Command Injection/ Basic Command Injection

Herkese selamlar bu yazımda Hackviser üzerinde bulunan Web security lablarından biri olan **Command Injection/ Basic Command Injection** labının çözümünü anlatacağım.

Öncelikle OS Command Injection açığının ne olduğunu kısaca bir hatırlayalım:

OS Command Injection (Komut Enjeksiyonu) zafiyeti de SQL Injection'a benzer bir mantıkla çalışır, ancak veritabanı sorguları yerine işletim sistemine ait komutlar manipüle edilir. Bu zafiyet, web uygulamasının bir kullanıcı girdisini doğrudan bir işletim sistemi komutuna dahil ettiği ve bu komutun güvenlik önlemleri olmadan çalıştırıldığı durumlarda ortaya çıkar.

Komut Enjeksiyonu Zafiyeti (OS Command Injection) Nedir?

Komut enjeksiyonu, web uygulamasının kullanıcıdan aldığı girdileri doğrudan sunucuda çalışan işletim sistemi komutlarına eklediği ve bunları güvenlik kontrolleri olmadan çalıştırdığı durumlarda meydana gelir. Saldırganlar, bu açıktan yararlanarak sistemde kendi istedikleri komutları çalıştırabilirler. Bu tür açıklar, genellikle sunucu üzerinde işlem yapan formlar, parametreler, dosya yolları ve diğer kullanıcı girdisi kabul eden alanlarda bulunur.

Komut Enjeksiyonu Zafiyeti Nerelerde Bulunabilir?

- **Form Alanları**: Örneğin, bir web formu üzerinden alan adı doğrulama veya IP adres kontrolü gibi işlemler yapılırken, girdi işletim sistemi komutuna eklenebilir.
- **URL Parametreleri**: URL üzerinden gelen parametreler bir işletim sistemi komutuna eklenip çalıştırılabilir.

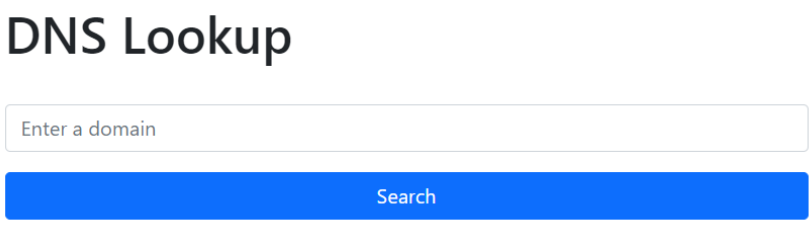
- **Dosya Yolları**: Dosya yükleme ve işlem yapma gibi senaryolarda, kullanıcı girişi bir komutun parçası olarak kullanılabilir.

OS Command Injection Açığını Bulmak İçin Nereleri Hedef Almalıyız?

