KARADENİZ TEKNİK ÜNİVERSİTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ
SİBER GÜVENLİK LABORATUVARI

Parola Saldırıları ve Güvenlik Yöntemleri

1. GİRİŞ

Bir sistemin güvenliği, en zayıf halkası kadar güçlüdür; çoğu zaman kullanıcı parolaları bu zayıf halka olabilmektedir. Parola güvenliği genellikle şifreleme veya hashleme yöntemleriyle sağlanır. Şifreleme (encryption), düz metin parolayı tersine çevrilebilir şekilde şifrelerken; hashleme, parolayı tek yönlü olarak dönüştürür ve geri açılması (orijinal parolanın elde edilmesi) matematiksel olarak çok zordur. Sistemler parolaları genellikle düz metin olarak değil, bu hash değerleri olarak depolar. Hash fonksiyonundan geçmiş bir parolanın kendisini bulmak, fonksiyonu tersine çevirmek yerine, olası aday parolaların hash'lerini hesaplayıp karşılaştırmayı gerektirir. Bu nedenle saldırganlar, hash değerini ele geçirdiklerinde parolayı kırmak için deneme-yanılma yoluyla hash üretip eşleştirme yöntemine başvururlar.

PAROLA SALDIRI TÜRLERİ

Parola saldırıları temel olarak çevrimdışı (offline) ve çevrimiçi (online) olarak iki kategoriye ayrılır:

- **Çevrimdışı Parola Saldırıları:** Saldırgan, hedef sistemden kullanıcıların parola hashlerini ele geçirir ve kırma işlemini kendi bilgisayarında yapar. Bu sayede hedef sisteme ardışık giriş denemeleri yapılmaz, dolayısıyla kilitlenme veya alarm tetiklenme riski yoktur. Ayrıca çevrimdışı ortamda, çok sayıda parola denemesi kısa sürede yapılabilir. Örneğin bir saldırgan veritabanı sızıntısından elde ettiği hashler üzerinde John the Ripper veya Hashcat gibi araçlarla deneme yapabilir. Hash değeri bilindiğinde yöntem, her deneme parolasının aynı hash fonksiyonuyla özetini alıp ele geçirilmiş hash ile karşılaştırmaktır; eşleşirse parola bulunmuş demektir. Bu süreç, hashleme işlemine tuz (salt) eklenmedikçe daha kolaydır; salt kullanımı aynı parolanın farklı kullanıcılar için farklı hashler üretmesini sağlayarak saldırıyı zorlaştırır. Çevrimdışı saldırılar için kullanılan araçlardan John the Ripper (JtR) ve Hashcat, çok çeşitli hash algoritmalarını destekler ve genellikle hızlı sonuç almak için optimize edilmişlerdir. Örneğin Hashcat GPU desteğiyle büyük arama alanlarını tarayabilir.
- **Çevrimiçi Parola Saldırıları:** Saldırgan, hedef sisteme doğrudan giriş denemeleri yaparak parolayı tahmin etmeye çalışır. Örneğin bir web sitesinin giriş formuna art arda şifreler deneyerek veya SSH/FTP gibi servislerde kullanıcı adı-parola denemeleri yaparak gerçekleştirilen saldırılar çevrimiçi saldırılardır. Bu tür saldırılarda araç olarak

Hydra sıklıkla kullanılır. Hydra, pek çok protokol (HTTP, SSH, FTP vb.) için kaba kuvvet ve sözlük saldırısı gerçekleştirebilen bir araçtır. Çevrimiçi saldırılar, her deneme hedef sisteme gittiği için yavaştır ve başarısız giriş denemeleri hedef loglarında iz bırakabilir. Hatta birçok sistem belirli sayıda hatalı denemeden sonra hesap kilitleme veya CAPTCHA gibi önlemlerle bu saldırıları engeller. Bu nedenle Hydra gibi araçlarla yapılan saldırılarda, isabetli kullanıcı adı/parola çiftini bulmak çevrimdışı kırmaya göre daha uzun sürebilir ve tespit edilme riski yüksektir. Hydra çevrimiçi uygulamalara yönelik olduğundan, ağ üzerinden çoklu deneme yaparken hedef sistemi kilitlememeye veya güvenlik alarmlarını tetiklememeye dikkat etmek gerekir.

