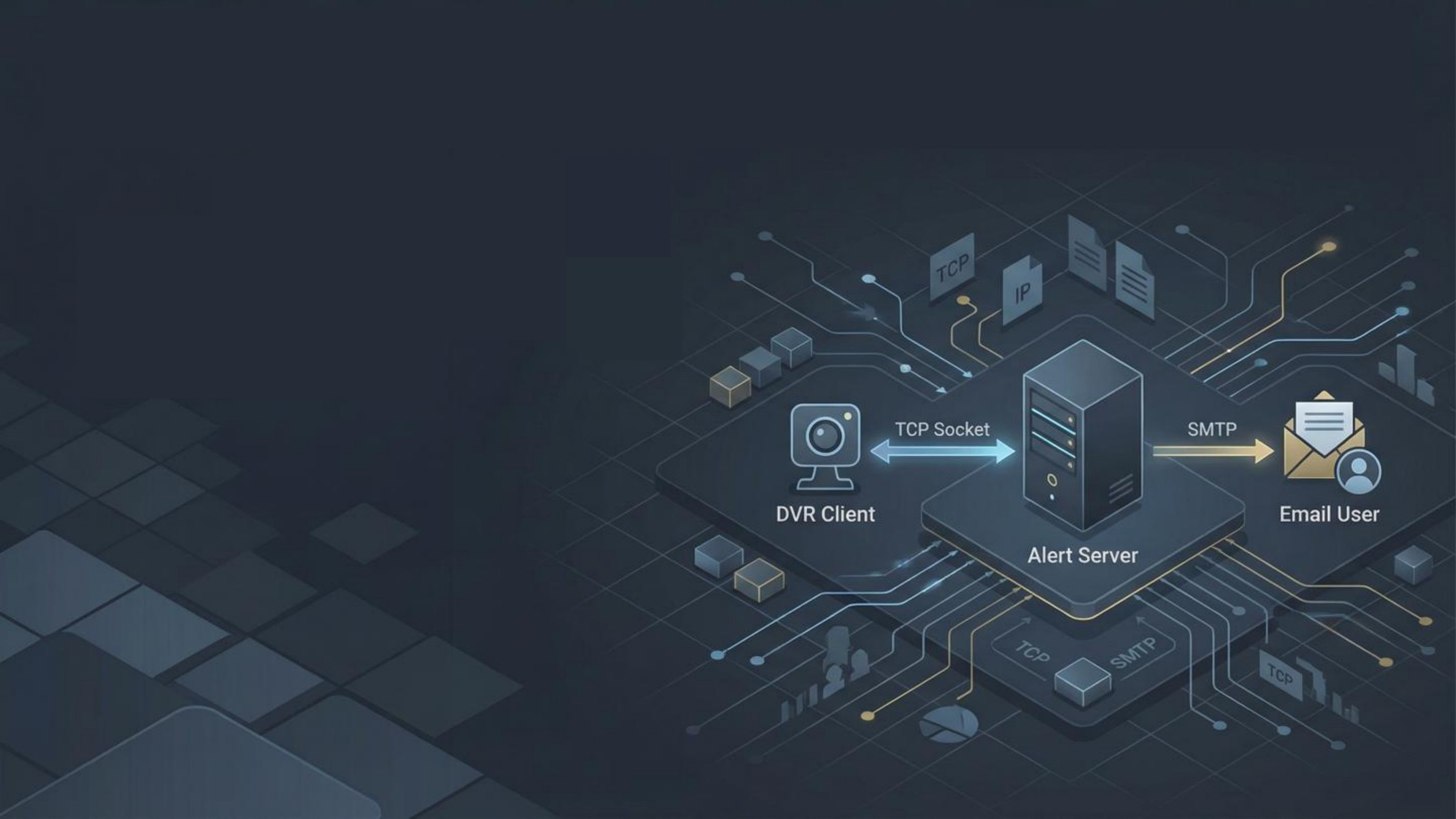