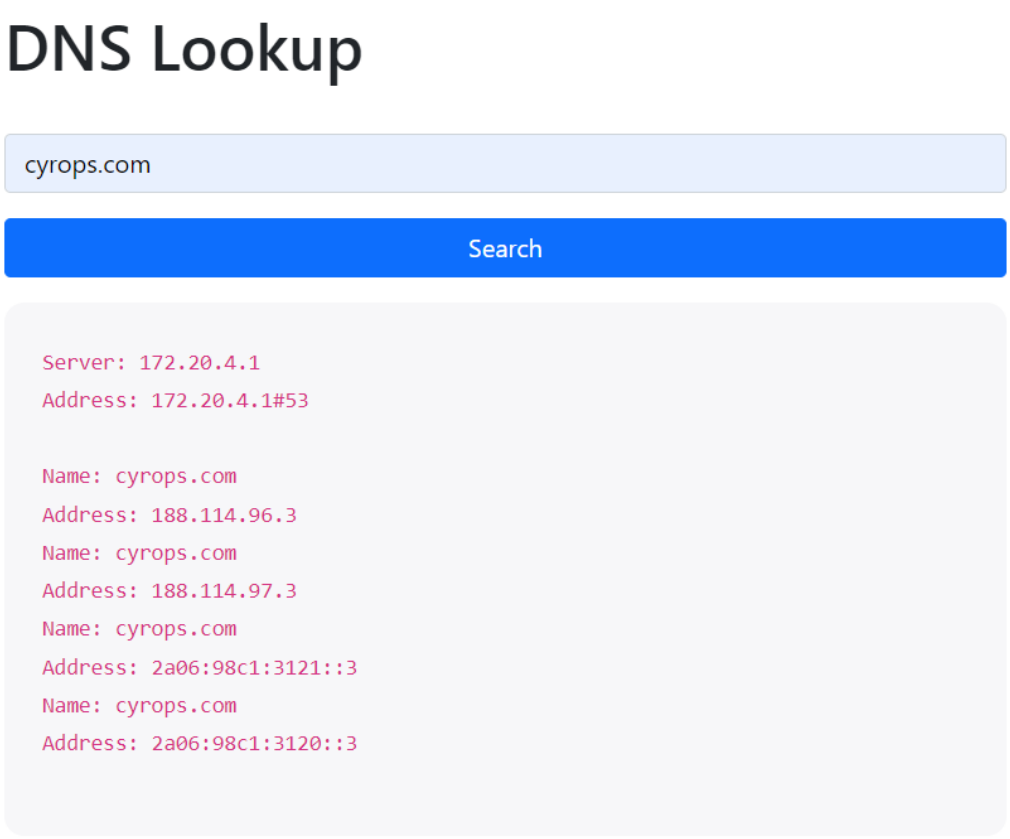
1. **Alan Adı veya IP Doğrulama**: `nslookup`, `ping`, `traceroute` gibi komutların çalıştığı yerlerde testler yapılabilir.
2. **Dosya İşleme**: Dosya upload veya path işlemi yapan uygulamalar potansiyel olarak açık verebilir.
3. **Herhangi Bir Komut Çalıştırma Noktası**: Bir uygulama, arka planda işletim sistemi komutu çalıştırıyorsa, bu komutun giriş parametrelerini kontrol etmek gerekir.

Sırada ise makinenin çözümü var.

Bizi aşağıdaki gibi bir sayfa karşılıyor.