PAROLA SALDIRI TEKNİKLERİ

Parola saldırı teknikleri genellikle kaba kuvvet veya sözlük tabanlı yöntemler olarak sınıflandırılır:

- **Kaba Kuvvet Saldırısı (Brute Force):** Mümkün olan tüm karakter kombinasyonlarını sistematik olarak deneyerek doğru parolayı bulmaya çalışır. Örneğin yalnızca küçük harflerden oluşan 6 karakterli bir parolanın **29^6 ≈ 595 milyon** olası kombinasyonu vardır. Parola uzunluğu arttıkça bu olasılıklar üstel olarak artar; 7 karakterli benzer bir parola için bu sayı **29^7 ≈ 17 milyar** kombinasyondur. Bu nedenle kaba kuvvet, parolanın karmaşıklığı ve uzunluğuna bağlı olarak çok zaman alabilir. Donanımlarla saniyede milyonlarca deneme yapılabilse de, güçlü parolalar için bu yöntem pratik olmayabilir. Kaba kuvvetin avantajı parola tahmininde hiçbir ön bilgiye ihtiyaç duymamasıdır; dezavantajı ise deneme sayısının çok yüksek olması nedeniyle zaman ve kaynak gerektirmesidir.
- **Sözlük Saldırısı (Dictionary Attack):** Bu yöntemde, saldırgan önceden derlenmiş bir parola listesi (sözlük) kullanarak deneme yapar. Sözlük, yaygın parolalar, sözlük kelimeleri, tarih veya isim gibi sık kullanılan kombinasyonlar içerir. Amaç, kullanıcıların zayıf veya tahmin edilebilir parolalar kullanma ihtimaline dayanarak bu listede doğru parolanın bulunmasıdır. Sözlük saldırısı, kaba kuvvete kıyasla çok daha az deneme gerektirir ve pratikte sıkça işe yarar çünkü pek çok kullanıcı tahmin edilebilir parolalar seçer. Örneğin "123456", "password", "qwerty" gibi parolalar listelerin üst sıralarında bulunur. Elbette sözlükte olmayan rastgele bir parola için bu yöntem başarısız olur. Sözlük saldırıları genelde melez (hybrid) saldırılarla da genişletilir: sözlük kelimelerinin sonuna sayı ekleme, harfleri sayı/simge ile değiştirme gibi basit varyasyonlar da denenir. John the Ripper gibi araçlar bu tür varyasyon kurallarını (rules) kendi sözlük modu kapsamında uygulayabilir.
- **Rainbow Tables:** Rainbow table, hash↔düz metin eşlemelerini önceden hesaplayıp saklayan büyük bir tablodur; saldırganlar ele geçirilmiş hashleri brute-force yapmak yerine bu tabloda arayarak çok daha hızlı sonuç elde edebilir. Dezavantajı, tabloyu oluşturmanın ve saklamanın yüksek disk maliyeti olmasıdır; ayrıca her kullanıcı için farklı bir salt kullanıldığında (saltlı hash) rainbow table'lar pratikte işe yaramaz.
- **Collision Attack:** Collision attack, bir hash fonksiyonu için farklı iki girdinin aynı hash değerini üretebilmesini hedefleyen saldırılardır; yani saldırgan m1 ≠ m2

olacak şekilde hash(m1) = hash(m2) sağlamaya çalışır. Collision'lar özellikle dijital imza ve belge bütünlüğü bağlamında tehlikelidir—çünkü aynı hash'e sahip iki farklı içeriği birbirinin yerine gösterebilirler. Parola kırmada ise esas problem preimage (verilen hash'e karşılık gelen orijinal girdiyi bulma) olduğundan collision doğrudan parola kırma çözümü sunmaz; yine de zayıf veya eskimiş hash algoritmaları (MD5, SHA-1) collision saldırılarına karşı savunmasız olduğundan bu algoritmalardan kaçınılmalı ve salt ile güçlü anahtar türetme fonksiyonları (bcrypt/Argon2) tercih edilmelidir.

