امنیت داده و شبکه

عرفان نصرتی

دکتر کامبیز میزانیان

فاز دوم پروژه

به نام خدا

۱ احراز هویت کاربران

در این قسمت طبق خواسته پروژه باید یک سیستم برای احراز هویت کاربران انجام می‌دادیم که مبتنی بر گذرواژه بود. برای اینکه گذرواژه بین سرور و کلاینت به صورت آشکار منتقل نشود و همچنین بتوانیم امنیت پیشرو داشته باشیم از دیفی هیلمن برای رد و بدل کردن یک کلید مشترک استفاده کردیم و پس از اجرای آن و رسیدن به یک کلید در اولین مرحله سرور پس از به اشتراک گذاشتن پارامترهای عمومی دیفی‌ـ هیلمن از کلاینت کلید عمومی او را درخواست می‌کند سپس با استفاده از کلید عمومی کاربر و کلید خصوصی خودش کلید مشترک را حساب کرده و سپس یک کلید متقارن تولید می‌کنید و کلید متقارن را به وسیله کلید مشترک رمز می‌کند و همراه کلید عمومی خود برای کاربر می‌فرستند. و کاربر هم با حساب کردم کلید مشترک، کلید متقارن را رمزگشایی می‌کند.

از اینجا به بعد هر دو سرور و کلاینت از این کلید مشترک برای رد و بدل کردن مسیج‌ها استفاده می‌کنند تا مهاجم نتواند آنها را تغییر دهد.همچنین برای دسترسی بهتر از یک پایگاه داده SQLite استفاده کرده ایم که مشخصات کابران را در آن ذخیره می‌کنیم تا زمانی که سرور را قطع کردیم و دوباره آن را راه اندازی کردیم بتوانیم اطلاعات را به وسیله‌ی این پایگاه داده بازیابی کنیم.

```python
def client_DH_key_exchange(server):
    client_DH = KeyEchange()
    client_pub_key = client_DH.get_pub_key()

    server_public_key, nonce, tag, cipher_text = server.server_DH_key_exchange(client_pub_key=client_pub_key)
    client_DH.calculate_share_key(server_public_key)
    session_key = client_DH.decrypt(cipher_text, tag, nonce)
    return session_key
```

۲ اعتبارسنجی درخواست ها

برای اعتبارسنجی درخواست‌ها از روش RBAC استفاده می‌کنیم. برای این کار سه اکشن R ، W، D داریم. و هر کاربری که وارد می‌شود ابتدا یک role با نام کاربری او ساخته می‌شود و به او نسبت داده می‌شود. همچنین کاربر را در لیست کاربران اد می‌کنیم. حال در صورتی که یک کاربر یک فولدر ایجاد کند یا فایلی را درست کند دسترسی کامل را به آن می‌دهیم. و از این به بعد با یک سه‌تایی $(role,\ action,\ resource)$ دسترسی به یک فایل را بررسی می‌کنیم اگر نقش درخواست دهنده بتواند دسترسی خواسته شده را به منبع داشته باشد این کار از طریق سرور بررسی می‌شود و نتیجه درخواست به کاربر بر می‌گردد. یک دیکشنری از سه‌تایی ها داریم که به وسیله آن وسیله آن می‌توانیم بررسی کنیم آیا یک کاربر با نقش مشخص می‌تواند به یک فایل دسترسی پیدا کند. این را با allowed و denied مشخص می‌کنیم.