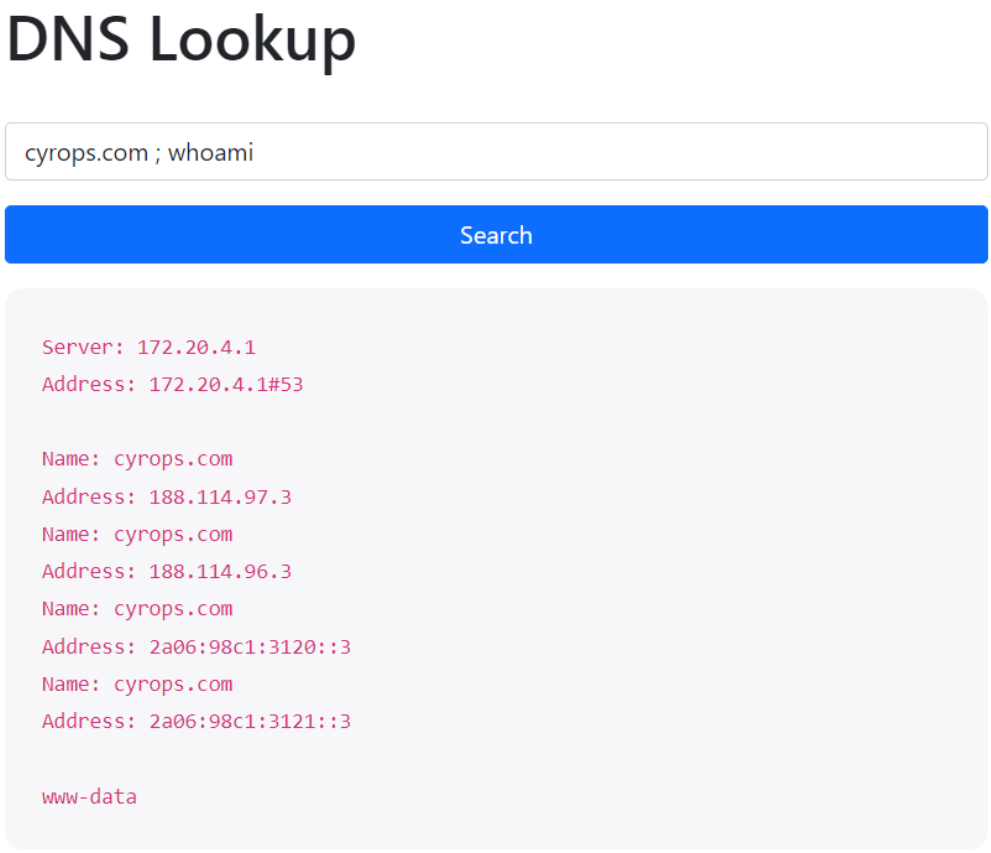


“cyrops.com” üzerinden deneme yaptığımızda aşağıdaki gibi bir DNS Lookup çıktısı elde ediyoruz.



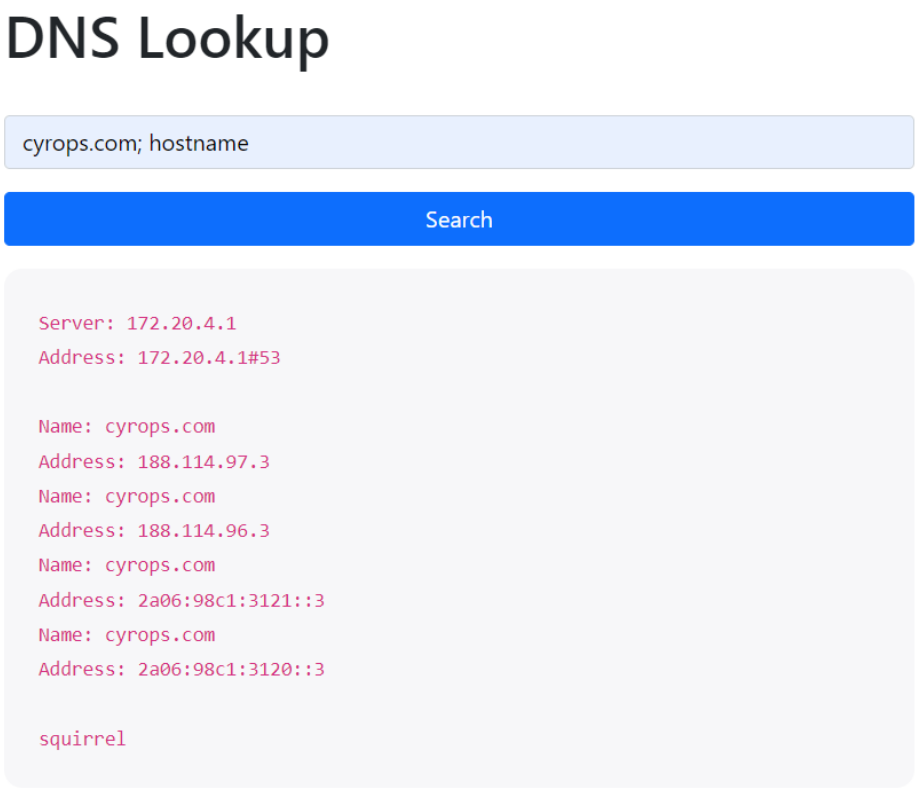
Bu url e nasıl bir komut eklersem istediğim çıktıyı alırım diye düşünmeye başlıyorum.

Burada aklıma “ ; “ kullanmak geliyor bunun sebebi ise noktalı virgülün komut ayırıcı işlevinin de olması.



Yukarıdaki gibi “whoami” komutunu kullanarak noktalı virgülün istediğim şekilde çalışıp çalışmadığını görüyorum ve işlem başarılı.

Sırada ise sunucunun ana bilgisayar adını bulmak var.



İşte istediğim bilgiye ulaştım.