PAROLA HASHLERİNİ ELE GEÇİRME YÖNTEMLERİ

Bir saldırganın çevrimdışı kırma yapabilmesi için önce bir şekilde parolaların hashlenmiş halini ele geçirmesi gerekir. Bunun çeşitli yolları vardır:

- **Ağ Trafiğini Dinleyerek:** Eğer bir kullanıcı parolasını şifrelenmemiş bir bağlantı üzerinden gönderiyorsa, bir saldırgan ağ dinleme (sniffing) yoluyla bu parolayı düz metin halde yakalayabilir. Örneğin HTTP ile korumasız bir giriş formundan gönderilen kullanıcı adı ve parolayı Wireshark ile yakalamak mümkündür. Günümüzde hassas servisler genelde TLS/SSL kullandığı için parolalar şifrelenir; ancak hala HTTP, Telnet, FTP gibi şifresiz protokoller kullanılıyorsa ağ dinleme ile parolalar doğrudan ele geçirilebilir (kırmaya bile gerek kalmaz). Bu nedenle "parola kırma" genellikle, parolanın şifreli veya hashli olarak elde edildiği durumlar için söz konusudur.
- **Veritabanı ve Sistem İhlalleri:** Bir sistemin veritabanı sızdırıldığında, kullanıcı hesaplarının hashlenmiş parolaları ele geçirilebilir. Örneğin bir web sitesinin kullanıcı tabloları internete sızdıysa, saldırganlar bu hashleri offline olarak kırmaya çalışır. Benzer şekilde, bir sunucu ele geçirildiğinde Linux ortamında /etc/shadow dosyasındaki hashler veya Windows ortamında NTLM hashleri saldırganın eline geçebilir.
- **Wi-Fi El Sıkışması Yakalama:** Kablosuz ağlarda WPA/WPA2 protokolü, cihaz ile erişim noktası arasında 4 yönlü el sıkışma (4-way handshake) adı verilen bir kimlik doğrulama alışverişi yapar. Bu süreçte şifrelenmiş olarak parolanın (WPA-PSK) doğruluğu karşılıklı doğrulanır. Saldırgan, radyo sinyallerini yakalayarak bu handshake mesajlarını kaydedebilir. Handshake, çözülemeyen bir anahtar içerse de, saldırgan bunu offline olarak sözlükle karşılaştırabilir. Yani her bir olası parola adayını alıp, WPA/WPA2'nin belirlediği hashleme yöntemine sokarak yakalanan handshake içindeki değere eşleşip eşleşmediğine bakar. Eşleşirse, doğru parolayı bulmuş olur.
- **Diğer Yöntemler:** Parola kırmanın dışında, parola öğrenmeye yönelik phishing (oltalama) saldırıları, keylogger yerleştirme gibi yöntemler de vardır ancak bunlar parolanın teknik olarak kırılması değil, kullanıcı aldatılarak veya sistemden izinsiz kaydedilerek elde edilmesidir.

