TEK KULLANIMLIK (ATILABİLİR) KİMLİK DOĞRULAMALI ANAHTAR DEĞİŞİMİ YAZILIM UYGULAMASININ GELİŞTİRİLMESİ

A SOFTWARE DEVELOPMENT OF ONE-TIME (DISPOSABLE) ID AUTHENTICATED KEY EXCHANGE APPLICATION

ALİ UMUT YÜKSEL

Hacettepe Üniversitesi

Lisansüstü Eğitim-Öğretim ve Sınav Yönetmeliğinin

ELEKTRİK ve ELEKTRONİK Mühendisliği Anabilim Dalı İçin Öngördüğü

YÜKSEK LİSANS TEZİ

olarak hazırlanmıştır.

2006

Fen Bilimleri Enstitüsü Müdürlüğü'ne,

Bu çalışma jürimiz tarafından **ELEKTRİK ve ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI 'nda YÜKSEK LİSANS** olarak kabul edilmiştir.

Başkan :….....(İmza).....
(Ünvanı, Adı ve Soyadı)

Üye (Danışman) :….....(İmza).....
(Ünvanı, Adı ve Soyadı)

Üye (Var ise Eş Danışman) :….....(İmza).....
(Ünvanı, Adı ve Soyadı)

Üye :….....(İmza).....
(Ünvanı, Adı ve Soyadı)

Üye :….....(İmza).....
(Ünvanı, Adı ve Soyadı)

ONAY

Bu tez ...../...../..... tarihinde Enstitü Yönetim Kurulunca belirlenen yukarıdaki jüri üyeleri tarafından kabul edilmiştir.

...../...../.....

Prof.Dr. Ahmet R.ÖZDURAL
FEN BİLİMLERİ ENSTİTÜSÜ MÜDÜRÜ

TEK ZAMANLI (ATILABİLİR) KİMLİK DOĞRULAMALI ANAHTAR DEĞİŞİMİ YAZILIM UYGULAMASININ GELİŞTİRİLMESİ

Ali Umut YÜKSEL

ÖZ

Günümüz gelişen teknolojisinde, özellikle internet tabanlı uygulamaların yaygınlaşmasıyla beraber yetkili kullanıcıların ayırt edilmesi ve herkese açık, güvensiz geniş alan ağları üzerinden güvenli iletişim gereksinimi büyük bir önem kazanmıştır.

Ağ uygulamalarında güvenli iletişim ve doğrulamayı sağlamak için geliştirilen ve sıklıkla kullanılmakta olan açık anahtar altyapısı ve Diffie-Helman temelli anahtar değişimi yöntemleri her ne kadar matematiksel olarak güvenilir (Computationally Secure) olarak kabul edilse de halen kaynağın güvenirliliğini hedef alan ve günümüzde bir çok ağ uygulamasının çalışabilirliğini engellemede en çok başvurulan yöntem olan DOS ataklarına, saldırganın elde ettiği geçerli kullanıcı kimliklerini kullanarak (impersonate) gerçekleştirdiği tekrar ataklarına (replay attacks) karşı açıklar bulundurmakta ve kullanıcı gizliliği (user privacy), kullanıcı erişilmezliği (user untraceability), kullanıcı kimlik aktarımının engellenmesi (leakage of users identiy) gibi gereksinimleri karşılayamamaktadır.

İnternet kullanıcıları, banka hesap numarası, kredi kart numarası gibi bilgilerini paylaştıkları, bankacılık işlemlerini yerine getirdikleri elektronik siteler için belli bir güvenlik seviyesine sahip boyuttaki parola ve kullanıcı kimlik bilgilerini hatırında tutmak yada her bir site için ayrı bir şifre üretme cihazı kullanmak zorunda kalmaktadır.

Hızla genişleyen ağ ortamında diğer bir gereksinim ise, kullanıcının ağ üzerinde gerçekleştirmek istediği işlemlerin verim ve güvenliğini arttırmak noktasında, elektronik hizmet siteleri arasındaki ortak çalışılabilirliğin arttırılmasıdır.

Bahsedilen gereksinimlerin özellikle e-devlet uygulamalarının, kablosuz iletişim ve internet teknolojilerinin gelişmesiyle paralel olarak ilerleyen zamanlarda daha büyük bir önem kazanacağı ise yadsınamaz bir gerçektir.

Bu tez kapsamında geliştirilen, tek kullanımlık parola-tek kullanımlık kimlik doğrulamalı anahtar değişimi ve haberleşme ortamında, yukarıda bahsedilen gereksinimleri karşılayan, kullanıcının sisteme kendisini dahil etmek için gizli bilgilerini sadece bir defa ilgili makama doğrulattığı, kullanımı basit ve etkili bir anahtar değişim sistemi oluşturulmuş, oluşturulan sistemin güvenliği, tek kullanımlık parola kullanma yeteneği ile desteklenmiştir. Sistemdeki İşlemlerin, tek kullanımlık parola ve kimlik değerleri gibi sayı değerleri üzerinden yürütülmesi ile haberleşen taraflar arasındaki ortak çalışılabilirlik ve haberleşmedeki verim arttırılmıştır.

Anahtar Kelimeler: Tek kullanımlık kimlik, tek kullanımlık parola, anahtar değişimi, ağ güvenliği, kullanıcı doğrulama.

Danışman: Yrd. Doç. Dr. Mehmet DEMİRER , Hacettepe Üniversitesi, Elektrik ve Elektronik Mühendisliği Anabilim Dalı

A SOFTWARE DEVELOPMENT OF ONE-TIME ID (DISPOSABLE) AUTHENTICATED KEY EXCHANGE APPLICATION

Ali Umut YÜKSEL

ABSTRACT

In Today's growing technology, the importance of legitimate user authentication and the need for secure communication on the insecure and public wide area networks is increasing rapidly, especially considering the expansion of internet based applications.

Although widely used Public Key Infrastructure and key exchange methods based on Diffie-Hellman algorithm are approved as computationally secure, they still have vulnerabilities such as denial of services attack and replay attack. Both denial of services attack, which is mostly used for interrupting the network application availability, and replay attack, on which valid user identities are used by the attacker, aims to damage source reliability. These approaches also cannot meet the necessity of the user privacy, user untraceability and identity leakage. Internet users have to keep their password and identification information on their mind or use an equipment like password generator for each electronic site where they expose the information like bank account or credit card number and do banking operations.

Another necessity on the fast expanding network area is to increase the web sites interoperability to improve the effectiveness and the security of the user operations.

It is undeniable truth that the importance of the above requirements will be increased especially by the progress of e-government applications and the improvement of the wireless communication technology .