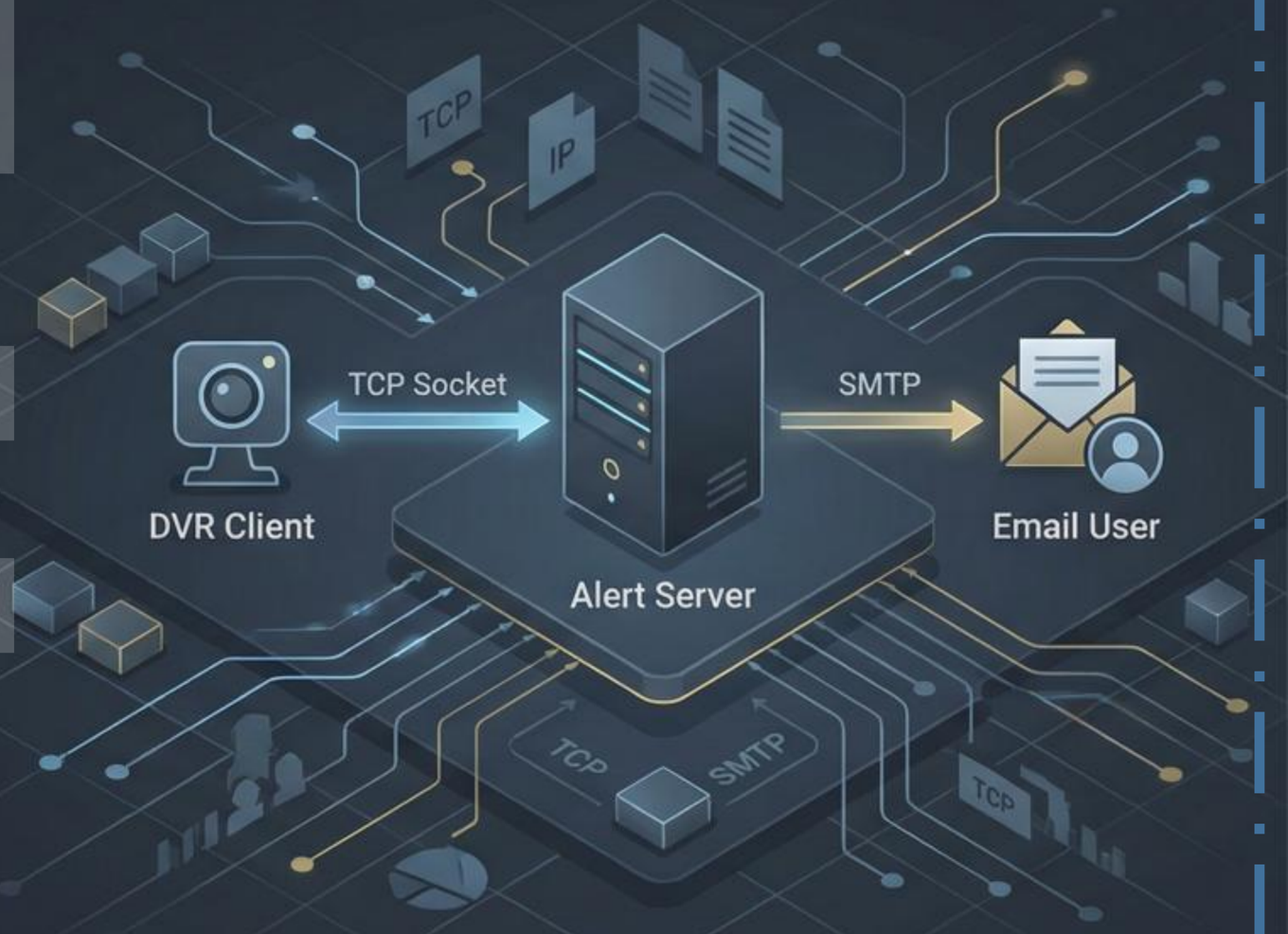