PAROLA GÜCÜ VE GÜVENLİĞİ

Parolanın kırılması, büyük ölçüde parolanın karmaşıklığına ve sistemin aldığı önlemlere bağlıdır. Zayıf parolalar (sözlük kelimeleri, ardışık sayılar, kullanıcı bilgileri vb.) kolaylıkla tahmin edilebilir ve kısa sürede kırılır. Güçlü parolalar ise yeterli uzunlukta, büyük/küçük harf, rakam ve sembol içeren rastgele dizilerdir; bunları kaba kuvvetle denemek pratikte imkânsıza yakındır. Örneğin yalnızca harflerden oluşan 8 karakterlik parola için olasılık **29^8** iken, harf-rakam-sembol karışık 8 karakterlik bir parolanın olasılık uzayı çok daha büyüktür. Ayrıca **tuzlama (salting)**, her kullanıcı parolasına özel rastgele bir değer ekleyerek hashlemeyi, sözlük ve rainbow table saldırılarına karşı daha güvenli hale getirir. Eğer bir Wifi ağının güvenliği söz konusu ise şifreleme yöntemi olarak destekleniyorsa WPA3 gibi daha güncel bir protokolü kullanmak, WPS özelliklerinin devre dışı bırakılması vb. işlemler ağı daha güvenli hale getirecektir. WPS genellikle herhangi bir şifre girme arayüzü bulunmayan cihazların (yazıcı, tarama cihazı vb.) ağa bağlanabilmesi için kullanılan bir özelliktir. Eğer evinizin ağında buna benzer cihazları kullanmıyorsanız kablosuz modeminizin WPS özelliğini kapatmanız güvenlik için önemlidir.

Sistem yöneticileri çevrimiçi saldırılara karşı da önlem alabilir: belli sayıda yanlış denemede hesabı kilitleme, CAPTCHA ile otomatik denemeleri engelleme, çok faktörlü kimlik doğrulama ekleme gibi yöntemler brute force saldırılarını başarısızlığa uğratır. Bu nedenle bir parolanın kırılabilmesi için sadece teorik yöntemler değil, pratikte bu önlemlerin varlığı da belirleyici olacaktır.

2. DENEYİN AMACI VE KAZANIMI

Bu deneyde parola saldırılarının (çevrimdışı ve çevrimiçi) nasıl işlediğini hem teorik hem pratik olarak görerek zayıf parola uygulamalarının neden risk oluşturduğunu fark etmek; hashleme, salt, handshake kavramlarını ve yaygın kırma tekniklerini (sözlük, kaba kuvvet, melez) deneyimlemek; ve Wireshark, John the Ripper, Hashcat, Aircrack-ng ve Hydra gibi araçları güvenli/izinli bir ortamda kullanarak parola saldırı ve güvenlik yöntemlerini uygulamalı bir şekilde anlamayı sağlamaktır.

- Parola hashlerinin ve farklı hash algoritmalarının (MD5, SHA serileri, vs.) temel farklarını ve hash formatlarından algoritma tespit etme mantığını açıklayabilme.
- Hashlerin nasıl ele geçirilebileceğini (veritabanı sızıntısı, ağ dinleme, Wi-Fi 4-way handshake vb.) anlayıp örneklerle ilişkilendirebilme.
- John the Ripper ve Hashcat ile çevrimdışı parola kırma iş akışını (sözlük, kural/variation, brute-force) uygulayarak yürütebilme; elde edilen sonuçları yorumlayabilme.
- Bir parolanın neden kırılabilir veya kırılması zor olduğunu (uzunluk, karmaşıklık, salt kullanımı, hash algoritması) somut örneklerle açıklayabilme ve güvenlik iyileştirme önerileri (güçlü parola politikaları, salting, 2FA vb.) sunabilme.

DENEYE HAZIRLIK

Deneye daha verimli ve güvenli katılabilmeniz için, laboratuvar öncesinde temel Linux/komut satırı becerileri, ağ ve kriptografi kavramları, parola hashleri ve saldırı

yöntemleri ile kullanılacak araçların (John the Ripper, Hashcat, Aircrack-ng, Wireshark, Hydra) temel çalışma mantığını öğrenmiş olmanız beklenmektedir.

