**Hız Canavarı Sahnede: Siber Güvenliğin İsviçre Çakısı FFUF Nedir?**

Siber güvenlik ve sızma testleri dünyasında hız, her şey demektir. Binlerce, hatta milyonlarca potansiyel zafiyeti kısa sürede tarayabilmek, bir adım önde olmanızı sağlar. İşte tam bu noktada, adını "Fuzz Faster U Fool" (Daha Hızlı Fuzz'la Seni Aptal) gibi iddialı bir ifadeden alan **FFUF** sahneye çıkıyor. FFUF, sadece bir dizin tarama aracı değil, bir web fuzzer'ı olarak HTTP isteklerinin neredeyse her parçasını inanılmaz bir hızla test etmenizi sağlayan çok yönlü bir güç merkezidir.

Bu yazıda, FFUF'un ne olduğunu, neden sızma testlerinin vazgeçilmezi haline geldiğini, nasıl kurulduğunu ve en önemlisi, bu hız canavarını bir profesyonel gibi nasıl kullanabileceğinizi etkili örneklerle keşfedeceğiz.

**1. FFUF Aracı Nedir?**

**FFUF**, Go programlama diliyle yazılmış, inanılmaz derecede hızlı, komut satırı tabanlı bir web fuzzer'ıdır. "Fuzzing," bir sisteme beklenmedik, geçersiz veya rastgele veriler göndererek nasıl tepki verdiğini gözlemleme tekniğidir. FFUF, bu tekniği web uygulamalarına uygular.

FFUF'un en temel ve sihirli konsepti **FUZZ** anahtar kelimesidir. Siz bir URL veya istek içinde FUZZ kelimesini nereye koyarsanız, FFUF belirlediğiniz kelime listesindeki (wordlist) her bir kelimeyi sırayla o FUZZ kelimesinin yerine koyarak binlerce istek gönderir. Bu sayede sadece dizin ve dosyaları değil; alt alan adlarını (subdomain), sanal konakları (virtual hosts), API parametrelerini ve hatta parametre değerlerini bile keşfedebilirsiniz.

**2. Neden Kullanılır?**

FFUF'un popülaritesinin arkasında yatan birçok neden vardır:

* **Akılalmaz Hız:** Go dilinin eş zamanlılık (concurrency) yetenekleri sayesinde, Python ile yazılmış Dirsearch gibi araçlardan katbekat daha hızlıdır. Bu, devasa kelime listeleriyle yapılan taramaları dakikalar içinde tamamlayabilmesi anlamına gelir.
* **Maksimum Esneklik:** FUZZ anahtar kelimesi sayesinde bir HTTP isteğinin herhangi bir yerini (URL, Host başlığı, POST verisi, vb.) fuzz'layabilirsiniz. Bu, onu basit bir dizin tarayıcısından çok daha fazlası yapar.
* **Gelişmiş Filtreleme:** Sonuçları HTTP durum koduna, yanıtın boyutuna, içerdiği kelimelere veya satır sayısına göre filtreleyebilirsiniz. Bu, alakasız sonuçları eleyerek sadece önemli bulgulara odaklanmanızı sağlar.
* **Düşük Kaynak Tüketimi:** Hızına rağmen sistem kaynaklarını verimli kullanır.
* **Aktif Geliştirme:** Modern web teknolojilerine ve saldırı vektörlerine uyum sağlayan, sürekli geliştirilen bir projedir.

**3. Nasıl Kurulum Yapılır?**

FFUF'u kurmak için birkaç basit yöntem bulunur.

**Yöntem 1: Go ile Kurulum (En Güncel ve Önerilen Yöntem)**

Eğer sisteminizde Go programlama dili kuruluysa, bu en kolay yoldur.

# Go kurulu değilse önce onu kurun

# Ardından aşağıdaki komutu çalıştırın

go install github.com/ffuf/ffuf/v2@latest

Bu komut, FFUF'un en son sürümünü indirip derleyecek ve kullanıma hazır hale getirecektir.

**Yöntem 2: Kali Linux (APT ile Kurulum)**

Kali Linux veya Debian tabanlı bir sistem kullanıyorsanız, paket yöneticisi apt ile kurabilirsiniz.

sudo apt update && sudo apt install ffuf

**Yöntem 3: GitHub Üzerinden Derlenmiş Sürümü İndirme**

Go kurmak istemiyorsanız, FFUF'un GitHub "Releases" (Sürümler) sayfasından kendi işletim sisteminize (Windows, Linux, macOS) uygun, önceden derlenmiş çalıştırılabilir dosyayı indirebilirsiniz.

