İSTANBUL TEKNİK ÜNİVERSİTESİ ★ BİLİŞİM ENSTİTÜSÜ

VERİTABANI GÜVENLİK RİSKLERİ, ŞİFRELEME ALGORİTMALARI VE ENJEKSİYON MODELLERİ

YÜKSEK LİSANS TEZİ

Ertürk ERDAĞI

Bilişim Uygulamaları Anabilim Dalı

Bilgi Güvenliği Mühendisliği ve Kriptografi Programı

ARALIK 2017

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ BİLİŞİM ENSTİTÜSÜ

VERİTABANI GÜVENLİK RİSKLERİ, ŞİFRELEME ALGORİTMALARI VE ENJEKSİYON MODELLERİ

YÜKSEK LİSANS TEZİ

Ertürk ERDAĞI
(707141004)

Bilişim Uygulamaları Anabilim Dalı

Bilgi Güvenliği Mühendisliği ve Kriptografi Programı

Tez Danışmanı: Doç. Dr. Enver ÖZDEMİR

ARALIK 2017

İTÜ, Bilişim Enstitüsü'nün 707141004 numaralı Yüksek Lisans Öğrencisi Ertürk ERDAĞI, ilgili yönetmeliklerin belirlediği gerekli tüm şartları yerine getirdikten sonra hazırladığı "VERİTABANI GÜVENLİK RİSKLERİ, ŞİFRELEME ALGORİTMALARI VE ENJEKSİYON MODELLERİ" başlıklı tezini aşağıda imzaları olan jüri önünde başarı ile sunmuştur.

Tez Danışmanı :	**Doç. Dr. Enver ÖZDEMİR** İstanbul Teknik Üniversitesi	.....
Jüri Üyeleri :	**Doç. Dr. M.Oğuzhan KÜLEKCİ** İstanbul Teknik Üniversitesi	.....
	Yrd. Doç. Dr. Erdinç ÖZTÜRK Sabancı Üniversitesi	.....

Teslim Tarihi : 17 Kasım 2017
Savunma Tarihi : 21 Aralık 2017

Kıymetli eşime,

ÖNSÖZ

Bu tezin oluşturulması sürecinde rehberliği ile büyük emek veren Doç Dr. Enver ÖZDEMİR'e, Milli Eğitim Bakanlığı bünyesinde birlikte çalışmakta olduğum meslektaşlarıma, bilgimin ve deneyimimin temelinde büyük katkıları olan Marmara Üniversitesi Bilgisayar ve Öğretim Teknolojisi bölümü hocalarıma, 2006 yılından bu yana yer aldığım özel sektördeki tüm çalışma arkadaşlarıma, meslek hayatımdaki en büyük moral motivasyon kaynağım olan GoyGoy grubuna, bugünlere gelmemde büyük emekleri olan annem, babam ve kardeşime teşekkür ederim.

Aralık 2017 Ertürk ERDAĞI

İÇİNDEKİLER

Sayfa

KISALTMALAR

SQL	: Structed Query Language
T-SQL	: Transact Structed Query Language
ASP.NET	: .NET Based Active Server Page
MSQSQL	: Microsoft SQL Server
DB	: Database
DBMS	: Database Management System
SQLi	: SQL Injection
ADO.NET	: ActiveX Data Objects.NET
PROC	: Procedure
SP	: Stored Procedure
ID	: Identity
HTML	: Hyper Text Markup Language
SYS	: System

ŞEKİL LİSTESİ

Sayfa

VERİTABANI GÜVENLİK RİSKLERİ, ŞİFRELEME ALGORİTMALARI VE ENJEKSİYON MODELLERİ

ÖZET

Belirli biçimlere uygun olarak düzenlenmiş verilerin saklanmasını sağlayan satır bazlı yapılara veritabanı denir. Veritabanında bulunan bilgiler, belirli bir düzende bulunan, belirli biçimde saklanmış verileri içerdirğinden kapsam alanı belirlidir. Bu noktada veritabanı kapsamı kullanım amacına göre belli bir konuda, birbiriyle ilişkili bilgilere dayalı kayıtların bir araya geldiği ve bir araya gelen bu verilerin dijital ortamda saklanan yapılar olarak da değerlendirilebilir. Bir öğrencinin eğitim hayatı süresince kendine ait eğitim bilgilerini ve sonuçlarını görüntüleme ve güncelleme ihtiyacı; bir marketteki ürünlerin stok durumunun, fiyat, satış ve elde edilen kazanç miktarı gibi parametrelerinin izlenmesi ve güncellenmesi; bir şirketin ürünlerini dünyanın her noktasından takip edebilmesiyle veritabanı kavramı ve bu veritabanlarındaki bilginin düzenli bir şekilde izlenme ve denetlenme ihtiyacı doğmuştur. Veritabanı üzerinde işlemler, yalnızca yetki verilen kullanıcıların sahip olduğu bir durumdur. Bu işlem okuma, yazma, silme ve güncelleme olarak sınıflandırılmaktadır. Veritabanındaki bu dört işlemin yapılmasını sağlayan ve kullanıcıya tasarımsal anlamda birçok kolaylık sağlayan sistemlere veritabanı yönetim sistemleri ismi verilmektedir. Kullanılacak veritabanı yönetim sistemindeki erişim denetimi, güvenik izleme prosedürleri ihtiyaç ve bilgi kaynağına göre farklılıklar göstermektedir. Veritabanı içerisindeki bilgiler, kurum ve bireyler için hassas nitelikte değerlendirilebilecek bilgileri sakladığından veritabanı ve bu veritabanının yönetimini sağlayan veritabanı yönetim sistemleri büyük önem taşımaktadır. Güvenlik faktörü yalnızca teknolojik etkenler çerçevesinde değil, bu veritabanına yetkisi çerçevesinde ulaşan veya uygulamyı aktif olarak kullanan kullanıcı sebebiyle insan etkenine de bağlıdır.

Veritabanı yönetim sistemlerindeki en önemli olgu veritabanında var olan bilginin yetkisiz kişi veya kişilerce silinme, değiştirilme, güncelleme ve bilgi ekleme işlemlerine karşı alınacak önlemlerdir.

Veritabanı yönetim sistemleri uygulamanın kullanacağı veritabanını sıfırdan başlayarak kullanıma hazır hale getirinceye dek geçecek süreci, kullanım esnasında oluşabilecek güvenlik risklerine karşın denetleme işlemlerini yerine getirmektedir. Veritabanı yönetim sistemleri kullanım şekli ve kullanım ölçeğine göre farklılık göstermektedir. Yalnızca birkaç kullanıcının bulunduğu bir veritabanının yönetimi ile büyük ölçekli ve fazla sayıda kullanıcı içeren bir veritabanının yönetimi farklı olmaktadır.

Veritabanı Yönetim Sistemleri veritabanı içerisindeki veriyi sağlıklı bir şekilde sağlamak üzere birçok türde araç sağlamaktadır. Bu araçlar başlangıç seviyesindeki kullanıcıdan, bu anlamda çalışan profosonel kişi ve kurumlarca kullanılabilecek çeşitlilik ve kapsamda yer almaktadır. Veritabanları yapısına göre Hiyerarşik Model, Ağ Modeli, İlişkisel Model ve Nesne Yönelimli Model olmak üzere dört farklı modele ayrılmaktadır.

Veritabanı Yönetim Sistemleri üzerindeki zafiyetlerin kullanılarak ya da veritabanını kullanan kişi ya da kişilerce yapılacak hatalı kullanım ya da kötü niyetli işlemler neticesinde bazı tehditler oluşabilir. Oluşan bu tehditlerin ölçeği küçük dahi olsa güvenlik noktasında büyük bir risk oluşturmaktadır. Veritabanı güvenliğini en fazla etkileyen etmenlerden biri insan faktörüdür. Bu noktada kişi ya da kişilerin bilerek ya da bilmeyerek yapmış olduğu işlemlerden ötürü veritabanı güvenliğinde ihlal olurken, aynı zamanda kritik bilgilerin kullanımına bağlı olarak başka zafiyetler ortaya çıkmaktadır. İnsan faktörünün etkili olmasında bilinç eksikliği ya da veritabanı güvenlik politikalarına uyulmaması, veritabanının bulunduğu makinaların fiziki güvenliğinin sağlanmaması, kullanıcının yeterli bilgi düzeyinin olmaması, yetki kuralına dikkat edilmemesi, veritabanındaki iş ve işlemlerin kayıt altına alındığı sistemlerin eksik olması, yedekleme ve bakım işlemlerinin yeterince yapılmaması etkilidir.

İnternet ortamında çevrimiçi çalışan bir uygulamanın veritabanı kullanımı için SQL kullanılmaktadır. Çevrimiçi bir uygulamanın hazırlanmasını sağlayan ASP.NET, PhP gibi internet tabanlı uygulama dilleri veritabanı yapısı üzerinde sorgulama ve ekleme işlemleri için SQL yapısal programlama dilini kullanmaktadır.

Kullanıcıların çevrimiçi bir uygulamada internet tarayıcısı üzerinden yaptıkları talepler veritabanı üzerinde veri girişi ya da veri sorgulamasını sağlamaktadır. Bu veri giriş ve sorgulama işlemleri bazı riskleri de taşımaktadır. Sorgulama sırasında kullanılan parametreler üzerinde yapılacak manipülasyonlar zafiyete neden olmaktadır. Bu zafiyet durumu veritabanındaki erişim yetkisi olmayan kişilerin bilgilere erişmesini sağlarken, aynı zamanda bu verilerin üzerinde değişiklik yapılmasını da sağlamaktadır. Oluşan bu duruma enejksiyon adı verilmektedir. Enjeksyon saldırısına karşı alınabilecek en temel önlem parametre değerlerini bazı özel karakterlere karşı bir fonksiyon ile kontrol edilmesidir. Veritabanı kullanan uygulama ve internet sitelerinin enjeksiyona korunaklı hale getirilmesi için , veritabanını kullanan uygulamanın mevcut durumunun analiz edilmesi, tüm açıkların tespit edilmesi yapılacak ilk adım olacaktır. İkinci adımda uygulamaya ait kodlar üzerinde standart bir yapı uygulanmalı ve veritabanına ait her işlem için güvenlik denetiminin uygulanması gerekmektedir. Üçüncü adımda ise veritabanı güvenliğini zaafiyete uğratabilecek noktaların tespit edilmesi ve ek güvenlik önlemlerinin alınması gerekmektedir.

DATABASE SECURITY RISKS, ENCRYPTION ALGORITHM AND INJECTION MODELS

SUMMARY

A database stores data organized in accordance with certain formats. The information contained in the database is organized and includes data generated for a specific purpose. At this point, the database scope can be regarded as a collection of related information-based records in a particular context, according to the purpose of use, and the structures that are contained in the digital environment. The need for a student to view and update his / her educational information and results during his / her education life; tracking and updating parameters such as stock status, price, sales and profits of products in a market; the ability to track a company's products from every corner of the world has created the need for a database concept and the regular monitoring and monitoring of information in these databases. As long as the database is authoritative, some users are authorized to perform transactions. This process is classified as read, write, delete and update. Database management systems are named for the systems that provide these four operations in the database and provide many convenience to the user in a design sense. Access control in the database depends on the security area of the database.. Since the information in the database stores is sensitive for institutions and individuals, database management systems that manage this database are of great importance. The security factor is not only based on technological factors, but also depends on the human factor because of the users who acces this database within the framework of authority or actively use the application.

The most important fact in database management systems is the measures to be taken against deletion, modification, update and addition of information in the database by unauthorized person or persons.

The database management systems will be able to use the application from scratch to ready to use, providing the ability to inspect against security risks that may occur during use. Database management systems differ in terms of usage. The management of a database with only a few users and the management of a database with a large numbers of users is different.

Database Management Systems provide many kinds of tools to input the data in the database in a healthy way. These tools are available in a variety of forms. There are four different types of databases according to their structure: Hierarchical Model, Network Model, Relational Model and Object Oriented Model.

Some threats may arise from the use of weaknesses in Database Management Systems or due to improper use or malicious actions to be made by the person or persons using the database. Even if the scale of these threats is small, it pose a great risk to safety. One of the most influential factors in database security is the human factor. The inadequate level of knowledge of the people using the system may lead to the failure

of security policies to be implemented correctly and the inability to regularly maintain and repair the machines in the system.

ASP.NET, PhP, etc., use SQL structured programming language for querying and adding operations on database structure. Requests made through the user's internet browser results in data entry or data query on the database. This data entry and inquiry process also carries some risks. The manipulations to be made on the parameters used during the query are causing the failure. This weakness allows non-authorized access to the database, while at the same time permitting modification of the data. This situation is called the injection. The most basic measure against injection attack is to control parameter values with a function against some special characters. The analysis of the current state of the application using the database will be the first step in identifying all the deficiencies. In the second step, a standard structure should be applied to the application codes and security audit must be applied for each transaction belonging to the database. It is also necessary to identify the points where the database security is weak and additional security precautions must be taken.

To reduce the damage of a successful SQL injection attack to a minimum, the privileges reserved for each database account must be restricted. If the database accounts used by application accounts are not database administrators, this will be an obstacle to providing any kind of access. Instead of the authorization to be given to the persons, the operation must be done according to the authorization required by the application. An account that has a read access needs to have only the read access. If an account needs to access to a table of information, then a structure must be created that limits access to that part of the table and provides access to the information in the table instead of the table below the account.

In this study, it is shown how to construct a structure to protect from injection. In the ASP.NET structure and in a web application prepared in C # programming language, it has been transferred as a processing application that needs to be done on the database in order to protect users from injection. For the functions that provide query and modification of the data on the database, the information sent from the application is kept with parameters. The manner in which this parameter and function is retained is largely preserved from the injection attack. This structure, which is constructed, covers the work done in the programming language stage and is not enough by itself. However, the forms of access to users' databases include authorization and all other actions that may be recorded, followed by monitoring methods and risks.

A project has been carried out on two different platforms in line with the research done. The project creates student classroom tracking software designed on Microsoft SQL Server Management. In this software, a web-based project has been realized on the database which is created by using .NET technology. In the project, records and inquiries of the data have been realized with the table and stored procedure structure. During this process, all security vulnerabilities in the thesis content are audited. In the project design section, the stored procedure and table structures are also detailed in the thesis.

The second project is based on the MYSQL database, which has a lot of usage space. The project theme is to analyze/monitor/audit the network traffic. The project was conducted by using MYSQL database and ASP.NET technology, two tables were used to query data and to update and add data. In the content of the project, the inquiries made within each page were taken to protect against the threat of injection.

1. GİRİŞ

Dijital ortamda kullanılan verilerin saklanması, güvenli bir biçimde ileriye dönük kalması ve her ortamdan rahatlıkla ulaşılabilmesi veritabanı ihtiyacını doğurmuştur. Veritabanı belirli biçimlere göre ayrılmış ve bütünlük içeren yapılar içerisinde saklanan dijital ortamlardır. Veritabanları tablolar içerisinde verileri barındırırlar. Tablodaki bilgilerin sorgulanması, silinmesi, güncellenmesi veya bu tablolara yeni bilgi eklenmesi işlemi sırasında oluşabilecek güvenlik riskleri verileri tehlikeye düşürebilir. Bu risklerin en aza indirgenmesi için gerekli önlemler titizlikle alınmalıdır. Veritabanı kavramı yalnızca bir internet sistesindeki verilerin saklandığı ortamı değil, bir marketin ürünlerinin stok ve fiyat bilgisinin de saklanmasını sağlar. Çok fazla kullanım alanı olduğundan güvenlik boyutu da önem taşımaktadır. Önem derecesi ne olursa olsun bir bilginin veritabanında saklanırken güvenliğini tehlikeye düşürecek bir durumun giderilmesi yönünde çalışmalar gün geçtikte gelişmektedir. Nitekim veritabanlarının karşılaşacakları tehlikeler gün geçtikçe farklılaşmaktadır. Risklere karşı koymak adına öncelikle veritabanı yapısının iyi bilinmesi ve karşılaşılabilecek olumsuzluklara ilişkin önlem alınması gerekmektedir. Bu noktada öncelikle veritabanının yapısı detaylandırılmış, sonrasında karşılaşılabilecek risklere ilişkin güvenlik tedbirleri örnekler ile sıralanmıştır.

1.1 Tezin Amacı

Bu tez kapsamında yerel ya da internet bazlı tüm bilgi kaynağının saklanmış olduğu veritabanı yapısının ne olduğu ve nasıl kullanıldığı açıklanmaya çalışılmıştır. Yapılan çalışma veritabanı güvenliği için öncelikle veritabanı yapısının detaylarının bilinmesi gerektiği ve çalışma prensiplerine dayalı güvenlik prosedürlerinin anlaşılması gerektiği prensibine dayalı olarak başlamıştır. Veritabanı içerisindeki güvenlik risklerinin teknik ve insan faktörü olarak ele alınarak, gerek teknik anlamda gerek insan faktörünü ilgilendiren bilinç düzeyinin artırılması gerektiği düşüncesiyle veri

kaynağı içerisinde karşılaşılabilecek tüm risklerin ana hatlarıyla ortaya çıkarılması ve bu risklere karşı alınabilecek önlemlerin sıralanması amaçlanmıştır.

Hazırlanan bu çalışma dışında veritabanları her geçen gün yeni ve daha gelişmiş güvenlik riskleri ile karşı karşıya kalmaktadır. Basit bir uygulama seviyesinden, bir devlete ait kaynakların yönetilmesine kadar geniş bir yelpazede kullanılan veritabanı kavramının günlük hayatta bu yapıyı kullanmayan kişiler için de bireysel bilgilerinin yetkisiz kişilerin eline geçmesi sonucu oluşabilecek olumsuzlukların önüne geçilmesi adına bir rehber niteliğinde hazırlanmıştır.

2. VERİTABANI GÜVENLİĞİ

2.1 Veritabanı Nedir ?

Kullanım amacına uygun olarak belirli formlatara uygun olarak düzenlenmiş veriler topluluğuna veritabanı denir. Veritabanında bulunan bilgiler belli bir konuyu kapsadığından kapsam alanı bulunmaktadır. Bu noktada veritabanı kapsamı belirlenmiş belli bir konuda, birbiriyle ilişkili kayıtların bir araya geldiği ve dijital ortamda saklandığı yapılarak olarak da değerlendirilebilir. Günlük hayatta kullanım alanı oldukça yaygın olan veritabanları ticari kaygı farketmeksizin birçok şekilde kullanılmaktadır. Bir öğrencinin kendine ait bilgileri kapsam doğrultusunda görüntüleme ve güncelleme ihtiyacı; bir marketteki ürünlerin stok, fiyat, satış ve kar durumlarının izlenmesi ve güncellenmesi; bir holdingin ürünlerini dünyanın her noktasından tespit edebilmesi ve ithalat ihracat dengesini görebilmek için izleme sistemleri veritabanı ihtiyacını doğurmuştur. Görüldüğü üzere büyük - küçük işletmelerde ya da ticari olmayan ortamlarda da kullanım alanı bulunmaktaır. Belli bir kapsamda bulunan verinin saklanması nedeniyle çeşitli ihtiyaçlar doğrultusunda işlenmesi gerekmektedir. Bu işlem okuma, yazma, silme ve güncelleme olarak değerlendirilebilir. Veritabanındaki bu dört ana işlemin yapılmasını sağlayan yapılara veritabanı yönetim sistemleri ismi verilmektedir. Veritabanı yönetim sistemleri temel olarak;

- Veritabanında bulunan bilgilerin saklandığı tabloları oluşturmak
- Tablolara bilgi eklemek, silmek veya güncellemek
- Bilgilerin koşullu ya da koşulsuz olmak üzere sorgulanması ve sorgu sonucuna göre raporlanması
- Bilgiye erişim izinlerini belirlemek
- Bilginin yedeklenmesini sağlamak

- Kendi içindeki ya da harici araçlar ile bilginin analiz edilmesi gibi işlemleri gerçekleştirir.