OS Command Injection açığını kullanarak neler yapabileceğimize bakalım. Bu açık sayesinde, saldırganlar sunucunun işletim sistemine doğrudan komut göndererek aşağıdaki gibi ciddi zararlar verebilirler:

1. **İşletim Sistemi Komutlarını Çalıştırmak**: Saldırgan, sunucunun işletim sisteminde komut çalıştırabilir, bu sayede sunucuya zarar verebilir, kritik sistem dosyalarına erişebilir.
2. **Dosya Erişimi ve Manipülasyonu**: Saldırgan, sunucunun dosya sistemine erişebilir, hassas dosyaları okuyabilir, silebilir veya değiştirebilir.
3. **Yetki Yükseltme**: Eğer işletim sisteminde çalışan komutlar yüksek yetkilere sahipse, saldırgan bu yetkileri kullanarak sistemin kontrolünü ele geçirebilir.
4. **Ağ Üzerinde Keşif**: Saldırgan, sunucudan başka ağdaki cihazlara erişim sağlayabilir ve ağ keşfi yapabilir.

5. **Arka Kapı Oluşturma**: Saldırgan, sunucu üzerinde arka kapı bırakarak gelecekte tekrar sisteme sızma girişiminde bulunabilir.

OS Command Injection Zafiyetini Nasıl Kapatabiliriz?

1. **Kullanıcıdan Alınan Verilerin Filtrelenmesi ve Doğrulanması**:
   - Kullanıcıdan alınan her türlü veri, komut satırına eklenmeden önce mutlaka filtrelenmelidir. Bu filtreleme işlemi sırasında, sadece beklenen ve güvenli karakterlere izin verilmelidir. Örneğin, `nslookup` gibi bir komutta sadece domain isimleri ya da IP adresleri kabul edilmelidir.
   - Girdiler, `input validation` yapılarak beklenen formata uygun olup olmadığı kontrol edilmelidir. Böylece komut enjekte etmeye çalışacak zararlı girişler engellenebilir.
2. **Komutları Doğrudan Çalıştırmaktan Kaçınmak**:
   - Mümkünse kullanıcı verisini doğrudan işletim sistemi komutlarına dahil etmemek gerekir. Bunun yerine, işletim sistemi ile iletişim kurmak için güvenli bir API veya kütüphane kullanılmalıdır.
   - Örneğin, PHP'de `shell_exec()` veya `system()` gibi fonksiyonlar yerine daha güvenli alternatifler kullanılmalıdır.
3. **Kullanıcı Girdisini Birleştirmeden Komut Çalıştırmak**:
   - Eğer işletim sistemi komutları mutlaka çalıştırılacaksa, parametreleri ayrı bir değişken olarak işletim sistemi komutuna geçiren güvenli yöntemler kullanılmalıdır. Örneğin, PHP'de `escapeshellarg()` ve `escapeshellcmd()` fonksiyonları ile komut satırı parametrelerini güvenli hale getirebiliriz.
4. **Yetkilerin Sınırlandırılması**:
   - Web uygulamasının işletim sistemi üzerinde çalıştırdığı komutların minimum yetkilere sahip olması sağlanmalıdır. Sunucuya verilen yetkiler, sadece gereken işlemleri yapabilecek şekilde sınırlanmalıdır.
   - Uygulamanın çalıştığı kullanıcı hesabının sadece gerekli dosya ve dizinlere erişim yetkisi olmalıdır.
5. **Güncellemeler ve Yama Yönetimi**:

- Sistem ve yazılım güncellemeleri düzenli olarak yapılmalıdır. Bilinen komut enjeksiyonu açıklarını kapatan yamaların sistem üzerinde uygulanması büyük önem taşır.

6. **Güvenlik Duvarı ve Ağ Kontrolleri**:
    - Sunucunun dışarıya istenmeyen ağ bağlantıları kurmasını engellemek için güvenlik duvarı kuralları oluşturulabilir. Böylece saldırganların sunucudan dışarıya veri sızdırma ya da başka bir sisteme saldırma girişimleri sınırlandırılabilir.

Umarım bu yazıyı sıkılmadan okumuşsunuzdur. Keyifli çalışmalar.