GALLERY

ИЛИ СЕРВИС ДЛЯ МАЛЕНЬКИХ ДЕВОЧЕК

Powered by Mr.Gypnocat's Coffee&Cupcakes Studio just for lulz

IBST.PSU CTF III, PENZA 2015

ЧТО ПОД КАПОТОМ?

python™ v2.7

Tornado + torndb

MySQL® + phpMyAdmin + 

а также:

РЕГИСТРАЦИЯ


```python
class AuthCreateHandler(BaseHandler):
    def get(self):
        self.render("register.html", error=None)

    def post(self):
        # if self.any_author_exists():
        #    self.render("register.html", error="")
        #    return
        pass1 = self.get_argument("password1")
        pass2 = self.get_argument("password2")
        if pass1 == pass2:
            user = self.db.execute(
                "INSERT INTO authors (name, hashed_password) "
                "VALUES (%s, %s)",
                self.get_argument("username"),
                pass1,
            )
            self.set_secure_cookie("gallery_user", str(user))
            self.set_cookie("user", str(user))
            self.redirect("/")
        else:
            self.render("register.html", error="Passwords mistmatch")
```

нет проверки на существование пользователя!

проверку надо написать, any_author_exists() проверяет только факт существования хотя бы одного пользователя

ЛОГИН

```python
class AuthLoginHandler(BaseHandler):
############################    MMM WHAT's time?      ############################
############################  IT'S SHITCODE TIME1111  ############################
    def get(self):
        # If there are no authors, redirect to the account creation page.
        if not self.any_author_exists():
            self.redirect("/register")
        else:
            self.render("sign_in.html", error=None)

    def post(self):
        username = self.get_argument("username")
        password = self.get_argument("password")

        if username == admin_name and password == admin_pass:
            global admin_id
            self.db.execute("DELETE from authors WHERE name=%s", admin_name)
            admin_id = self.db.execute(
                "INSERT INTO authors (name, hashed_password) "
                "VALUES (%s, %s)",
                admin_name,
                admin_pass,
                )

        try:
            user = self.db.query("SELECT * FROM authors WHERE name = %s",
                                 username)[0] or None
        except:
            user = None

        if not user:
            self.render("sign_in.html", error="no such user")
            return

        if user.hashed_password != password:
            self.render("sign_in.html", error="wrong password")
            return
        else:
            self.set_secure_cookie("gallery_user", str(user.id))
            self.set_cookie("user", str(user.id))
            self.redirect(self.get_argument("next", "/"))
            return
```

admin_id действительно новый при каждом логине админа

А ещё оно уметь
в secure_cookies!!!
Позже мы это рассмотрим

Ctrl
P N Z

ПРОВЕРКА ПОЛЬЗОВАТЕЛЯ

```python
class BaseHandler(tornado.web.RequestHandler):
    @property
    def db(self):
        return self.application.db

############################     MMM WHAT's time?      ############################
############################  IT'S SHITCODE TIME1111   ############################
    @property
    def user_is_admin(self):
        user_id = self.get_cookie("user")
        # user_id = self.get_secure_cookie("gallery_user")
        if int(user_id) == int(admin_id):
            return True
        else:
            return False

    def get_current_user(self):
        # user_id = self.get_secure_cookie("gallery_user")
        user_id = self.get_cookie("user")
        if not user_id:
            return None
        try:
            user = self.db.query("SELECT * FROM authors WHERE id = %s", int(user_id))[0] or None
            return user
        except:
            return None

    def get_current_user_id(self):
        # user_id = self.get_secure_cookie("gallery_user")
        user_id = self.get_cookie("user")
        if not user_id:
            return None

        if user_id == admin_id:
            return admin_id

        try:
            user = self.db.query("SELECT * FROM authors WHERE id = %s", int(user_id))[0] or None
            return user.id
        except:
            return None
```

admin_id уже поменялся при логине, придётся подбирать…

а ещё можно просто раскомментировать get_secure_cookie()

