# S.C.A.R.

## System for Countering Automated Reconnaissance

Falito Eriano N., Luklu Miranda, Reiza Gerrard R.

Politeknik Siber dan Sandi Negara

Semester Gasal 2025/2026

# Latar Belakang & Masalah

- **Eskalasi Serangan Web**: Peningkatan serangan otomatis terhadap server HTTP (SQLi, Recon).
- **Pertahanan Konvensional**: Firewall bersifat statis dan pasif (log-only).
- **Kebutuhan Sistem Aktif**: Server yang tidak hanya mendeteksi, tapi juga merespons balik penyerang secara cerdas.

## Solusi S.C.A.R.

Menggabungkan **Active Defense** (Tarpit) dengan **Multi-Layer AI** untuk menjebak dan menguras sumber daya penyerang.

# Eksploitasi Network Stack & Protokol

S.C.A.R. memanfaatkan berbagai lapisan dalam sistem pemrograman jaringan Python:

- **High-Level Handling**: Modul `http.server` (Custom Handler untuk analisis request).
- **Server Concurrency**: Modul `socketserver` (`ThreadingTCPServer`).
- **Low-Level Socket**: Penulisan data langsung ke *raw socket buffer* (`self.wfile`) saat Tarpit.
- **Transport Layer**: Manipulasi status koneksi TCP (Keep-Alive & Chunked Transfer).

# Konsep: Active Defense & Tarpit

**Paradigma Pertahanan Aktif**

- Mengubah server menjadi umpan (*Honeypot*).
- Manipulasi protokol untuk tujuan defensif.

**Mekanisme Tarpit**

- *TCP State Exhaustion*.
- *HTTP Chunked Stream Manipulation*.
- Mengirim garbage data secara lambat (*Slow Drip*).

*"Menjebak penyerang dalam koneksi tak berujung."*

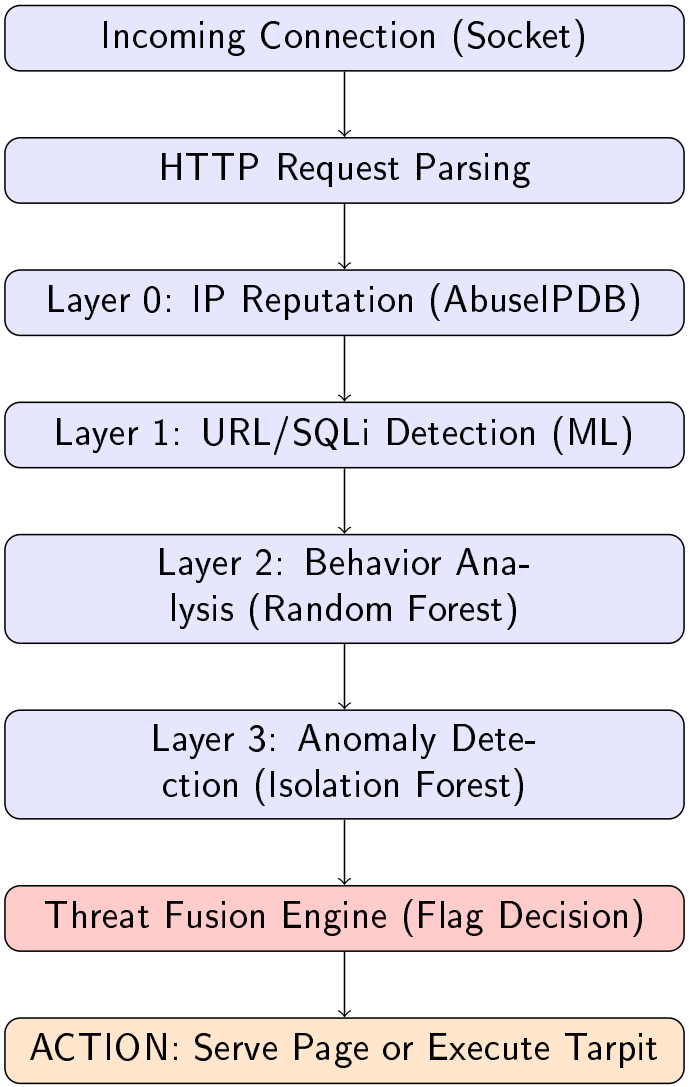
# Analisis Perbandingan: Blocking vs Tarpitting

| Fitur | Blocking (Firewall) | Tarpitting (SCAR) |
|---|---|---|
| *Awareness* | Penyerang langsung tahu | Penyerang "tertipu" (HTTP 200) |
| *Cost* | Nol bagi penyerang | Tinggi (Menghabiskan CPU/Thread) |
| *Inteligensi* | Data minimal | Pengumpulan data berkelanjutan |
| *Mitigasi* | Ganti IP/Target | Tersangkut di satu target |

## Keunggulan SCAR

Menguras *resource* bot pemindai hingga 99% lebih efektif dibanding blokade statis.

# Arsitektur Multi-Layer Threat Fusion



# Threat Fusion: Hard vs Soft Flags

Sistem menggunakan kombinasi deteksi untuk meminimalkan *False Positive*:

- **Hard Flag**: Serangan eksplisit (SQLi terdeteksi URL model). $\rightarrow$ **Tarpit Langsung**.
- **Soft Flag**: Indikasi mencurigakan (Anomali atau Reputasi buruk). $\rightarrow$ **Konsensus**: Butuh 2 Soft Flag untuk memicu Tarpit.

## Hasil

Request normal dari pengguna asli dipastikan lolos (Zero False Positive), sementara bot berbahaya terjebak.