- Temel Linux komutları (dosya gezme, dosya görüntüleme, nano/vim kullanımı).
- TCP/IP, IP–MAC adresi, port kavramları; Wireshark ile paket yakalamanın mantığı.
- Hash kavramı: MD5, SHA-1, SHA-256; salt nedir, neden kullanılır.
- Parola saldırı türleri: çevrimdışı vs çevrimiçi, sözlük ve brute-force farkı.
- Araçların temel kullanımı:
    - John the Ripper (john hashes.txt, john --show).
    - Hashcat (hash türü seçme, wordlist ile deneme).
    - Aircrack-ng (handshake .cap kırma).
- Sözlük dosyaları (ör. rockyou.txt) ve neden önemli oldukları.
- Etik/yasal kurallar: sadece izinli ortamda kullanım, gizlilik ve güvenlik bilinci.

3. DENEYİN YAPILIŞI

Deney, tüm gerekli araçların (ve örnek dosyaların) yüklü olduğu bir Kali Linux sistem üzerinde gerçekleştirilecektir. Her öğrenci kendi sanal makinesi üzerinde tek başına çalışacaktır. Aşağıdaki adımlar sırasıyla gerçekleştirilmelidir. Adımların mantığını anlamak, sadece sonuca odaklanmaktan daha önemlidir. Aşağıda ortam ve gereksinimler verilmiştir:

Araçlar:

- **Wireshark:** Ağ trafiğini incelemek için.
- **John the Ripper:** Çevrimdışı parola kırma için.
- **Hashcat:** Hash türlerine göre parola deneme için (CPU kullanımıyla).
- **Aircrack-ng:** WPA/WPA2 handshake içeren .cap dosyalarının parola denemesi için.

Hazır Dosyalar:

- hashes.txt → Ele geçirilmiş parola hash örnekleri.
- wifi_handshake.cap → WPA handshake örneği.
- rockyou.txt veya daha küçük özel parola listeleri.

1) Hash Oluşturma ve Tanıma

Amaç: Farklı algoritmalarla üretilen hash değerlerinin nasıl göründüğünü incelemek. Ayrıca hash uzunluğu ve formatına bakarak algoritmayı tanıyabilmeyi öğrenmek.

MD5 : `echo` -n 'password123' | md5sum
SHA-1 : `echo` -n 'password123' | sha1sum
SHA-256 : `echo` -n 'password123' | sha256sum
Hash türü belirleme : `hashid` -em <hash>

- **Adım 1.a:** Terminali açın. Basit bir metnin çeşitli hash algoritmalarıyla özetini oluşturmak için Linux komutlarını kullanın. Örneğin, *echo -n 'password123' | md5sum* komutu "password123" metninin MD5 hash değerini üretir. Benzer şekilde:
  - echo -n ' password123' | sha1sum (SHA-1 hash)
  - echo -n ' password123' | sha256sum (SHA-256 hash) komutlarını çalıştırın.
- **Adım 1.b:** Çıktılardaki hash değerlerine dikkat edin. Hepsi farklı uzunluklarda ve karakterdedir. Örneğin MD5 32 hex haneden oluşurken SHA-256 64 hex haneden oluşur. Hash uzunluğu ve formatından, hangi algoritma olduğu genelde anlaşılabilir. (Örn: 32 karakterlik bir hex dizesi çoğunlukla MD5'tir, 40 karakter SHA-1, 64 karakter SHA-256 vb.)
- **Adım 1.c :** Eğer Kali Linux kullanıyorsanız hash-identifier veya hashid aracını deneyin. Bu araçlar, verdiğiniz hash değerinin muhtemel algoritmasını tahmin eder. Örneğin: *hashid -em 5f4dcc3b5aa765d61d8327deb882cf99* (bu değer "password" stringinin MD5 hashidir). Araç çıktısında MD5 olasılığını göreceksiniz.

Bu kısımda, farklı hash algoritmalarının çıktılarının nasıl göründüğünü ve hash tanımlama araçlarının kullanımını görmüş oldunuz. Hash bir kez oluşturulunca teorik olarak geri dönüşü yoktur; ancak sonraki adımda yapacağımız gibi, aynı algoritmayla aday parolalar üretip karşılaştırarak doğru parolayı bulabiliriz.