As a result , a One-Time Password-One-Time ID Authenticated Key Exchange and Communication software environment is developed to meet the requirements that mentioned above. The software need to verify users information by synchronization authority just once. An easy to use, effective key exchange and communication system is built to make the software more user-friendly and effectual. The system security is supported by the ability of using one-time

password. By using only numerical quantities like One-Time ID and fixed ID in the processes of the system, interoperability and efficiency of the communication are increased. .

Keywords: One-Time ID, One-Time Password, key exchange, network security, user authentication.

Advisor: Asst. Prof. Dr. Mehmet DEMİRER , Hacettepe University, Department of Electirical and Electronics Engineering

TEŞEKKÜR

Bu çalışmanın gerçekleştirmesi esnasında her zaman iyi niyet, sabır ve hoşgörüsünü koruyan sayın danışmanım Yrd. Doç. Dr Mehmet DEMİRER'e,
Verdiği desteği her zaman minnetle anacağım T.C Maliye Bakanlığı Bilgi İşlem Daire Başkanı sayın Mahmut GENÇAĞA'ya,
Destek ve hoşgörüsü için sevgili eşime,
Son olarak öğrenim hayatımda geldiğim noktada en büyük pay sahibi sayın aileme teşekkürü bir borç bilirim.

Hacettepe Üniversitesi 2006 Ali Umut YÜKSEL

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

SİMGELER VE KISALTMALAR DİZİNİ

KDC:Key Distribution Center

AS:Authentication Server

EKE:Encrypted Key Exchange

SPEKE:Strong Password-Only Authenticated Key Exchange

A-EKE:Augmented Encrypted Key Exchange

DOS:Denial of Service

OID:One-Time ID

OTP:One Time Password

DH:Diffie-Hellman

NONCE:Number Once

GUID:Globally Unique Identifier

1 GİRİŞ

İnternet teknolojisinin insan hayatına her geçen gün daha da fazla girmesiyle beraber, insanların yaşamları boyunca gerçekleştirdikleri çeşitli işlemler elektronik ortama aktarılır ve internet kullanıcılarının hizmetine sunulur hale gelmiştir.

Özellikle bankacılık işlemleri ve e-devlet gibi uygulamaların internet üzerinden erişilebilir hale gelmesiyle, kullanıcı tarafında gizlilik ve güvenlik, sunucu tarafında ise güvenlik, yüksek erişilebilirlilik, birlikte çalışılabilirlik ve hizmet verme performansı gibi kavramların önemi giderek artış göstermektedir. Bu tez kapsamında gerçekleştirilecek olan Tek Kullanımlık Kimlik Doğrulamalı Anahtar Değişimi yazılımı ile yukarıda bahsedilen gereksinimlere cevap veren bir haberleşme ortamı ortaya konulacaktır.

Elektronik olarak Tek Kullanımlık Kimlik; güvenilir kullanıcıların güvenilir sunucularla haberleşmeleri esnasında üretip sunucu ile senkronize ettikleri ve kendilerini sunucuya tanıtarak güvenilir kılmak ve haberleşme ortamını şifrelemek için bir defaya mahsus kullandıkları kimlik olarak tanımlanabilir. Tek Kullanımlık kimlik oturumunun kullanılması ile beraber hat dinleniyor dahi olsa saldırgan kimin haberleşmekte olduğunu saptayamamaktadır.

Tek Kullanımlık kimlik anlık olarak sunucu ve istemci arasında paylaşılan gizli bir anahtardır ve saldırgan zor da olsa hattı dinleyerek geçerli bir tek kullanımlık kimlik elde etse de bunu daha sonra kullanamamaktadır. Tek Kullanımlık kimlik kullanılmasıyla beraber sunucunun cevap vereceği geçerli istek sayısı sınırlanmakta, kaynak güvenilirliğini zedeleme ve tekrarlama atakları engellenmektedir.

Tez çalışmasının 2. bölümünde temel kriptografi kavramları, 3. bölümünde matematiksel açıklamalar, 4. bölümünde doğrulama protokolleri, çeşitli anahtar değişim yöntemleri ve DH Temelli Tek Kullanımlık Kimlik Doğrulamalı Anahtar Değişimi, 5. bölümünde tez yazılımın geliştirilmesinde kullanılan ortam ve teknolojiler 6. bölümünde tez kapsamında geliştirilen uygulama yazılımı açıklanmaktadır. 7. bölümünde ise geliştirilen yazılım ile elde edilen sonuçlar sunulmaktadır.

2 KRİPTOGRAFİK TEMEL

Bu bölümde, kriptoloji bilimi, ağ haberleşmesinde güvenlik, muhtemel tehdit ve saldırılar, şifreleme sistemleri, anahtar yönetimi ve değişimi, tek yönlü fonksiyonlar, sayısal imza ve elektronik sertifika konuları hakkında temel bilgiler yer almaktadır.

2.1 Temel Bilgiler

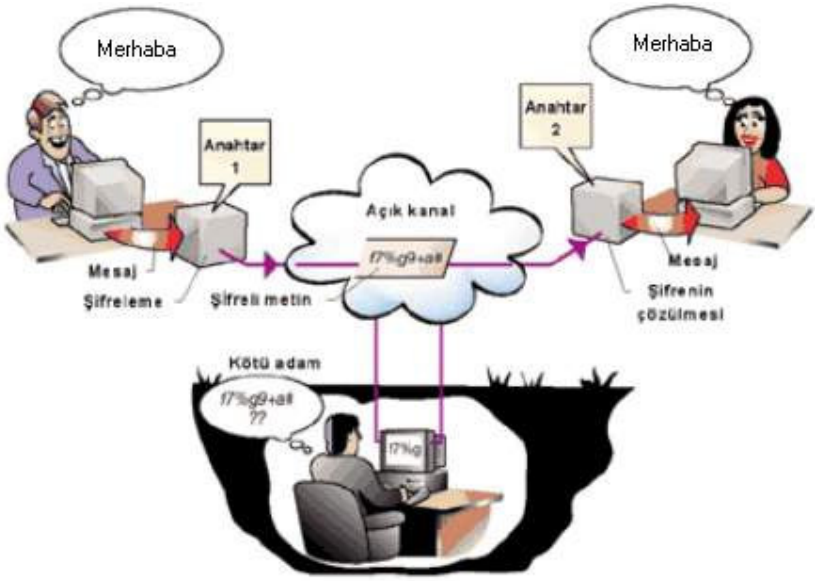
Kriptoloji kelimesi, köken olarak eski Yunanca'da yer alan "kryptos logos" kelimelerinden gelmektedir. Kryptos kelimesi gizli anlamını, logos ise dünya anlamını taşımaktadır. Kelimenin birçok dünya dilindeki karşılığı da bu orijinal halini korumaktadır.