2.2 Veritabanı Yönetim Sistemleri

Kurum ve bireylerin hassas olabilecek nitelikte bilgileri tutulduğundan veritabanı ve bu veritabanının yönetimini sağlayan veritabanı yönetim sistemleri büyük önem taşımaktadır. Gün geçtikçe gerek web tabanlı, gerek mobil tabanlı yazılımların artması ve bu yazılımların internet ortamı üzerinden bilgi transferi gerçekleştirdiğinden veritabanı güvenliğini daha önemli kılmaktadır. Güvenlik faktörü yalnızca teknolojik imkanlar çerçevesinde değil, bu veritabanına ulaşan veya aktif kullanan kullanıcıdan ötürü insan etkenine de bağlıdır. Bugüne dek veritabanının güvenliğini sağlama noktasında çeşitli mimari yapılar kullanılmıştır. Bu yapılar [1];

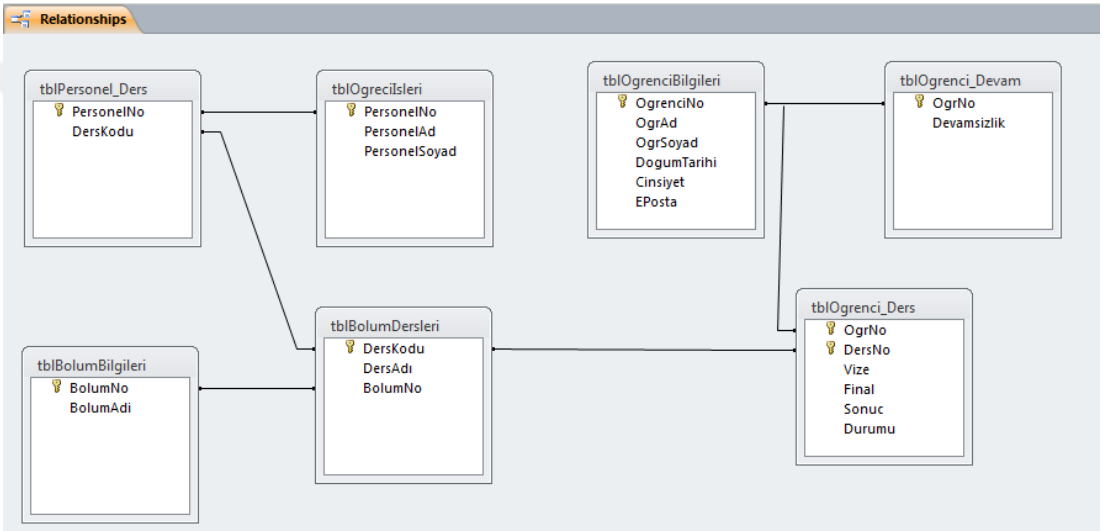
- Sayfaları ve kapsamlarını mimarisi
- Dosya (filegroup) mimarisi
- İşlem günlüğü mimarisi
- Tablo ve dizin veri yapıları mimarisi
- Sorgu işleme mimarisi
- Bellek yönetimi mimarisi
- İş parçacığı ve görev mimarisi

şeklinde sıralanmaktadır. İhtiyaç ve yapıların çeşidine göre bu güvenlik önlemlerinin sayısı gün geçtikçe artmaktadır. Veritabanı yönetim sistemlerindeki en önemli olgu var olan bilginin yetkisiz kişi veya kişilerce silinme, değiştirilme, güncelleme ve bilgi ekleme işlemlerine karşı alınacak önlemlerdir. Bir veritabanındaki bilginin yetkisiz kişilerin eline geçmesi veya bahsedilen manipülasyon biçimleri ile değiştirilmesi veritabanı güvenliğinin ihlalini ortaya çıkarmaktadır.

Yeni bir veritabanını oluşturmak, oluşturulan bir veritabanını düzenlemek, geliştirmek, bakımını yapmak gibi işlemleri gerçekleştirmek için kullanılan yazılımlara veritabanı yönetim sistemleri denir. 1960'lı yıllarda tasarlanan ilk veritabanı yönetim sistemi sonrasında farklı modelleri de içerisinde bulunduran veritabanı yönetim sistemleri ortaya çıkmıştır. 1970'li yılların sonunda ise ilişkisel veritabanı modeli şu an kullanılan veritabanı yönetim sistemlerinin temelini oluşturmuştur. [2]

Veritabanı yönetim sistemleri uygulamanın kullanacağı veritabanını hazır hale getirmek, denetleme işlemlerini sağlamak ve veritabanının güvenliğini sağlamak adına bazı araçlar sunmaktadır. Veritabanı yönetim sistemleri kullanım şekli ve kullanım ölçeğine göre farklılık göstermektedir. Yalnızca birkaç kullanıcının bulunduğu bir veritabanının yönetimi ile büyük ölçekli ve fazla sayıda kullanıcı içeren bir veritabanının yönetimi farklı olmaktadır.

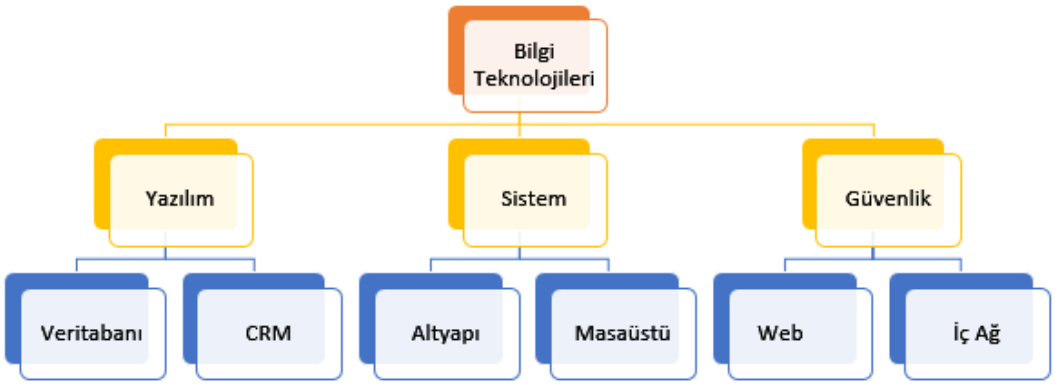
Veritabanı Yönetim Sistemleri bilgiyi sağlamak üzere birçok türde nesne, verilere erişmek üzere kullanılacak rol ve grup modellerini barındırmaktadır. Veritabanları yapısına göre dört farklı modele ayrılmaktadır.



Şekil 2.1 : Örnek bir veritabanı modeli

3. VERİTABANI MODELLERİ

3.1 Hiyerarşik Veritabanı Modeli



Şekil 3.1 : Örnek hiyerarşik veritabanı modeli

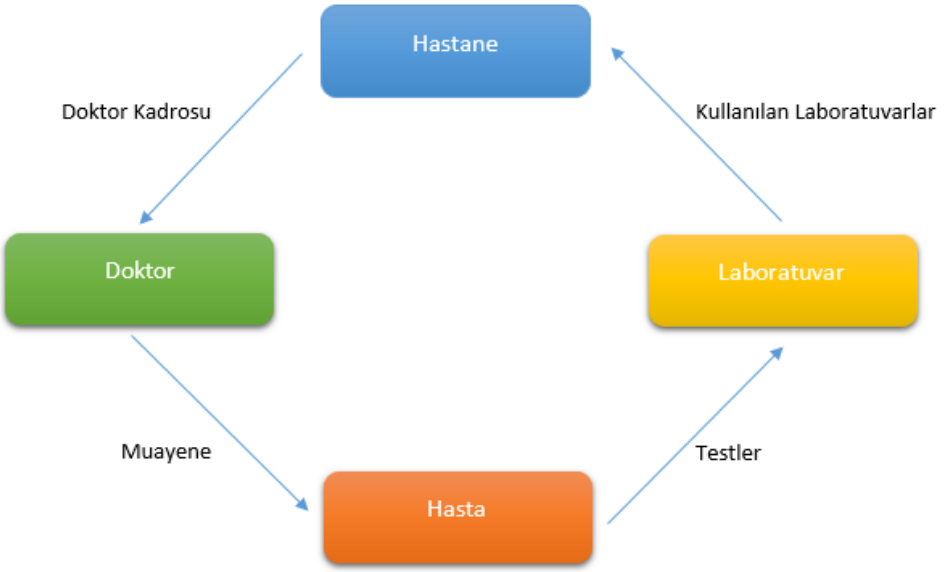
1960 önesinde Charles Bachman tarafndan ilk Veritabanı Yönetim Sistemi geliştirilmişti [3]. Bu veritabanı yönetim sistemi temel alınarak oluşturulan Hiyerarşik Veritabanı Modeli verinin depolanması ve işlenmesi için oluşturulan ilk veritabanı modelidir. İlk kullanılan model olarak gelişmiş bir yapı bulunmaması sebebiyle yaygınlaşmamıştır. Bu model ilk olarak 1968 yılında IBM tarafından geliştirilmiş ve yalnızca o tarihlerde kullanılan ana bilgisayarlarda kullanılmıştır. Belirtilen yıllarda PC (Personal Computer)'lerin yaygın olmaması sebebiyle mainframe üzerinde kullanılmıştır.

Hiyerarşik Veritabanı Modeli içerisinde bilgi bir ağaç şeklinde depolanmaktadır. Bir kaydın kök olarak en üstte olması ve bu bilgiye bağlı alt bilgilerin oluşturduğu yapıyı içermektedir. Alt dallara doğru inilirken kullanılan yapı ata ve çocuk olarak isimlendirilmektedir. Bir kaydın üst kısmına o bilginin atası, bir kaydın alt bilgisine o bilginin çocuğu adı verilir. Bu modelde 'birden çok' yapı kullanılır. Bu yapı içerisinde bir kaydın başka bir veya daha fazla tabloda karşılığı olabilen (kullanıcı tablosuna karşılık olarak, siparişler tablosunda bu kullanıcı için birden fazla kaydın bulunması),

herbir alt kaydın kendisine bağlı bir üst kaydının olduğu bilgi bulunmaktadır. Burada örnek olarak bankalar ve şubeler arasındaki ilişki verilebilir. Bir bankanın birden çok şubesi bulunabilir fakat her şube yalnızca bir bankaya aittir. Her şubede birçok çalışan olabilir fakat her çalışan bir bankaya bağlıdır. Bu modelde bir bilgiye ulaşmak için çoğu zaman ana bilgiden başlanarak alt bilgilere doğru araştırma yapılması gerekmektedir.

3.2 Ağ Tabanlı Veritabanı Modeli

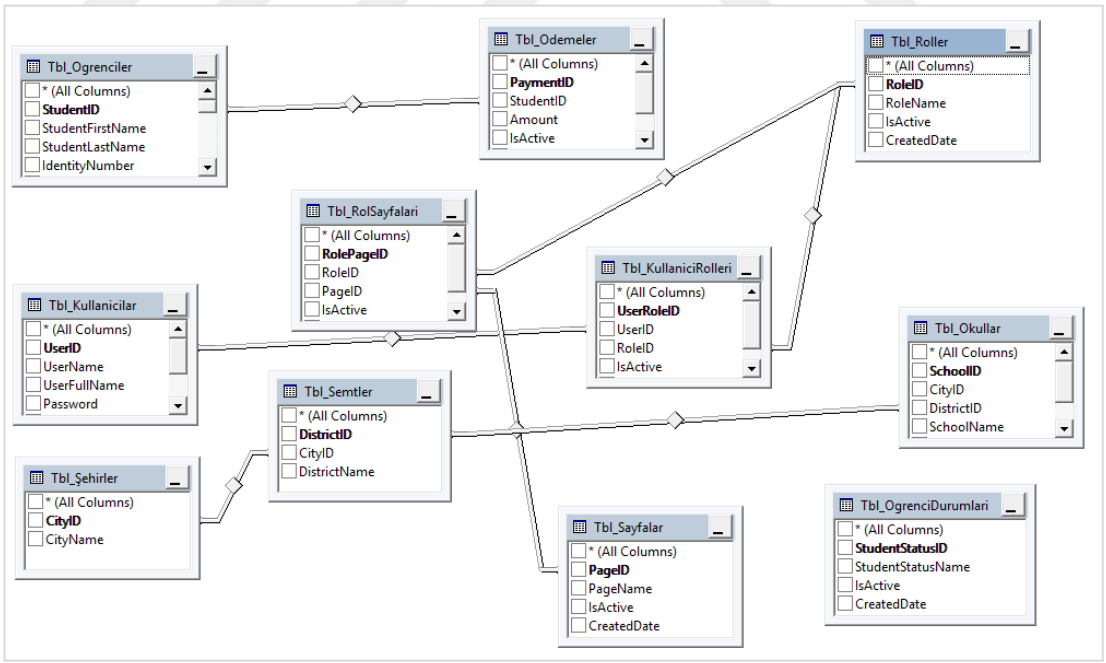
1969'da ilk olarak ortaya çıkan Ağ Tabanlı Veritabanı Modeli 1980'li yılların ilk yarısına kadar kullanılmıştır. İlk olarak 1960'lı yılların sonunda CODASYL (Conference On DAta Systems Languages) modeli kapsamında tanıtılan bu yapı, Hiyerarşik Veritabanı Modeli'nin yetersiz kalması sebebiyle ortaya çıkmıştır [3]. Bu modelde ağ şeması, yazılımsal dil ve alt şema olmak üzere üç yapı bulunmaktadır. Tablo ve grafik temeline dayanan bu modelde veri tipleri tablolar şeklinde temsil edilir. Temelinde bulunan grafik okları ilişkileri temsil eder ve bağlantı olarak yer alır. Kayıt tipi ve bağlantı olmak üzere iki ayrı yapılandırma aracı bulunmaktadır. Bilginin türünü kayıt kısmı belirlerken, ilişki tipimi ise bağlantılar belirler. Yapısal olarak hiyerarşik modele benzemektedir. Farklı yönü ise bağlantı açısından herhangi bir sınırlamasının olmamasıdır. Çoklu ilişki modelinin bulunmamasından ötürü kısıtlıdır.



Şekil 3.2 : Ağ tabanlı veritabanı modeli

3.3 İlişkisel Veritabanı Modeli

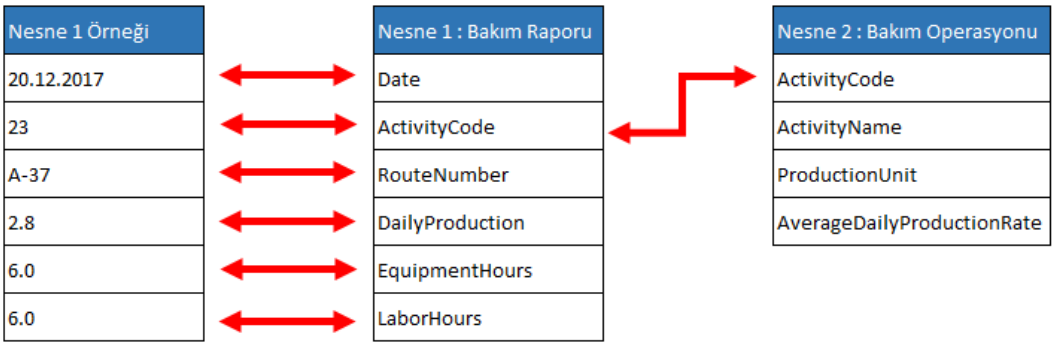
1970 yılında Ted Codd'un, IBM San Jose Laboratuvarı'nda tanımladığı ilişkisel veri modeline dayalı dijital veritabanı modelidir. Günümüzde en fazla kullanılan veritabanı modeli olan İlişkisel Veritabanı Modeli öncesinde kullanılan klasik dosya sisteminin ilişkileri oluştururken ortaya çıkan performans ve kullaınm olumsuzluğu bu modelin oluşmasında büyük bir etken oluşturmuştur. Modelde veriler biçimlenmiş ve belirli yapılarda bulunan tablolarda yer almaktadır. Oluşturulan tablolar arasındaki ilişki modelinde satırlar bilgiye ait kaydın kendisini, sütunlar bu kayda ait herbir özelliği temsil etmektedir. İlişkisel model için örnek olarak banka, şube ve çalışanlar arasındaki yapı gösterilebilir. Burada üç farklı veri tipi üç farklı tabloda saklanmakta olup, şube tablosunda hangi bankaya ait olduğuna ilişkin bilgiyi tutan banka bilgisi bulunarak banka tablosu ile ilişkilendirilmekte, çalışanlar tablosunda ise şube ve banka bilgileri ile iki tabloyla ilişki oluşturulmaktadır. Bir bankanın birden fazla şubesinin olması, bir banka şubesinin birden fazla çalışanının olmasına olanak veren bu yapı, günümüzde en fazla kullanılan modellerden biridir.



Şekil 3.3 : İlişkisel veritabanı modeli

3.4 Nesne Yönelimli Modeli

1976 yılında Peter Chen tarafından tanımlanan bu modelde veriler nesne şeklinde tutulmaktadır. Diğer modellerde herbir kayıt için diğer tablolarda bulunan kayıtlar ile ilişki kullanılırken, bu modelde veriler nesne ve alt özellikleri şeklinde yapılandırılmaktadır. Bu model ile birlikte kullanılacak yazılımda nesne yönelimli programlama kullanıldığında oluşacak yapı nesne yönelimli veritabanı modeli olarak adlandırılmaktadır. Yapı olarak aramanın hızlı olması büyük bir avantaj olarak karşımıza çıkmaktadır. Saklanan nesneler kendi içlerinde başka nesneler de barındırabilir. Bu model özellikle son yirmi yılda fazlasıyla kullanılan bir model olup, giderek yaygınlaşmaktadır. Hiyerarşik Veritabanı Modeli ve Ağ Tabanlı Veritabanı Modeli günümüz şartlarında kendilerini koruyamadığı gibi ihtiyaçlar doğrultusunda gelişen durumlara İlişkisel Veritabanı Modeli de karşı koyamayarak yerini Nesne Yönelimli Model'e bırakmaktadır. Bu model nesne yönelimli programlama modelinin son yıllarda popüler olması sebebiyle kullanım alanı olarak daha fazla yer bulmaktadır.



Şekil 3.4 : Nesne tabanlı veritabanı modeli

4. ÖRNEK VERİTABANI SALDIRILARI

Veritabanı yönetim sistemlerine yapılan saldırılar yapılan saldırının ve hedef edinilen veritabanının büyüklüğüne göre literatürde kendisine yer bulmaktadır. Büyük ölçekli veritabanları üzerinde yapılan saldırılar edinilebilecek kayıpların fazla olması sebebiyle önemli olsa da bu ölçekte veritabanları kurumsal anlamda güvenlik denetimlerine tabi tutulmaktadır.

