**Proje Adı:** Ağ Tarayıcısı

**Giriş**

**Projenin Amacı:**

Bu proje, belirli bir IP adresi aralığındaki veya ağdaki aktif cihazları ve bu cihazlardaki açık portları tespit etmek amacıyla bir ağ tarayıcısı geliştirmeyi hedeflemektedir. Bu sayede, ağ yöneticilerinin ağlarını daha iyi anlamalarına, güvenlik açıklarını belirlemelerine ve ağ performansını optimize etmelerine yardımcı olacak bir araç sağlanacaktır.

**Projenin Kapsamı:**

* **Temel Özellikler:** 
  + Belirli bir IP adresi aralığını veya alt ağ maskesini taramak.
  + Aktif cihazları (örneğin, bilgisayarlar, sunucular, yönlendiriciler) tespit etmek.
  + Her cihaz için açık TCP ve UDP portlarını belirlemek.
  + Tarama sonuçlarını kullanıcı dostu bir şekilde görüntülemek (örneğin, tablo veya liste formatında).
  + Tarama sonuçlarını bir dosyaya kaydetmek (örneğin, CSV veya TXT formatında).
* **Gelişmiş Özellikler (isteğe bağlı):** 
  + Her cihaz için MAC adresini ve işletim sistemini belirlemek.
  + Belirli hizmetlerin (örneğin, HTTP, SSH, FTP) açık portlarda çalışıp çalışmadığını kontrol etmek.
  + Tarama hızını ve doğruluğunu artırmak için çoklu iş parçacığı kullanmak.
  + Kullanıcıların tarama ayarlarını özelleştirmelerine olanak tanımak (örneğin, tarama portları, zaman aşımı süresi).
  + Grafiksel kullanıcı arayüzü (GUI) eklemek.

**Teknolojiler:**

* **Programlama Dili:** Python
* **Kütüphaneler:** nmap, socket, threading, tkinter (GUI için)
* **İşletim Sistemi:** Windows, Linux, macOS

**Tasarım**

1. **Mimari**

Projenin anlaşılabilir olması için modüler bir mimari tasarlanması amaçlanmaktadır. Bu sayede proje anlaşılabilir, bakımı kolay ve genişletilebilir olacaktır. Modüller şunları içerecektir:

* tarama\_motoru.py: Ağ taramasını gerçekleştirecek temel fonksiyonları içerecektir.
  + hedef\_belirle(): Hedef IP adresi aralığını veya alt ağ maskesini belirler.
  + cihaz\_tara(): Belirtilen hedeflerdeki aktif cihazları tespit eder.
  + port\_tara(): Her cihaz için açık portlar taranır
* sonuc\_islemleri.py: Tarama sonuçlarını işleyen ve görüntüleyen fonksiyonları içerir.
  + sonuclari\_goruntule(): Tarama sonuçlarını kullanıcı dostu bir şekilde gösterir.
  + sonuclari\_kaydet(): Tarama sonuçlarını bir dosyaya kaydeder.
* gui.py: Grafiksel kullanıcı arayüzünü oluşturan fonksiyonları içerir.
  + pencere\_olustur(): Ana penceriyi ve GUI elemanlarını oluşturur.
  + tarama\_baslat(): Kullanıcının girdiği bilgilere göre tarama\_motoru.py modülündeki fonksiyonları çağırır.
  + sonuclari\_goster(): sonuc\_islemleri.py modülünden alınan sonuçları GUI’de görüntüler.
* main.py: Tüm modülleri birleştiren ve uygulamayı başlatan ana program dosyası.
* GUI modülünü içe aktarır ve GUI penceresini oluşturur.
* Kullanıcı etkileşimlerini (örneğin, buton tıklamaları) GUI modülüne yönlendirir.
  1. **GUI Tasarımı**

 **Hedef Belirleme:**

* IP adresi aralığı veya alt ağ maskesi girmek için bir metin kutusu.
* "Tara" butonu ile taramanın başlatılması.

 **Tarama Ayarları (isteğe bağlı):**

* Taranacak port aralığını belirlemek için bir metin kutusu veya liste.
* Zaman aşımı süresi gibi diğer ayarlar için seçenekler.

 **Sonuç Görüntüleme:**

* Aktif cihazları ve açık portları listelemek için bir tablo veya ağaç yapısı.
* Her cihazın detaylı bilgilerini (MAC adresi, işletim sistemi vb.) göstermek için bir panel.

 **İlerleme Göstergesi:**

* Tarama işlemi sırasında ilerlemeyi göstermek için bir ilerleme çubuğu.

 **Durum Çubuğu:**

* Tarama durumu, hata mesajları ve diğer bilgileri göstermek için bir durum çubuğu.

Örnek temel tasarım taslağı aşağıda verilmiştir.

+-------------------------------------------+ | Ağ Tarayıcısı | |---------------------------------------| Hedef IP Aralığı: [metin kutusu] |

| Tarama Ayarları:

| \* Port Aralığı: [metin kutusu]

| \* Zaman Aşımı: [seçenekler]

| [Tara] butonu |

|---------------------------------------------------------------------------------------------------------|

Aktif Cihazlar: | | [Tablo veya ağaç yapısı]

|--------------------------------------------------------------------------------------------------------|

Cihaz Detayları: | | [Panel] |

|-------------------------------------------|

| İlerleme: [ilerleme çubuğu] |

|-------------------------------------------|

| Durum: [durum çubuğu] |

+-------------------------------------------+

1. **Veri Yapıları**

Tarama sonuçlarını depolamak için uygun veri yapıları kullanılmalıdır. Örnek vermek gerekirse, her cihaz için bir sözlük kullanılabilir. Bu sözlük, cihazın IP adresi, mac adresi , açık portları, işletim sistemi gibi bilgileri içerebilir.

cihaz\_bilgileri = {

"ip\_adresi": "192.168.1.10",

"mac\_adresi": "00:11:22:33:44:55",

"acik\_portlar":,

"isletim\_sistemi": "Windows 10"

}

Bütün cihazlar için bir liste kullanılabilir. Bu liste , her cihaz için bir sözlük içerebilir:

tarama\_sonuclari = [

cihaz\_bilgileri\_1,

cihaz\_bilgileri\_2,

...

]

1. **Algoritmalar**

Aktif cihaz tespiti için ICMP Echo Request (ping) paketleri gönderilecek . Yanıt veren cihazlar aktif olarak kabul edilecek.

Açık portları taramak için TCP SYN taraması yapılacak. Bu yöntemle hedef porta bir SYN paketi göndererek gönderilecek ve cevap beklenecektir.

İşletim sistemi tespiti için TCP/IP yığın parmak izi kullanılacaktır. Bu yöntem, hedef cihaza farklı TCP paketleri göndererek ve yanıtları analiz ederek çalşacaktır.

1. **Tasarım Şeması**

Aşağıdaki şemada projenin temel bileşenleri ve çalışma şekli verilmiştir.

**Şekil 1.1.** Çalışma Şekli