Kriptoloji, kavram olarak şöyle tanımlanabilir:
"Haberleşen iki veya daha fazla tarafın bilgi alışverişini emniyetli olarak yapmasını sağlayan, temeli matematiksel zor problemlere dayanan tekniklerin ve uygulamaların bütünüdür [1]."
Günümüzde kriptoloji, matematik, elektronik, optik, bilgisayar bilimleri gibi birçok disiplini kullanan özelleşmiş bir bilim dalı olarak kabul edilmektedir. Kriptografi, belgelerin şifrelenmesi ve şifresinin çözülmesi için kullanılan yöntemlere verilen addır.

Kriptoanaliz, kriptografik sistemlerin kurduğu mekanizmaları inceler ve çözmeye çalışır. Kriptoanalizin kriptoloji içindeki önemi çok büyüktür. Ortaya konan bir şifreleme sistemini inceleyerek, zayıf ve kuvvetli yönlerini ortaya koymak için kriptoanaliz kullanılır. Günümüzde elektronik bilgi sistemlerinin yaygınlaşması kriptolojinin önemini çok fazla arttırmıştır. Kriptolojinin başlıca kullanım alanı hareket halindeki veya depolanmış bilginin şifrelenmesi ve istendiğinde bu şifrenin çözülmesidir. Kriptolojinin temel malzemesi bilgi olduğu için neredeyse sınırsız sayıda uygulamada kullanılması söz konusu olmuştur.

Şekil 2.1 de tipik bir bilgi iletim ortamı gözlenmektedir. Kriptografik iletim esnasında, kullanıcı bir mesajı (m) göndermeden önce bir anahtar (k1) kullanarak

şifreler. Şifreli metin (c) yasadışı dinleyicilere açık olan bir kanaldan gönderilir. Mesajı okumak için alıcı bir anahtar (k2) kullanarak şifreyi çözer ve m mesajını elde eder. Aktif düşmanlar araya girip iletişimi dinleyebilir. Eğer k1 ve k2 eşitse, sistem simetriktir. Aksi takdirde bu sistem asimetrik olarak nitelenir. Güvenliğin garantilenmesi için k2 her zaman gizli olmalıdır, ancak k1'i kullanarak k2'yi elde etmek mümkün olmadığı sürece k1 açıklanabilir. Bu durumda sisteme açık anahtarlı sistem (public key system) adı verilir.



Şekil 2-1Bilgi İletim Ortamı

2.2 Güvenlik ve Tehditler

Bilgi güvenliği temel olarak üç yönden ele alınabilir [2].

1- Güvenlik Saldırıları: Bilgi güvenliğini bozmaya yönelik yapılan her türlü eylem bir güvenlik saldırısı olarak kabul edilebilir.

2- Güvenlik Mekanizması: Bilgiye karşı yapılan saldırıları algılayan, bu saldırılara karşı koruyan veya bilgiyi kurtarmaya yönelik olarak geliştirilmiş olan bir mekanizmadır.

3- Güvenlik Servisi: Bilgi işlem sistemlerinin ve iletilen bilginin güvenliğini sağlamlaştıran servistir. Güvenlik saldırılarına karşı tasarlanan bu servisler bir veya birden fazla güvenlik mekanizması tarafından sağlanırlar.

2.2.1 Servisler

Bilgi güvenliğinde gerçekleştirilmek istenen hedefler, temel olarak dört ana servisin üzerinde yükselmektedir [2].

1- Gizlilik: Bilginin içeriğini yetkili olan kısımlar haricindekiler için gizli tutmaktır. Gizliliği sağlamak için fiziksel korumadan matematiksel algoritmalara kadar birçok yöntemin varlığından bahsedilebilir.

2- Veri Bütünlüğü: Verinin yetkisiz kısımlar tarafından değiştirilememesidir. Veri bütünlüğünün sağlandığından emin olabilmek için yetkisiz kısımların veri üzerinde yaptığı işlemler belirlenmelidir. Bu işlemler ekleme, silme, çıkarma, tekrarlama yada geciktirme gibi eylemler şeklinde olabilir.

3- Kimlik Doğrulama: Bu servis tanımlama ile ilişkilidir, verinin bütün içeriği ve kendisine uygulanır. Veri iletimine katılan bütün taraflar birbirlerini tanımlayabilmelidirler. Kanal üzerinden iletilen veri; kaynağı, kaynak tarihi, veri içeriği, gönderilme zamanı gibi özellikleri için doğrulanabilmelidir. Bu nedenle kriptografi görüşü, içerik kimlik doğrulaması ve kaynak veri doğrulaması olmak üzere iki alt sınıfta ele alınmaktadır. Veri kaynağı kimlik doğrulaması aynı zamanda veri bütünlüğü servisini de gerçeklemektedir, çünkü veri üzerinde yetkisiz kısım tarafından yapılan bir değişiklik, veri kaynağının da değişmesini sağlayacaktır.

4- İnkar Edememezlik: Veriyi gönderen ve işleyen kişinin yaptıkları işlemleri sonradan inkar edememesidir.

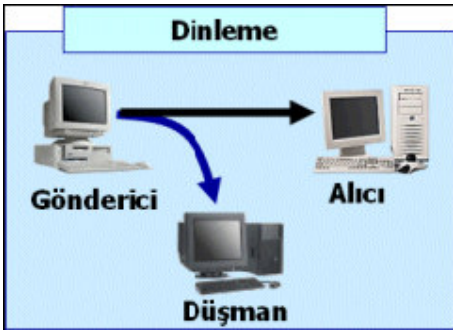
2.2.2 Mekanizmalar

Yukarıda bahsedilen servisleri gerçekleştirmek için bir çok mekanizmanın varlığından bahsedilebilir, örneğin kriptografik teknikler bir güvenlik mekanizmasıdır.

2.2.3 Tehditler

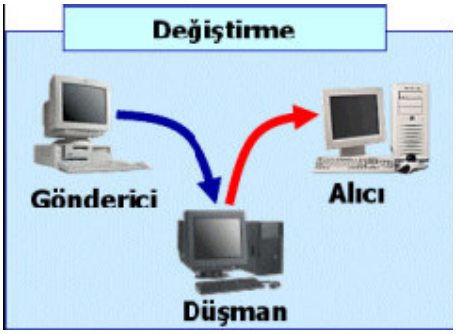
Güvensiz olan ağ üzerinde haberleşerek mesaj alışverişinde bulunan taraflar, haberleşme oturumuna karşı gerçekleştirilebilecek çeşitli tehditlerle karşı karşıya kalabilirler. Bu tehditler temel olarak aşağıdaki gibi sıralanabilir [1].

1- **Gizlilik İhlali:** Dinleme tehdidi olarak bilinen bu eylemde, haberleşme kanalını dinleyen saldırgan, gönderici ile alıcı arasındaki mesaj trafiğini dinleyebilir ve elde ettiği mesajları okuyarak bu haberleşmenin gizliliğini bozabilir.