Amerika Birleşik Devletleri'nin iç istihbarat ve güvenlik gücü olan FBI, Rus mafyası ile ilgili yaptığı açıklamada yirmili yaşlardaki mafyanın bilgisayar korsanlarının kullanılarak e-ticaret şirketlerinin bilgisayar ve uygulamalarındaki kredi kartı, banka hesabı, kişisel bilgilerin ele geçirdigini açıklamıştı.

Have I Been Pwned (HIBP) adlı internet sitesinin kurucusu, siber güvenlik uzmanı Troy Hunt, çeşitli zamanlarda, çeşitli yöntemlerle ele geçirilmiş 306 milyon şifreyi dijital ortamda yayınladı. Bu şifreler çeşitli uygulamarın kendi güvenlik zafiyetlerinin yanında, kullanıcıların bilerek ya da bilmeyerek yapmış oldukları işlemlere dayalı tuzakların neticesinde ele geçirilmiştir. Uygulama içerisinde gizlilği yüksek derecede olması gereken şifrelerin herhangi bir şifreleme algoritması ile şifrelenmeden olduğu gibi yayımlaması ve veritabanı ele geçirildiğinde kullanıcıların var olan şifrelerinin de ele geçirilmesi tespit edilen önemli bir nokta olarak kayıtlara geçti. ABD Ulusal Teknoloji ve Standartlar Enstitüsü ve Birleşik Krallık Siber Güvenlik Merkezi tarafından bu açıklama da desteklenmiş oldu.

2000 yılında CD Universe isimli internet tabanlı müzik satış sitesine bir hacker tarafından yapılan saldırı sonrasında sitenin aktif olarak kullandığı veritabanındaki kişilere ulaşılmış, buradaki kredi kartı ve kişisel bilgiler kopyalanmıştır. Bilgilerin iade edilmesi karşılığında para talep eden hacker, isteği gerçekleşmeyince sitedeki tüm kullanıcıların kişisel bilgileri ve finansal bilgilerini içeren kredi kartı ve hesap bilgilerini bir siteye yüklemiştir.

2010 yılında Türkiye Cumhuriyeti'nde bulunan tüm vatandaşların kimlik bilgilerinin saklandığı MERNİS (Merkezi Nüfus İdare Sistemi) içerisindeki veritabanı üzerinden çalındığına yönelik haberler gündemde yer bulmuştu. Burada vatandaşların kimlik bilgilerinin yer aldığı sistemin güvenliği tartışılır hale gelmişti. Kurulan uzman bir ekip ile sistemin güvenliği kontrol edilirken var olan güvenlik açığının yanında birçok açık da tespit edilmiş ve giderilmesi için çalışma başlatılmıştır.

5. VERİTABANI RİSKLERİ

Veritabanı risklerinin oluşmasında teknik boyutun haricindeki insan faktörünün etkili olmasında bilinç eksikliği ya da veritabanı güvenlik politikalarına uyulmaması, veritabanının bulunduğu makinaların fiziki güvenliğinin sağlanmaması, kullanıcının yeterli bilgi düzeyinin olmaması, yetki kuralına dikkat edilmemesi, veritabanındaki iş ve işlemlerin kayıt altına alındığı sistemlerin eksik olması, yedekleme ve bakım işlemlerinin yeterince yapılmaması etkilidir. [2]

Şekil 5.1 : Veritabanı riskleri

Veritabanlarına ilişkin riskler 10 başlık altında toplanmıştır:

5.1 Aşırı Yetki Tehditi

Veritabanı üzerinde bir kullanıcının var olan yetki çerçevesinde yapabileceği işlemler dışında farklı alanlara da ulaşabildiği durumlar meydana gelebilir. Bu durumda ilgili veritabanı için erişim denetiminin sağlıklı olmadığı sonucu çıkarılabilir. Örneğin bir hastaneye ait otomasyon sisteminin veritabanı üzerinde hastaların bilgileri üzerinden yalnızca kullanıcıların randevularının bulunduğu tabloyu görmesi gereken bir kullanıcı, erişim denetiminin sağlıklı yapılmaması nedeniyle kullanıcıların kişisel bilgilerine de ulaşması bu tehditin meydana geldiğini göstermektedir. Bu durumda kullanıcı veritabanı içerisindeki kullanıcı bilgilerini değiştirebilir, silebilir ya da kötü niyetli kişi ya da kişilere verebilir. Bu durumu gidermek adına veritabanı üzerinde sağlıklı erişim denetimi yaplması gerekmektedir. Birçok veritabanı yönetim sisteminde sorgu seviyesinde basit anlamda erişim denetimi yapılmaktadır. Fakat yapılan bu erişim denetimi basit düzeyde olmaktadır. Bu erişim denetiminin sağlıklı olması için, denetim işleminin uygulamalardaki kullanıcı düzeyinin yanı sıra, sunucu düzeyinde de yapılması gerekmektedir. Gelişmiş yapılar üzerinde uygulama tarafından gelen bir sorgunun kullanıcının yetkisi dahilinde olup olmadığı kontrol edilir, eğer yetkisini aşan bir durum var ise yapılan işlem bildirilmek üzere kayıt altına alınır ve talep edilen bilgi verilmeyerek iptal edilir.

5.2 SQL Enjeksiyonu

Sorguların çalıştırılmasına dayanan bu tehdit tipinde kullanılan sorgu içerisinde yapılan müdahaleler neticesinde gerçekleşir. Veritabanı ve uygulama üzerindeki önlemlerin yetersiz olması ya da bilgi eksikliğinden kaynaklanan durumlardan ötürü oluşan bir durumdur. SQL Enjeksiyonu ve NoSQL Enjeksiyonu olmak üzere iki tipi bulunmaktadır. SQL Enjeksiyonu geleneksel veritabanı üzerinde giriş bilgilerine yapılan müdahalelerden oluşmaktadır. Burada yetkisiz kullanımı da içeren bir tehdit bulunmaktadır. NoSQL Enjeksiyonu ise büyük veri içeren veritabanlarını hedef almaktadır. Burada sorgu içerisine kötü niyetli ifadeler enjekte edilerek işlem gerçekleştirilir. Bu tehdit tezin ana konusunu oluşturduğundan daha sonraki bölümde detaylı bir şekilde açıklanacaktır.

5.3 Yetki Suistimali

Veritabanı üzerinde yetkisi dahilindeki işlemlerin olumsuz sonuçlar doğuracak şekilde sonuçlanmasına neden olan tehdittir. Burada erişim denetimi sağlıklı bir şekilde ilerlemeyen bir veritabanında yetkisi kısıtlı bir kullanıcının uygulama ya da veritabanındaki bir açıklıktan faydalanarak kendi yetkisini artırması ve bu yetki çerçevesinde yapmış olduğu işlemler bu tehdit içerisinde değerlendirilebilir. Bu tehditler sunucu bazlı seviyede IPS adı verilen atak önleme sistemleri ve sorgu düzeyinde yapılacak erişim kontrolleriyle engellenmektedir. IPS üzerindeki trafiğin incelenmesi sonucunda oluşan anormal durumlar analiz edilerek kaynak noktasından önlem alınabilir.

Bu tehdit üzerinde örnek olarak sorgu seviyesinde denetimin yapılmadığını düşünelim. Bu kapsamda sorgu içerisinde yetkisi kısıtlı bir kullanıcı SQL ile var olan kullanıcı yetkisini veritabanı sahibi ya da veritabanı yöneticisi olarak güncelleyebilir. Güncellemenin erişim denetimine takılmaması sonucunda kullanıcıda var olan sınırsız yetki ile veritabanı üzerinde manipülasyonlar gerçekleştirilebilir ya da var olan bilgi yetkisiz olarak ele geçirilebilir.

5.4 Kötü Amaçlı Yazılımlar

Hassas verilerin çalınması ya da kullanılması için e-posta, website ve kişisel bilgisayarlar üzerinde kullanılan yöntemler veritabanındaki hassas bilginin elde edilmesi için de kullanılabilir. Bu noktada bilginin çalınması ya da deforme edilmesi bu kapsamda değerlendirilebilir. Bu yöntemden korunmak için veritabanının bulunduğu sunucuda güvenlik duvarının etkin hale gelmesi, antivirüs sisteminin kurulması ve etkin şekilde erişim denetiminin yapılması gerekmektedir.

5.5 Zayıf Denetim

Veritabanı üzerindeki güvenlik politikasının gereken düzeyde olmaması risklere açık olduğu anlamına gelir. Önemli verileri içeren veritabanı işlemlerinde yapılan her türlü işlem kayıt altına alınarak sonrasında analiz edilmek üzere izleme faaliyetleri yürütülmektedir. Veritabanında gerçekleşen işlemlere ilişkin ayrıntılı denetim kayıtlarının olmaması ciddi bir risk oluşturmaktadır. Zayıf veritabanı denetleme mekanizmaları uygulama ve kuruluşlar için risklerin artması anlamını taşımaktadır.

Yönetici yetkisine sahip bir kullanıcının erişim denetim kontrolünü kapatması neticesinde kontrollerin devre dışı kalması ve veritabanının savunmasız hale gelmesi bu duruma örnek gösterilebilir. Bu noktada yalnızca veritabanı seviyesinde değil, veritabanının bulunduğu sunucu üzerinde de denetimin yapılmasını zorunlu kılmaktadır.

5.6 Yedek Açıklığı

Veritabanına ait bilgilerin yedeklendiği depolama ortamı genellikle dış ortamdan gelecek saldırılara karşı korunmasızdır. Bu saldırılar yalnızca dijital ortam üzerinden yapılan saldırılar ile kısıtlı olmayıp, veritabanı yedekleme diskleri ve kasetlerinin fiziki olarak zarar görmesi, kayıp ya da çalıntı durumları da dahildir. Ayrıca önemli bilgilerin düşük seviyede erişime sahip veritabanı kullanıcısına açık olması yedekleme faaliyetlerini denetleme ve izleme noktasında eksiklik oluşturabilir. Önemli verileri içeren veritabanına ait yedek kopyalarını korumak yalnızca bütünlüğü koruma amaçlı olmayıp birçok şirket ve uygulamanın güvenlik politikası haline gelmiştir. Yedekleme işleminin belirli zamanlarda ve yalnızca erişim yetkisi bulunan kullanıcılar tarafından yapılması önemlidir. Kurumsal yapılar üzerinde yedekleme işlemleri günlük, haftalık ve aylık periyotlarda otomatik olarak alınmaktadır. Alınmış bir yedeğin uygulamaya ait aynı fiziksel ya da sanal makina üzerinde bulundurulmaması esastır. Nitekim meydana gelen olumsuz bir durumda hem uygulamanın, hem de yedeğin zarar görmesine neden olacaktır.

5.7 Güvenlik Açığı ve Yanlış Yapılandırma

Veritabanı üzerinde kötü niyetli kullanım için güvenlik açıkları bulunabilir. Bu noktada güvenlik açıklarının kaynağının çözüm bulunabilmesi adına uygulama, sunucu ve işletim sistemi üzerinde araştırılarak belirlenmesi gerekmektedir. Herbir noktada yapılan analizler için oluşturulacak önlemler birbirlerinden farklıdır. Bu önlemler kendi alanında uzmanlık isteyen birçok işlemi kapsamaktadır. Kullanılan veritabanına ait üretici firma tarafından yayınlanan güvenlik yamalarının düzenli olarak uygulanması, açıklanan tehditlere ilişkin veritabanı seviyesinde önlem alınması büyük önem taşımaktadır. Bunun yanında uygulamanın kullandığı veritabanına ilişkin yanlış yapılandırma neticesinde kontrollerin devre dışı kalması neticesinde güvenlik riskleri artacaktır. Gerek işletim sistemi seviyesinde gerek veritabanı seviyesinde

güncellemelerin takip edilmesi ve sistematik olarak uygulanması bu riskin engellenmesini sağlayacaktır.

5.8 Yönetilmeyen Hassas Veri

Hassas veri kavramı veritabanı içerisinde bulunan, yetkisiz kişi ya da kişilerin eline geçtiğinde kritik bir risk oluşturabilecek bilgiyi kapsamaktadır. Bu noktada bir veritabanında bulunan kişilerin adres, telefon ve T.C. kmlik numaraları bu hassas bilgiye örnek olarak verilebilir. Kurumsal ölçekli veritabanlarının doğru bir şekilde yönetilmesi ve veritabanlarında bulunan kritik verileri korunması için erişim denetimleri yapılmaktadır. Anlık olarak verilere yapılacak müdahalenin normal şartların dışına çıkması durumunda gereken işlem yapılarak risk seviyesi minimuma indirilir. Kontrol mekanizmaları yalnızca yapısal alana değil veri alanına da erişmelidir. Hassas bilginin içerisinde kullanıcıya ait şifrenin herhangi bir şifreleme algoritması ile şifrelenmeden eklenmesi, veritabanını ele geçiren kişinin kullanıcıların şifreleriyle denetimsiz olarak işlem yapabilmesine olanak sağlamaktadır.

5.9 Hizmet Aksatma

Hizmet aksatma, kullanılan uygulamalara, sistemlere veya verilere erişimin durdurulduğu saldırı tipidir. Bu risk üzerinde kullanıcının erişiminin engellenmesi ve sistemlerin stabil çalışmasının engellenmesi hedef olarak gözetilmektedir. Genellikle dış kaynak tarafından yapılan saldırı neticesinde gerçekleşmektedir. Uygulama ya da sunucuya bağlantı noktaları ve port üzerinde gerekli erişim kontrollerinin yapılmasına olanak sağlayacak yapılandırmanın yapılması bu saldırının engellenmesinde büyük rol oynamaktadır. Sunucu bazlı yapılan çalışmanın yanında veritabanının kullanıldığı web uygulamasına ait erişim kaynaklarının yönetilmesi de bu saldırının engellenmesini sağlayacaktır.

5.10 Sınırlı Güvenlik Uzmanlığı ve Eğitim

Güvenlik boyutu yalnızca teknik anlamda ele alınacak bir konu değildir. Teknik anlamda tüm önlemler alınsa dahi kullanıcının yapabileceği bir hatanın etkisi de düşünülmelidir. Burada kullanıcının bilerek ya da bilmeyerek yapmış olduğu bir hatanın doğurabileceği sonuçlar çoğu zaman teknik önlemlerin de ötesine geçmiş

bulunmakta. Kullanıcının konu ile ilgili teknik anlamdaki bilgi yetersizliği de bu kategoride değerlendirilebilir. Bu noktada kurumsal ya da bireysel olarak kullanıcıların bilinçlendirilmesi ve alanında uzman kişiler ile çalışılması, yeni bir ürün veya hizmete ilişkin hızlı bir adaptasyon sürecinin sağlanması bu riski en aza indirecektir. Günümüzde çalışanların alanlarındaki yeniliği takip etmesindeki sıkıntı bireysel bir sorun olarak değerlendirilecekken, bir şirket içerisindeki büyük çapta bir riski de ortaya çıkarmış olacaktır. Bu bağlamda şirket olarak hem yeniliğin, hem de edinilen ürün ya da hizmetin verimli kullanımı adına eğitimler düzenlenmeli ve bu eğitimler sonrasında kullanıcılardan geri bildirim alınmalıdır.

6. SQL DİLİ

1975 yılında SQL (Structured Query Language – Yapısal Sorgulama Dili), veritabanlarına erişmek ve onları kullanmak ve geliştirmek için oluşturulmuştur. Bu dil ile veritabanı üzerinde veri ekleme, silme, güncelleme, okuma gibi işlemler gerçekleştirilebilmektedir. Basit düzeyde yapılacak bu işlemlerin yanında büyük yapılara ilişkin çeşitli fonksiyonlar da oluşturulabilir. SQL dili kullanılarak veritabanı üzerinde çeşitli zararlara neden olabilecek riskler de oluşturulabilir. Bir kullanıcının gönderdiği parametreler ışığında oluşturulan SQL yapısı, kullanıcının doğrulama ve yetkilendirme mekanizmalarının sağlıklı yürütülebilmesi için kullanılan erişim denetimlerini geçerek SQL yapısını bozabilir. Bu zafiyet kullanılarak veritabanında yetkisiz okuma, ekleme, silme ve değiştirme işlemleri yapılabilir. Oluşturulan bu risk ile veritabanına herhangi bir zarar vermeksizin, sonrasında girişlerinin normal bir kullanıcı olarak yapabilecek ve bu kullanıcı ile sahip olmadığı yetkiler ışığında işlemler gerçekleştirebilecektir.

SQL kullanılarak yapılan sorgular uygulamaların anlık bilgi edinmesi için dinamik olarak yapılandırılabilir.

*SELECT * FROM Tbl_Users*

SQL sorgusu ile veritabanındaki Tbl_Users tablosundaki tüm kayıtlar herhangi bir filtreleme olmadan istenmektedir. Herhangi bir kriter belirtmeksizin edinilen bu sorguya parametre eklenerek özelleştirilebilir ve yalnızca istenen bilgi elde edilebilir.

6.1 Meta Karakter

Meta karakter kavramı bir uygulama dili için özelleştirilmiş karakterleri kapsamaktadır. Bu karakterler kullanılan yazılım diline göre kendinden sonra gelecek karaktere göre işlem yapar. SQL için ‘ karakter meta karakter olarak tanımlanır. Çünkü iki ‘ işareti arasındaki ifade düz metin olarak algılanır.

7. VERİTABANI GÜVENLİK ÖNLEMLERİ

Veritabanı Yönetim Sistemleri veritabanına gelecek tehditlerden korumak adına bazı önlemler içermektedir. Alınması gereken önlemler yalnızca veritabanı üzerinde değil, veritabanının bulunduğu bilgisayardaki güvenliği de kapsamalıdır. Ayrıca veritabanını kullanan bir web uygulamasının güvenliği de bu kapsama dahil edilebilmelidir. Web tabanlı uygulamalar güvenlik boyutunu görebilmek adına üç farklı katmana ayrılmıştır. İlk katman kullanılan web uygulamasının kullanıcı tarafından ilk taleplerinin alındığı internet tarayıcılarıdır. İnternet tarayıcıları üzerinden kullanıcıların uygulama üzerindeki talepleri suucuya iletilir. İkinci katman dinamik olarak oluşan web uygulama katmanıdır. Üçüncü katman ise uygulama tarafından kullanılan verilerin saklandığı veritabanı yönetim sistemleridir.

Veritabanı güvenliğinde önemli etkenlerden biri verinin saklandığı kaynağı kimin, nasıl ve ne şekilde ulaşabileceğinin kesin olarak belirlenmesidir. Bu noktada yetkisiz erişim ve değiştirme işlemlerinin engellenmesi için erişim denetimi mekanizası hem veritabanı hem de sunucu seviyesinde oluşturulur. Erişim denetimi yalnızca kullanıcı adı ve şifre olarak değil, aynı zamanda veritabanı suncusundaki fiziki ya da sanal olarak yer alan bilgisayara erişimi de kapsamaktadır.

Veritabanında güvenlik mekanizmasının kontrolü aşağıdaki başlıklar halinde değerlendirilecektir.

7.1 Erişim Kontrolü

Erişim kontrolü yalnızca veritabanı ile sınırlı olmayan, Veritabanı Yönetim Sisteminin bütüncül bir yaklaşım ile sağlaması gereken temel hizmetlerden biridir. Korunması gereken verilerin önem derecesi ne olursa olsun yetkisiz okuma ve yazma işlemlerinden muhafaza edilmesi öncelikle veritabanı üzerinde başlar, ardından üzerinde bulunduğu sunucuya kadar devam eder. Veritabanına ve veritabanındaki tüm nesnelere ilişkin erişimin sağlanması demek bu erişim süresince bu nesneler üzerinde

istenilen işlemin yapılabileceği anlamına gelmez. Burada çeşitli seviyelerde oluşturulan güvenlik gruplarının ve erişim tanımlamaları devreye girmektedir.

