مستند B

درس مهندسی اینترنت

گزارش فنی: پیاده سازی ساختار Client-Server برای ارسال هشدار، مبتنی بر SMTP

محمد پهلوانیان

ترم مهر ۱۴۰۴ (۴۰۴۱)

مقدمه

در سامانه‌های نظارت تصویری خانگی، یکی از نیازهای مهم، امکان دریافت هشدارهای فوری در زمان بروز رخدادهایی مانند تشخیص حرکت، باز شدن درب یا قطع ارتباط دوربین است. در سناریوی واقعی، بسیاری از دستگاه‌های **DVR** و **NVR** از پروتکل **SMTP** برای ارسال اعلان‌های ایمیلی استفاده می‌کنند. هدف این پروژه، شبیه‌سازی همین رفتار واقعی اما در قالب یک سیستم ساده و قابل فهم مبتنی بر معماری فنی **Client– Server** و برنامه‌نویسی سوکتی است.

در این پروژه، یک کلاینت (شبیه‌ساز دستگاه **DVR**) روی شبکه، رخدادهای نظارتی را تولید و از طریق یک اتصال **TCP** به سرور ارسال می‌کند. سرور پس از دریافت پیام، آن را تحلیل کرده و با استفاده از یک ماژول **SMTP** ، هشدار متنی (به همراه تصویر در صورت وجود) را برای کاربر نهایی ارسال می‌کند. این ساختار، نمونه‌ای عملی از ارتباطات لایه کاربرد **(Application Layer)** در مهندسی اینترنت است.

۱. اهداف پروژه

اهداف اصلی:

۱. **درک عملی معماری Client–Server** و نحوه تبادل داده بر بستر TCP.
۲. **پیاده‌سازی ارسال ایمیل با SMTP** و آشنایی با رفتار واقعی سیستم‌های اعلان هشدار.
۳. **مدیریت تصاویر و داده‌ها** از طریق رمزنگاری Base64 و انتقال در قالب JSON .
۴. **ساخت یک سناریوی کامل** از دریافت رویداد تا ثبت، پردازش و ارسال هشدار.

خروجی‌های کلیدی پروژه:

- ماژول SMTP مستقل برای ارسال ایمیل.
- سرور چندنخی **(Multi-Threaded)** برای دریافت رویدادها.
- کلاینت تولیدکننده رویداد با ارسال اختیاری تصویر.
- قابلیت اجرا در حالت تست **(Debug SMTP)** یا حالت واقعی **(Gmail / SMTP Server)** .

۲. معماری کلی سیستم

ساختار کلی سیستم در چهار بخش تعریف شده است:

DVR Client → TCP Socket → Alert Server → SMTP → User Email Inbox

پهلوانیان محمد
@PahlavanianMohammad

اجزای اصلی:

- کلاینت **(client.py)**:
  - ارتباط **TCP** با سرور برقرار می‌کند.
  - رویدادها را در قالب **JSON** ارسال می‌کند.
  - توانایی ضمیمه کردن تصویر **(Base64)** را دارد.
- سرور **(server.py)**:
  - روی پورت **TCP** مشخصی شنود می‌کند.
  - برای هر اتصال، یک **Thread** جداگانه ایجاد می‌کند.
  - پیام **JSON** را خوانده، تحلیل کرده و با **SMTP** هشدار ارسال می‌کند.
- ماژول **SMTP (smtp_module.py)**:
  - ارسال ایمیل با پیکربندی از طریق متغیرهای محیطی.
  - پشتیبانی از **TLS**، **SSL** و حالت **Debug**.

۳. طراحی پروتکل پیام

برای جلوگیری از پیچیدگی و نیز قابلیت توسعه، قالب پیام‌ها **JSON** انتخاب شده است. هر پیام یک خط **JSON** است که شامل فیلدهای زیر می‌شود:

فیلد	توضیح
event	نوع رویداد مثلاً **MOTION_DETECTED**
camera	**نام یا شناسه دوربین**
timestamp	**زمان UTC رویداد**
image_filename	**نام فایل تصویر (اختیاری)**
image_b64	**محتوای تصویر به صورت Base64**

* استفاده از **Base64** باعث می‌شود تصویر بدون نیاز به ارسال فایل جداگانه، در قالب **JSON** قابل انتقال باشد.