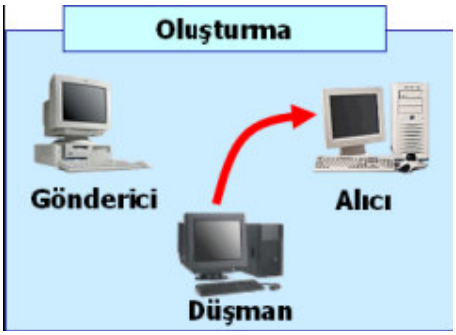
Şekil 2-2 Dinleme

2- **Bütünlük İhlali:** Değiştirme tehdidi olarak bilinen bu eylemde, haberleşmeye müdahale eden saldırgan, göndericinin mesajlarını değiştirerek, alıcıya giden mesajı istediği şekle sokabilir.



Şekil 2-3 Değiştirme

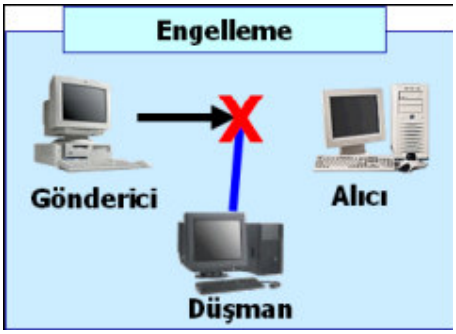
3- **Kimlik Doğrulama İhlali:** Oluşturma tehdidi olarak bilinen bu eylemde saldırgan, alıcıya göndericinin kimliğini taklit ederek mesajlar gönderebilir. Bu durumda eğer alıcı güvenilir bir kimlik doğrulaması yapmıyorsa yanlış mesajlarla saldırganın istediği noktalara yönlendirilebilir.



Şekil 2-4 Oluşturma

4- **İnkar Edememezlik İhlali:** Mesajı gönderen veya alan tarafın sonradan gerçekleştirdikleri eylemi inkar etmeleri söz konusu olabilir.

5- **Süreklilik İhlali:** Saldırgan, haberleşen iki taraf arasındaki hattı veya haberleşme araçlarını kullanılmaz hale getirerek haberleşmenin sürekliliğini engellemeye çalışır.



Şekil 2-5 Engelleme

Yukarıda maddeler halinde verilen veri güvenliğine karşı yapılan saldırıları iki grupta toplamak mümkündür, Pasif Saldırılar ve Aktif Saldırılar.

Pasif Saldırılar genellikle veri içeriğini ifşa etme ve trafiği dinleme eylemleri şeklinde gerçekleşir [2] ve veri üzerinde herhangi bir değişiklik gerçekleşmediği için algılanması zor saldırılardır.

Günümüz geniş alan ağı iletişim ortamında saldırganlar tarafından daha çok tercih edilen saldırılar Aktif Saldırılardır. Bu saldırı çeşitlerinden en önemlileri aşağıda açıklanmaktadır.

1- Tekrarlama (Replay) Saldırıları:

Geçerli veri iletişiminin kötü niyetli ve hileli bir şekilde tekrar edilmesi yada geciktirilmesi şeklinde gerçekleşen saldırılardır. Bu saldırı senaryosunda A tarafı ile B tarafı arasındaki haberleşme de A tarafı kimliğini B tarafına ispat etmek istemektedir, bunun için B tarafı A tarafından şifresini göndermesini ister, C tarafı ise saldırgandır ve iletim ortamını dinleyerek şifreyi elde eder, A ile B arasındaki iletişimin ardından C tarafı B tarafına elde ettiği şifreyi göndererek kendini A tarafı gibi gösterir ve B tarafı ile iletişime geçer. Bu saldırı tipini engellemek için oturum jetonları ve zaman damgalarının kullanılması gibi yöntemler geliştirilmiştir.

2- Hizmeti Yalanlama (Denial of Service) Saldırıları:

Kullanıcılara verilen hizmeti engellemek için bilgisayar sistemlerine yada ağ'a karşı yapılmaktadır. Yapılan saldırı, hizmet veren sistemin ağ bant genişliğini düşürme veya kaynaklarını meşgul etme (Disk alanı, Hafıza alanı, işlemci zamanı bakımından) şeklinde kendini göstermektedir.

3- Ortadaki Kulak (Man-in-the-middle attack)

Ağ ortamında iki taraf arasında gerçekleşen mesaj alışverişi esnasında tarafların haberi olmaksızın mesaj içeriğinin saldırgan tarafından dinlenmesi, değiştirilerek taraflara gönderilmesi yada mesaj iletiminin kesintiye uğratılması gibi eylemleri kapsayan saldırı türüdür. Bu tip saldırılar özellikle doğrulamasız Diffie-Hellman Anahtar Değişimi gibi protokollerde etkili olabilmektedir. Anahtar değişim protokolünü bu tip saldırılara karşı güvenilir yapmak için taraflar arasında ek bilgilerin taşınması gerekli olup bu konu tez kapsamında ayrıntılı olarak ele alınmaktadır.

4- Sözlük Saldırıları (Dictionary Attack)

Sözlük saldırıları, saldırganın şifreli metne veya doğrulama mekanizmasına karşı şifreleme anahtarını veya iletişim yapan kişilere ait parolaları elde etmek için bir çok olasılığı deneyerek gerçekleştirdiği saldırılardır. Bütün olasılıkların denendiği kaba kuvvet saldırılarından (brute force attack) farkı, saldırı esnasında sadece dile ait sözlüğün içerdiği kelime olasılıklarının denenmesidir. Bir çok ağ kullanıcısının kendi dillerine özgü, kolay hatırda kalır parolalar seçmesi nedeniyle bu saldırı çeşidi büyük oranda başarıya ulaşabilmektedir. Sözlük saldırıları şifreleme anahtarını veya şifreli metnin belli bir bölümünü ele geçirme ve bilgisayar parolalarını elde ederek bilgisayar güvenliğini tehdit etme olmak üzere iki ana şekilde uygulanmaktadır.