Erişim kontrolü kısmında yapılacak bir hata uygulama bazlı kalmayıp kuruma ait tüm kaynakların tehlike altında girmesine neden olabilir. Veritabanına erişim isteyen kullanıcıların yapacakları işlemin niteliğine göre kontrollü erişim hakları verilebilir. Erişim kontrolünün etkin uygulanması bilerek ya da bilmeyerek yapılan olumsuz bir durumun geri alınmasında önem taşıyacaktır. Çünkü yapılan erişimlerin ve işlemlerin kayıt altına alınması oluşan olumsuz durumda sorun tespiti noktasında büyük rol oynamaktadır. Eğer yetkisiz bir kullanıcı bu işlemi gerçekleştirmiş ise erişim kontrolünün uygulanmadığı ya da güvenlik ilkelerine tam olarak uyulmadığı anlamı çıkabilir.

7.2 Çıkarsama Politikası

Güvenlik politikları kurumsal olarak veri ve uygulamaların sağlıklı bir şekilde ilerlemesi ve korunması adına önemlidir. Bu politika doğrultusunda veritabanı sisteminin ve veritabanındaki bilgilerin korunması sistematik olarak yapılacak iç-dış analizler ve değerlendirmeler ile sağlıklı şekilde yapılabilir. Veritabanında bulunan bilgilerin yetkisiz kişi ya da kişilerin elince geçmemesi ve bilgilerin ifşa edilmesi sağlanacak politika ile önlenmektedir. Bilgilerin dolaylı olarak ifşa edilmesinde eğer veritabanı modeli olarak ilişkisel veritabanı kullanılıyor ise alanların ilişkili olduğu veriler üzerinden ifşa edilip edilmediğinin tespiti yapılabilir.

7.3 Kullanıcı Tanımlama / Kimlik Doğrulama

Veritabanın üzerinde güvenlik prosedürlerinin oluşturulmasında ilk düşünülmesi gereken nokta veritabanına erişen kullanıcıların bilinmesi ve yönetilebilmesidir. Hangi kullanıcının veritabanına eriştiğinin bilinmesi, kullanıcının gerçekleştireceği işlemlerin bilinmesi ve kontrol altında tutulabilmesi anlamını taşımaktadır. Bu kontrol yapılan tüm işlemlerin kaydedilmesi ve işlem basamaklarının analiz edilmesi esasına dayanır. Veritabanında kullanıcıların erişim izni olup olmayacağının belirlendiği tanımlama alanının yanında, giriş yapan bir kullanıcının doğrulanarak sistem içerisindeki rollerinin ve erişim alanlarının da yönetilebilir olması gerekmektedir. Günümüzde Veritabanı Yönetim Sistemleri bu ihtiyacı büyük ölçüde karşılamaktadır.

Veritabanı Yönetim Sistemi'nin hükmettiği veritabanı dışında işletim sistemi üzerinde de geçerli olabilecek güvenlik politikalarının uygulanması üçüncü parti programlar ile gerçekleştirilebilir. Veritabanını kullanan uygulamaya ait servis, uygulama ve veritabanı alanlarının farklı sunucu ve işletim sistemleri üzerinde olması yönetimi kolaylaştırmaktadır.

7.4 Hesap Verebilirlik ve Denetim

Denetim işlemi, veritabanı kullanıcısına bağlı olarak ve veritabanı kullanıcısından bağımsız iki şekilde gerçekleştirilebilen veritabanındaki eylemlerinin izlenmesi ve kaydedilmesidir. Hesap verilebilirlik kavramı ise sistemdeki kullanıcı eylemleri için bir denetim izinin sürdürülmesi işlemidir. Hesap verebilirlik ve denetim, veritabanlarına tanımlı kullanıcıların erişim işleminden itibaren sistem içerisinde yaptıkları tüm eylemlerin kayıt altında olmasını ve sonrasında olumsuz bir durum karşısında bu kayıtların analiz edilmesi ile problem tespitini sağlamaktadır. Denetim işlemi yalnızca giriş yapmış bir kullancının sistem içerisindeki eylemlerini değil, sisteme giriş yaparken başarısız denemeleri de denetlemektedir. Bu durum veritabanına ya da sunucuya yapılan sistematik bir saldırının tespiti noktasında büyük önem taşımaktadır.

7.5 Şifreleme

Şifreleme, bir bilginin sistematik olarak belirlenen bazı kıstaslar çerçevesinde bir şifreye dönüştürme işlemidir. Bir bilginin şifrelenmesi tam anlamıyla korunduğunu göstermez. Şifreleme yönteminin basit düzeyde olması, anahtar sisteminin uygun olmaması, anahtarin kısa olması, gizli anahtarın ifşa edilmesi ya da insancıl faktörler şifreleme işleminin güvenli olduğunu göstermez. Veri şifreleme noktasında kullanılacak şifreleme yönteminin doğru seçimi gereğinden fazla maliyet ve zamanın harcanmasının önüne geçecektir.

Şifreleme işlemi ilk olarak SQL Server 2005 ve SQL Server 2008 ile birlikte desteklenmeye başlandı. Veritabanındaki verilerin korunması için şifreleme işlemi genel olarak üç alanda yapılmaktadır.

7.5.1 İşletim sistemi seviyesinde şifreleme

Microsoft, Windows 2000 işletim sistemini piyasaya çıkarmasıyla birlikte verilerin şifrelenmesi konusunda üçüncü parti uygulama araçları ile destek vermeye başladı. Data Protection API isimli kütüphane ile yazılım geliştiriciler kullanıcı bilgilerinden türetilen bir simetrik anahtarı kullanarak verileri şifreleyebiliyor ve bunları aynı anahtarla çözebiliyorlar.

7.5.2 Veritabanı sunucusu seviyesinde şifreleme

Microsoft SQL Server servisi kurulurken Windows Data Protection API kullanılarak Service Master Key oluşturulur. Bu anahtar, SQL Server içerisindeki ana veritabanı olan ‘master’ içerisinde Security altında bulunan Symmetric Key klasörünün içinde yer alır. Sys.symmetric_keys ile bu anahtarın ne zaman oluşturulduğu ve kullanılan algoritma ile ilgili bilgileri görüntülemek için aşağıdaki T-SQL kodu çalıştırılabilir :

```
SELECT * FROM sys.symmetric_keys
WHERE name = '##MS_ServiceMasterKey##';
```

Service Master Key kullanılan veritabanından tamamen kaldırılamaz. Ancak yeni bir anahtar üretilmesi sağlanabilir. Bu işlem aşağıdaki T-SQL kodu ile gerçekleştirilebilir.

```
ALTER SERVICE MASTER KEY REGENERATE
```

Oluşturulan anahtarın yedeği aşağıdaki T-SQL kodu ile alınabilir.

```
BACKUP SERVICE MASTER KEY
TO FILE = 'c:\BackUp\service_master_key'
ENCRYPTION BY PASSWORD = 'passw@rd123';
```

Yedeklenmiş bir anahtarın geri yüklenmesi aşağıdaki T-SQL kodu ile gerçekleştirilebilir.

```
RESTORE SERVICE MASTER KEY
```

TO FILE = 'c:\BackUp\service_master_key'
ENCRYPTION BY PASSWORD = 'passw@rd123';

7.5.3 Veritabanı seviyesinde şifreleme

Her veritabanı tek bir "Database Master Key" içermektedir. Verilerin şifrelenme işlemi için kullanılan Certificate ve Asymmetric Key'ler bu anahtar yardımıyla şifrelenip korunmaktadır. Certificate ve Asymmetric Anahtarları oluştururken nasıl şifreleneceği belirtilmezse otomatik olarak bunların korunması için "Database Master Key" kullanılır.

"Database Master Key", "Service Master Key" tarafından korunan simetrik anahtarlı şifredir. AES_256 algoritması ve kullanıcı tarafından belirtilen bir anahtar yardımı ile korunur. Database Master Key anahtar elde edilmek istenen veritabanı üzerinde şu T-SQL kodu çalıştırılır.

CREATE MASTER KEY
ENCRYPTION BY PASSWORD ='passw@rd123'

Database Master Key'i görüntülemek için ilgili veritabanı üzerinde şu T-SQL kodu kullanılabilir.

*SELECT * FROM sys.symmetric_keys*
WHERE name='##MS_DatabaseMasterKey##'

Veri şifreleme mekanizmasının çalıştırılmasını sağlayan fonksiyonlar:

T-SQL fonksiyonu	:	EncryptByPassPhrase	- DecryptByPassPhrase
Asymmetric Keys	:	EncrypByAsymKey	- DecryptByAsymKey
Symmetric Keys	:	EncrypByKey	- DecryptByKey
Certificates	:	EncrypByCert	- DecryptByCert

7.5.3.1.1 Always encrpyted

Microsoft SQL Server 2016 ile gelen yeni bir veritabanı şifreleme yöntemidir. Veritabanı şifreleme yöntemleri ile kritik öneme sahip veriler şifrelenip korunabilir. Şifrelenmiş kredi kartı bilgileri, kişisel kimlik bilgilerini veya uygulamada hassas veri niteliğinde olabileceği tüm veriler şifrelenebilir.

Veri şifreleme ve şifrelenmiş bir verinin çözülme iş yükü, kullanım yoğunluğuna ve şifreleme tekniğine bağlı olarak kritik seviyelere ulaşabilir. SQL Server 2016 öncesinde bu iş yükü tamamen veritabanı üzerindedir. Kritik seviyede (örneğin bir banka) bulunan bir veritabanı sunucusu bu iş yükünü kaldıramayabilir.

SQL Server 2016 ile birlikte duyurulan Always Encrypted yöntemi şifreleme ve şifrelenmiş verinin çözülmesi iş yükünü SQL Server'a değil, istemcilere yükler. Veritabanında şifrelenmesini istenilen alanlar verinin şifrelenmiş ya da şifrelenmemiş gönderilmesine bağlı olmaksızın daima şifreli olarak durur. En yetkili kullanıcılar bile izin verilmediği sürece şifrelenmiş bu verilerin açık halini göremez. [4]

Veritabanında bulunan veriler istemciye şifreli gider. Kullanıcı tarafındaki bulunan anahtar kullanılarak şifre çözülür ve verinin istemci uygulamasının açık şekilde görülebilmesi sağlanır. Uygulama tarafından açık şekilde gönderilen veri aynı şekilde istemci tarafında şifrelenir ve veritabanına şifreli olarak gider.

İstemci ve veritabanı arasındaki şifreleme süreci otomatik olarak çalışır. .NET 4.6 ile gelen geliştirilmiş ADO.NET kütüphanesi üzerindeki bağlantı cümlesine (connectionstring) eklenen bir ifade ve depolanan anahtar yardımıyla işlemler otomatik olarak yerine gelmektedir.

8. SQL ENJEKSİYONU

ASP.NET, PhP gibi web tabanlı programlama dilleri veritabanı üzerinde gerçekleştirilecek sorgulama işlemleri için SQL dilini kullanmaktadır. Kullanıcının uygulama üzerinden yaptığı talepler veritabanı üzerinde veri girişi, veri sorgulaması, veri silinmesi gibi işlemleri sağlamaktadır. Bu veri işlemleri bazı riskleri de taşımaktadır. Projenin oluşturulması sürecinde gerek veritabanı, gerek proje üzerinde yapılan hatalardan ötürü veritabanı üzerinde saklanan bilgiler üzerinde izinsiz erişim ve kullanımı gibi olumsuz sonuçlar meydana gelebilir.

Uygulama üzerinde kullancının SQL ile yaptığı sorgulamada bazı parametre değerlerinin değiştirilmesi olumsuzluklara sebep olmaktadır. Bu olumsuzluk veritabanında erişim yetkisi olmayan kişilerin bilgilere erişmeyi sağlaması, erişim sağlanabilen veriler üzerinde değişiklik yapılması gibi istenmeyen sonuçlar doğuracaktır. Bu zararı önlemek adına alınabilecek en temel önlem parametre değerlerini bazı özel karakterlere karşı kontrol edilmesi olabilir.

SQL Enjeksiyon konusu ile ilgili tez içerisinde yer alan uygulamanın oluşturulması ve teorik kısmın açıklanması Misrosoft SQL Server veritabanı ile ASP.NET yapısı üzerinden açıklanacaktır.

SQL enjeksiyonu kullanım şekline göre sınıflandırılmaktadır :

8.1 İmleç Kullanılarak Yapılan SQL Enjeksiyonu

SQL dilinde meta karakter olarak adlandırılan tırnak işareti (‘) bu tipteki enjeksiyonun temelini oluşturmaktadır. Kullanılan meta karakter sayesinde sorgulama ifadesinde manipülasyon yapılmaktadır. Örneğin oluşturulan bir ASP.NET sayfasındaki kullanıcı adı ve şifre bölümleri veritabanına gönderilerek böyle bir kullanıcı olup olmadığı kontrol edilecek, edinilen sonuca göre kullanıcıya mesaj verilecektir. Sayfadaki alanlara kullanıcı adı olarak erturk, şifre olarak ise 12345 girildiğinde oluşturulan SQL sorgusu şu şekilde olacaktır:

*SELECT * FROM Tbl_Users WHERE UserName = ‘erturk AND Password = ‘12345’*

Burada sayfadaki kullanıcı adı kısmına erturk, şifre kısmına ‘ OR 1 = 1 – bilgisi girildiği takdirde oluşacak sorgu şu şekilde olacaktır:

*SELECT * FROM Tbl_Users WHERE UserName = ‘erturk AND Password =" OR 1=1 --'*

Bu oluşan yeni sorgu kullanıcı adı olarak erturk için şifre girişine gerek duymaması sağlanmış olacaktır. Sorgu sayesinde 1=1 eşitliği sağlanacak ve yetkisiz bir kullanıcının giriş yapmasına neden olacaktır. Burada Password alanı için eklenen ‘ işaretinin kapanmamasının nedeni ilgili cümle sonuna getirilen – karakterlerinin açılan ve kapanan ‘ işaretlerinin önemsenmemesini sağlamaktır.

8.2 Tetiklenen Yordamlar

Erişim denetiminin sağlıklı olarak yürütülemediği bir veritabanı üzerinde veritabanına giriş yapmış bir kullanıcının SQL Server üzerinde varsayılan şekilde bulunan ve sistemsel işlemleri yapmayı sağlayan yordamları da çalıştırabilme yetkisi bulunmaktadır. Bu kısımda örnek olarak sp_delete_backuphistory isimli sistem yordamı verilebilir. Bu yordam SQL Server içerisinde daha önce alınmış olan veritabanı yedeklerinin silinmesini sağlamaktadır. Kullanıcı enjeksiyon saldırısı sırasında şifre alanına sistemsel yordamları girerek oluşturacağı sorguda erişim denetimi olmadan sistemsel yordam da çalışacaktır..

8.3 SQL Server Sistem Odaklı Enjeksiyon

SQL Server içerisindeki özel fonksiyonların çalıştırılmasına bağlı çalıştırılan enjeksiyon türüdür. Burada örnek olarak SQL Server hizmetini kapatmayı sağlayan SHUTDOWN WITH NOWAIT yordamı örnek verilebilir. Oluşacak yeni yordam şu şekildedir:

*SELECT * FROM Tbl_Users WHERE UserName = 'erturk AND Password=';*
EXEC sp_delete_backuphistory –

Bu komut SQL Server'ın servislerinin kapanmasını sağlayacaktır. Bu durum servise bağlı tüm hizmetlerin aksamasına neden olacaktır.

8.4 Yanlış Tip İşleme

Kullanıcı tarafından girilen bir verinin tip kontrolünün yapılmamasına dayanan enjeksiyondur. Örneğin sayısal bir değer bekleyen alan için metin bazlı bir değer girildiğinde gerçekleşebilir.

*SELECT * FROM Tbl_Users WHERE id =" + idValue + "; "*

Sorguda idValue kısmına sayısal bir değer gelmesi beklenmektedir. Eğer bu ifade yerine

Delete From Tbl_Users

yazılırsa tablodaki tüm bilgiler silinecektir. Bu saldırının önlenmesi adına beklenen parametrenin tip kontrolünün yapılarak işleme alınması gerekmektedir.

8.5 Blind SQL Enjeksiyonu

Bu enjeksiyon modelinde bir web uygulaması enjeksiyona karşı korumasız olduğunda varolan sistem yapısını anlamak ve sonraki saldırıların temelini oluşturmak için kullanılır. Örneğin bir e-ticaret sitesinde ürünün gösterilmesini sağlayan;

www.abc.com?Product.aspx?ProductID=1

linkinde 1 ID değerine sahip ürünün detayının gösterildiği bir sayfa gelecektir. Burada ID değerinin karşılığını bulan ve ihtiyaç duyulan bilgiler sunucu üzerinde oluşturulan sorgu ile gösterilir. Örnek sorgu;

*SELECT * FROM Tbl_Products WHERE ID=IDVALUE*

şeklinde oluşmaktadır. Burada parametere olarak gönderilen 1 değerine sahip ürün bilgisi gelecektir. Sorgu, sunucu üzerinde yapılandırıldığından saldırganın herhangi bir müdahalesi söz konusu değildir. Fakat link üzerinden gönderilen değer ile parametreye müdahale edilebilir. İkinci denemede;

www.abc.com?Product.aspx?ProductID=1 OR 1=1

linki ile yeni sorgu

*SELECT * FROM Tbl_Products WHERE ID=1 OR 1=1*

şeklinde olacaktır. Burada istenilen ürüne ait sayfanın gelmesi enjeksiyona açık olduğunu göstermektedir. Üçüncü denemede;

www.abc.com?Product.aspx?ProductID=1 AND 1=2

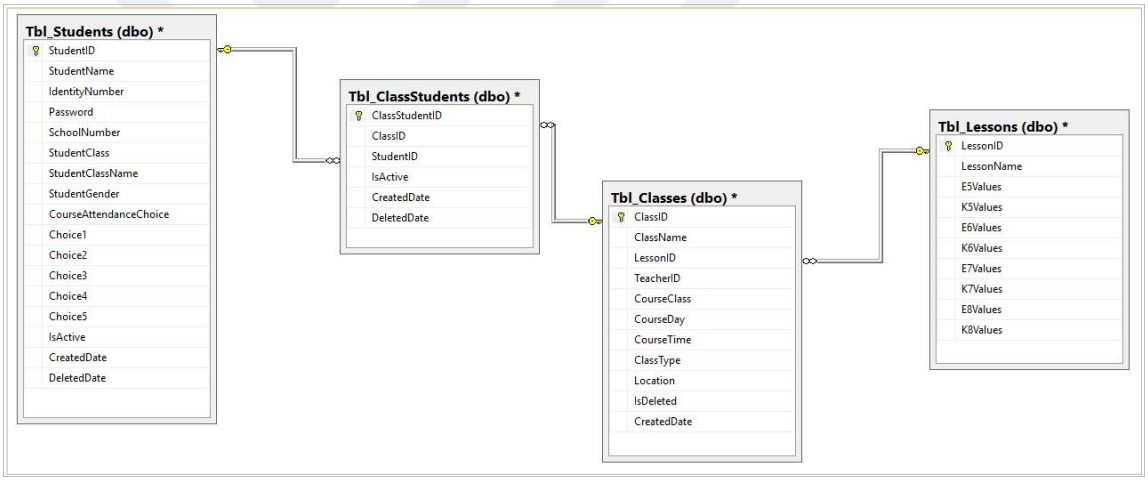
linki ile yeni sorgu

*SELECT * FROM Tbl_Products WHERE ID=1 AND 1=2*

şeklinde olacaktır. Burada boş sayfa ya da hata sayfası gelmesi durumunda da enjeksiyona açık olduğu görülmektedir. Bu noktada saldırgan enjeksiyona açık bir veritabanı üzerinden daha fazla bilgi edinmek adına çalışma yürütebilir.

