THREAT INTELLIGENCE AGGREGATOR – PROJE RAPORU

22290082
Tuğrul Özgün

1. Giriş

Bu proje, IP adresleri, domainler ve URL'ler üzerinde tehdit istihbaratı (Threat Intelligence) analizi yapan web tabanlı bir uygulamadır. Sistem; VirusTotal, AbuseIPDB ve Shodan gibi dış tehdit istihbarat servislerinden veri toplayarak birleştirir, kullanıcının risk seviyesini incelemesini sağlar ve sonuçları MongoDB'ye kaydeder.

2. Amaç

Bu uygulamanın temel amacı:

- Kullanıcının tek bir yerden çoklu kaynaklı tehdit analizi yapabilmesini sağlamak
- IP/domain/URL sorgularını merkezi bir API üzerinden yürütmek
- Farklı istihbarat servislerinin sonuçlarını yorumlamak ve risk puanı hesaplamak
- Oluşturulan raporları MongoDB üzerinde saklamak ve listelemek
- Basit, responsive bir web arayüzü üzerinden analizi kolaylaştırmak

Uygulama, temel bir Threat Intelligence Aggregator olarak tasarlanmıştır.

3. Sistem Mimarisi

Proje üç ana bileşenden oluşur:

3.1 Backend API (ASP.NET Core)

İki önemli controller içerir:

a) ThreatIntelController

Görevleri:

- `/api/ThreatIntel/lookup` → IP/Domain analiz eder
- `/api/ThreatIntel/save` → Analiz raporunu MongoDB'ye kaydeder

- `/api/ThreatIntel/saved` → Kayıtlı raporları listeler

Bu controller, aşağıdaki servislerle entegre çalışır:

- **VirusTotal API**
- **AbuseIPDB API**
- **Shodan API**

Toplanan tüm veriler harmanlanır ve tek bir JSON sonuç döndürülür.

b) AnalysisController
Görevleri:

- API'lerden gelen sonuçları yorumlamak
- Risk skoru hesaplamak
- Analiz modelini frontend için normalize etmek

3.2 MongoDB (Kalıcı Depolama)

MongoService.cs yapısı:

- `SaveReport()` → Bir ThreatReport nesnesi ekler
- `GetReports()` → Tüm raporları tarih sırasına göre getirir

Koleksiyon: ThreatReports
Veriler UTC timestamp ile saklanır.

3.3 Frontend (index.cshtml)

Tek sayfalı bir arayüzden oluşur:

- `/api/ThreatIntel/lookup` endpointi ile anlık analiz
- Sonuçların card formatında gösterimi
- Kaydetme butonu
- Kayıtlı raporların listesi

JS tarafında:

- Fetch API ile backend’e istek
- Sonuç gösterme
- Risk badge (Low/Medium/High)
- JSON pretty print
- localStorage fallback sistemi

4. Veri Modeli: ThreatReport.cs

Aşağıdaki alanlar bulunur:

Alan	Açıklama
`Id`	MongoDB ObjectId
`Query`	Analiz edilen IP/domain
`ResolvedIP`	DNS çözümlemesi sonucu
`VirusTotal`	Raw JSON
`AbuseIPDB`	Raw JSON
`Shodan`	Raw JSON
`OverallRisk`	Low, Medium, High
`Timestamp`	UTC zamanı

JSON’un JsonElement olarak tutulması esneklik sağlar.

5. Backend İş Akışı

1. Kullanıcı bir IP girer → `/lookup` endpointi tetiklenir
2. Controller üç API’ye de istek gönderir
3. Dönüşler AnalysisController’da yorumlanır
4. Ortak sonuç → risk analizi hesaplanır
5. Frontend’e JSON olarak döner
6. Kullanıcı isterse raporu `/save` ile kaydeder
7. `/saved` ile geçmiş raporlar listelenir

6. Frontend İş Akışı

1. Sorgu gönderme

Kullanıcı IP/domain yazar → fetch() ile:

```
GET /api/ThreatIntel/lookup?query=1.1.1.1
```

2. Sonuç gösterme

JSON → HTML card içinde gösterilir
Risk seviyesi renkli badge olarak sunulur.

3. Kaydetme

```
POST /api/ThreatIntel/save
```

4. Kayıtlı raporları listeleme

```
GET /api/ThreatIntel/saved
```

7. Risk Derecelendirme

AnalysisController içinde:

Servis	Değeri	Risk
VirusTotal	malicious ≥ 3	High
AbuseIPDB	AbuseConfidence > 75	High
Shodan	Açık port sayısı fazla	Medium
Hiç tehdit yok		Low

Genel risk → en yüksek çıkan servise göre belirlenir.

8. Sonuç

Bu proje, bir kullanıcının **ç**oklu tehdit istihbarat kaynağını tek bir panelden analiz etmesini, bu verileri risk derecesiyle birlikte yorumlayabilmesini ve sonuçları kalıcı olarak saklayabilmesini sağlayan tam işlevsel bir web uygulamasıdır.

ASP.NET Core ile modern, ölçeklenebilir bir API yapısı oluşturulmuş, MongoDB ile kalıcı kayıt yönetimi sağlanmış ve Bootstrap destekli sade bir arayüz ile kullanıcı deneyimi geliştirilmiştir.