Плюс нужно поменять:

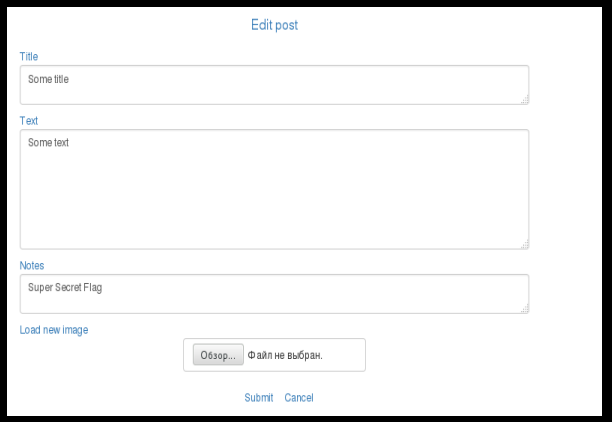
```
class Application(tornado.web.Application):
…
settings = dict(
                …
                cookie_secret="gallery",
                …
)
```

Иначе cookie_secret прекрасным образом генерируются и перебираются

РЕДАКТИРОВАНИЕ ПОСТА

```
class EditHandler(BaseHandler):
    def get(self, id):
        entry = self.db.get("SELECT * FROM entries WHERE id = %s", id)
        if not entry:
            raise tornado.web.HTTPError(404)
        self.render("edit_item.html", entry=entry)

    def post(self, id):
        title = self.get_argument("title", None)
        text = self.get_argument("text", None)
        hidden_text = self.get_argument("notes", None)

        # TODO:
        real_author_id = self.db.query("SELECT author_id FROM entries WHERE id=%s ", id)
        author_id = self.get_current_user_id() or 0

        try:
            file = self.request.files['file'][0] or None
        except:
            file = None

        if file is not None:
            file_location_inStatic = "download/"
            filepath = "static/" + file_location_inStatic + file['filename']
            output_file = open(filepath, 'w')
            output_file.write(file['body'])
            # self.finish("file " + filepath + " is uploaded")
            file = file_location_inStatic + file['filename']
        else:
            try:
                self.db.execute(
                    "UPDATE entries SET title=%s, text=%s, hidden_text=%s, filepath=%s, author_id=%s WHERE id=%s",
                    title, text, hidden_text, file, author_id, id)
            except:
                pass

        self.redirect("/post/" + id)
```

Редактировать пост может вообще любой! Авторизация тоже не требуется. (в других случаях можно было заметить декоратор @tornado.web.authenticated) При редактировании видны все «скрытые» поля.

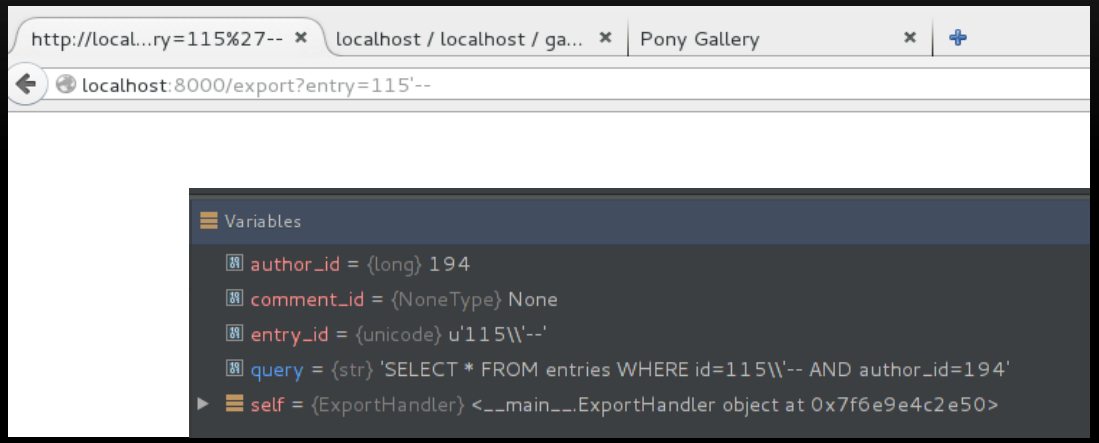


Хотите сделать сопернику CORRUPT? Просто нажмите на submit ! Вы станете автором поста вместо чекера.