TCP
IP
TCP Socket
SMTP
DVR Client
Alert Server
Email User
TCP
SMTP
TCP

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

ارائه دهنده: محمد پهلوانیان

درس: مهندسی اینترنت

ترم مهرماه 1404 (4041)



پیاده‌سازی ارتباط TCP قابل‌اعتماد

- تضمین تحویل داده
- حفظ ترتیب
- کنترل خطا

درک عملی معماری Client–Server

تبادل داده دوطرفه

شبیه‌سازی سناریوی واقعی هشدار امنیتی

تشخیص حرکت، باز شدن درب

استفاده از SMTP در لایه کاربرد

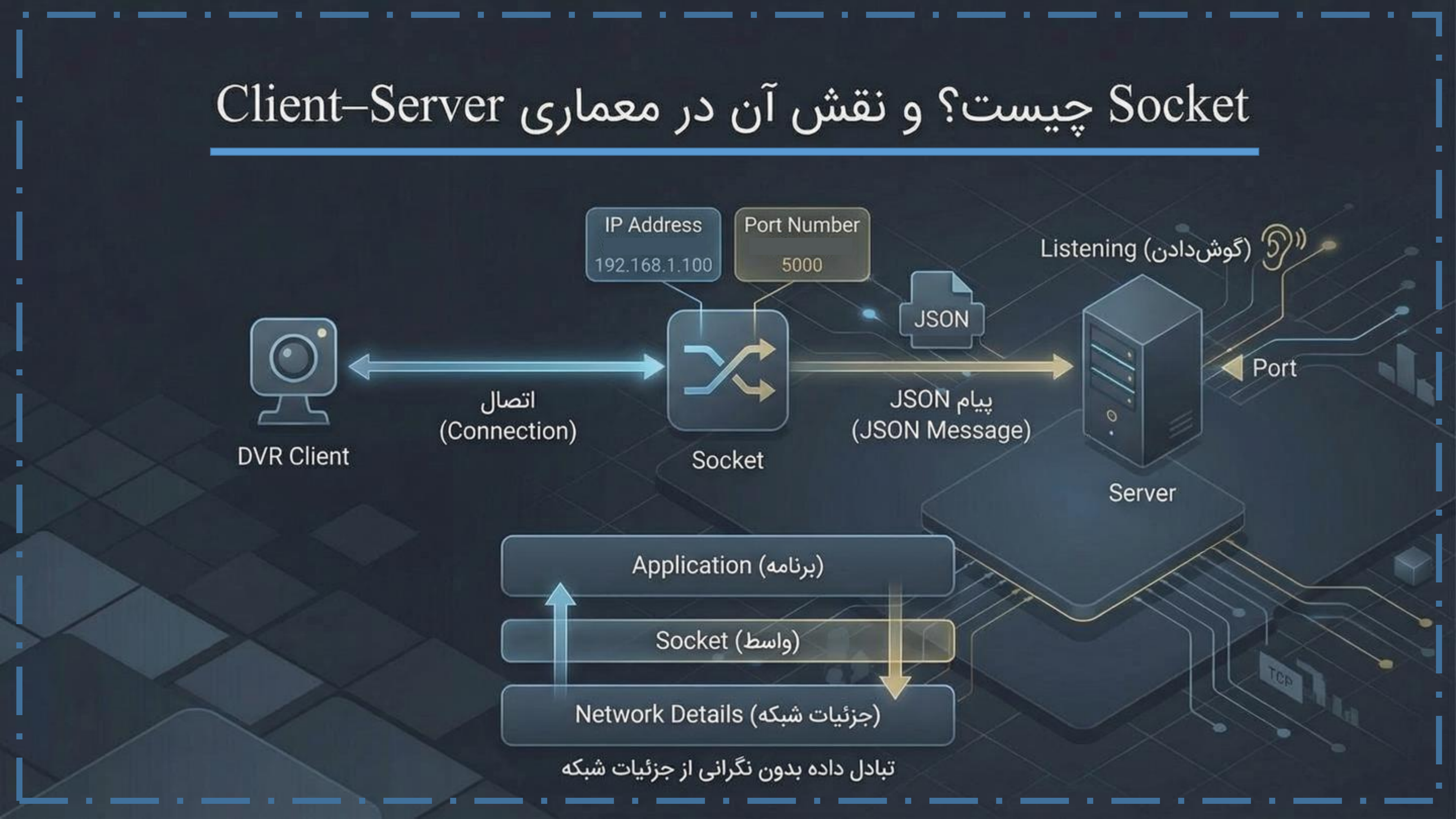
ارسال ایمیل هشدار با TLS/SSL

پیاده‌سازی کامل با کد قابل اجرا و مستندسازی جهت توسعه و تست

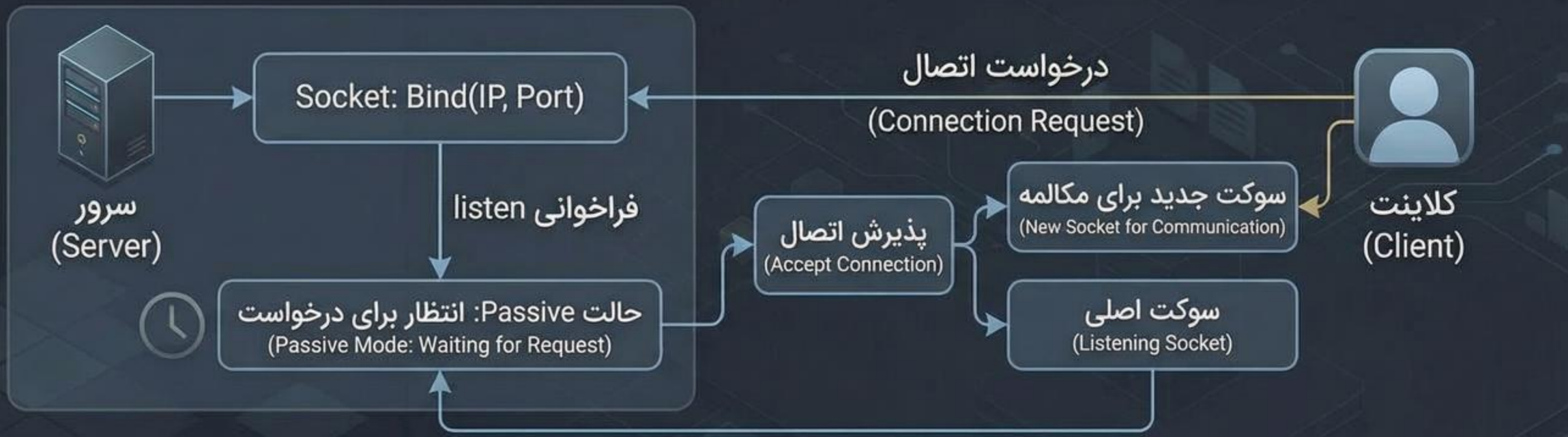
پروتکل TCP: تضمین ارسال قابل‌اعتماد داده‌ها

این ویژگی‌ها TCP را به انتخابی طبیعی برای پروتکل‌هایی مانند SMTP تبدیل می‌کند که به ارسال بدون خطا نیاز دارند.

Socket چیست؟ و نقش آن در معماری Client–Server



مفهوم Listening در سرور



- ورود به حالت Passive و انتظار برای درخواست کلاینت
- بلاک شدن برنامه تا اتصال کلاینت، بدون اشغال منابع سیستم