۳ دستورات مربوط به پوشه و فایل

برای پیاده‌سازی دستورات مربوط به فایل از کتابخانه os استفاده کرده‌ایم تا در تمام محیط ها قابل اجرا باشد. همچنین برای مپ کردن و رمز کردن آدرس و نام فایل ها از یک کلید مستر در سرور استفاده می‌کنیم و تمام مسیرها و نام فایل‌ها را به وسیله آن رمز می‌کنیم. توجه داشته باشید که رمز کردن خود فایل‌ها به وسیله کلیدی که برای هر فایل به صورت جداگانه تولید می‌شود رمز می‌شود. همچنین با استفاده از os.path.absolute مسیرهایی را که می‌گیریم را به مسیر کامل تبدیل می‌کنیم تا هم به صورت سراسری و هم به صورت محلی در دسترس باشد.

```python
def mkdir(path, server, client):
    path = server.encrypt_path(path)

    path = path_list(os.path.abspath(path))
    reverse_path = []
    for index in range(len(path)):
        if index == len(path) - 1:
            reverse_path.append(path[index])
            continue
        if path[index] in path[index + 1 :]:
            continue
        reverse_path.append(path[index])
    path = os.path.join(*reverse_path)
    if os.path.exists(path):
        return
    os.makedirs(path)
    server.access_control.allow(client.username, "R", [str(path)])
    server.access_control.allow(client.username, "W", [str(path)])

    return
```

همانطور که در بالا دیده می‌شود ابتدا مسیر به encrypt-path داده می‌شود تا با استفاده از کلید سرور رمز شود و در مخزن فایل مسیر فایل‌ها و نام فایل‌ها هم رمز شود. سپس فولدرهایی که برای کاربر وجود ندارد را می‌سازیم و سپس

دسترسی‌های لازم را به کاربران می‌دهیم.

۴ ویرایش محتوای فایل ها

برای ویرایش فایل از کتابخانه subprocess و ویرایشگر پیش‌فرض متنی استفاده می‌کنیم. برای اینکه مخاطرات امنیتی را درنظر بگیریم ابتدا یک فایل temp ایجاد می‌کنیم و محتوای رمز شده را رمزگشایی کرده و در آن میریزیم و سپس با استفاده از ایدتور آن را باز کرده و تغییرات را در آن می‌دهیم در این صورت در صورتی که کاربر دسترسی w به یک فایل نداشته باشد در صورتی که آن را تغییر هم بدهد نمی‌تواند آن را سیو کنیم.

```python
def text_editor(path, client, server):
    temp_path = os.path.join(os.getcwd(), "temp")
    temp = open(temp_path, "w")
    path = server.encrypt_path(path)        erfiboy, 6 hours ago •
    if pathlib.Path(path).is_file():
        temp.write(client.decrypt_file(path, server))

    try:
        editor = os.environ["EDITOR"]
    except KeyError:
        if platform == "linux" or platform == "linux2":
            editor = "nano"
        elif platform == "win32":
            editor == "notepad"
        elif platform == "darwin":
            editor = "nano"

    temp.close()
    subprocess.call([editor, temp_path])
    temp = open(temp_path, "r")
    content = temp.read()
    client.encrypt_file(content, path, server)
    os.remove(temp_path)
```

۵ نمایش اطلاعات

برای نمایش اطلاعات از پرینت و کتابخانه termcolor استفاده کرده‌ایم تا بتوانیم پرینت‌های رنگی داشته باشیم و محیطی شبیه لینوکس داشته باشیم.

۶ ذخیره سازی امن اطلاعات در سرور

برای ذخیره‌سازی امن در سرور علاوه بر ارتباط امن مبتی بر session-key، برای هر فایل یک کلید ایجاد می‌کنیم سپس کلیدهارا در یک فایل جیسون نگه داری می‌کنیم. که کلید دیکشنری نام فایل و value ان برابر کلید استفاده شده برای رمز کردن آن فایل است و بعد از اینکه کاربر خارج شد کلیدهای او را در یک جیسون ذخیره می‌کنیم و وقتی که دوباره وارد شد آن کلیدها را برای او بارگزاری می‌کنیم تا بتواند به کلیدها دسترسی داشته باشد.

```
def dump_dict(self):
    with open(os.path.join(self.base_path, f"{self.username}_keys.json"), "w") as outfile:
        json.dump(self.map_keys, outfile)

def load_dict(self):
    with open(os.path.join(self.base_path, f"{self.username}_keys.json"), "r") as outfile:
        self.map_keys = json.load(outfile)
```