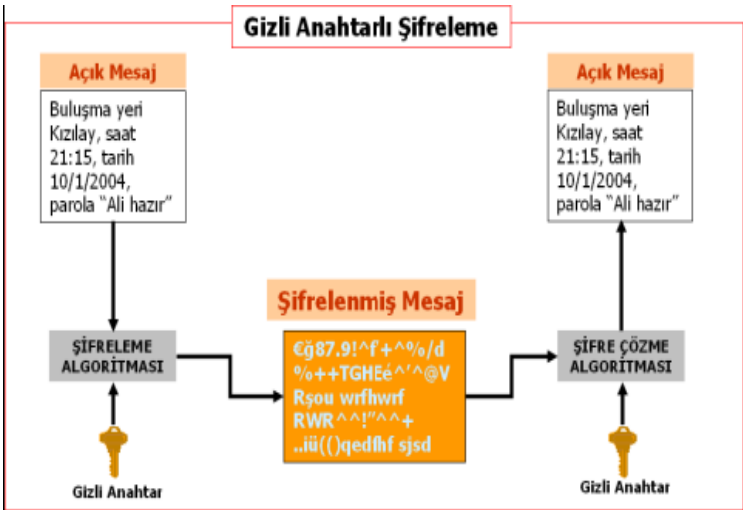
2.3 Simetrik Kripto Sistemleri

Simetrik kriptografide, şifreleme ve şifre açma işlemi taraflar arasında gizli olarak saklanan bir anahtar ile gerçekleştirilmektedir. Bu nedenle, bu tip sistemlere gizli anahtarlı kriptografi sistemi adı da verilmektedir.

Bu sistemde haberleşen taraflar:

- Aynı şifreleme algoritmasını,
- Birbirine uyumlu gerçeklemeleri,
- Aynı anahtarı

Kullanırlar.



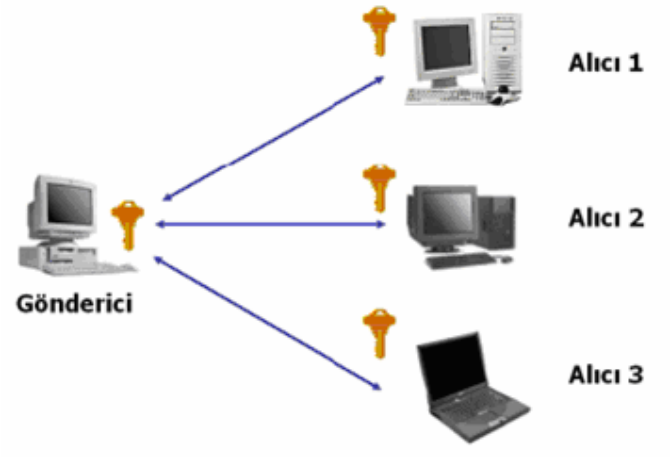
Şekil 2-6 Gizli Anahtar Şifreleme Ortamı

Simetrik kriptografinin en önemli öğesi anahtar gizliliği olduğu için birden fazla kişinin haberleştiği bir ortamda anahtar yönetimi büyük dikkat gerektirmektedir.

Simetrik kriptografi de temel olarak iki çeşit anahtar yönetiminden bahsedilebilir.

1- Birden-Çoğa (One-to-Many) Anahtar Yönetimi

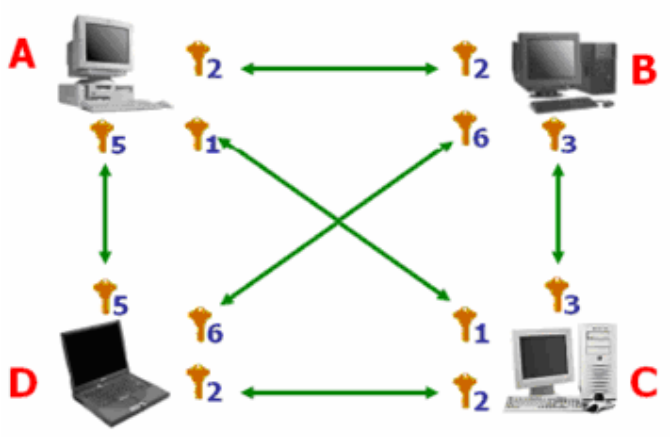
Bu yöntemde haberleşen tüm taraflar aynı gizli anahtarı kullanırlar. Bu nedenle herkes birbirinin şifreli mesajlarını açıp okuyabilir.



Şekil 2-7 Birden-Çoğa Anahtar Yönetimi

2- Çoktan-Çoğa (Many-to-Many) Anahtar Yönetimi

Bu yöntemde her bir taraf kendi arasında gizli bir anahtar kullanarak haberleşmektedir.



Şekil 2-8 Çoktan-Çoğa Anahtar Yönetimi

Bu yöntem sistemdeki kişi sayısına bağlı olarak çok fazla anahtar üretimini gerektirdiği için çok kullanışlı değildir. Anahtar sayısının kullanıcı sayısına bağlı olarak artışı Tablo 1 de görülmektedir [1].

Kullanıcı Sayısı	Anahtar Sayısı
3	3
4	6
10	45
100	4,950
1,000	499,500
10,000	49,995,000
n	n*(n-1) /2

Tablo 1 Anahtar Sayısının Kullanıcı Sayısına Bağlı Olarak Artışı

Bu tez kapsamında geliştirilecek olan platformda kullanılacak olan anahtar yönetimi birden-çoğa anahtar yönetiminin gelişmiş bir türevi olarak kabul edilebilir.

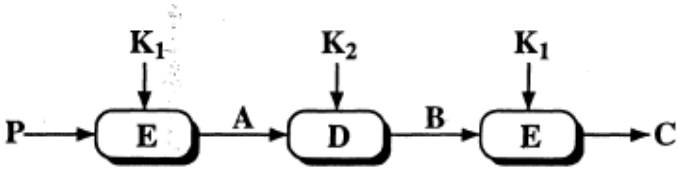
Birden-Güvenilir Guruba Anahtar Yönetimi olarak adlandırılabilecek bu yöntemde istemci güvenilir guruba üye bir sunucu ile haberleşmek istediğinde bir kereliğe mahsus olmak üzere kendini doğrulayarak değişken anahtarını Senkronizasyon Makamı ile senkronize etmektedir. Gruba dahil olan sunucularla istemci arasındaki haberleşme esnasındaki anahtar değişimi ve doğrulama işlemleri Senkronizasyon Makamı üzerinden gerçekleştirilmektedir.