9. MICROSOFT SQL SERVER TABANLI ÖRNEK UYGULAMA

Örnek bir ugulama ile adım adım SQL Enkesiyon işleminin nasıl gerçekleştirildiğini açıklanacaktır. Yapılacak çalışma, ASP.NET tabanlı C# uygulama dilini kullanan bir web uygulaması ve Microsoft SQL Server üzerinde gerçekleşecektir. Yapılan uygulama bir öğrenci otomasyon sisteminin yalnızca bir bölümünü oluşturacak küçük ölçekli bir uygulama olacaktır. Öncelikle veritabanı üzerinde gerekli tabloların tasarlanması adımı gerçekleştirilir. Burada öğrenciler, sınıflar, dersler, öğrencilerin hangi sınıfta bulunduklarında dair dört tablo tasarlanacaktır.



Şekil 9.1: Örnek uygulama veritabanı tabloları

Oluşturulan veritabanında öğrencilerin bilgilerinin saklanacağı Tbl_Students tablosu, derslerin saklanacağı Tbl_Lessons tablosu, sınıfların saklancağı Tbl_Classes tablosu, sınıfta bulunacak öğrencilerin saklandığı Tbl_ClassStudents tablosu tasarlanmıştır. Bu dört tablo birbirlerine öğrenciye ait ID bilgisi, sınıfa ait ID bilgisi ve derse ait ID bilgisi ile ilişkilendirilerek ilişkisel veritabanı oluşturulmuştur.

```sql
CREATE PROC [dbo].[Sp_StudentLogin]
(
    @IdentityNumber nvarchar(500),
    @Password nvarchar(500)
)

as

select  StudentID, StudentName, IdentityNumber, Password, SchoolNumber, StudentClass,
        StudentClassName, StudentGender, CourseAttendanceChoice,
        Choice1, Choice2, Choice3, Choice4, Choice5, IsActive, CreatedDate, DeletedDate

from    [dbo].[Tbl_Students]

where   (IdentityNumber = @IdentityNumber) and
        ([Password] = @Password) and
        IsActive = 1
```

Şekil 9.2 : Kullanıcı girşine ilişkin veritabanı yordamı

Kullanıcı girişinin kontrolünü sağlayacak olan veritabanı yordamı iki parametre alacak şekilde hazırlanmıştır. İlk parametre kimlik numarasını alan @IdentitiyNumber, ikinci parametre ise @Password olarak tasarlanmıştır. Bu iki koşulu sağlayan ve durumunu belirten kısmın aktif olduğu kullanıcı bilgilerinin getirilmesi sağlanmıştır.

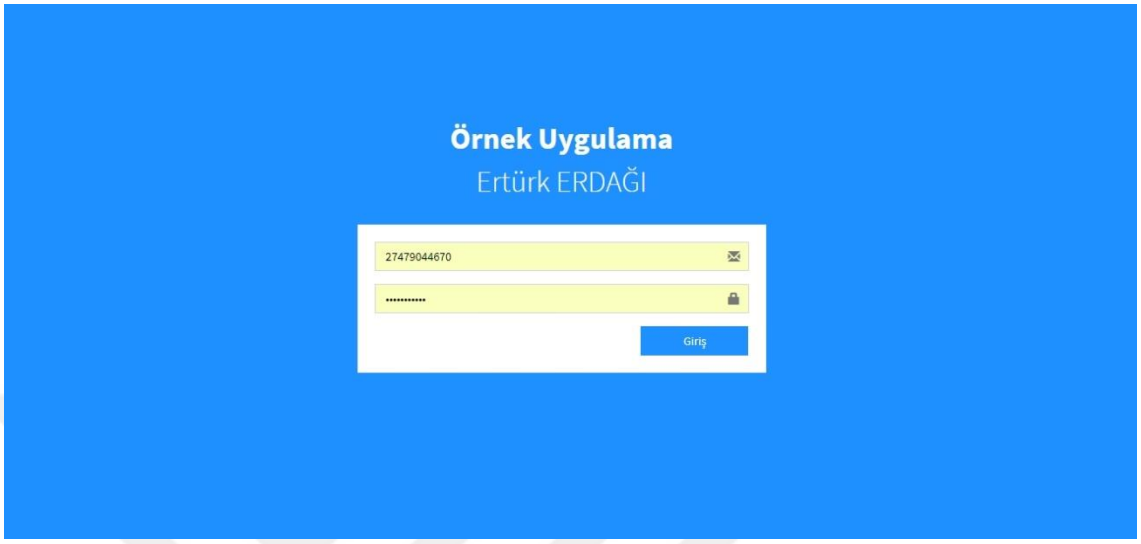
```aspx
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Login.aspx.cs" Inherits="AgaziKurs.Login" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Örnek Uygulama - Ertürk ERDAĞI</title>
    <meta charset="utf-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta content="width=device-width, initial-scale=1, maximum-scale=1, user-scalable=no" name="viewport" />
    <link rel="stylesheet" href="Bootstrap/css/bootstrap.min.css" />
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/font-awesome/4.4.0/css/font-awesome.min.css" />
    <link rel="stylesheet" href="https://code.ionicframework.com/ionicons/2.0.1/css/ionicons.min.css" />
    <link rel="stylesheet" href="Style/AdminLTE.min.css" />
    <link rel="stylesheet" href="Plugins/iCheck/square/blue.css" />
</head>
<body class="hold-transition login-page">
    <form id="form1" runat="server">
        <div class="login-box">
            <div class="login-logo">
                <a href="../../index2.html"><b>Örnek Uygulama</b><br />Ertürk ERDAĞI</a>
            </div>
            <div class="login-box-body">
                <div class="form-group has-feedback">
                    <asp:TextBox runat="server" ID="txtUserName" CssClass="form-control"></asp:TextBox>
                    <span class="glyphicon glyphicon-envelope form-control-feedback"></span>
                </div>
                <div class="form-group has-feedback">
                    <asp:TextBox runat="server" ID="txtPassword" TextMode="Password" CssClass="form-control"></asp:TextBox>
                    <span class="glyphicon glyphicon-lock form-control-feedback"></span>
                </div>
                <div class="row">
                    <div class="col-xs-8">
                        <asp:Label runat="server" ID="lblStatus"></asp:Label>
                    </div>
                    <div class="col-xs-4">
                        <asp:Button runat="server" ID="btnLogin" Text="Giriş" CssClass="btn btn-primary btn-block btn-flat" OnClick="btnLogin_Click" />
                    </div>
                </div>
            </div>
        </div>
    </form>
</body>
</html>
```

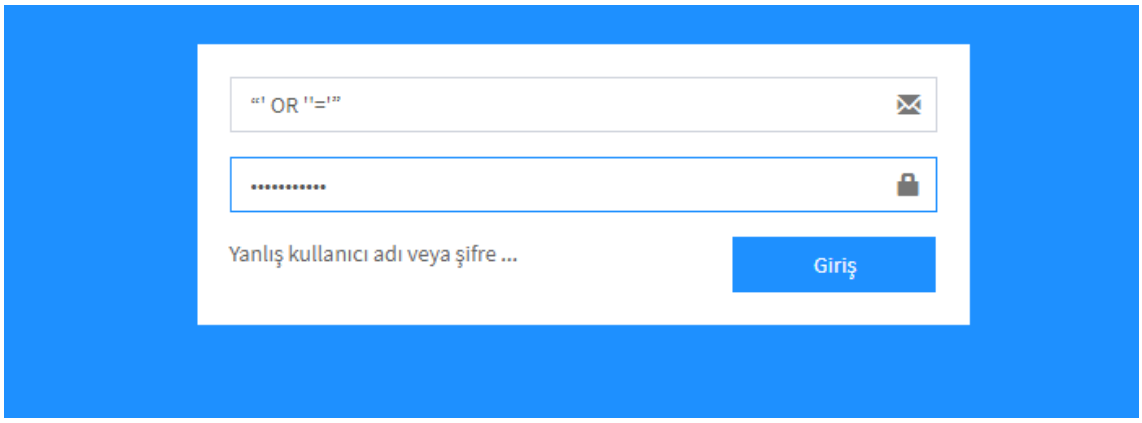
Şekil 9.3: Kullanıcı giriş sayfası HTML tasarım kodları

ASP.NET sayfası için kullanıcı adının ve şifrenin girilebileceği birer adet TextBox eklenmiştir. Şifre alanının güvenli olması için TextMode özelliği Password olarak belirlenerek şifre girişinde güvenlik sağlanmıştır.



Şekil 9.4 : Kullanıcı giriş sayfası önyüzü

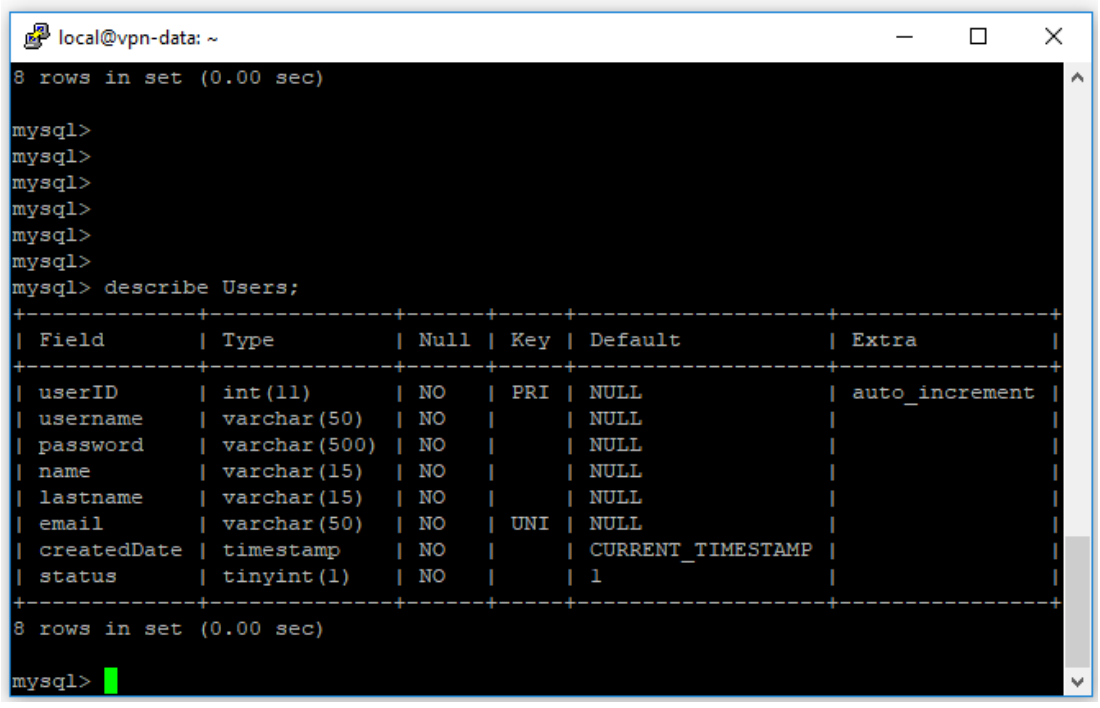
SQL içerisindeki Stored Procedure yardımıyla sorgulanan bilgiler ile bu şartları taşıyan bir kayıt olması durumunda kullanıcı anasayfaya yönlendirilecektir. Bu noktada yazılan fonksiyondaki parametre isimleri ve kullanış biçimi SQL enjeksiyonu engelleyecek şekilde uygulanmıştır. Kullanıcı adı ve şifre kısmına enjeksiyon işlemini gerçekleştirmek için ilgili değerler girildiğinde aşağıdaki şekilde işlemin başarısız olduğu mesajı gelecektir. Burada yazılan fonksiyonun enjeksiyona karşı korunaklı olması önemlidir.



Şekil 9.5 : SQL enjeksiyon denemesi

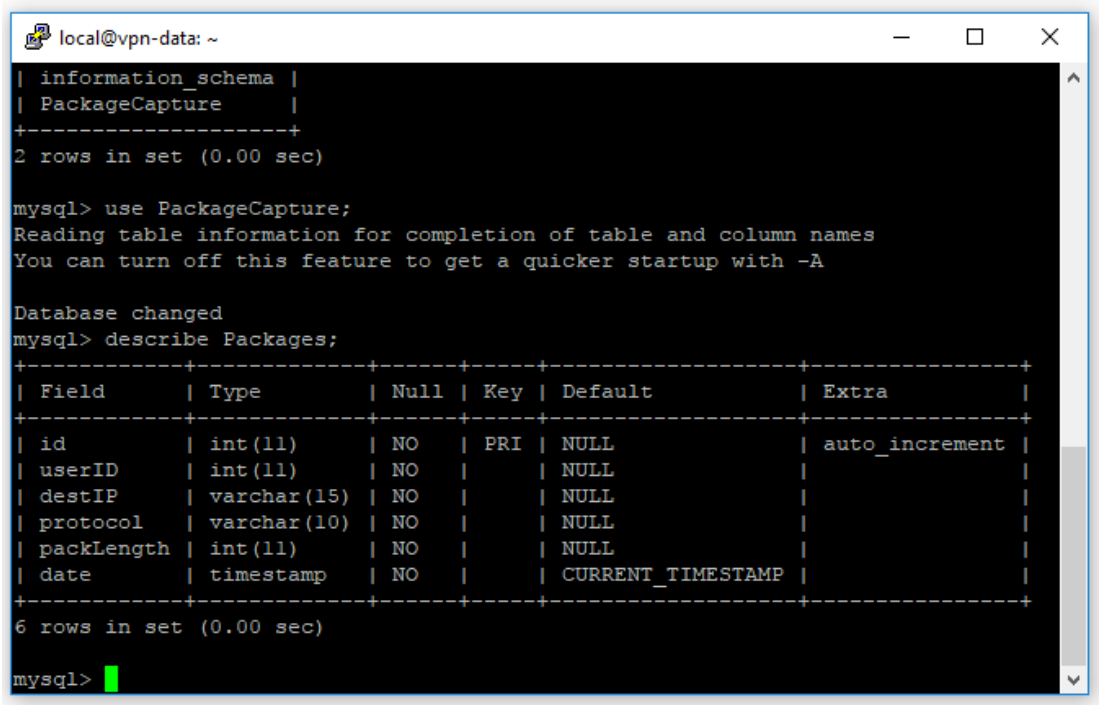
10. MYSQL TABANLI ÖRNEK UYGULAMA

Microsoft SQL Server ile yapılan uygulama sonrasında kullanım alanının fazla olması sebebiyle MYSQL üzerinde bir uygulama gerçekleştirilecektir. Uygulama içeriği kullanıcıların bilgisayar, tablet, mobil cihazlarının ağ üzerindeki paketlerinin denetleme işlemlerinin analiz edilmesine dair uygulamanın bir parçasını oluşturacaktır. Projede ağ paketlerinin yakalanması ve veritabanına kayıt işlemi üçüncü parti uygulamalar ile gerçekleştirildiğinden burada yer almayacaktır. Projenin ilk adımı olarak veritabanı tasarlanacaktır. Veritabanında kullanıcıların kimlik bilgilerinin yer aldığı Users isminde bir tablo ile paketlere ilişkin detayların bulunduğu Packages tablosu yer alacaktır. Tablolara verilerin eklenmesi, sorgulanması, silinmesi veya güncellenmesi gibi işlemler saklı yordamlar (stored procedure) üzerinden ve projenin programlama dili olan C# üzerinden yapılacaktır. Users tablosu kullanıcıların uygulama üzerinden kayıt, profil güncelleme ve sorgulama işlemler için kullanılacaktır.



Şekil 10.1 : Veritabanındaki users tablosunun yapısal görünümü

Tablo içerisinde id değeri olarak belirlenen ve herbir kullanıcı için benzersiz değeri temsil eden sayısal bir alan ayrılmıştır. Bu alandaki bilginin otomatik verilmesi için AUTO_INCREMANT özelliği açılmıştır. Username alanında kullanıcı adı, password kısmında sisteme girişi sağlayan şifre bilgisi, name alanında ad, lastname alanında soyad, email alanında ise e-posta adresi tutulacak şekilde yapılandırılmıştır. Belirtilen bu alanlar metin bazlı alanlar olarak ve içerebilecekleri maksimum uzunluğa göre yapılandırılmıştır. Password alanı şifreleme algoritmaları ile oluşturulacağından uzun tutulmuştur. CreatedDate alanı ile bu tabloya eklenecek bir bilginin sistemdeki mevcut zamanı kullanarak hangi tarih ve saatte eklendiği görülebilecektir. Burada otomatik olarak bu değerin verilmesi için CURRENT_TIMESTAMP kısmı açılmıştır. Status alanı ise kullanıcının aktif ya da pasif olmasına dair bir bilgiyi içeren sayısal değer olarak yapılandırılmıştır.