همانطور که در بالا نشان داده شده است پس از رمز کردن یک فایل ابتدا آن کلید را در یک دیکشنری با نام فایل ذخیره می‌کنیم سپس تابع save-file سرور را فراخانی می‌کنیم تا محتوای رمز شده را بگیرید و با رمز کردن نام و مسیر فایل آن را داخل مخزن فایل‌ها قرار دهد.تابع retrieve کاری دقیقا برعکس آن را انجام می‌دهد و نام یک فایل را می‌گیرید و محتوای آن را برمی‌گرداند و سپس کاربر با کلید متناظر آن فایل آن فایل را رمزگشایی کرده و استفاده می‌کند.

```
def encrypt_file(self, content, path, server):
    path = os.path.abspath(path)

    if path in self.map_keys.keys():
        key = self.map_keys[path].encode()
        fernet = Fernet(key)
        encrypted_content = fernet.encrypt(content.encode()).decode()
        save_file(encrypted_content, path, server, self)

    else:
        key = Fernet.generate_key()
        fernet = Fernet(key)
        self.map_keys[path] = key.decode()
        encrypted_content = fernet.encrypt(content.encode()).decode()

    save_file(encrypted_content, path, server, self)
```

```
def save_file(content, path, server, client):      erfiboy, 2 hours ago • final a
    mkdir(pathlib.Path(path).parent, server, client)
    with open(path, "w") as file:
        file.write(content)

    server.access_control.allow(client.username, "R", [str(path)])
    server.access_control.allow(client.username, "W", [str(path)])
    return True


def retrieve_file(path, server, client):
    if not server.access_control.is_allowed(client.username, "R", str(path)):
        return
    if os.path.exists(path):
        with open(path, "r") as file:
            content = file.read()
            return content
```

۷ اشتراک گذاری امن فایل

این قسمت به دلیل اینکه همگروهی بنده از زدن پروژه در یک روز مانده به تحویل پروژه خودداری کردند انجام نشده است. لازم به ذکر است تمام پروژه به صورت تک نفره زده شده است و این ناهماهنگی به دلیل این است که من دانشجوی دانشکده برق هستم و هیچ شناختی نسبت به همگروهی خود نداشتم و ایشان نیز تا روزهای پایانی کار خود را به تعویق انداختند و در نهایتا آن را انجام ندادند.

با توجه به توضیحات بالا این قسمت به صورت کامل انجام نشده است و فقط در کنترل دسترسی تابع های مورد نیاز تست و پیاده‌سازی شده اند. این توابع اجازه دسترسی به یک منبع را توسط شخص دیگری صادر می کنند اما می دانیم برای اینکه فایل توسط هر دو نفر قابل دیدن باشد باید با یک کلید که هر دو به آن دسترسی دارد رمز شده باشد به این منظور با استفاده از مکانیزم دیفی_هیلمن که در ابتدا برای رد و بدل کردن session-key استفاده کرده بودیم یک کلید مشترک به وسیله سرور بین آنها به اشتراک می گذاریم از آنجایی که سرور یک موجود معتمد است پس مقدار فرستاده شده توسط کاربران را تغییر نمی‌دهد و همچنین کنجکاو بودن آن به علت روش الگوریتم دیفی_هیلمن نمی‌تواند مشکل ساز شود و کلید به صورت امن جابه‌جا خواد شد.

```
def allow(self, role, action_type, resources=[]) -> None:
    if not (role or role in self._roles):
        return
    if not (action_type or action_type in self._action_types):
        return

    for resource in resources:
        if not (resource or resource in self._resources):
            continue

        self._allowed[role, action_type.upper(), str(resource)] = True

def deny(self, role, action_type, resources=[]) -> None:
    if not (role or role in self._roles):
        return
    if not (action_type or action_type in self._action_types):
        return

    for resource in resources:
        if not (resource or resource in self._resources):
            continue

        self._denied[role, action_type.upper(), str(resource)] = True
```