- ایجاد سوکت اختصاصی برای هر کلاینت و امکان سرویس‌دهی هم‌زمان

پیاده‌سازی ساختار Client–Server برای ارسال هشدار مبتنی بر SMTP

این زنجیره سه‌لایه‌ای یعنی شبکه، کاربرد و ارائه خدمت، دقیقاً الگویی است که در سیستم‌های واقعی DVR/NVR مشاهده می‌شود.

ساختار ماژولار سیستم

client.py:
تولید و ارسال رویداد

تولید رویداد
(Event Generation)
ارسال از طریق TCP
(Send via TCP)

server.py:
مدیریت و پردازش پیام

پذیرش چند کلاینت
(Threading)
تجزیه پیام
(Message Parsing)

smtp_module.py:
ارسال ایمیل و اعتبارسنجی

ارتباط با سرور ایمیل
(Email Server Connection)
اعتبارسنجی TLS
(TLS Authentication)
ارسال واقعی یا دیباگ
(Actual/Debug Send)

تفکیک ماژول‌ها: امکان تست واحد و توسعه مستقل

طراحی پیام JSON

این طراحی امکان افزودن سناریوهای جدید مثل تشخیص حرکت یا قطع دوربین را می‌دهد و با جاسازی تصویر به صورت Base64 در همان JSON، نیاز به پروتکل اضافه برای فایل را از بین برده و ارتباط را در یک TCP واحد در واحد حفظ می‌کند.

