TOR Scraper Görev Tanımı Raporu

1. Proje Amacı

Siber tehdit aktörleri,izlerini kaybettirmek için Tor ağını kullanmaktadır. Tekil analizler manuel yapılabilse de, yüzlerce .onion adresini (sızıntı siteleri, forumlar, marketler) düzenli olarak taramak insan gücüyle imkansızdır.

Bu projenin amacı; öğrencilerin **Go (Golang)** dilini kullanarak, **toplu hedef listesini (yaml)** işleyebilen, trafiği Tor ağı üzerinden anonim olarak yönlendiren ve elde edilen istihbaratı raporlayan bir **otomasyon aracı** geliştirmeleridir. Bu proje, CTI süreçlerindeki "Collection" (Toplama) ve "Automation" (Otomasyon) yetkinliklerini kazandırmayı hedefler.

2. Proje Tanıtımı ve İşleyişi

Öğrencilerden beklenen, yerel Torservisi(SOCKS5 Proxy) üzerinden trafiği yönlendirerek, verilen bir dosyadaki tüm .onion adreslerini sırayla (veya eşzamanlı) ziyaret edip veri toplayan bir Go uygulaması geliştirmeleridir.

Proje şu 4 ana modülden oluşmalıdır:

1. **Dosya Okuma Modülü (Input Handler):**
   1.1. Program, komut satırından verilen bir metin dosyasını (örn: targets.yaml) okumalıdır.
   1.2. Dosya içindeki her bir satır (URL), temizlenerek (whitespace trimming) işleme alınmalıdır.
2. **Tor Proxy Yönetimi (Go Proxy Client):**
   2.1. Go'nun net/http kütüphanesi, varsayılan internet çıkışı yerine yerel 127.0.0.1:9050/9150 SOCKS5 proxy'sine yönlendirilmelidir.
   2.2. **Zorunluluk:** IP sızıntısını önlemek için özel bir http.Transport ve http.Client yapılandırılmalıdır.
3. **İstek ve Hata Yönetimi:**
   3.1. Listede çalışmayan veya kapanmış (dead) onion siteleri programı durdurmamalıdır. Program hatayı loglayıp bir sonraki URL'e geçmelidir.
4. **Veri Kayıt (Output Writer):**
   4.1. Her başarılı istekten dönen HTML verisi, URL adına veya tarih damgasına göre ayrı dosyalar halinde (veya yapılandırılmış tek bir JSON dosyasında) saklanmalıdır.

3. Beklenen Çıktılar

Buprojetamamlandığındaaşağıdaki somut çıktılar elde edilecektir:

- **Otomatize Tarama Aracı:** Yüzlerce linki tek komutla tarayabilen derlenmiş bir Go binary dosyası (exe veya linux binary).
- **Toplu Veri Seti:** targets.yaml içindeki erişilebilir tüm sitelerden çekilmiş screenshot verileri.
- **Durum Raporu:** Hangi URL'lerin aktif, hangilerinin pasif olduğunu gösteren bir özet log (örn: scan_report.log).

4. Kullanılacak Dil ve Teknolojiler

Projedeperformansveconcurrencyavantajlarınedeniyle**sadece Go** dili kullanılacaktır.

- **Programlama Dili:** Go (Golang)
- **Kritik Kütüphaneler:**
  - net/http: HTTP istekleri için.
  - golang.org/x/net/proxy: SOCKS5 proxy desteği için (standart kütüphane dışı tek izin verilen paket).
  - os, bufio: Dosya okuma ve yazma işlemleri için.
- **Ağ Altyapısı:** Tor Service (Arka planda çalışır durumda).

5. Ekran Görüntüleri ve Örnek Kullanım

Rapordaprojeninçalıştığınıkanıtlayan aşağıdaki ekran görüntüleriyeralmalıdır:

1. **Girdi Dosyası:** İçinde 5-10 adet .onion adresi bulunan targets.yaml dosyasının içeriği.
2. **Tarama Süreci:** Program çalışırken terminalde akan loglar.
   - *Örnek:* [INFO] Scanning: http://hss3d...onion -> SUCCESS
   - *Örnek:* [ERR] Scanning: http://badurl...onion -> TIMEOUT
3. **IP Kontrolü:** Programın istek atarken Tor IP'si kullandığını gösteren bir doğrulama (örn: check.torproject.org çıktısı).
4. **Sonuç Dosyaları:** Tarama bittikten sonra oluşan klasördeki veri dosyaları.

Örnek Kullanım Komutu:

go run main.go

Eğitmen’in Seyir Defteri:

> *"Godili(Golang),modern bulut ve ağ araçlarının dilidir. Bu projede Python yerine Go kullanmamızın sebebi, ileride binlerce siteyi aynı anda taramak istediğinizde Go'nun 'Goroutines' yapısının size sağlayacağı performansı şimdiden hissetmenizdir. Bu ödevde basit bir döngü kullanabilirsiniz, ancak meraklıları 'goroutine' ile taramayı hızlandırmayı deneyebilir!"*