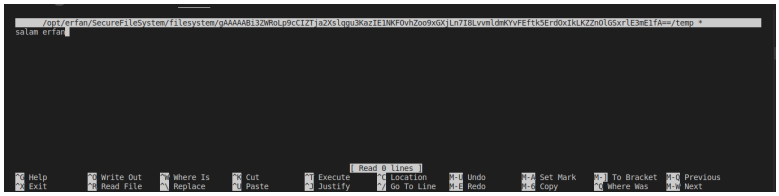
به وسیله این دوتابع می‌توان دسترسی یک فایل را به یک نقش داد و یا از او گرفت از آنجایی که ما برای هر کاربر یک نقش با نام کاربری او درست کرده این پس برای اینکار کافی است که دسترسی لازم را به نقش با نام کاربری مورد نظر بدهیم و سپس الگوریتم دفی_هیلمن را به وسیله سرور اجرا کنیم تا دو نفر به یک فایل رمز شده یکسان برسند.

۸ روش اجرا

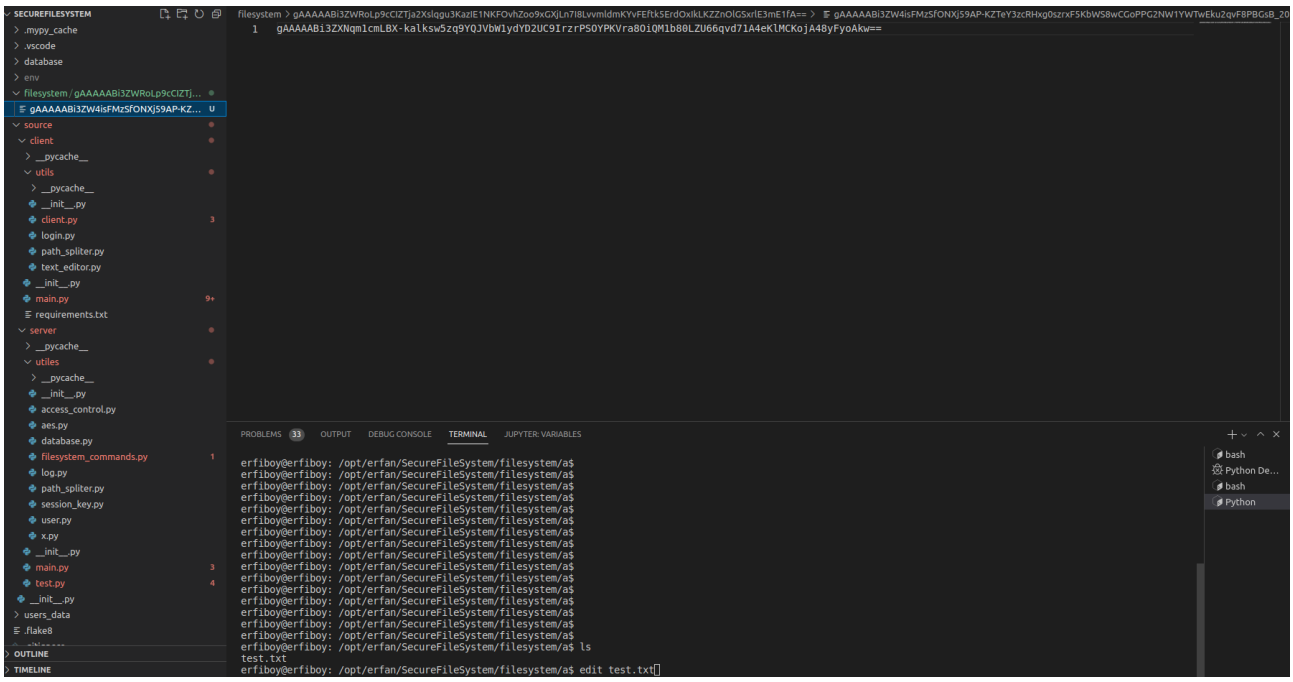
روش اجرا به این صورت است که ابتدا کتابخانه‌های موجود در فایل requirement.txt را نصب می‌کنیم و سپس فایل main.py را در client اجرا می‌کنیم. در ابتدا دو گزینه داریم یا باید وارد شویم یا در صورت نداشتن اکانت باید یک اکانت جدید بسازیم: همانطور که در زیر دیده می‌شود یک کاربر با نام خودم می‌سازم و همچنین پسورد را بدلیل امنیت نشان نمی‌دهیم تا طول آن نیز مشخص نشود و همواره پسوردها را به صورت hash نگه‌داری می‌کنیم.

