

Forum:[Hack Forums](#)

Thread: [Need help in NoSQL injection exploitation](#)

wolises584 - August 12, 2024, 2:26 am

Hello Hacker,

I've a source code in nodejs, now I know that it is vulnerable to NoSQL injection. I know that input field username or phone can be injected, But i'm not quite sure how to exploit this to login bypass or exfiltrated the data from the app.

Need your hacker minds. ?

source code:

```
Code const { default: axios } = require('axios');
const bcrypt = require('bcryptjs');
const { WS_SRV_URL, SHARED_SECRET_KEY, IS_DEV } = require('.././config.js');
const { SUPER_USER_ROLES, REGEX } = require('.././constants.js');
const { logToChannel } = require('.././helper/tgBot.js');
const Token = require('.././models/jwt.js');
const User = require('.././models/user.js');
const TokenHandler = require('.././TokenHandler.js');
const { verifyOTP } = require('.././services/admin-otp-verify.js');
const { getSiteStatus } = require('.././admin/modifySiteStates.js');
```

```
function Login() {
  return async (req, res) => {
    try {
      const siteStatus = await getSiteStatus();
      if (
        (siteStatus?.userLogin === false ||
          siteStatus?.userLogin === 'false') &&
        !SUPER_USER_ROLES.LEVEL_2.includes(user.privillage)
      ) {
        res.json({
          err: 'website is under maitainance please come back later',
          success: false,
        });
        return;
      }
    }
  }
}
```

```
//search for the username recieved the req.body object
const { phone, username, otp, password } = req.body;
let user;
let mode;
```

```
if (phone) {
  if (!REGEX.TEN_DIGIT_NUMBER.test(phone)) {
    return res.json({
```

```

        err: 'Please Make Sure your WhatsApp Number is correct',
    });
}
user = await User.findOne({
    phone: phone,
    otpConfirmed: true,
});
mode = 'phone';
} else if (username) {
    if (!REGEX.ALPHA_NUMERIC.test(username)) {
        return res.json({
            err: 'Please Make Sure username is correct',
        });
    }
    user = await User.findOne({
        username: username,
        otpConfirmed: true,
    });
    mode = 'username';
} else {
    return res.json({
        err: 'Please Make Sure Details are correct',
    });
}

if (!user) {
    console.error(
        '[LOGIN TRY]: user not found',
        req.headers['x-forwarded-for'],
        {
            phone,
            username,
            password,
        }
    );
    return res.json({
        err: 'Please Make Sure your WhatsApp Number is correct',
    });
}

if (user.isBlocked) {
    console.log('account is blocked', user.username);
    res.json({
        isBlocked: true,
        err: 'Your Account has been Blocked because you violated our terms and policy. Kindly
Contact Support for Resolution',
    });
    return;
}

if (mode === 'phone') {
    const date = new Date();
    const last = new Date(date.getTime() - 15 * 60 * 1000);

```

```

if (user.otp?.updatedAt < last) {
  return res.json({
    err: 'Your OTP Code has expired',
    success: false,
  });
}
if (user.otp.code !== parseInt(otp)) {
  if ([ '8888888888' ].includes(phone)) {
    if (otp !== '123456') {
      return res.json({
        err: 'invalid OTP',
      });
    } else {
      logToChannel(`MOCK OTP LOGIN: ${phone}`);
    }
  } else {
    console.error(
      '[LOGIN TRY]: INVALID OTP',
      req.headers['x-forwarded-for'],
      {
        phone,
        username,
        password,
      }
    );
    return res.json({
      err: 'invalid OTP',
    });
  }
}
}
if (user.privillage === 'user') {
  await Token.deleteMany({ userId: user._id });
  try {
    await axios(`${WS_SRV_URL}/logout-user/${user._id}`, {
      headers: {
        'shared-secret-key': SHARED_SECRET_KEY,
      },
    });
  } catch (err) {
    console.error('[LOGOUT_USER', user.phone, err.message);
  }
}
}
if (mode === 'username') {
  if (!SUPER_USER_ROLES.LEVEL_2.includes(user.privillage)) {
    return res.json({
      err: 'Unauthorized!!!',
    });
  }
}
const [otp, pass] = IS_DEV
  ? ['123456', password]
  : password.split('+');
if (otp.length !== 6) {

```

```

logToChannel(
  `[ADMIN LOGIN TRY]: OTP not valid ${
    req.headers['x-forwarded-for']
  } ${JSON.stringify({
    phone,
    username,
    password,
  })}`
);
res.json({
  err: 'Please Make Sure your Password is Correct',
});
return;
}
if (!IS_DEV && !verifyOTP(otp)) {
  logToChannel(
    `[ADMIN LOGIN TRY]: OTP not valid ${
      req.headers['x-forwarded-for']
    } ${JSON.stringify({
      phone,
      username,
      password,
    })}`
  );
  res.json({
    err: 'Please Make Sure your Password is Correct',
  });
  return;
}