**4. Kullanım Parametreleri Nedir?**

FFUF'un gücü, parametrelerinin akıllıca kullanılmasında yatar. İşte en temel ve önemli olanlar:

* -u: Hedef URL. **FUZZ** anahtar kelimesini içermelidir.
* -w: Taramada kullanılacak kelime listesinin (wordlist) yolu.
* -c: Çıktıyı renklendirir. Okunabilirliği büyük ölçüde artırır. (Şiddetle tavsiye edilir!)
* -t: Eş zamanlı çalışacak iş parçacığı (thread) sayısı. (Örn: -t 100)
* -mc: Sadece belirtilen HTTP durum kodlarını gösterir. (Örn: -mc 200,301,302 sadece başarılı ve yönlendirme sonuçlarını gösterir.)
* -fc: Belirtilen HTTP durum kodlarını sonuçlardan gizler. (Örn: -fc 404 ile "Bulunamadı" hatalarını görmezsiniz.)
* -fs: Belirtilen boyuttaki (byte) yanıtları gizler. Sabit bir "Sayfa Bulunamadı" sayfası olan sitelerdeki gürültüyü engellemek için mükemmeldir. (Örn: -fs 1250)
* -H: İsteğe özel HTTP başlıkları (Header) ekler. (Örn: -H "Cookie: session=12345")
* -X: Kullanılacak HTTP metodunu belirtir (GET, POST, PUT, vb.).
* -d: POST metodu ile gönderilecek veriyi belirtir.

**5. Etkili Kullanım İçin 5 Örnek ve Açıklamaları**

FFUF'un gerçek gücünü bu pratik örneklerle görelim.

**Örnek 1: Temel Dizin ve Dosya Fuzzing**

Bu, FFUF'un en basit kullanımıdır. Belirli uzantılara sahip dosyaları ve dizinleri ararız.

ffuf -u https://hedefsite.com/FUZZ -w /usr/share/wordlists/dirb/common.txt -e .php,.html,.bak -c

* **Açıklama:**
  + -u https://hedefsite.com/FUZZ: FFUF'a, kelime listesindeki her kelimeyi hedefsite.com/ adresinin sonuna ekleyerek denemesini söyler.
  + -w ...common.txt: Tarama için yaygın olarak kullanılan common.txt listesini belirtir.
  + -e .php,.html,.bak: Her kelimeyi hem kendisiyle hem de sonuna bu uzantılar eklenmiş haliyle dener (örneğin admin, admin.php, admin.html, admin.bak).
  + -c: Sonuçları renklendirerek (200 OK için yeşil, 403 için turuncu vb.) analizi kolaylaştırır.

**Örnek 2: Alt Alan Adı (Subdomain) Keşfi**

FFUF, alt alan adlarını bulmak için de harikadır. FUZZ kelimesini bu kez URL'nin başına koyarız.

ffuf -u https://FUZZ.hedefsite.com -w /path/to/subdomains.txt -H "Host: FUZZ.hedefsite.com" -fc 404 -c

* **Açıklama:**
  + -u https://FUZZ.hedefsite.com: FUZZ bu kez alan adının başına yerleştirilir. subdomains.txt listesindeki www, api, dev gibi kelimelerle https://www.hedefsite.com, https://api.hedefsite.com gibi istekler oluşturulur.
  + -H "Host: FUZZ.hedefsite.com": Bu çok önemlidir. Sunucuya hangi alt alan adını istediğimizi Host başlığı ile belirtiriz.
  + -fc 404: Var olmayan alt alan adlarından dönen 404 hatalarını gizleyerek çıktıyı temizleriz.

**Örnek 3: Gizli Parametreleri Fuzz'lama (GET İsteği)**

Bir sayfada (users.php gibi) gizli kalmış parametreleri (id, debug, admin gibi) bulmaya çalışalım.

ffuf -u "https://hedefsite.com/users.php?FUZZ=test" -w /path/to/params.txt -fs 1450 -c

* **Açıklama:**
  + -u "...?FUZZ=test": Bu kez FUZZ, bir parametre adı olarak kullanılır. params.txt listesindeki her kelime, users.php?id=test, users.php?debug=test şeklinde denenir.
  + -fs 1450: Parametre geçersiz olduğunda sayfanın standart yanıt boyutunun 1450 byte olduğunu varsayalım. Bu komut, bu boyuttaki yanıtları gizleyerek sadece farklı bir yanıt döndüren (yani işe yarayan) parametreleri görmemizi sağlar.

**Örnek 4: POST Verisi Değerini Fuzz'lama (IDOR Zafiyeti Tespiti)**

Bir kullanıcının sadece kendi verisini görmesi gereken bir sayfada, başka kullanıcıların verilerini görüp göremediğimizi test edelim.

ffuf -u https://hedefsite.com/my-account -X POST -d "user\_id=FUZZ" -w /path/to/numbers\_1\_to\_1000.txt -H "Content-Type: application/x-www-form-urlencoded" -mc 200 -c

* **Açıklama:**
  + -X POST: İsteğin bir POST isteği olduğunu belirtir.
  + -d "user\_id=FUZZ": FUZZ bu kez user\_id parametresinin *değeri* olarak kullanılır.
  + -w ...numbers\_1\_to\_1000.txt: 1'den 1000'e kadar sayıların olduğu bir liste ile user\_id'yi dener.
  + -mc 200: Sadece "Başarılı" (200) yanıtlarını gösterir. Eğer kendi ID'niz dışındaki ID'ler için de 200 yanıtı alıyorsanız, bu bir IDOR zafiyetine işaret edebilir.