همانطور که نشان داده شده است در ابتدا کاربر یک فولدر خالی مشاهده می کند حال می تواند به وسیله دستورات فایل سیستم تغییرات لازم را بدهد برای مثال من یک فولدر جدید ساخته ام و در آن وارد شده ام.

سپس یک فایل با نام test.txt ایجاد کرده ام و داخل آن salam erfan را نوشته ام و ذخیره کرده ام.



همانطور که در بالا سمت چپ دیده می شود تمام مسیر فایل‌ها رمز شده است. هچنین در صفحه ادیتور پیام erfan salam را مشاهده می کنیم که رمز شده است.



سپس از این کاربر خارج می‌شویم و سعی می‌کنیم با یک کاربر ثبت نام نشده وارد شویم و می‌بینیم که پیغام خطا برای ما ارسال می شود سپس یک کاربر دیگر به نام ali را ثبت نام می کنیم و مشاهده می‌کنیم که کاربر ali نمی‌تواند محتویات فایل ها و پوشه های erfiboy را مشاهده کند.

```
erfiboy@erfiboy: /opt/erfan/SecureFileSystem/filesystem/a$
erfiboy@erfiboy: /opt/erfan/SecureFileSystem/filesystem/a$
erfiboy@erfiboy: /opt/erfan/SecureFileSystem/filesystem/a$
erfiboy@erfiboy: /opt/erfan/SecureFileSystem/filesystem/a$ ls
test.txt
erfiboy@erfiboy: /opt/erfan/SecureFileSystem/filesystem/a$ exit
SecureFileSystem > [1] SignUp [2] Login: 2
Enter your user username: l
Enter your user password:
ERROR:root:User with this username doesn't existed.
l@l: /opt/erfan/SecureFileSystem/filesystem$ The username or password is wrong!
SecureFileSystem > [1] SignUp [2] Login: 1
Enter your user first name: ali
Enter your user last name: nosrati
Enter your user username: ali
Enter your user password:
ali@ali: /opt/erfan/SecureFileSystem/filesystem$ ls

ali@ali: /opt/erfan/SecureFileSystem/filesystem$
```

دوباره با کاربر erfiboy وارد می‌شویم و مشاهده می کنیم که اطلاعات وی دست نخوره باقی مانده است.

```
SecureFileSystem > [1] SignUp [2] Login: 1
Enter your user first name: ali
Enter your user last name: nosrati
Enter your user username: ali
Enter your user password:
ali@ali: /opt/erfan/SecureFileSystem/filesystem$ ls

ali@ali: /opt/erfan/SecureFileSystem/filesystem$ exit
SecureFileSystem > [1] SignUp [2] Login: 2
Enter your user username: erfiboy
Enter your user password:
ERROR:utiles.log:(1, 'erfan', 'nosrati', 'erfiboy', 'cc7e4412564ba8a761bd32ab4cc6086bac3c2c9e580367e0b0eb32a4316f9154')
erfiboy@erfiboy: /opt/erfan/SecureFileSystem/filesystem$ ls
a
erfiboy@erfiboy: /opt/erfan/SecureFileSystem/filesystem$
```