2.3.1 DES Şifreleme Algoritması:

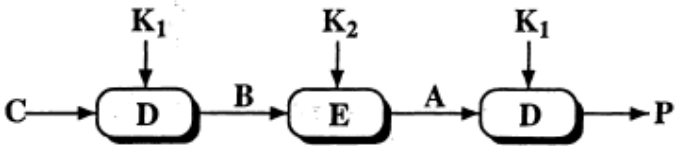
Bankacılık ve finans sektöründe ağırlıklı olarak kullanılan bu algoritma IBM firması tarafından 1974 yılında bulunmuş ve 1977 yılında Amerikan Standardı olarak kabul edilmiştir [3]. Üzerinde en çok çalışılmış olan algoritmadır. Günümüzde bu algoritma 3'lü DES (triple DES) şeklinde, üç farklı anahtarla aynı bloğa 3 defa DES uygulanarak da kullanılmaktadır. Algoritmanın kullanılması için herhangi bir lisans ödenmesi gerekmemektedir. Algoritmanın temel özellikleri şöyledir:

- 64-bit blok şifreleme, 56-bit anahtar uzunluğu, 16 tur
- Anahtar 16 değişik anahtar parçasına dönüştürülür.
- Karıştırma için S-kutuları kullanılır.
- Dağıtma için permütasyonlar kullanılır.
- Donanımsal olarak 1 Gbit/sn hızda çalışabilir (250 MHz hızındaki donanımda)

Des algoritmasının S-Box'lar gibi zayıf yönlerini kapatabilmek için 3-DES algoritması geliştirilmiştir. Bu algoritmada anahtar uzunluğu 192 bit'e kadar çıkmaktadır. Aşağıdaki şekilde görüldüğü gibi 3-DES algoritmasında ilk adımda metin K1 anahtarı kullanılarak şifrelenmekte, ikinci adımda K2 anahtarı kullanılarak deşifre edilmekte, üçüncü adımda ise K1 anahtarı ile tekrar şifrelenerek şifreli metin oluşturulmaktadır.



Şekil 2-9 3-DES Şifreleme



Şekil 2-10 3-DES Şifre Çözme

Tez kapsamında geliştirilecek olan yazılımda da şifreleme için 3-DES algoritması kullanılmıştır.

2.4 Asimetrik (Açık Anahtar Kripto Sistemleri)

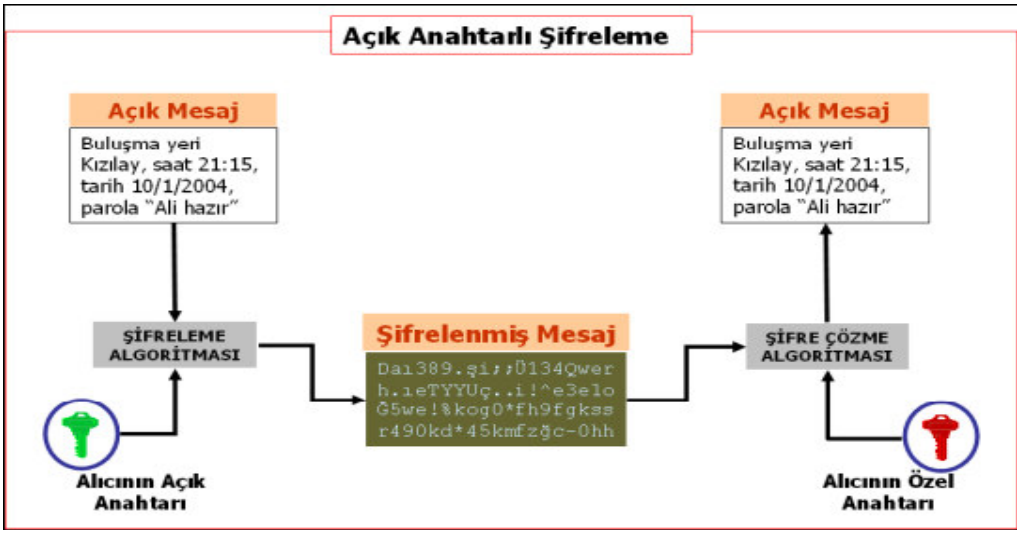
Asimetrik kriptografide, şifreleme ve şifre çözme işlemi farklı anahtarlar ile yapılır. Bu anahtar çiftini oluşturan anahtarlara açık ve özel anahtar adı verilir. Bu kriptografi yönteminde özel anahtar gizli tutulmalıdır fakat açık anahtar gerekli kişilere verilebilir ve başka kişilerle paylaşılabilir. Bu özelliğinden dolayı asimetrik

kriptografi, açık anahtarlı şifreleme adıyla da anılır. Bu sistemi kullanarak haberleşen taraflar aynı şifreleme algoritmasını kullanırlar, birbiriyle uyumlu gerçeklemeler kullanır ve gerekli anahtarlara erişebilirler.

Asimetrik kriptografi için anahtar yönetimi çok önemlidir. Anahtar yönetimi için dikkat edilmesi gereken noktalar şöyle sıralanabilir:

- Açık anahtarlar kontrollü olarak güvenilir bir merkez tarafından dağıtılmalı ve değiştirilmeleri önlenmelidir.
- Anahtar çiftleri güvenilir bir merkez tarafından üretilebileceği gibi kullanıcı bireysel olarak kendi anahtar çiftini üretebilir.
- Şifreleme ve imzalama işlemleri için ayrı anahtar çiftleri olmalıdır. Çok özel durumlarda imzalama ve şifreleme anahtar çiftleri aynı olabilir.
- Anahtar iptalleri kontrollü bir şekilde gerçekleştirilmeli ve yayınlanan silme listeleri ile dışarıya iletilmelidir.

Asimetrik kriptografi için anahtar yönetimi simetrik kriptografiye göre daha kolaydır çünkü bir kullanıcıyla şifreli haberleşmek isteyen kişi karşı tarafın açık anahtarına ihtiyaç duyar. Bu açık anahtar kamuya açık olarak yayınlandığı için sisteme giren bir kişi için sadece bir anahtar çifti üretmek yeterli olmaktadır [1].



Şekil 2-11 Açık Anahtar Şifreleme Sistemi

Asimetrik kriptografinin kuvvetli tarafları aşağıdaki gibi özetlenebilir:

- Anahtar yönetimi ölçeklenebilirdir.
- Kırılmaları oldukça zordur.
- Asimetrik kriptografi algoritmaları ile bütünlük, kimlik doğrulama ve inkâr edememezlik güvenlik hizmetleri gereksinimleri gerçekleştirilebilmektedir..

Asimetrik kriptografinin zayıf yönleri ise aşağıdaki gibidir:

- Simetrik kriptografi algoritmalarına oranla 1500 kat daha yavaştırlar.
- Anahtar uzunluğu mobil cihaz gibi ortamlarda kullanmak için elverişli değildir.

2.5 Simetrik ve Asimetrik Şifreleme Algoritmalarının Karşılaştırılması

Her ne kadar işlemlerin yapıldığı platforma bağlı olsa da, tek anahtarlı simetrik şifreleme algoritmaları açık anahtar şifreleme algoritmalarına göre çok daha hızlıdırlar.