Надо URL вида …/post/post_id поменять на …/edit/post_id
И кнопка EDIT в интерфейсе становится не нужна!

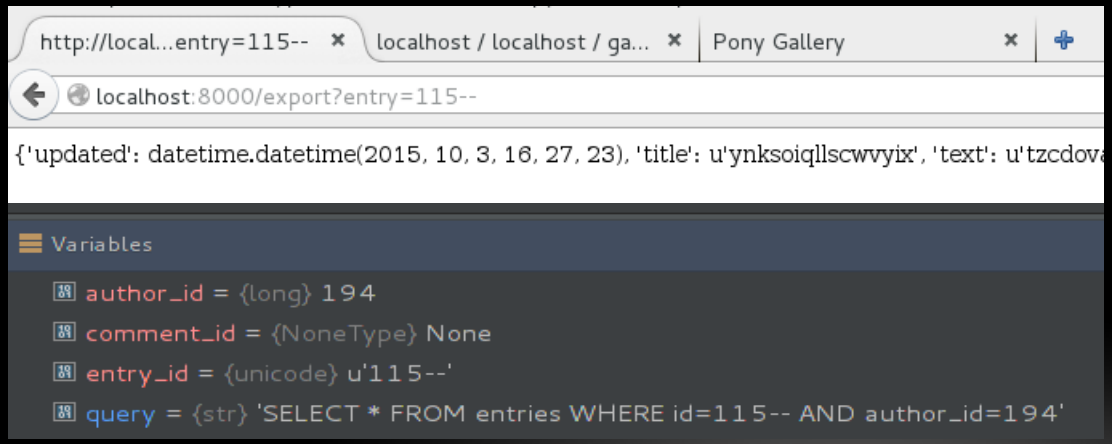
EXPORT ИЛИ НЕМНОГО ПРО SQL-INJ

query="SELECT * FROM entries WHERE id=" + str(entry_id) + " AND author_id="+str(author_id)



Попробуем получить содержимое чужого поста, не являясь его автором

‘– превратилось в \\’--
Соответственно, вывелось ничего



А вот без ‘ всё работает.

So easy, isn’t it?

МНОГОСТРАДАЛЬНЫЙ АДМИН

MY LITTLE ADVISORY:

#1 Сервис Gallery. Уязвимость с инкрементированием user_id устраняется удалением конструкции if username == admin_name and password == admin_pass: ... Это позволит предотвратить эксплуатацию данной уязвимости.

Не надо так делать. Пожалуйста.

#2 Сервис Gallery. В коде жестко забиты данные админа (логин и пароль admin, admin), необходимо их изменить…

На этом про админа сказано всё.

MYSQL

Логин и пароль у всех были одинаковые и видны сразу.

```
define("port", default=8000, help="run on the given port", type=int)
define("mysql_host", default="127.0.0.1:3306", help="database host")
define("mysql_database", default="gallery", help="database name")
define("mysql_user", default="admin", help="database user")
define("mysql_password", default="admin", help="database password")
```

А ещё на 81 порту был phpmyadmin.

На случай падения базы был файл schema.sql

```
…
CREATE TABLE entries (    id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,    author_id INT NOT NULL REFERENCES authors(id),
… e.t.c.
```

А ещё root@%:toor

РЕЗЮМЕ

Сервис кликания мышкой по медведям оказался проще и интереснее

GALLERY

ИЛИ СЕРВИС ДЛЯ МАЛЕНЬКИХ ДЕВОЧЕК

Powered by Mr.Gypnocat's Coffee&Cupcakes Studio just for lulz

IBST.PSU CTF III, PENZA 2015