**Örnek 5: Hızlı, Odaklı ve Sessiz Tarama**

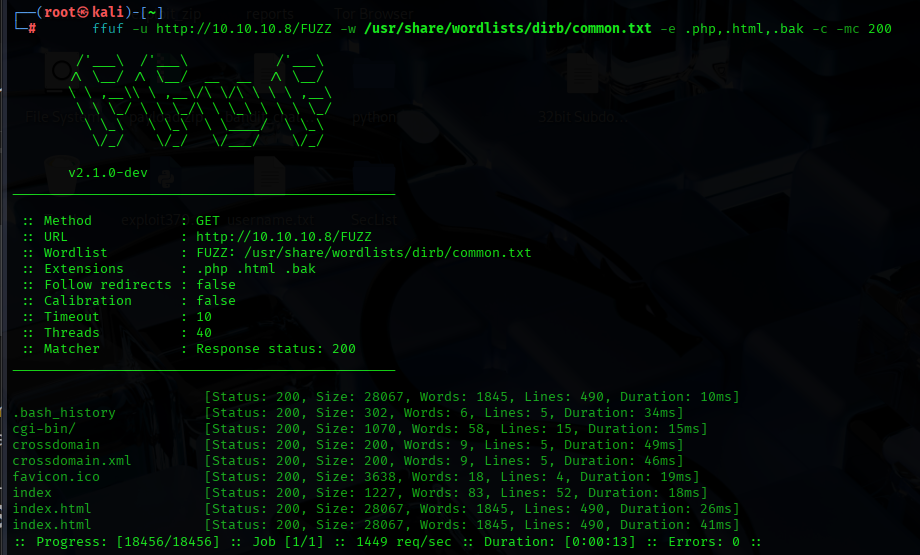
Gerçek bir sızma testi senaryosunda, hızı sonuna kadar açıp sadece ilgilendiğimiz sonuçları görmek isteriz.

ffuf -u https://hedefsite.com/FUZZ -w /usr/share/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt -t 200 -mc 200,301,302 -o ffuf\_results.json

* **Açıklama:**
  + -w ...directory-list-2.3-medium.txt: SecLists gibi daha kapsamlı ve profesyonel bir kelime listesi kullanılır.
  + -t 200: Tarama 200 thread ile inanılmaz bir hıza çıkarılır (sunucunun kaldırabileceğinden emin olun!).
  + -mc 200,301,302: Sadece erişilebilir (200), kalıcı yönlendirme (301) ve geçici yönlendirme (302) gibi kesinlikle ilginç olan durum kodlarına odaklanır.
  + -o ffuf\_results.json: Sonuçları sessizce, JSON formatında bir dosyaya kaydeder. Bu, sonuçları daha sonra başka araçlarla işlemek veya raporlamak için idealdir.

**🚀 Örnek: FFUF ile Hızlı Dizin ve Dosya Keşfi**

FFUF, web uygulamalarındaki dizinleri, dosyaları veya parametreleri hızlı bir şekilde brute-force yöntemiyle tespit etmek için kullanılan bir fuzzing aracıdır. Aşağıda, http://10.10.10.8/ adresine karşı gerçekleştirilen bir FFUF taramasının çıktısı yer almaktadır.

  
Bu çıktıda FFUF aracı -w parametresi ile common.txt wordlist’ini kullanarak, .php, .html, .bak uzantılı dosyaları keşfetmeye çalışmaktadır. -mc 200 parametresi ile yalnızca HTTP 200 (başarılı) cevapları filtrelenmiştir. Tarama sonucunda .bash\_history, cgi-bin/, crossdomain.xml, index.html gibi önemli dosyaların ve dizinlerin bulunduğu görülmektedir. Bu dosyalar sistem hakkında bilgi sızdırabilir ya da saldırı yüzeyi oluşturabilir. 40 iş parçacığıyla çalışan FFUF, saniyede 1400'den fazla isteği işleyerek oldukça hızlı bir tarama gerçekleştirmiştir.

**Sonuç**

FFUF, modern web güvenliği testlerinin vazgeçilmez bir parçasıdır. Hızı, esnekliği ve güçlü filtreleme yetenekleri sayesinde, bir hedefin saldırı yüzeyini dakikalar içinde haritalandırmanızı sağlar. FUZZ anahtar kelimesinin sihrini bir kez anladığınızda, FFUF'un sadece bir dizin bulucudan çok daha fazlası olduğunu, adeta bir siber güvenlik İsviçre çakısı olduğunu göreceksiniz.

Her zaman olduğu gibi, bu gücü büyük bir sorumlulukla kullanmak esastır. FFUF'u ve benzeri araçları **yalnızca yasal yetkiniz ve izniniz olan sistemler üzerinde** kullanarak bilginizi ve yeteneklerinizi etik sınırlar içinde geliştirmelisiniz.