**🧠 1. Smuggler Nedir?**

**Smuggler**, HTTP Request Smuggling (HRS) güvenlik açıklarını tespit etmek için geliştirilmiş, Python tabanlı bir sızma testi aracıdır.

* HTTP başlıklarının yanlış yorumlanmasından kaynaklanan zafiyetleri bulur.
* Proxy, CDN veya load balancer arkasındaki sistemleri test edebilir.
* Otomatik payload üretme ve test etme yeteneğine sahiptir.

📌 Özellikle **CL.TE**, **TE.CL** ve diğer request parsing uyumsuzluklarını hedef alır.

**🎯 2. Ne İşe Yarar?**

| **Özellik** | **Açıklama** |
| --- | --- |
| 🔍 HTTP Request Smuggling taraması | CL.TE, TE.CL ve hibrit zafiyet tespiti |
| 🛠️ Otomatik payload üretimi | Farklı HTTP başlık kombinasyonları |
| 📡 Proxy/CDN testleri | Arka uç (backend) ile ön uç (frontend) iletişim hataları |
| 🧪 Zafiyet sömürme | Yetkisiz erişim, cache poisoning, credential theft |

**⚙️ 3. Kurulum (Kali / Linux)**

**✅ 1. GitHub’dan indir:**

metin, ekran görüntüsü, yazı tipi içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

✅ 2. Python bağımlılıklarını yükle:

metin, ekran görüntüsü, yazı tipi, grafik içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

**🚀 4. Kullanım Komutları**

**✅ 1. Tek hedef taraması:**

python3 smuggler.py -u https://example.com

**✅ 2. Liste halinde hedef taraması:**

python3 smuggler.py -l targets.txt

**✅ 3. Belirli test türünü seçme (örnek: CL.TE):**

python3 smuggler.py -u https://example.com -m CL.TE

**✅ 4. Çıktıları dosyaya kaydetme:**

python3 smuggler.py -u https://example.com -o results.txt

**📋 5. Örnek Terminal Çıktısı**

metin, ekran görüntüsü, yazı tipi içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

**🧠 6. HTTP Request Smuggling Nedir?**

HTTP Request Smuggling, bir **ön uç sunucu (proxy, load balancer)** ile **arka uç sunucu** arasındaki HTTP istek yorumlama farklarını istismar ederek, ek isteklerin “smuggle” edilmesi tekniğidir.

* CL.TE → Content-Length & Transfer-Encoding uyumsuzluğu
* TE.CL → Transfer-Encoding & Content-Length uyumsuzluğu
* Yetkisiz işlemler, cache poisoning, WAF atlatma

**🧪 7. Kullanım Senaryoları**

| **Senaryo** | **Açıklama** |
| --- | --- |
| 🛡️ Web güvenlik testi | HTTP Request Smuggling tespiti |
| 🚀 WAF/Proxy bypass | Filtre atlatma senaryoları |
| 🧠 Red Team keşif | Backend’e doğrudan erişim sağlama |
| 🔍 Cache poisoning | Yanıltıcı veri cache’e yükleme |

**🎯 8. Öğrenim Kazanımları**

* HTTP başlıklarının nasıl çalıştığını ve parsing farklarını öğrenme
* HTTP Request Smuggling zafiyet türlerini tanıma
* Python tabanlı güvenlik testi aracını kullanma
* Web güvenliği testlerinde proxy/CDN yapılarını analiz etme
* Red Team bakış açısıyla backend erişimi sağlama