Şekil 10.2 : Veritabanındaki packages tablosunun yapısal görünümü

Veritabanındaki ikinci tablo Packages tablosunda ise tüm ağ paketlerinin kaydı tutulacaktır. Bu tabloda id değeri otomatik olarak artan, benzersiz değeri temsil eden sayısal alanı temsil edecektir. UserID alanı ise bir önceki tablomuz olan Users tablosundaki id alanını saklayacaktır. Bu sayede iki tablo arasındaki ilişki kurulacaktır. Bir kullanıcıya ait paketlerin tespiti bu alan sayesinde gerçekleştirilecektir. destIP,

protocol, packLength ile trafiğe ilişkin IP adresinin, protokol bilgisinin ve paketin boyutuna ilişikin bilgiler saklanacaktır. Date alanı ile bu tablodaki bir paket bilgisinin hangi tarih ve saatte eklendiği görülebilecektir. Burada otomatik olarak bu değerin verilmesi için CURRENT_TIMESTAMP kısmı açılmıştır.

```
USE `PackageCapture`;

/* SET QUOTED_IDENTIFIER ON */

DELIMITER //

CREATE PROCEDURE Sp_AddUser
(
    p_username VARCHAR(50),
    p_password VARCHAR(50),
    p_name VARCHAR(15),
    p_lastname VARCHAR(15),
    p_email VARCHAR(50)
)

BEGIN

DECLARE v_Status INT;
DECLARE v_StatusText NVARCHAR(100);

IF EXISTS(SELECT * FROM Users WHERE username = p_username AND status = 1)
    THEN
        SET v_Status = 0;
        SET v_StatusText = 'UserName exists.';

ELSEIF EXISTS(SELECT * FROM Users WHERE email = p_email AND status = 1)
    THEN
        SET v_Status = 0;
        SET v_StatusText = 'Email exists.';
ELSE
        INSERT INTO Users (username, password, name, lastname, email, status) VALUES (p_username, p_password, p_name, p_lastname, p_email, 1);
        SET v_Status = 1;
        SET v_StatusText = 'User created sucessfully';
    END IF;
END IF;


SELECT  v_Status AS Status, v_StatusText AS StatusText;
END;
//

DELIMITER ;
```

Şekil 10.3 : Veritabanındaki Sp_AddUser yordamının yapısal görünümü

Web projesinde oluşturulacak kayıt sistemi ile kullanıcının Users tablosuna eklenmesini sağlayan saklı yordam (stored procedure) içerisinde sistemdeki kullanıcı adı ve email kontrolü yapıldıktan sonra ekleme işlemi yapılmaktadır. Oluşturulan iki değişken ile bu işlemin sonuç bilgisinin görüntülenmesi sağlanmıştır.

```
USE `PackageCapture`;

/* SET QUOTED_IDENTIFIER ON */

DELIMITER //

CREATE PROCEDURE Sp_UpdateUser
(
    p_userID int,
    p_username VARCHAR(50),
    p_password VARCHAR(50),
    p_name VARCHAR(15),
    p_lastname VARCHAR(15),
    p_email VARCHAR(50)
)

BEGIN

DECLARE v_Status INT;
DECLARE v_StatusText NVARCHAR(100);

IF EXISTS(SELECT * FROM Users WHERE (username = p_username AND status = 1) AND userID <> p_userID)
THEN SET v_Status = 0;
SET v_StatusText = 'UserName exists.';

ELSEIF EXISTS(SELECT * FROM Users WHERE (email = p_email AND status = 1) AND userID <> p_userID)
THEN
SET v_Status = 0;
SET v_StatusText = 'Email exists.';
ELSE
UPDATE Users

SET username = p_username,
    password = p_password,
    name = p_name,
    lastname = p_lastname,
    email =  p_email

WHERE userID = p_userID;


SET v_Status = 1;
SET v_StatusText = 'User updated sucessfully';
END IF;
END IF;

SELECT  v_Status AS Status, v_StatusText AS StatusText;
END;
//

DELIMITER ;
```

Şekil 10.4 : Veritabanındaki Sp_UpdateUser yordamının yapısal görünümü

Kullanıcının bilgilerinde güncelleme yapılmasını sağlayan Sp_UpdateUser isimli saklı yordam üzerinde istenen parametreler doğrultusunda işlem yapılacaktır. Talep edilen parametreler doğrultusunda güncelleme işlemi yapılacaktır. Girilen parameterelerin sistemde başka bir kullanıcıya ait kullanıcı adı ve e-posta adresi olup olmadığı da kontrol edilecektir. İşlem durumuna göre mesaj bilgisi ile sonuç görüntlenmektedir.

```
USE `PackageCapture`;

/****** Object:  StoredProcedure [dbo].[Sp_UserLogin]    Script Date: 17.12.2017 13:26:59 ******/
/* SET ANSI_NULLS ON */

/* SET QUOTED_IDENTIFIER ON */

DELIMITER //

CREATE PROCEDURE Sp_UserLogin
(
    p_username varchar(15),
    p_password varchar(50)
)

BEGIN

SELECT  id, userID, username, name, lastname, email, createdDate, status

FROM    Users

WHERE   username = p_username AND password = p_password AND status = 1;
END;
//

DELIMITER ;
```

Şekil 10.5 : Veritabanındaki Sp_UserLogin yordamının yapısal görünümü

Hazırlanacak web uygulamasında kullanıcıların, kullanıcı adı ve şifre bilgisi ile sisteme girişlerini sağlayacak yordam Sp_UserLogin adıyla hazırlanmıştır. Burada iki parametrenin sistemdeki karşılığı kontrol edilerek eğer böyle bir kullanıcı mevcutsa kullanıcı bilgilerinin sonuç olarak verilmesi sağlanmıştır. Bu noktada şifrelerin gönderilme ve sorgulama işlemleri tamamen şifrelenmiş yöntemle yapılmıştır. Buradaki şifreleme yöntemi web projesi bölümünde açıklanacaktır.

```
USE `PackageCapture`;

/****** Object:  StoredProcedure [dbo].[Sp_GetPackagesByDateAndUser]    Script Date: 17.12.2017 13:25:04 ******/
/* SET ANSI_NULLS ON */

/* SET QUOTED_IDENTIFIER ON */

DELIMITER //

CREATE PROCEDURE Sp_GetPackagesByDateAndUser
(
    p_StartDate datetime(3),
    p_EndDate datetime(3),
    p_userID int
)

BEGIN

SELECT id, userID, destIP, protocol, packLength, date

FROM    Packages

WHERE   (date >= p_StartDate AND date <= p_EndDate) AND (userID = p_userID)

ORDER BY date ASC;
END;
//

DELIMITER ;
```

Şekil 10.6 : Veritabanındaki Sp_GetPackagesByDateAndUser yordamı

Sisteme giriş yapan kullanıcının bilgisine ve verilecek iki tarih aralığına göre Packages tablosu üzerinde yapılacak sorgulama doğrultusunda paketlerin bilgisi gelecektir. Bu bilginin web uygulaması üzerinde listelenmesi sağlanacaktır.

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site1.Master" AutoEventWireup="true" CodeBehind="Default.aspx.cs"
  Inherits="PackageCapture.Default" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" runat="server">
    <div class="content-wrapper">
        <section class="content-header">
            <h1>Home Page
            </h1>
        </section>
        <section class="content">
            <div class="row">
                <div class="box box-info">
                    <div class="box-body">
                        <div class="table-responsive">
                            <table class="table no-margin">
                                <div style="background-color: bisque; color: white; padding: 10px; margin-bottom: 20px">
                                    <div style="width: 100%">
                                        <asp:Label runat="server" Text='Administrator' ForeColor="Black" Font-Size="16px" Font-Bold="true"></
                                    asp:Label>  |  
                                                <asp:Label runat="server" Text='30.12.2017 16.31' ForeColor="Gray" Font-Size="16px" Font-
                                    Bold="true"></asp:Label><br />
                                        <br />
                                        <asp:Label runat="server" Text='You can get package information using "Packages" menu' ForeColor="Red"
                                    Font-Bold="true" Font-Italic="true"></asp:Label>
                                    </div>
                                </div>
                                <div style="background-color: bisque; color: white; padding: 10px; margin-bottom: 20px">
                                    <div style="width: 100%">
                                        <asp:Label runat="server" Text='Administrator' ForeColor="Black" Font-Size="16px" Font-Bold="true"></
                                    asp:Label>  |  
                                                <asp:Label runat="server" Text='28.12.2017 12.13' ForeColor="Gray" Font-Size="16px" Font-
                                    Bold="true"></asp:Label><br />
                                        <br />
                                        <asp:Label runat="server" Text='You can change profile information with "Profile" menu' ForeColor="Red"
                                    Font-Bold="true" Font-Italic="true"></asp:Label>
                                    </div>
                                </div>
                            </table>
                        </div>
                    </div>
                </div>
            </div>
        </section>
```

Şekil 10.7 : Anasayfa (Default.aspx) tasarımsal HTML kodları - 1

```
        <section class="content-header">
            <h1>Last 5 Package Information
            </h1>
        </section>
        <section class="content">
            <div class="row">
                <div class="box box-info">
                    <div class="box-body">
                        <div class="table-responsive">
                            <table class="table no-margin">
                                <thead>
                                    <tr>
                                        <th>Destination IP</th>
                                        <th>Protocol</th>
                                        <th>PackLength</th>
                                        <th>Date</th>
                                    </tr>
                                </thead>
                                <tbody>
                                    <asp:Repeater runat="server" ID="rptPackages">
                                        <ItemTemplate>
                                            <asp:Label runat="server" ID="lblID" Text='<%# DataBinder.Eval(Container.DataItem, "id") %>'
                                        Visible="false"></asp:Label>
                                            <tr style="height: 45px">
                                                <td><%# DataBinder.Eval(Container.DataItem, "destIP") %></td>
                                                <td><%# DataBinder.Eval(Container.DataItem, "protocol") %></td>
                                                <td><%# DataBinder.Eval(Container.DataItem, "packLength") %></td>
                                                <td><%# DataBinder.Eval(Container.DataItem, "date") %></td>
                                            </tr>
                                        </ItemTemplate>
                                    </asp:Repeater>
                                </tbody>
                            </table>
                        </div>
                    </div>
                </div>
            </div>
        </section>
    </div>
</asp:Content>
```

Şekil 10.8 : Anasayfa (Default.aspx) tasarımsal HTML kodları - 2

```
using DataLayer;
using MySql.Data.MySqlClient;
using System;
using System.Collections.Generic;
using System.Configuration;
using System.Data;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace PackageCapture
{
    - references
    public partial class Default : BasePage
    {
        - references
        protected void Page_Load(object sender, EventArgs e)
        {
            if (Session["UserID"] == null)
            {
                Response.Redirect("Login.aspx");
            }
            if (!IsPostBack)
            {
                BindPackages();
            }
        }

        - references
        private void BindPackages()
        {
            MySqlConnection con = new MySqlConnection(ConfigurationManager.ConnectionStrings["ConStr"].ConnectionString);
            string SQLText = "SELECT id, destIP, protocol, packLength, date FROM Packages WHERE(userID = " + Session["UserID"].ToString() + ") 
               ORDER BY date DESC LIMIT 5";
            MySqlCommand cmd = new MySqlCommand(SQLText, con);
            con.Open();
            DataSet dsPackages = new DataSet();
            MySqlDataAdapter daUser = new MySqlDataAdapter(SQLText, con);
            daUser.Fill(dsPackages, "Packages");
            rptPackages.DataSource = dsPackages.Tables[0];
            rptPackages.DataBind();
        }
    }
}
```

Şekil 10.9 : Anasayfa (Default.aspx.cs) C# kodları

Kullanıcıların giriş yaptıktan sonra ulaşacakları anasayfanın (Default.aspx) hem tasarım hem de program kodları paylaşılmıştır. Burada anasayfanın üst kısmında giriş yapan kullanıcılara bilgilendirici notların bulunduğu bir bölüm ile alt kısmında son beş paketin görüntülenmesini sağlayan bir bölüm hazırlanmıştır. Bu sayfada kullanıcının sisteme giriş yapmadan erişmesini engellemek için Session üzerinden UserID bölümünün kontrolü yapılmıştır. Bu yapı siseme giriş sayfasında açıklanacaktır.

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site1.Master" AutoEventWireup="true" CodeBehind="ChangePassword.aspx.cs"
  Inherits="PackageCapture.ChangePassword" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" runat="server">
    <asp:ScriptManager runat="server"></asp:ScriptManager>
    <div class="content-wrapper">
        <section class="content-header">
            <h1>Change Password</h1>
        </section>
        <section class="content">
            <div class="row">
                <section class="content" style="min-height: 100px !important">
                    <div>
                        Current Password
                    </div>
                    <div>
                        <asp:TextBox runat="server" ID="txtCurrentPassword" CssClass="form-control" Width="460px" TextMode="Password"></
                          asp:TextBox>
                    </div>
                    <div>
                        New Password
                    </div>
                    <div>
                        <asp:TextBox runat="server" ID="txtPassword" CssClass="form-control" Width="460px" TextMode="Password"></asp:TextBox>
                    </div>
                    <div>
                        Re-enter New Password
                    </div>
                    <div>
                        <asp:TextBox runat="server" ID="txtPasswordRe" CssClass="form-control" Width="460px" TextMode="Password"></asp:TextBox>
                    </div>
                    <br />
                    <br />
                    <asp:Label runat="server" ID="lblStatus" ForeColor="Blue" Font-Size="20px" Font-Bold="true"></asp:Label>
                    <div style="height: 44px; margin-top: 12px">
                        <asp:Button runat="server" ID="btnSave" Text="Save" CssClass="btn btn-sm btn-info btn-flat pull-left" Width="100px"
                          OnClick="btnSave_Click" />
                    </div>
                </section>
            </div>
        </section>
    </div>
</asp:Content>
```

Şekil 10.10 : Şifre değiştirme (ChangePassword.aspx) tasarımsal HTML kodları

```
    0 references
    protected void Page_Load(object sender, EventArgs e)
    {

    }

    0 references
    protected void btnSave_Click(object sender, EventArgs e)
    {
        if (txtCurrentPassword.Text == string.Empty)
        {
            lblStatus.Text = "Current password can not be empty";
        }
        else if (txtPassword.Text == string.Empty)
        {
            lblStatus.Text = "New passowrd can not be empty";
        }
        else if (txtPassword.Text != txtPasswordRe.Text)
        {
            lblStatus.Text = "New passwords not match";
        }
        else
        {
            MySqlConnection con = new MySqlConnection(ConfigurationManager.ConnectionStrings["ConStr"].ConnectionString);
            string SQLText = "SELECT userID, username, name, lastname, email, createdDate, status FROM Users WHERE password = '" +
              Encryption.EncryptSHA512Managed(txtCurrentPassword.Text) + "'";
            MySqlCommand cmd = new MySqlCommand(SQLText, con);
            con.Open();
            DataSet dsUser = new DataSet();
            MySqlDataAdapter daUser = new MySqlDataAdapter(SQLText, con);
            daUser.Fill(dsUser, "User");
            con.Close();
            if (dsUser.Tables[0].Rows.Count > 0)
            {
                lblStatus.Text = "You entered wrong current password.";
            }
            else
            {
                SQLText = "UPDATE Users SET password = '" + Encryption.EncryptSHA512Managed(txtPassword.Text) + "' WHERE userID = " +
                  Session["UserID"].ToString();
                cmd = new MySqlCommand(SQLText, con);
                con.Open();
                cmd.ExecuteNonQuery();
                con.Close();
                lblStatus.Text = "Password changed successfully";
            }
        }
    }
}
```

Şekil 10.11 : Şifre değiştirme (ChangePassword.aspx) C# kodları

Sisteme giriş yapan kullanıcının şifresini değiştirebilmesini sağlayan sayfa tasarımına ve program kodlarına ait bilgilerde şifreleme işleminin yapıldığı sınıfa ait referans bilgisi görüntülenebilmektedir. Bu yapıda öncelikle kullanıcının mevcut şifresinin kontrolü yapılmakta, eğer sistemdeki şifre ile tutarlı ise şifre değişiklik işlemi gerçekleştirilmektedir. Bu durum ek bir güvenlik işlemi için yapılmıştır.

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Login.aspx.cs" Inherits="PackageCapture.Login" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Package Capture</title>
    <meta charset="utf-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta content="width=device-width, initial-scale=1, maximum-scale=1, user-scalable=no" name="viewport" />
    <link rel="stylesheet" href="Bootstrap/css/bootstrap.min.css" />
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/font-awesome/4.4.0/css/font-awesome.min.css" />
    <link rel="stylesheet" href="https://code.ionicframework.com/ionicons/2.0.1/css/ionicons.min.css" />
    <link rel="stylesheet" href="Style/AdminLTE.min.css" />
    <link rel="stylesheet" href="Plugins/iCheck/square/blue.css" />
</head>
<body class="hold-transition login-page">
    <form id="form1" runat="server">
        <div class="login-box">
            <div class="login-logo">
                <a href="../../Default.aspx"><b>PACKAGE CAPTURE</b></a>
            </div>
            <div class="login-box-body">
                <div class="form-group has-feedback">
                    <asp:TextBox runat="server" ID="txtUserName" CssClass="form-control"></asp:TextBox>
                    <span class="glyphicon glyphicon-envelope form-control-feedback"></span>
                </div>
                <div class="form-group has-feedback">
                    <asp:TextBox runat="server" ID="txtPassword" TextMode="Password" CssClass="form-control"></asp:TextBox>
                    <span class="glyphicon glyphicon-lock form-control-feedback"></span>
                </div>
                <div class="row">
                    <div class="col-xs-4">
                        <asp:Button runat="server" ID="btnRegister" Text="Register" CssClass="btn btn-primary btn-block btn-flat"
                          OnClick="btnRegister_Click" BackColor="Green" />
                    </div>
                    <div class="col-xs-4">
                        <asp:Button runat="server" ID="btnLogin" Text="Login" CssClass="btn btn-primary btn-block btn-flat"
                          OnClick="btnLogin_Click" />
                    </div>
                </div>
                <br />
                <div class="col-xs-8">
                    <asp:Label runat="server" ID="lblStatus"></asp:Label>
                </div>
            </div>
        </div>
    </form>
</body>
</html>
```

Şekil 10.12 : Kullanıcı girişi (Login.aspx) tasarımsal HTML kodları

```
2 references
public partial class Login : BasePage
{
    0 references
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            txtUserName.Attributes.Add("placeholder", "User name");
            txtPassword.Attributes.Add("placeholder", "Password");
        }
    }

    0 references
    protected void btnLogin_Click(object sender, EventArgs e)
    {
        try
        {
            MySqlConnection con = new MySqlConnection(ConfigurationManager.ConnectionStrings["ConStr"].ConnectionString);
            string SQLText = "SELECT userID, username, name, lastname, email, createdDate, status FROM Users WHERE username = '" +
              txtUserName.Text + "' AND password = '" + Encryption.EncryptSHA512Managed(txtPassword.Text)  + "' AND status = 1";
            MySqlCommand cmd = new MySqlCommand(SQLText, con);
            con.Open();
            DataSet dsUser = new DataSet();
            MySqlDataAdapter daUser = new MySqlDataAdapter(SQLText, con);
            daUser.Fill(dsUser, "User");
            if (dsUser.Tables[0].Rows.Count > 0)
            {
                Session["UserID"] = dsUser.Tables[0].Rows[0]["userID"].ToString();
                Session["UserName"] = dsUser.Tables[0].Rows[0]["username"].ToString();
                Session["Name"] = dsUser.Tables[0].Rows[0]["name"].ToString();
                Session["LastName"] = dsUser.Tables[0].Rows[0]["lastname"].ToString();
                Session["FullName"] = Session["Name"].ToString() + " " + Session["LastName"].ToString();
                Session["Email"] = dsUser.Tables[0].Rows[0]["email"].ToString();
                Response.Redirect("Default.aspx");
            }
        }

        catch (Exception ex)
        {
            Response.Write(ex.ToString());
        }
    }

    0 references
    protected void btnRegister_Click(object sender, EventArgs e)
    {
        Response.Redirect("Register.aspx");
    }
}
```

Şekil 10.13 : Kullanıcı girişi (Login.aspx) C# kodları

Kullanıcının sisteme giriş yapacağı sayfa içerisinde kullanıcı adı ve şifrenin girileceği bri TextBox alanı bulunmaktadır. Şifrenin girişini sağlayan TextBox alanının şifre girişine uygun olması için TextMode özelliği Password olarak ayarlanmıştır. Girilen şifrenin veritabanına hash edilerek gönderilmesini sağlanmıştır. Kullanıcı girişi başarılı olduktan sonra ad, soyad, e-posta gibi bilgilerin daha sonra kullanılabilmesi için Session üzerinde tutulması sağlanmıştır. Kullanıcının giriş yapıp yapmadığı Session üzerindeki UserID değeri ile kontrol edilecektir.