2) Çevrimdışı Parola Kırma

Amaç: Hash dosyasındaki zayıf parolaları offline araçlarla kırmayı denemek. Böylece farklı modların (single, wordlist) nasıl çalıştığını gözlemlemek.
Senaryo: Bir sistemden ele geçirilmiş hash değerlerini kırmaya çalışacağız. Size verilen hashes.txt dosyasında bir veya daha fazla kullanıcının parolasına ait hashler bulunmaktadır. Bu hashler örnek olması için oluşturulmuştur ve görece zayıf parolalardır.

- **Hash dosyası kontrolü :** `cat` hashes.txt
- **Önceki sonuçları gösterme :** `john` --show hashes.txt
- **Kırma başlatma :** `john` hashes.txt
- **Tür belirtmek gerekirse :** `john` --format=sha512crypt hashes.txt
- **Sonuç görüntüleme :** `john` --show hashes.txt
- **Sözlük ile deneme :**

```
gunzip /usr/share/wordlists/rockyou.txt.gz
john --wordlist=/usr/share/wordlists/rockyou.txt --rules
hashes.txt
```

- **Adım 2.a:** Hash formatının anlaşılması: Önce dosyanın içeriğine göz atın. Örneğin dosyada şu formatta satırlar olabilir:
  - alice:6Q4...$N9uGp... (devamı) ...:18090:0:99999:7:::

- bob:6Kk...$z1HV... (devamı) ...:18090:0:99999:7:::

Bu örnek Linux sistemlerinde */etc/shadow* dosyasındaki satırlara benzer. alice: kısmı kullanıcı adını, 6 ile başlayan kısım hashli parolayı (SHA-512 algoritması 6 ile belirtilir), sonraki sayılar ise parola politikası bilgileridir. John the Ripper, bu formatı doğrudan kabul edebilir. Eğer farklı bir format veya sadece düz hash verilmişse (örneğin yalnızca 5f4dcc3b5... gibi), o durumda hash türünü belirleyip John parametrelerini ona göre ayarlamak gerekir.

- **Adım 2.b:** John ile kırma – Basit deneme: Terminalden John the Ripper'ı çalıştırın. Öncelikle John'un doğru şekilde çalıştığını ve hash dosyasını okuyabildiğini test edelim. john --show hashes.txt komutu, dosyadaki hashler daha önce kırıldıysa sonuçlarını gösterir; ilk defa çalıştırdığımız için muhtemelen "no password hashes left to crack" gibi bir ifade göreceksiniz (veya hiç çıktı vermeyebilir). Ardından john hashes.txt komutunu çalıştırın. John, belirtilen dosyadaki hashleri kırmaya başlayacaktır. Not: John, hash türünü otomatik algılayamazsa veya bir uyarı verirse, --format= parametresiyle hash türünü belirtmek gerekebilir (örneğin SHA-512 için --format=sha512crypt).
- **Adım 2.c:** İşlemi izleme: John çalışırken, konsolda denenen yöntemlerle ilgili satırlar görebilirsiniz. John, önce "single crack" mod denilen bir modda hızlıca basit tahminler dener (kullanıcı adı bilgisiyle ilgili varyasyonlar gibi). Ardından kendi küçük yerleşik sözlüğünü ve kurallarını kullanarak denemeler yapabilir. Zayıf parolalar çok kısa sürede bulunacaktır. John'u birkaç dakika çalıştırdıktan sonra işlemi durdurmak için Ctrl+C yapın.
- **Adım 2.d:** Sonuçları alma: John durdurulduktan sonra john --show hashes.txt komutunu tekrar çalıştırın. Bu komut, kırılan hashlerin artık açık parola karşılıklarını gösterecektir. Örneğin:
  - alice: password123