پیاده‌سازی کلاینت (Client Implementation)

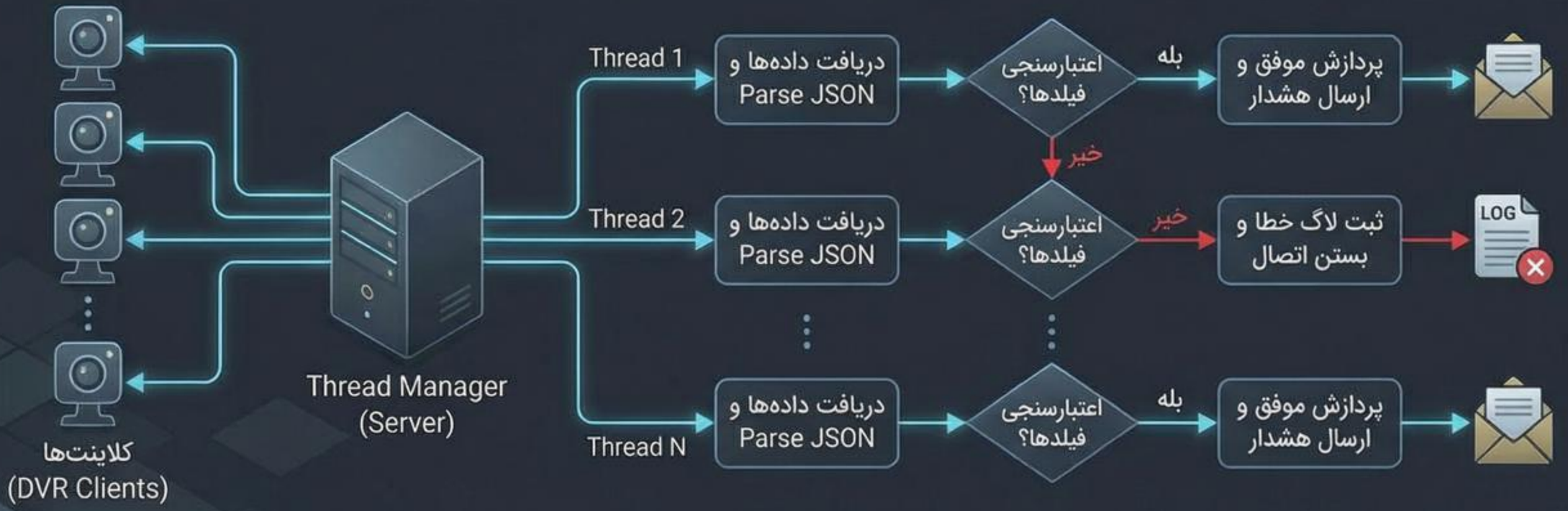
۱. **ایجاد اتصال:** کلاینت با ایجاد TCP Socket به آدرس سرور وصل می‌شود.

۲. **ارسال داده:** پیام JSON و تصویر (Base64) در یک ارسال واحد ارسال می‌شوند.

۴. **قطع اتصال:** در پایان، اتصال به‌درستی بسته می‌شود تا منابع آزاد شوند.

۳. **تکرار:** حلقه ارسال مداوم امکان تست تکراری و شبیه‌سازی هشدارهای متوالی را فراهم می‌کند.

مکانیزم پردازش هم‌زمان و اعتبارسنجی در سرور



مقیاس‌پذیری بالا: پشتیبانی از صدها کلاینت هم‌زمان در محیط واقعی.

اعتبارسنجی دقیق داده‌ها: تضمین صحت و امنیت ساختار JSON.

Thread مجزا برای هر کلاینت: جلوگیری از بلوکه شدن و افزایش کارایی.