۴. پیاده‌سازی سرور

سرور بر پایه **TCP** و با قابلیت **Multi-Thread** طراحی شده است. مراحل عملکردی سرور:

پهلوانیان محمد
@PahlavanianMohammad

۱. ایجاد سوکت و Bind روی پورت مشخص.
۲. Listen برای اتصالات ورودی.
۳. قبول اتصال جدید و ایجاد Thread مجزا.
۴. خواندن پیام‌ها از طریق Stream خطی.
۵. تحلیل JSON و استخراج اطلاعات رویداد.
۶. مدیریت تصویر Decode و ذخیره موقت.
۷. ارسال هشدار ایمیلی از طریق SMTP

* این مدل به سرور اجازه می‌دهد چندین کلاینت را به طور هم‌زمان مدیریت کند.

۵. پیاده‌سازی کلاینت

کلاینت برای سادگی، یک برنامه خط فرمان است که امکان ارسال رویدادهای زیر را فراهم می‌کند:

- ارسال یک پیام واحد
- ارسال تعداد نامحدود پیام با فاصله زمانی مشخص (حالت Loop)
- ارسال تصویر همراه رویداد

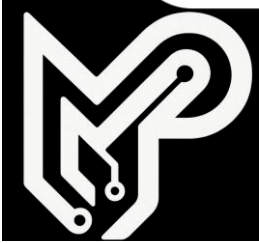
این ابزار مانند یک DVR واقعی عمل می‌کند که در لحظه تشخیص حرکت، اطلاعات را به مرکز پردازش ارسال می‌نماید.

۶. طراحی و پیاده‌سازی SMTP

ویژگی‌ها:

- ارسال ایمیل با TLS یا SSL
- پشتیبانی از Gmail App Password
- امکان اجرا با SMTP Debug Server بدون ارسال واقعی ایمیل
- امکان ضمیمه‌کردن تصویر

در صورت عدم تنظیم پیکربندی SMTP ، ماژول به صورت خودکار از سرور محلی Debug روی پورت ۱۰۲۵ استفاده می‌کند. این قابلیت، فرآیند تست را بسیار ساده می‌کند.



پهلوانیان محمد
@PahlavanianMohammad

۷. نحوه اجرای سیستم

اجرای حالت تست (بدون ارسال ایمیل واقعی)

۱. اجرای سرور SMTP محلی:

```
python -m smtpd -c DebuggingServer -n localhost:1025
```

۲. اجرای سرور اصلی:

```
python server.py
```

۳. ارسال رویداد:

```
python client.py --event MOTION_DETECTED --camera cam1
```

اجرای حالت واقعی (ارسال ایمیل)

تنظیم متغیرهای محیطی:

```
export SMTP_HOST="smtp.gmail.com"
export SMTP_PORT="465"
export SMTP_USER="your@gmail.com"
export SMTP_PASS="app_password"
export ALERT_RECIPIENT="someone@example.com"
```

سپس:

```
python server.py
python client.py --event DOOR_OPENED --camera cam2
```

۸. تحلیل امنیتی و ملاحظات عملیاتی

- اطلاعات SMTP هرگز نباید داخل کد قرار گیرد؛ استفاده از متغیر محیطی روش ایمن‌تری است.
- ارتباط TCP در این نسخه رمزگذاری نشده است و برای محیط واقعی باید روی TLS یا SSH Tunneling قرار گیرد.
- امکان افزودن توکن امنیتی برای احراز هویت کلاینت‌ها وجود دارد.

پهلوانیان محمد

@PahlavanianMohammad

- ارسال تصویر باعث افزایش حجم پیام می‌شود؛ در پروژه‌های دنیای واقعی معمولاً از FTP/HTTPS برای انتقال فایل‌ها استفاده می‌شود و فقط لینک آن در پیام ارسال می‌گردد.

جمع بندی

در این پروژه برگرفته از یک تجربه واقعی، تمام مفاهیم اصلی مهندسی اینترنت — شامل ارتباط TCP، ساختار پیام JSON، معماری چندنخی، و پروتکل SMTP — به صورت عملی پیاده‌سازی شده‌اند. این ساختار نه‌تنها قابل توسعه است، بلکه بسیار نزدیک به فرآیندهای واقعی در تجهیزات نظارت خانگی مورد استفاده قرار می‌گیرد.

تجربه پیاده‌سازی عملی این پروژه، درک عمیق‌تری از لایه کاربرد، مدیریت ارتباطات شبکه و سازوکار ارسال هشدار در سیستم‌های نظارتی فراهم کرد و توانست پایه‌ای برای درک هرچه بهتر مفاهیم تئوری کلاس درس باشد.

کد و پروژه در گیت هاب

لینک دسترسی:

https://github.com/PahlavanianAzadUni/smtp-alert-engine/

پهلوانیان محمد
@PahlavanianMohammad