```
namespace DataLayer
{
    4 references
    public static class Encryption
    {
        4 references
        public static string EncryptSHA512Managed(string password)
        {
            UnicodeEncoding uEncode = new UnicodeEncoding();
            byte[] bytPassword = uEncode.GetBytes(password);
            SHA512Managed sha = new SHA512Managed();
            byte[] hash = sha.ComputeHash(bytPassword);
            return Convert.ToBase64String(hash);
        }
    }
}
```

Şekil 10.14 : Şifreleme işlemi için kullanılacak sınıfa ait C# kodları

```
<div class="content-wrapper">
    <section class="content-header">
        <h1>Package Information</h1>
    </section>
    <section class="content">
        <div class="row">
            <section class="content" style="min-height: 100px !important">
                <asp:UpdatePanel runat="server">
                    <ContentTemplate>
                        <div style="width:100%; height:250px">
                            <div style="float: left; width: 300px">
                                <div>
                                    Start Date
                                </div>
                                <div>
                                    <asp:Calendar runat="server" ID="calStartDate" CssClass="form-control" Width="250px" BackColor="Transparent" BorderStyle="None"
                                      Height="200px">
                                        <SelectedDayStyle BorderColor="#FF3300" ForeColor="White" />
                                    </asp:Calendar>
                                </div>
                            </div>
                            <div style="float: left">
                                <div style="width: 100%">
                                    End Date
                                </div>
                                <div style="width: 100%">
                                    <asp:Calendar runat="server" ID="calEndDate" CssClass="form-control" Width="250px" BackColor="Transparent" BorderStyle="None"
                                      Height="200px"></asp:Calendar>
                                </div>
                            </div>
                        </div>
                        <div>
                            <div style="float: left; width: 300px">
                                <div style="width: 100%">
                                    Start Time
                                </div>
                                <div style="width: 100%; height: 60px">
                                    <asp:DropDownList runat="server" ID="ddlStartHour" CssClass="form-control" Width="100px" Style="float: left"></asp:DropDownList> 
                                    <asp:DropDownList runat="server" ID="ddlStarMinute" CssClass="form-control" Width="100px" Style="float: left"></asp:DropDownList> 
                                </div>
                            </div>
                            <div>
                                <div>
                                    End Time
                                </div>
                                <div>
                                    <asp:DropDownList runat="server" ID="ddlEndHour" CssClass="form-control" Width="100px" Style="float: left"></asp:DropDownList> 
                                    <asp:DropDownList runat="server" ID="ddlEndMinute" CssClass="form-control" Width="100px" Style="float: left"></asp:DropDownList> 
                                </div>
                            </div>
                        </div>
                        <br />
                        <br />
                    </ContentTemplate>
                </asp:UpdatePanel>
                <asp:Label runat="server" ID="lblStatus" ForeColor="Blue" Font-Size="20px" Font-Bold="true"></asp:Label>
                <div style="height: 44px; margin-top: 12px">
                    <asp:Button runat="server" ID="btnList" Text="List" CssClass="btn btn-sm btn-info btn-flat pull-left" Width="100px" OnClick="btnList_Click" />
                </div>
            </section>
```

Şekil 10.15 : Paketler sayfasının (Packages.aspx) tasarımsal HTML kodları – 1

```
        <div class="row">
            <div class="box box-info">
                <div class="box-body">
                    <asp:Label runat="server" ID="lblRecordCount" ForeColor="Red" Font-Bold="true" Font-Size="15px"></asp:Label><br />
                    <div class="table-responsive">
                        <table class="table no-margin">
                            <thead>
                                <tr>
                                    <th>Destination IP</th>
                                    <th>Protocol</th>
                                    <th>PackLength</th>
                                    <th>Date</th>
                                </tr>
                            </thead>
                            <tbody>
                                <asp:Repeater runat="server" ID="rptPackages">
                                    <ItemTemplate>
                                        <asp:Label runat="server" ID="lblID" Text='<%# DataBinder.Eval(Container.DataItem, "id") %>' Visible="False"></asp:Label>
                                        <tr style="height: 45px">
                                            <td><%# DataBinder.Eval(Container.DataItem, "destIP") %></td>
                                            <td><%# DataBinder.Eval(Container.DataItem, "protocol") %></td>
                                            <td><%# DataBinder.Eval(Container.DataItem, "packLength") %></td>
                                            <td><%# DataBinder.Eval(Container.DataItem, "date") %></td>
                                        </tr>
                                    </ItemTemplate>
                                </asp:Repeater>
                            </tbody>
                        </table>
                    </div>
                </div>
            </div>
        </div>
    </section>
</div>
</asp:Content>
```

Şekil 10.16 : Paketler sayfasının (Packages.aspx) tasarımsal HTML kodları - 2

```
2 references
public partial class Packages : System.Web.UI.Page
{
    0 references
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            for (int i = 0; i < 24; i++)
            {
                string TextVal = i.ToString();
                if (i < 10)
                {
                    TextVal = "0" + i.ToString();
                }
                ddlStartHour.Items.Add(new ListItem(TextVal, TextVal));
                ddlEndHour.Items.Add(new ListItem(TextVal, TextVal));
            }
            for (int j = 0; j < 60; j++)
            {
                string TextVal = j.ToString();
                if (j < 10)
                {
                    TextVal = "0" + j.ToString();
                }
                ddlStarMinute.Items.Add(new ListItem(TextVal, TextVal));
                ddlEndMinute.Items.Add(new ListItem(TextVal, TextVal));
            }
            calStartDate.SelectedDate = DateTime.Now.AddDays(-1);
            calEndDate.SelectedDate = DateTime.Now;
        }
    }

    0 references
    protected void btnList_Click(object sender, EventArgs e)
    {
        DateTime datStart = calStartDate.SelectedDate.AddHours(Convert.ToInt32(ddlStartHour.SelectedValue)).AddMinutes(Convert.ToInt32
          (ddlStarMinute.SelectedValue));
        DateTime datEnd = calEndDate.SelectedDate.AddHours(Convert.ToInt32(ddlEndHour.SelectedValue)).AddMinutes(Convert.ToInt32
          (ddlEndMinute.SelectedValue));
        MySqlConnection con = new MySqlConnection(ConfigurationManager.ConnectionStrings["ConStr"].ConnectionString);
        string SQLText = "SELECT id, destIP, protocol, packLength, date FROM Packages WHERE (userID = " + Session["UserID"].ToString() + ") AND (date >=
          '" + datStart.ToString("yyyy-MM-dd HH:mm:ss") + "' AND date <= '" + datEnd.ToString("yyyy-MM-dd HH:mm:ss") + "')";
        MySqlCommand cmd = new MySqlCommand(SQLText, con);
        con.Open();
        DataSet dsPackages = new DataSet();
        MySqlDataAdapter daUser = new MySqlDataAdapter(SQLText, con);
        daUser.Fill(dsPackages, "Packages");
        rptPackages.DataSource = dsPackages.Tables[0];
        lblRecordCount.Text = "Total " + dsPackages.Tables[0].Rows.Count.ToString() + " record";
        rptPackages.DataBind();
    }
}
```

Şekil 10.17 : Paketler sayfası (Login.aspx) C# kodları

Packages sayfası içerisinde ASP.NET kontroller içerisindeki takvim aracı ile başlangıç ve bitiş tarihlerinin seçiminin yapılabilmesi ve saat ayrıntısının seçilebilmesi için dört adet dropdownlist kullanılmıştır. Başlangıç tarihi ve saati ile bitiş tarihi ve saati doğrultusunda sisteme giriş yapan kullanıcıya ait Session üzerindeki id değeri kullanılarak sorgulama yapılmakta ve gelen sorgular ilgili sayfada listelenmektedir. Listenin üst kısmında kaç kayıt geldiine ilişkin bir bilgi de sayısal olarak eklenmiştir.

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site1.Master" AutoEventWireup="true" CodeBehind="Profile.aspx.cs" Inherits="PackageCapture.Profile" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" runat="server">
    <asp:ScriptManager runat="server"></asp:ScriptManager>
    <div class="content-wrapper">
        <section class="content-header">
            <h1>Profile Information</h1>
        </section>
        <section class="content">
            <div class="row">
                <section class="content" style="min-height: 100px !important">
                    <div>
                        Name
                    </div>
                    <div>
                        <asp:TextBox runat="server" ID="txtName" CssClass="form-control" Width="460px"></asp:TextBox>
                    </div>
                    <div>
                        Surname
                    </div>
                    <div>
                        <asp:TextBox runat="server" ID="txtLastName" CssClass="form-control" Width="460px"></asp:TextBox>
                    </div>
                    <div>
                        E-mail
                    </div>
                    <div>
                        <asp:TextBox runat="server" ID="txtMail" CssClass="form-control" Width="460px"></asp:TextBox>
                    </div>
                    <br />
                    <br />
                    <asp:Label runat="server" ID="lblStatus" ForeColor="Blue" Font-Size="20px" Font-Bold="true"></asp:Label>
                    <div style="height: 44px; margin-top: 12px">
                        <asp:Button runat="server" ID="btnSave" Text="Save" CssClass="btn btn-sm btn-info btn-flat pull-left" Width="100px"
                          OnClick="btnSave_Click" />
                    </div>
                </section>
            </div>
        </section>
    </div>
</asp:Content>
```

Şekil 10.18 : Kullanıcı profil sayfasının (Profile.aspx) tasarımsal HTML kodları

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        MySqlConnection con = new MySqlConnection(ConfigurationManager.ConnectionStrings["ConStr"].ConnectionString);
        string SQLText = "SELECT userID, username, name, lastname, email, createdDate, status FROM Users WHERE userID = " + Session["UserID"].ToString
          ();
        MySqlCommand cmd = new MySqlCommand(SQLText, con);
        con.Open();
        DataSet dsUser = new DataSet();
        MySqlDataAdapter daUser = new MySqlDataAdapter(SQLText, con);
        daUser.Fill(dsUser, "User");
        con.Close();
        if (dsUser.Tables[0].Rows.Count > 0)
        {
            txtName.Text = dsUser.Tables[0].Rows[0]["name"].ToString();
            txtLastName.Text = dsUser.Tables[0].Rows[0]["lastname"].ToString();
            txtMail.Text = dsUser.Tables[0].Rows[0]["email"].ToString();
        }
    }
}
0 references
protected void btnSave_Click(object sender, EventArgs e)
{
    if (txtName.Text == string.Empty)
        lblStatus.Text = "Username can not be empty";
    else if (txtName.Text == string.Empty)
        lblStatus.Text = "Name can not be empty";
    else if (txtLastName.Text == string.Empty)
        lblStatus.Text = "Last name can not be empty";
    else if (txtMail.Text == string.Empty)
        lblStatus.Text = "Mail name can not be empty";
    else
    {
        MySqlConnection con = new MySqlConnection(ConfigurationManager.ConnectionStrings["ConStr"].ConnectionString);
        string SQLText = "SELECT userID, username, name, lastname, email, createdDate, status FROM Users WHERE email = '" + txtMail.Text + "' and
          userID <> " + Session["UserID"].ToString();
        MySqlCommand cmd = new MySqlCommand(SQLText, con);
        con.Open();
        DataSet dsUser = new DataSet();
        MySqlDataAdapter daUser = new MySqlDataAdapter(SQLText, con);
        daUser.Fill(dsUser, "User");
        con.Close();
        if (dsUser.Tables[0].Rows.Count > 0)
            lblStatus.Text = "E-mail exists in our system. Please enter a different e-mail";
        else
        {
            SQLText = "UPDATE Users SET name = '" + txtName.Text + "', lastname = '" + txtLastName.Text + "', email = '" + txtMail.Text + "' WHERE
              userID = " + Session["UserID"].ToString();
            cmd = new MySqlCommand(SQLText, con);
            con.Open();
            cmd.ExecuteNonQuery();
            con.Close();
            lblStatus.Text = "Profile information updated successfully";
        }
    }
```

Şekil 10.19 : Kullanıcı profil sayfasının (Login.aspx) C# kodları

```
<body class="hold-transition login-page">
    <form id="form1" runat="server">
        <div class="login-box">
            <div class="login-logo">
                <a href="../../Default.aspx"><b>PACKAGE CAPTURE</b></a>
            </div>
            <div class="login-box-body">
                <div style="height: 40px; width: 100%">
                    <div style="float: left; width: 150px; padding-top:7px">
                        User Name :
                    </div>
                    <div style="float: left; width: 270px"><asp:TextBox runat="server" ID="txtUserName" CssClass="form-control"></asp:TextBox> </div>
                </div>
                <div style="height: 40px; width: 100%">
                    <div style="float: left; width: 150px; padding-top:7px">
                        Password :
                    </div>
                    <div style="float: left; width: 270px"><asp:TextBox runat="server" ID="txtPassword" CssClass="form-control" TextMode="Password"></
                      asp:TextBox>
                    </div>
                </div>
                <div style="height: 40px; width: 100%">
                    <div style="float: left; width: 150px; padding-top:7px">
                        Confirm Password :
                    </div>
                    <div style="float: left; width: 270px"><asp:TextBox runat="server" ID="txtPasswordRe" CssClass="form-control" TextMode="Password"></
                      asp:TextBox></div>
                </div>
                <div style="height: 40px; width: 100%">
                    <div style="float: left; width: 150px; padding-top:7px">
                        Name :
                    </div>
                    <div style="float: left; width: 270px"><asp:TextBox runat="server" ID="txtName" CssClass="form-control"></asp:TextBox></div>
                </div>
                <div style="height: 40px; width: 100%">
                    <div style="float: left; width: 150px; padding-top:7px">
                        Last Name :
                    </div><div style="float: left; width: 270px"><asp:TextBox runat="server" ID="txtLastName" CssClass="form-control"></asp:TextBox></div>
                </div>
                <div style="height: 40px; width: 100%">
                    <div style="float: left; width: 150px; padding-top:7px">
                        E-mail :
                    </div>
                    <div style="float: left; width: 270px"><asp:TextBox runat="server" ID="txtMail" CssClass="form-control"></asp:TextBox></div>
                </div>
                <br />
                <asp:Label runat="server" ID="lblStatus" ForeColor="Red"></asp:Label><br /><br />
                <div style="height: 40px; width: 100%"><asp:Button runat="server" ID="btnSubmit" Text="Register" OnClick="btnSubmit_Click" CssClass="btn btn-
                  primary btn-block btn-flat" /></div>
            </div>
        </div>
    </form>
</body>
```

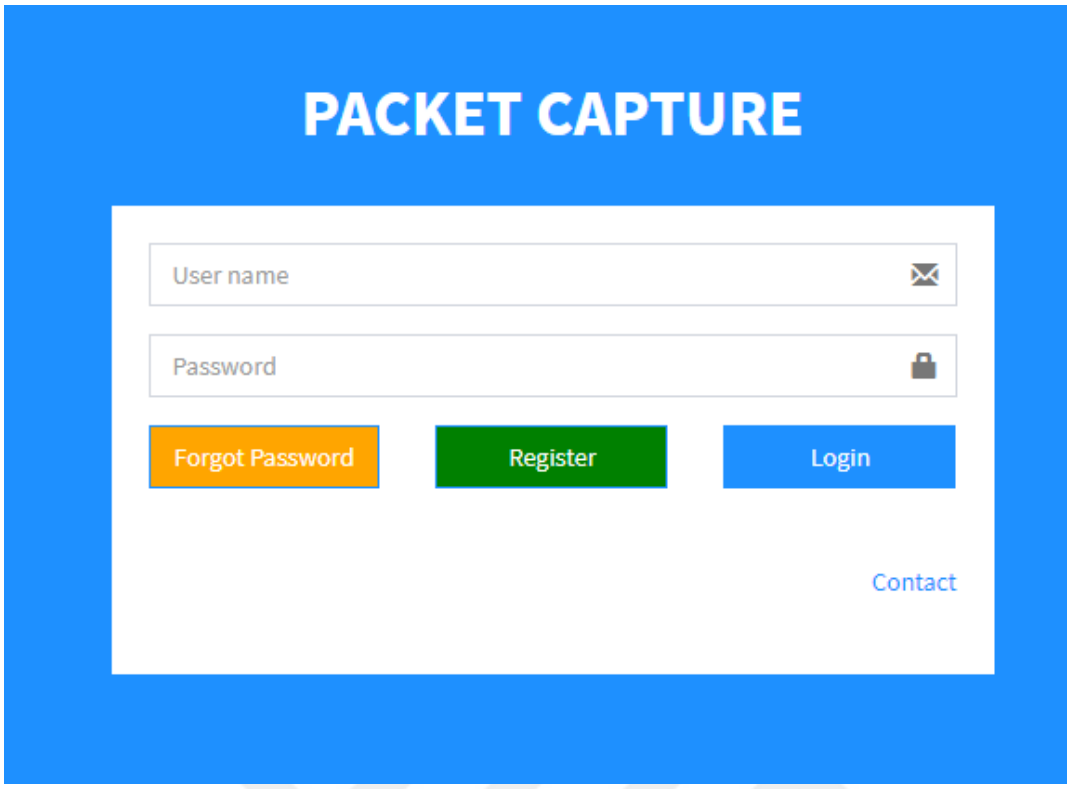
Şekil 10.20 : Kullanıcı kayıt sayfasının (Register.aspx) tasarımsal HTML kodları