# Core Logic: Threat Fusion Decision

Inti dari sistem pengambilan keputusan (Simplifikasi):

```
def is_threat(results):
    hard_flags = count(r for r in results if r.type == 'HARD')
    soft_flags = count(r for r in results if r.type == 'SOFT')

    if hard_flags >= 1: return True  # SQLi/Recon terdeteksi
    if soft_flags >= 2: return True  # Konsensus anomali & reputasi
    return False
```

*Logic ini memastikan akurasi tinggi tanpa memblokir trafik legitimate.*

# Detail Teknis: Tarpit Execution

Pemanfaatan **Transfer-Encoding: chunked** (RFC 2616) pada level socket TCP:

```
SERVER << HTTP/1.1 200 OK
SERVER << Transfer-Encoding: chunked
SERVER << X-Trap-ID: a3f8c1... [+0s]
[Jeda 5 detik]
SERVER << X-Trap-ID: 7bc2e5... [+5s]
... berulang hingga penyerang timeout ...
```

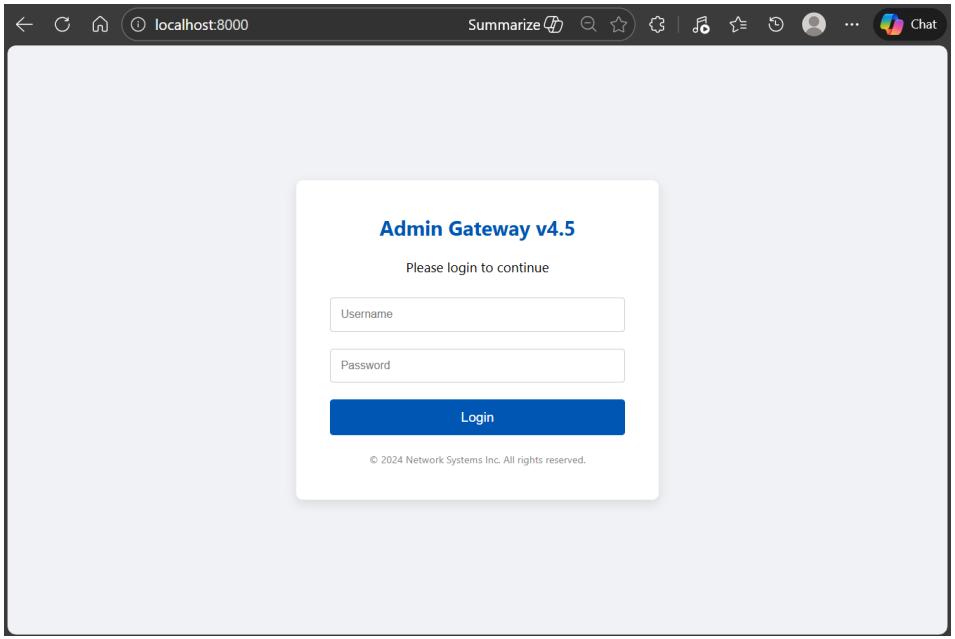
- **Socket-Level Write**: Data ditulis langsung ke socket.
- **Concurrency**: Menggunakan *threading* agar satu tarpit tidak memblokir klien lain.

# Manajemen Concurrency & Threading

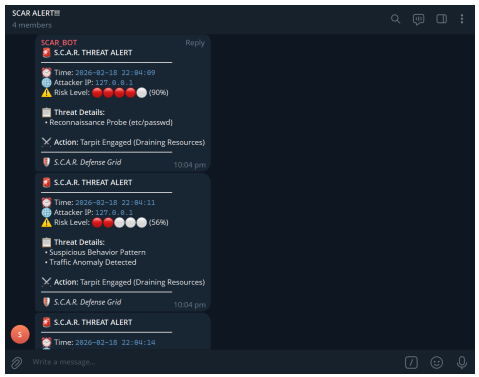
Salah satu tantangan terbesar pemrograman jaringan adalah *Scalability*:

- **ThreadingTCPServer**: Menciptakan satu *thread* per koneksi baru.
- **Daemon Threads**: S.C.A.R. memastikan thread serangan dapat dibersihkan secara otomatis saat server utama berhenti.
- **Thread Safety**: Penggunaan `threading.Lock()` untuk akses cache reputasi IP secara bersamaan.
- **Isolation**: Tarpit yang berjalan lambat tidak akan menggunakan CPU secara intensif, hanya memakan *socket descriptor*.

# Implementasi & Bukti Visual




Gambar: **Halaman Honeypot**




Gambar: **Notifikasi Alert Telegram**

- Arsitektur **Server & Client Simultan**.
- Notifikasi *Real-Time* via Telegram Bot API.

# Hasil Pengujian & Analisis

| Skenario | Jenis Serangan | Hasil | Aksi Sistem |
|---|---|---|---|
| GET / | Normal | Lolos | HTTP 200 OK |
| Recon (../) | Path Traversal | Tarpit | Garbage Stream |
| SQLi payload | SQL Injection | Tarpit | Garbage Stream |
| Unusual Pattern | Anomali | Tarpit | Garbage Stream |

**Analisis Efektivitas:**

- **Detection Rate**: 100% terhadap skenario serangan.
- **Tarpit Duration**: Berhasil menahan bot selama 10–30 detik per sesi.
- **Performance**: Latensi analisis AI $<1$ detik.

# Kesimpulan

1. **Inovasi Pertahanan**: S.C.A.R. berhasil mengubah paradigma dari pasif ke aktif menggunakan manipulasi protokol HTTP.
2. **Cerdas & Akurat**: Integrasi *Threat Fusion Engine* memastikan deteksi akurat dengan nol *False Positive*.
3. **Efisien**: Penggunaan *multi-threading* menjamin sistem tetap stabil dalam melayani pengguna normal meskipun sedang menjebak penyerang.

# Terima Kasih!