```

```

// compare the given password with the stored hashed password
let validPassword = bcrypt.compareSync(pass, user.password);
// check if the password is valid
if (!validPassword) {
  console.error(
    '[LOGIN TRY]: password not valid',
    req.headers['x-forwarded-for'],
    {
      phone,
      username,
      password,
    }
  );
  logToChannel(
    `[ADMIN LOGIN TRY]: password not valid ${
      req.headers['x-forwarded-for']
    } ${JSON.stringify({
      phone,
      username,
      password,
    })}`
  );
  res.json({

```

```

        err: 'Please Make Sure your Password is Correct',
    });
    return;
}
}
//initialize the tokenHandler

let userToken = new TokenHandler();
await userToken.getToken(user._id);

let extraProps = {};

const origin = req.get('origin');
if (origin.includes('admin') || mode === 'username') {
    if (!SUPER_USER_ROLES.LEVEL_2.includes(user.privillage)) {
        console.log('user is not an admin', user.username);
        logToChannel(
            `${user.username} is trying to login as an admin`
        );
        return res.json({
            err: 'You are not an admin',
        });
    } else {
        logToChannel(`${user.username} logged into admin`);
    }
    extraProps = {
        privillage: user.privillage,
    };
}

//setting the token as response header
res.setHeader('id', userToken.id);
res.setHeader('key', userToken.key);
res.cookie('cookieName', 'hi there fucking user', {
    maxAge: 900000,
    httpOnly: true,
});
//also returning the id and key in response body
res.json({
    id: userToken.id,
    key: userToken.key,
    ...extraProps,
});
} catch (error) {
    console.error(
        'login',
        req.headers['x-forwarded-for'],
        req.body,
        error
    );
    res.json({
        success: false,
        err: 'Something went wrong!',
    });
}

```

```

    });
  }
};
}
module.exports = Login;

```

SilentNuke - August 12, 2024, 11:09 am

what part did you say was vulnerable? Can you also post that part separately please?

wolises584 - August 13, 2024, 12:47 am

This is a login page for the application, and for login it is accepting 2 different methods, with username and with the phone no. in below code it's taking user supplied input and using it without any validation.

Code //search for the username recieved the req.body object

```

const { phone, username, otp, password } = req.body;
let user;
let mode;

```

Then it's checking if user is logging with phone or username, but using all the values without validation, so there is NoSQL operator injection there.

Code

```

    if (phone) {
    if (!REGEX.TEN_DIGIT_NUMBER.test(phone)) {
    return res.json({
    err: 'Please Make Sure your WhatsApp Number is correct',
    });
    }
    user = await User.findOne({
    phone: phone,
    otpConfirmed: true,
    });
    mode = 'phone';
  } else if (username) {
    if (!REGEX.ALPHA_NUMERIC.test(username)) {
    return res.json({
    err: 'Please Make Sure username is correct',
    });
    }
    user = await User.findOne({
    username: username,
    otpConfirmed: true,
    });
    mode = 'username';
  } else {
    return res.json({
    err: 'Please Make Sure Details are correct',
    });
  }
}

```

i'm able to inject NoSQL operator and can bypass all the checks. but it's simply responding with an empty response because i'm not able to bypass other logics of the application and it's not giving me auth token in response.

like this below code snippet, i can bypass this check, with simple input of {\$ne:"invalid"}.

```
Code if (mode === 'username') {
  if (!SUPER_USER_ROLES.LEVEL_2.includes(user.privillage)) {
    return res.json({
      err: 'Unauthorized!!!',
    });
  }
  const [otp, pass] = IS_DEV
    ? ['123456', password]
    : password.split('+');
  if (otp.length !== 6) {
    logToChannel(
      `[ADMIN LOGIN TRY]: OTP not valid ${
        req.headers['x-forwarded-for']
      } ${JSON.stringify({
        phone,
        username,
        password,
      })}`
    );
    res.json({
      err: 'Please Make Sure your Password is Correct',
    });
    return;
  }
}
```

but there is another check in the code, which is not letting me enter in the application.

```
Code let validPassword = bcrypt.compareSync(pass, user.password);
// check if the password is valid
if (!validPassword) {
  console.error(
    `[LOGIN TRY]: password not valid`,
    req.headers['x-forwarded-for'],
    {
      phone,
      username,
      password,
    }
  );
  logToChannel(
    `[ADMIN LOGIN TRY]: password not valid ${
      req.headers['x-forwarded-for']
    } ${JSON.stringify({
      phone,
      username,
      password,
    })}`
  );
  res.json({
    err: 'Please Make Sure your Password is Correct',
  });
  return;
}
```

}

Hope this can help to understand!