Ancak, kırılmalarındaki zorluk ve anahtar dağıtım işlemlerindeki kolaylık açısından açık anahtar şifreleme sistemleri günümüzde daha geniş uygulama alanı bulmaktadırlar. Hız problemini azaltmak içinse, mesaj imzalanırken kendisi değil tek blok halindeki hash değeri imzalanmakta, şifrelenirken ise mesajın tümü tek yönlü simetrik şifreleme algoritması kullanılarak şifrelenmekte ve bu şifreleme de kullanılan anahtar, açık anahtar tabanlı bir algoritma ile şifrelenmektedir.

Açık anahtar kripto sistemi kullanılarak, anahtar dağıtım işlemi kolaylaştırılmasına rağmen anahtar dağıtımı esnasında karşılaşılan sorunlar tam olarak ortadan kaldırılamamaktadır. Bunlardan en önemlisi, açık anahtarlar kullanıcılara dağıtılırken hangi açık anahtarın hangi kullanıcıya dağıtılacağını belirleyen sertifikaların kullanılması zorunluluğudur [4].

Sertifikalar bir açık anahtar ile sahibi arasında ilişkinin belgesidir. Güvenilir sertifikasyon makamları tarafından üretilen bu sertifikaların içerdiği bilgilerin doğru olduğu kabul edilir.

2.6 Tek Yönlü Fonksiyonlar ve Ayrık Logaritma Problemi

Tez kapsamında ele alınmakta olan Hash fonksiyonları ve Diffie-Hellman anahtar değişimi sistemi temel olarak kriptoloji biliminde tek yönlü fonksiyon olarak adlandırılan çözümü zor bir probleme dayanmaktadır.

Tek yönlü fonksiyonda, tek yönde çözümü hesaplamak kolay olmasına karşılık aksi yönde çözümü hesaplamak oldukça zordur. Örnek vermek gerekirse, normal yönde çözümü hesaplamak saniyeler mertebesinde sürmekte, aksi yönde çözümü hesaplamak mümkün olsa bile süresi aylar yada yıllar sürebilmektedir.
Tuzak kapısı tekyönlü fonksiyon ise verilecek ek bir bilgi sayesinde tersine çözümün kolay olduğu tek yönlü fonksiyondur [5]. Örneğin, açık anahtar kripto sistemlerinde açık olan anahtar, fonksiyon örneği hakkında belirli bir bilgi verirken özel anahtar ise tuzak kapı bilgisini vermektedir.

Ayrık logaritma problemi ise Diffie-Hellman anahtar değişiminin temelini oluşturmaktadır. Problem tek yönlü bir fonksiyondur ve bir sonraki bölümde anlatılacak olan, *grup G*'de logaritma bulma zorluluğu temeline dayanmaktadır.

Matematiksel olarak ayrık logaritma problemi, *G* sonlu grubundan seçilen *g* ve *h* elemanları için $g^x = h$ eşitliğini sağlayabilecek bir x sayısının bulunduğunu destekler.

Ayrık logaritma problemi hesabı için birçok algoritma bulunmaktadır. Tez yazılımında kullanılan kütüphanede ise verimli bir genel maksat algoritma olan, tekrarlı karesini alma algoritması (repeated square-and-multiply) kullanılmaktadır.

2.7 KIYIM (HASH) Fonksiyonları

Hash fonksiyonları, ağ haberleşmesinde alıcının aldığı mesajın gerçekten güvenli bir kaynaktan geldiğinin ve herhangi bir değişikliğe uğramadığının kontrolü için modern kriptoloji de sıkça kullanılan fonksiyonlardır.

Bu fonksiyonlar $h = H(M)$ şeklinde ifade edilebilir. Burada *M* değişken uzunluktaki mesaj, *h* ise bu mesajdan oluşturulan sabit uzunluktaki hash değeridir.

Hash fonksiyonlarının güvenli ve verimli bir şekilde kullanılabilmeleri için birtakım gereksinimleri yerine getirmeleri gerekmektedir. Bu gereksinimler aşağıdaki gibi maddelendirilebilir [2].

- *H* fonksiyonu uzunluğuna bağımlı olmaksızın bütün mesajlara uygulanabilmelidir.
- *H* fonksiyonu her türlü girdi için sabit bir çıkış üretmelidir.
- Herhangi bir *(x)* girdisi için *H(x)* kolayca hesaplanabilir olmalı, yazılımsal ve donanımsal olarak uygulanabilir olmalıdır.
- *h*'nin bilinen değerleri için $H(x) = h$ olacak şekilde bir *x* değerinin hesaplanabilmesi zor olmalıdır. Bu, fonksiyonun tek yön özelliği olarak da bilinmektedir. Bu özellik olmazsa gizli değer kullanılarak gerçekleştirilen bir iletişim ortamında, hattı dinleyerek mesaj ve şifrelenmiş mesajdan elde edilen hash değeri bilgisine sahip olan saldırgan, fonksiyonu tersi yönüne işleterek gizli değer bilgisini elde edebilir.
- *x*'in bilinen değerleri için *x*'e eşit olmayan ancak $H(y) = H(X)$ eşitliğini sağlayan bir *y* değerinin hesaplanması güç olmalıdır. Bu durum zayıf çarpışma direnci olarak adlandırılmaktadır. Bir mesajın, alternatifi olan bir mesaj için birbirine eşit hash değerlerinin hesaplanamayacağını garanti eder.
- $H(x) = H(y)$ olacak şekilde bir *(x,y)* çiftinin hesaplanması güç olmalıdır. Bu durum güçlü çarpışma direnci olarak da bilinmektedir.

Hangi tür hash fonksiyonu olursa olsun, bütün hash fonksiyonları aynı genel prensipleri kullanmaktadır.
Bit blokları olarak gruplanan mesaj ve dosya gibi girdiler için her defasında bir bloğa yinelemeli (iteratif) bir işlem uygulanarak *n* bitlik bir hash kod çıktısı üretilir.

En basit hash fonksiyonu, her bir bit bloğuna, bit bit *XOR* işleminin uygulandığı fonksiyondur.