ماژول SMTP: جزئیات پیاده‌سازی

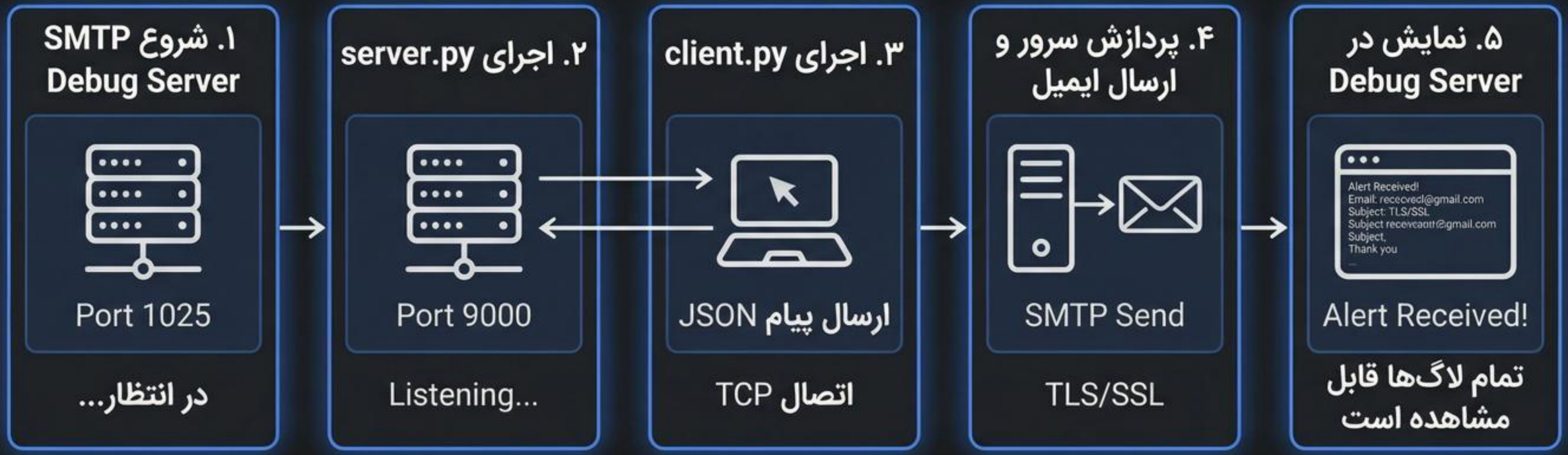
اتصال و امنیت

- استفاده از کتابخانه استاندارد Built-in
- اتصال به Gmail (پورت ۴۶۵) با STARTTLS
- رمز عبور از متغیر محیطی (ENV)

حالت دیباگ و تست

- حالت Debug فعال (بدون ارسال واقعی)
- لاگ کامل هدر و بدنه پیام در سرور محلی

اجرای سیستم (Demo)



کل زنجیره در کمتر از پنج ثانیه تکمیل می‌شود

بررسی کد

سناریوهای تست شده

لیست سناریوها

- تشخیص حرکت
- باز شدن درب
- قطع ارتباط دوربین
- ارسال مکرر هشدار
- ارسال تصویر Base64

نتایج و عملکرد سیستم

ملاحظات امنیتی

حریم خصوصی و پایداری سیستم

- جلوگیری از نوشتن اطلاعات حساس در لاگ‌ها.
- مقاوم‌سازی سرور در برابر حملات ReDoS و بمب ایمیل.

رمزنگاری و احراز هویت

- فعال‌سازی TLS در لایه TCP برای رمزنگاری ارتباط کلاینت-سرور.
- اعتبارسنجی کلاینت با استفاده از توکن یا گواهی (Certificate).

مدیریت ایمن رمز عبور

- استفاده از متغیرهای محیطی (Environment Variables) به جای کد سخت‌شده.

جمع‌بندی

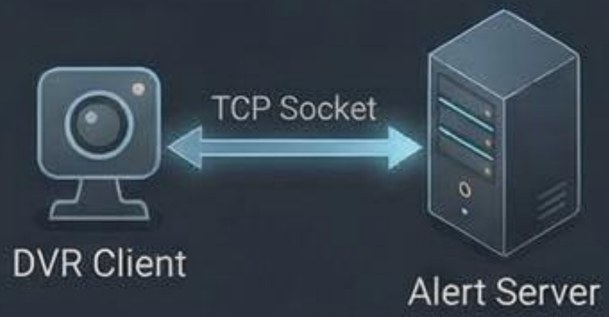
شبیه‌سازی واقعی

- شبیه‌سازی نزدیک به سیستم‌های واقعی
- کد قابل توسعه

پروتکل‌ها و ابزارها

- استفاده واقعی از TCP, Socket, SMTP
- لایه انتقال و کاربرد

معماری Client–Server



- پیاده‌سازی عملی
- درک مفاهیم شبکه

با این پیاده سازی عملی، معماری **Client-Server** را در دنیای واقعی بررسی کردیم.
اکنون می توان درک کامل و درستی از این معماری داشت و در ادامه مسیر با افزودن رمزنگاری و پایگاه داده، این پروژه را به یک محصول در دنیای واقعی تبدیل کرد.

TCP/IP

لایه	در پروژه
Application Layer	SMTP, JSON, Email alerts
Transport Layer	TCP sockets
Network Layer	IP
Data Representation	Base64 encoding

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