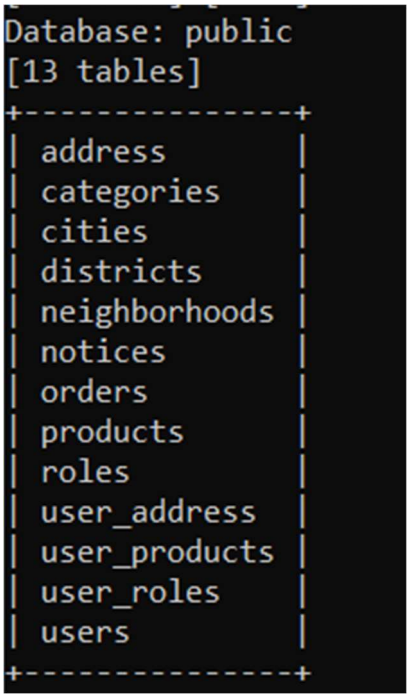


Şekil 1.6 SQLi cevabı

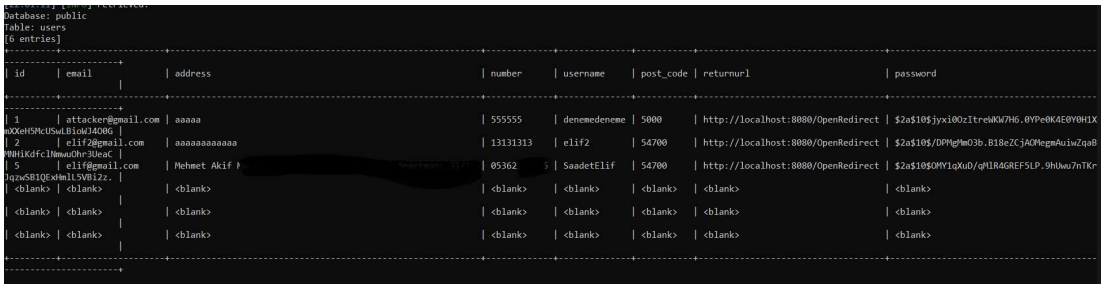
Bu durumun ardından ise saldırgan SQLMAP aracını kullanarak bütün veri tabanı üzerindeki bilgileri elde edebilir. SQLMAP aracıyla elde edilemeyecek bulgular için ise manuel ilerlemeyi veya kendi scriptini yazmayı deneyebilir. Biz kanıt olarak veri tabanındaki tabloların adını SQLMAP aracıyla elde ettik .



Şekil 1.7 Sqlmap sorgusu



Şekil 1.8 Tablo isimleri



Şekil 1.9 Users tablosunun verileri

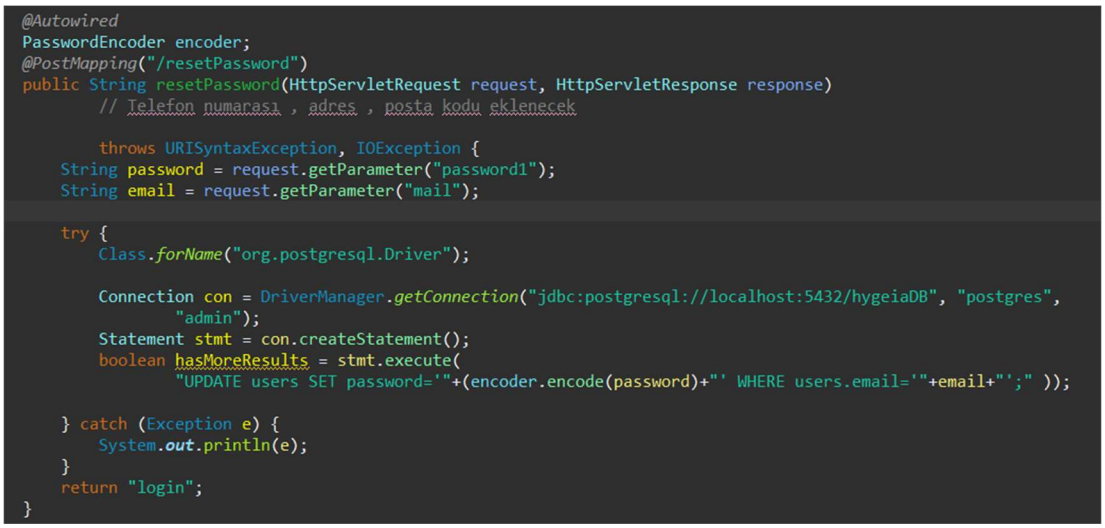
Bu kodunun güvenli olması için Java üzerinde bulunan PreparedStatement nesnesi kullanılabilir. Bu nesne ile uygulama verilen girdileri kendi içinde filtreler ve zararlı girdileri sorgu içinden çıkartır

```
Statement stmt = con.createStatement();
String query = "SELECT US.id AS \"id\", US.email AS \"mail\", US.password AS \"parola\", US.returnurl, "
        + "US.username AS \"Username\", US.address AS \"address\", US.number AS \"phoneNumber\", US.post_code AS \"postCode\" "
        + "FROM users US WHERE username = ?";

PreparedStatement guvenli = con.prepareStatement(query);
guvenli.setString(1, username);
ResultSet resultSet = guvenli.executeQuery();
```

Şekil 1.10 prepared statement ile filtrelenmiş kod

İkinci örneğimizi inceleyecek olursak uygulamada bir parola sıfırlama öğesi bulunuyor. Bu parola sıfırlama bölümünde, kullanıcıdan alınan mail adresi ve parola değeri java kodlarının içine güvenli bir şekilde yazılmamıştır.



```
@Autowired
PasswordEncoder encoder;
@PostMapping("/resetPassword")
public String resetPassword(HttpServletRequest request, HttpServletResponse response)
        // Telefon numarası , adres , posta kodu eklenecek

        throws URISyntaxException, IOException {
    String password = request.getParameter("password1");
    String email = request.getParameter("mail");

    try {
        Class.forName("org.postgresql.Driver");

        Connection con = DriverManager.getConnection("jdbc:postgresql://localhost:5432/hygeiaDB", "postgres",
                "admin");
        Statement stmt = con.createStatement();
        boolean hasMoreResults = stmt.execute(
                "UPDATE users SET password='"+(encoder.encode(password)+"' WHERE users.email='"+email+"';" ));

    } catch (Exception e) {
        System.out.println(e);
    }
    return "login";
}
```

Şekil 1.11 Parola sıfırlama SQLi görseli