```
0 references
protected void btnSubmit_Click(object sender, EventArgs e)
{
    try
    {
        if (txtUserName.Text == string.Empty)
            lblStatus.Text = "Username can not be empty";
        else if (txtPassword.Text == string.Empty)
            lblStatus.Text = "Password can not be empty";
        else if (txtPassword.Text != txtPasswordRe.Text)
            lblStatus.Text = "Passwords not match";
        else if (txtName.Text == string.Empty)
            lblStatus.Text = "Name can not be empty";
        else if (txtLastName.Text == string.Empty)
            lblStatus.Text = "Last name can not be empty";
        else if (txtMail.Text == string.Empty)
            lblStatus.Text = "Mail name can not be empty";
        else
        {
            MySqlConnection con = new MySqlConnection(ConfigurationManager.ConnectionStrings["ConStr"].ConnectionString);
            string SQLText = "SELECT userID, username, name, lastname, email, createdDate, status FROM Users WHERE username = '" + txtUserName.Text +
              "'";
            MySqlCommand cmd = new MySqlCommand(SQLText, con);con.Open();
            DataSet dsUser = new DataSet();
            MySqlDataAdapter daUser = new MySqlDataAdapter(SQLText, con);
            daUser.Fill(dsUser, "User");con.Close();
            if (dsUser.Tables[0].Rows.Count > 0)
                lblStatus.Text = "Username exists in our system. Please enter a different username";
            else
            {
                SQLText = "SELECT userID, username, name, lastname, email, createdDate, status FROM Users WHERE email = '" + txtMail.Text + "'";
                cmd = new MySqlCommand(SQLText, con);
                con.Open();dsUser = new DataSet();
                daUser = new MySqlDataAdapter(SQLText, con);
                daUser.Fill(dsUser, "User");con.Close();
                if (dsUser.Tables[0].Rows.Count > 0)
                    lblStatus.Text = "Email exists in our system. Please enter a different e-mail";
                else
                {
                    SQLText = "INSERT INTO Users(username, password, name, lastname, email) VALUES ('" + txtUserName.Text + "','" +
                      Encryption.EncryptSHA512Managed(txtPassword.Text) + "','" + txtName.Text + "','" + txtLastName.Text + "','" + txtMail.Text +
                      "')";
                    cmd = new MySqlCommand(SQLText, con);
                    con.Open();
                    cmd.ExecuteNonQuery();
                    con.Close();
                    lblStatus.Text = "User created successfully. <a href=\"Login.aspx\">Please click here to login</a>";
                }
            }
        }
    }
    catch (Exception ex)
    {
        lblStatus.Text = ex.ToString();
    }
}
```

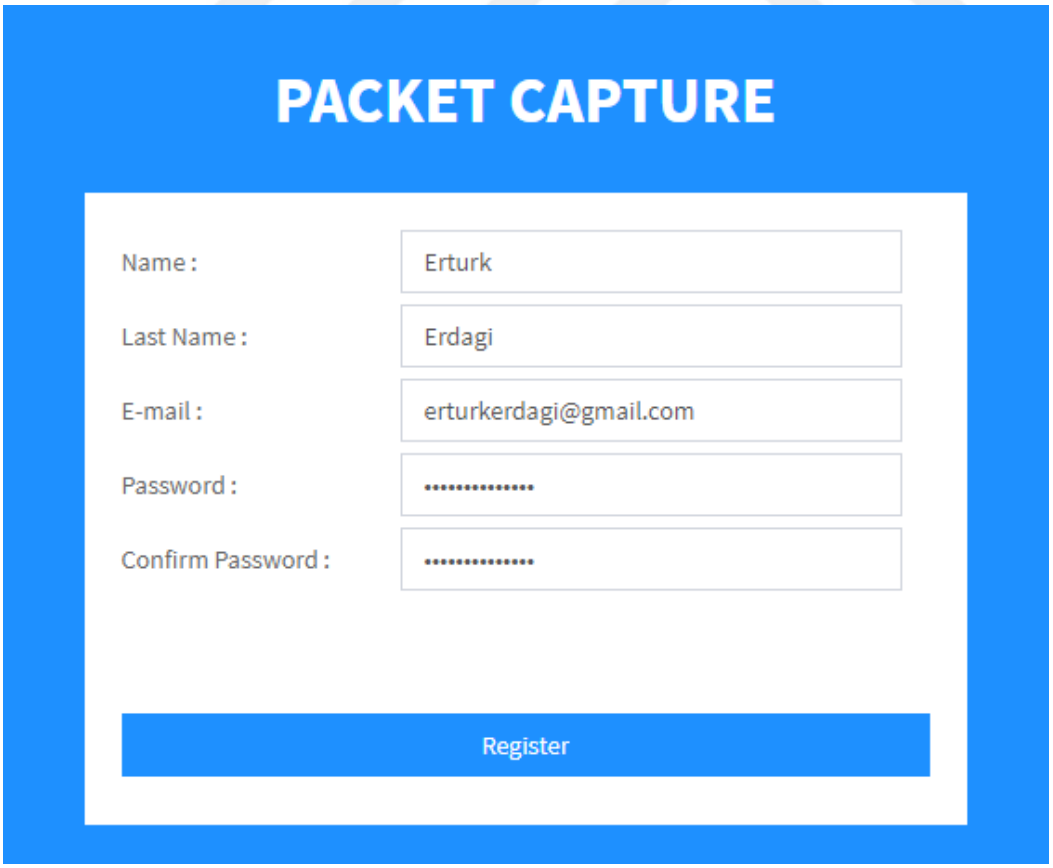
Şekil 10.21 : Kullanıcı kayıt sayfasının (Register.aspx) C# kodları

Kullanıcı giriş sayfası üzerindeki Register butonu ile ulaşılabilecek bu sayfada sistemde bulunmayan bir kullanıcı adı ve e-posta adresi girilerek yeni bir kullanıcı oluşturulabilmekte. Belirlenen şifre diğer modüllerde olduğu gibi hash edilerek veritabanına gönderilmektedir.

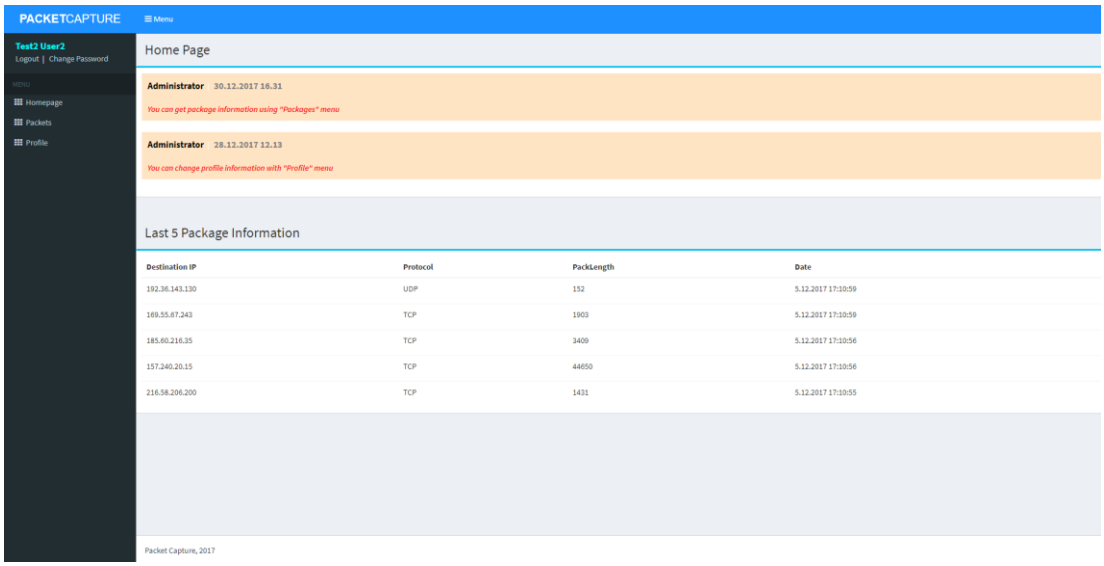
Tasarım ve kodlama aşaması tamamlandıktan sonra projenin adım adım modüllerinin ekran görüntüleri paylaşılacaktır.



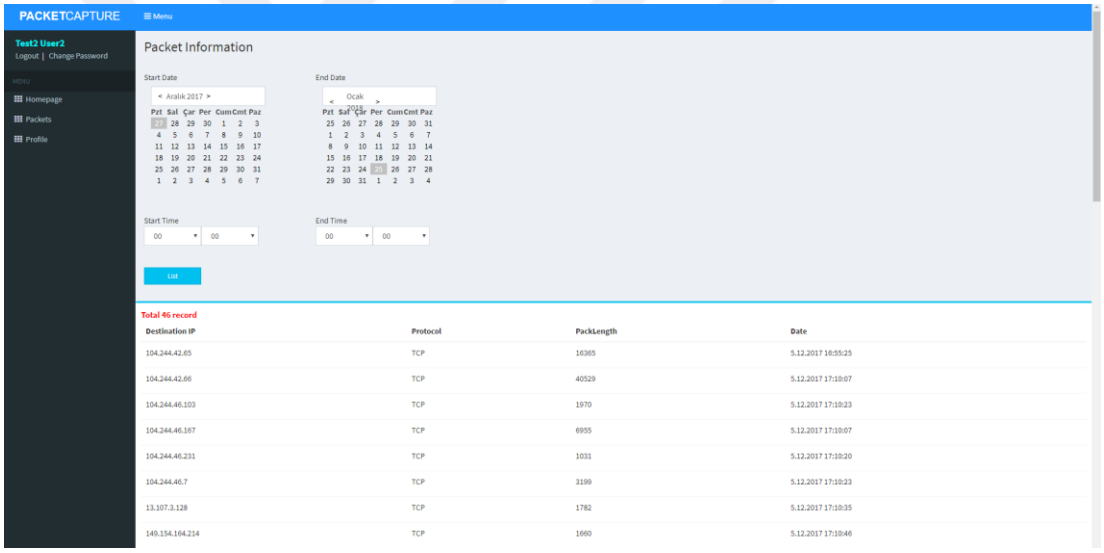
Şekil 10.22 : Kullanıcı giriş sayfası



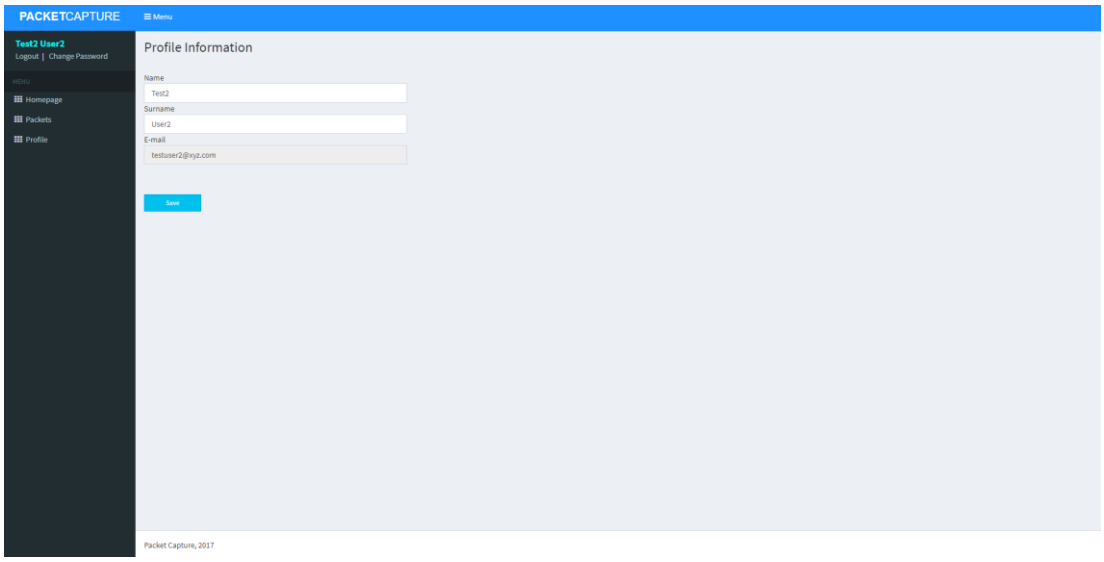
Şekil 10.23 : Kullanıcı kayıt sayfası



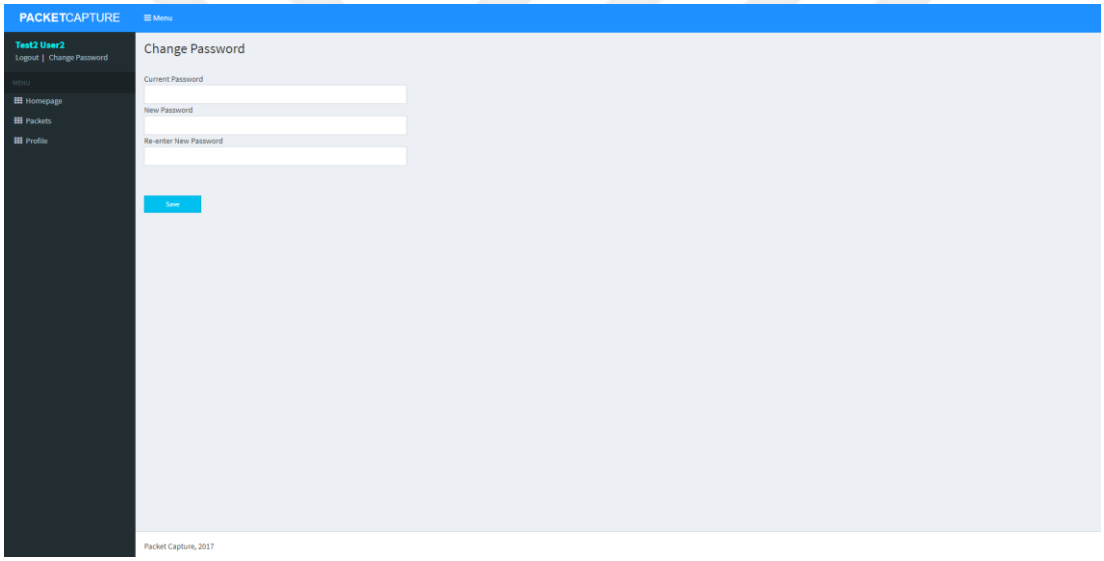
Şekil 10.24 : Anasayfa



Şekil 10.25 : Paketler sayfası



Şekil 10.26 : Profil sayfası



Şekil 10.27 : Şifre değiştirme sayfası

11. SONUÇ VE ÖNERİLER

Website ve uygulamaların kullanmış olduğu veritabanlarının güvenli hale getirilmesi için yapılması gerekenler birkaç adımdan oluşmaktadır.

- Uygulamanın mevcut durumunun analiz edilerek varolan tüm açıklarının tespit edilmesi gerkmektedir. Açıklara ait sonrasında yapılacak işlemler için bu adım bir temel niteliğindedir.

- Uygulamanın geliştirilme sürecinde yazılan kodlar belirli bir standart temeline oturtulmalıdır.

- Gerek veritabanı, gerek uygulama üzerinede ypılacak her ekleme ve düzenleme işlemi için güvenlik denetiminin uygulanması gereklidir.

- Güvenliğin zaafiyete uğrayabileceği noktaların analiz edilmesi ve ek güvenlik önlemlerinin alınması gerekmektedir.

Temel olarak enjeksiyon kontrolü uygulamanın denetimi ile olmaktadır. Fakat bu durum çok fazla maliyet ve zaman gerektirmektedir. Uygulama kodlarının denetlenmesi sırasında alanında uzman olmayan bir kişinin yapacağı denetim tam anlamıyla korunma stratejisine uymayacaktır. Bu noktada otomatik olarak denetim yapan tarayıcıların kullanılması zaman ve maliyet açısından büyük fayda sağlayacaktır. Bu noktada kullanılacak tarayıcılar URL üzerinden ve sayfa içerisinde çalıştırılan sorguların denetimini yaparak enjeksiyon uygulamasına karşın dayanıklılık testlerini gerçekleştirir.

Uygulama kullanım sırasında oluşabilecek hataya ait gösterilen açıklamalar bu noktayı kullanarak sızmak isteyen kullanıcılar için yol gösterici nitelikte olabilir. Dikkat edilmesi gereken nokta çıkan hatanın kaydının tutulması ve sonrasında bu hata ile ilgili güvenlik önleminin alınmasıdır. Hata içeriğinde satır numarası ve sorgu işlemlerine ilikin programlama metinlerinin verilmesi sızma işlemleri için ipucu niteliğinde olabilir.

Veri içeriği ve miktarına göre büyük ölçekli ve önemli uygulamaların sahip olduğu veritabanları tek noktada tutulması güvenlik zafiyetine sebep olabilir. Burada önemli nokta bir veritabanındaki bilgiye erişildiğinde oluşacak olumsuzluktan diğer veritabanındaki bilgilerin etkilenmemesidir. Bu durum güvenlik politikasının gereği olarak yapılmasının yanında sistemin daha hızlı çalışmasını sağlamak için kullanılabilecek bir yöntem olarak da yapılabilir. Uygulama içerisinde aynı işlevleri görseler dahi farklı veritabanlarının bulunması herbir veritabanındaki kullanıcı haklarının birbirlerinden farklı olarak oluşturulmuş ve denetlenmiş olmasını gereklidir.

Erişim denetiminin tek bir noktadan yapılması önemlidir. Farklı yerlerden, farklı kullanıcılar tarafından yapılan yetkilendirmeler karışıklığa sebep olabilir.

Uygulamanın kullanımı esnasında yapılacak her türlü hamlenin kayıt altına alınması ve sonrasında yapılacak bir analiz için anlaşılır şekilde olması gerekmektedir. Sistematik olarak ilerleyen bir hata ya da açıklığın uygulama geliştirici tarafından fark edilmemesi mümkündür. Dolayısıyla bu analizin sağlıklı yapılabilmesi adına adım adım kayıt işlemi yapılmalıdır.

SQL enjeksiyon saldırısına ait hasarı minimuma indirgemek için, her bir veritabanı kullanıcısı için verilecek yetkilendirmenin sağlıklı bir şekilde yapılandırılması - gerekmektedir. Uygulamanın veritabanındaki işlemleri gerçekleştirmesi için açılmış olan kullanıcı hesabının veritabanı yöneticisi olmaması olmaması, her türlü erişimin sağlanması yönünde engel teşkil edecektir. Kişilere verilecek yetki yerine uygulamanın ihtiyaç duyacağı işlemler için yetkilendirme en sağlıklısıdır. Okuma erişimine gereksinim duyan bir hesaba yalnızca erişim yapabileceği tablolara okuma erişimi sağlanması gerekmektedir.

KAYNAKLAR

[1] **Url-1** < *https://technet.microsoft.com/tr-tr/library/cc280361(v=sql.100).aspx*>, 25.09.2017 tarihinde erişildi.

[2] **Y. VURAL and Ş. SAĞIROĞLU** (2010) Veritabanı Yönetim Sistemleri Güvenliği: Tehditler ve Korunma Yöntemleri, Politeknik Dergisi.

[3] **Url-2** < *https://en.wikipedia.org/wiki/Charles_Bachman* >, 15.10.2017 tarihinde erişildi.

[4] **Url-3**<*https://social.technet.microsoft.com/wiki/contents/articles/24995.sifreleme-encryption-esaslar-bolum-1-tr-tr.aspx*>, 10.09.2017 tarihinde erişildi.

[5] **M. A. SALAHLI** (2016) BM315 Veritabanı Yönetim Sistemleri (2016), MEB

[6] **L. Nichols** (2007) A Comparison of Object-Relational and Relational Databases, the Faculty of California Polytechnic State University, San Luis Obispo.

[7] **M. Aljarallah** (2014) Comperative Study of Database Modelling Approaches, Algoma University.

[8] **A. Carlsson** (2003) Object oriented databases – a natural part of object oriented software development.

[9] **Asieh Mokarian, Ahmad Faraahi, Arash Ghorbannia Delavar** (2013) False Positives Reduction Techniques in Intrusion Detection Systems-A Review, IJCSNS International Journal of Computer Science and Network 128 Security, VOL.13 No.10, October 2013.

[10] **Milli Eğitim Bakanlığı** (2012) Ağ Veritabanı Yönetimi, Bilişim Teknolojileri, Ankara.

[11] **L. Wichman** (2010) An motivational approach to self: Integration in personality. In R. Dienstbier (Ed.), Nebraska Symposium on Motivation: Vol. 38. Perspectives on motivation (pp. 237-288). Lincoln, University of Nebraska Press.

[12] **Barikat İnternet Güvenliği Bilişim Tic.Ltd.Şti.** (2013) En Önemli On Veritabanı Güvenliği Tehdidi.

[13] **R. D. M. B. Doygun DEMİROL** (2013) SQL Enjeksiyon Saldırı Uygulaması ve Güvenlik Önerileri, International Symposium on Digital Forensics and Security (ISDFS'13), Elazığ.

[14] **URL-4** <*https://ferruh.mavituna.com/sql-injection-a-giris-ve-sql-injection-nedir-oku*>, 15.05.2017 tarihinde erişildi.

ÖZGEÇMİŞ

Ad-Soyad **:** Ertürk ERDAĞI

Doğum Tarihi ve Yeri **:** 06.04.1987 – Kars

E-posta **:** erdagie@itu.edu.tr

ÖĞRENİM DURUMU:

- **Lisans** **:** 2009, Marmara Üniversitesi, Bilgisayar ve Öğretim Teknolojileri

YAYINLAR, SUNUMLAR VE PATENTLER:

- **Erdağı E.**, 2017: Milli Eğitim Bakanlığı Eğitimde Bilişim Ağı.Konferansı, Bilgi Güvenliği Semineri, 2017 İstanbul, Türkiye.