  gibi çıktı alırsanız, alice kullanıcısının parolasının "password123" olduğunu bulmuşsunuz demektir. Bulunan parolaları not edin. (John, bulduğu parolaları kendi john.pot dosyasına kaydeder, böylece aynı hashleri tekrar kırmaya çalışmaz.)
- **Adım 2.e:** Sözlük kullanımı: Eğer John bazı hashleri çözememişse, daha geniş bir sözlük ile deneme yapabiliriz. Kali Linux ile gelen rockyou.txt sözlüğü popüler bir seçenektir. Ancak bu dosya sıkıştırılmıştır. Öncelikle gunzip /usr/share/wordlists/rockyou.txt.gz komutuyla açın (bir kez yapmak yeterli). Ardından John'u bu sözlükle çalıştırmak için:

  john --wordlist=/usr/share/wordlists/rockyou.txt --rules hashes.txt

  komutunu verin. Bu komutu çalıştırdığınızda, rockyou.txt listesindeki milyonlarca kelime ve John'un kendi kuralları (--rules) ile denemeler yapılır. Bu işlem, alice kullanıcısının "password123" gibi basit şifresini bulsa bile, bob kullanıcısının parolasını bulmakta başarısız olabilir veya laboratuvar süresi için çok uzun sürebilir.

- **Adım 2.f:** Kırılan parolaların analizi: alice'in parolası, rockyou.txt listesinde bulunan çok yaygın bir paroladır. Ancak bob'un parolası kırılamamıştır. Bu, bob'un parolasının ya listede olmadığını ya da John'un varsayılan kurallarının üretemeyeceği kadar karmaşık bir varyasyon olduğunu gösterir. Örneğin, parola "alicebob" kelimesinin "A" harfi büyük ve sonunda "!" olan "AliceBob!" gibi bir varyasyonu olabilir. Bu tür spesifik ve karmaşık bir varyasyonu denemek için John'un --rules seti yetersiz veya çok yavaş kalabilir. Bulunan parolalara bakarak birkaç çıkarım yapın. Örneğin parolalar kısa mı, sözlük kelimesi mi, kullanıcı adıyla ilişkili mi? Hangi yöntemle bulundu (varsayılan tekil mod mu, sözlük mü)?

3) Hedef Odaklı Kural Tabanlı Saldırı (Custom Rule Attack):

John the Ripper ile bob'un parolasını bulamadık çünkü varsayılan kurallar senaryomuza (ilk harf büyük, sonuna özel karakter) uymadı ve tüm olasılıkları denemek (brute-force) yıllar sürebilir. Çok daha esnek kural setlerini kullanarak bulmaya çalışalım. bob'un parolasının, diğer kullanıcı adı olan "alice" ile kendi kullanıcı adı "bob"un birleşiminden oluşan "alicebob" kelimesinin bir varyasyonu olduğundan şüpheleniyoruz. Hedefimiz: "Alicebob!".

- **Adım 3.a: Kırılmamış Hash'i Ayıklama:** Tüm hashleri tekrar denememesi için, sadece kırılmamış olan bob'un hash'ini hashes.txt dosyasından alıp yeni bir dosyaya kaydedeceğiz.

```
grep "bob" hashes.txt > hashes_for_john.txt
cat  hashes_for_john.txt
```

- **Adım 3.b:** Hedef Odaklı Sözlük ve Kural Oluşturma: rockyou.txt gibi 14 milyonluk bir listeyi denemek yerine, hipotezimize (parolanın "alicebob" varyasyonu olduğu) dayanan çok küçük, özel bir sözlük oluşturacağız.

```
echo "alicebob" > custom_dict.txt
```

- **Adım 3.c:** Şimdi, "alicebob" kelimesini nasıl "Alicebob!" haline getireceğini söyleyen özel bir kural dosyası (my_john.conf) oluşturacağız. Kural sözdizimine (syntax) göre:

  c : İlk harfi büyük yapar (Capitalize).

  $! : Kelimenin sonuna "!" karakterini ekler (Append !).

  T5: "Alicebob" -> "AliceBob" (6. pozisyondaki 'b' harfini 'B' yap, karakter sayısını 0. indeksten başlayarak sayar.)

Aşağıdaki komutla bu kuralları dosyamıza yazalım:

```
echo '[List.Rules:MyCustomRule]' > my_john.conf
echo 'c T5 $!' >> my_john.conf
```

- **Adım 3.d: Saldırıyı Başlatma**: Artık John'u çalıştırarak ona hash dosyasını, sözlüğümüzü ve yeni kural setimizi gösterebiliriz. John, 6 ile başlayan hash'leri bazen otomatik olarak yanlış algılayabilir.
  Hash formatını --format=sha512crypt parametresiyle manuel olarak belirteceğiz.

```
john --wordlist=custom_dict.txt --rules=MyCustomRule --
config=my_john.conf --format=sha512crypt hashes_for_john.txt
```

- **Adım 3.d:** İşlem bittiğinde, kırılan parolayı görmek için --show parametresini kullanırız:

```
john --show hashes_for_john.txt
```

4) WPA Wi-Fi Parolası Kırma (Aircrack-ng)

Amaç: Yakalanmış handshake dosyasını kullanarak WPA ağ parolasını bulmaya çalışmak. Sözlük saldırısı ile doğru parolanın sözlükteyse kolayca tespit edilebileceğini görmek.

- **Sözlükle kırma:** `aircrack-ng` -w custom_wifi_list.txt wifi_handshake.cap
- **Gerekirse BSSID ekle:** -b <BSSID>
- **Doğru parola bulunduğunda:** KEY FOUND! [parola]

Elimizde, bir kablosuz ağa ait handshake capture dosyası var (sağlanan **wifi_handshake.cap** dosyası). Bu adımda, bu handshake içerisindeki WPA parolasını bulmaya çalışacağız. Bu işlem offline olarak gerçekleştirilecek ve bir sözlük saldırısı uygulanacaktır.

- **Adım 4.a:** Sözlük Hazırlığı: Bizim hedef ağımızın (Siber Güvenlik Laboratuvarı Modemi) parolasının rockyou.txt gibi genel bir listede olma ihtimali düşüktür. Adım 3'te öğrendiğimiz gibi, hedef hakkında bildiklerimize dayanarak (örneğin "laboratuvar" adı ve "yıl" bilgisi) özel bir sözlük dosyası oluşturacağız.

```
echo "siberlab" > custom_wifi_list.txt
echo "siberlab2024" >> custom_wifi_list.txt
echo "SiberLab" >> custom_wifi_list.txt
echo "password" >> custom_wifi_list.txt
echo "12345678" >> custom_wifi_list.txt
echo "siberlab2025" >> custom_wifi_list.txt
```

- **Adım 4.b:** Aircrack-ng ile Kırma: Aircrack-ng, WPA handshake dosyasını doğrudan alıp her bir sözlük kelimesini deneyebilir. Temel komut formatı şöyledir:

  aircrack-ng -w <wordlist.txt> -b <BSSID> <capture.cap>

  Burada -b <BSSID> opsiyoneldir; belirtilirse sadece ilgili erişim noktasına ait handshake değerlendirilir. Tek bir handshake varsa opsiyonel olabilir. Bizim örneğimizde:

  aircrack-ng -w custom_wifi_list.txt wifi_handshake.cap

  şeklinde çalıştırınız. (Eğer çıktı birden fazla ağ bulduğunu belirtirse, -b parametresi ile dosyada bulunan hedef ağın BSSID'sini girmeniz gerekir. Wireshark ile .cap içine bakılarak bulunabilir.)

- **Adım 3.c:** Aircrack-ng komutu çalışmaya başlayınca sözlükteki kelimeleri tek tek denemeye başlar. Denenen anahtar sayısını ve tahmini kalan süreyi görürsünüz. Eğer doğru parola listede varsa, eşleştiğinde "KEY FOUND!" mesajı belirecektir.