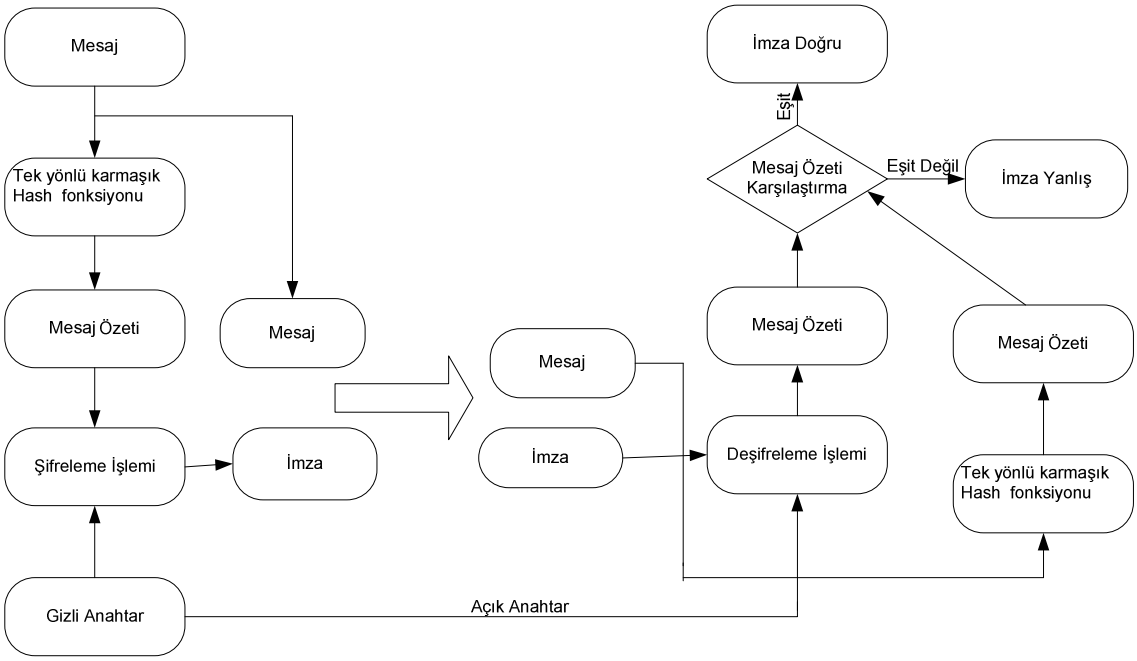
2.8 Sayısal İmza

Sayısal imza açık anahtar kripto sistemi uygulamalarından biridir. Bir verinin imzası veriye ve imzalayan kişinin özel anahtarına bağlı olarak değişiklik gösterir. Sayısal imza, elektronik ortamda tam anlamıyla ıslak imzanın yerine kullanılacağından aşağıdaki gereksinimleri karşılamak zorundadır.

- Gerçeklik: Sayısal imza ait olduğu nesnenin orijinal olduğunu ispatlar.
- Taklit Edilememe: İmzanın kendisi, verinin imzanın gerçek sahibi tarafından işaretlendiğinin ispatıdır.
- Yeniden Kullanamama: Veri üzerindeki sayısal imza başka bir nesne üzerine taşınarak yeniden kullanılamaz.
- Değiştirilememe: Veri bir defa imzalandıktan sonra imzası çözülmeden bir daha değiştirilemez.
- İnkar Edilememe: Veriyi imzalayan kişi sonradan imzayı inkar edemez.

Sayısal imzada imzayı atan taraf veriyi kendi gizli anahtarı ile şifreler. Şifrelenen veri düz verinin sonuna eklenerek karşı tarafa gönderilir. Veriyi alan taraf imzayı atan kişinin açık anahtarını kullanarak şifreyi çözer. Şifresi çözümlenen veri gerçek veriyle aynı ise verinin imzası doğrulanmış demektir.

Yukarıda bahsedilen kullanımda karşı tarafa gönderilen mesaj büyüklüğü imza ile beraber iki katına çıkmaktadır. Performans ve veri yolunu efektif kullanmak açısından dezavantaj getiren bu durumu önlemek için Şekil 2-12 de görüldüğü gibi imzalama işleminde verinin tamamen kendisi değil, tek yönlü hash fonksiyonu ile oluşturulan özeti imzalanır.



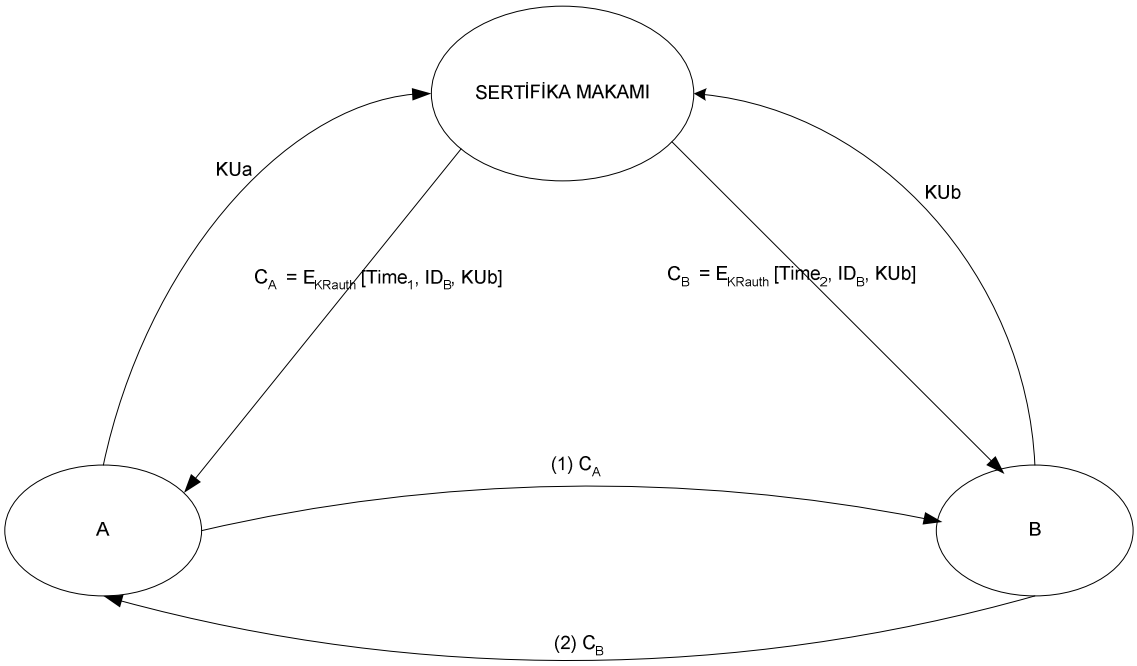
Şekil 2-12 Mesajın İmzalanması ve Doğrulanması

2.9 Elektronik Sertifikalar

Açık anahtar şifreleme yöntemlerinde sıklıkla kullanılan elektronik sertifikalar, sertifika otoritesi tarafından üretilen, açık anahtar ve diğer bazı bilgileri taşıyan, sertifika isteminde bulunan tarafa açık anahtar bilgisine karşılık gelen gizli anahtar bilgisi ile gönderilen elektronik bileşenlerdir. Haberleşen her bir taraf, kendi anahtar bilgisini, karşı tarafa sertifika bilgisini ileterek aktarır. Elektronik sertifika kullanımı ile taraflar arasında güvenli anahtar iletimi için aşağıda sıralanan gereksinimler karşılanabilmektedir:

- Haberleşen herhangi bir taraf isim ve açık anahtar bilgisini sertifikayı okuyarak elde edebilir.
- Taraflar sertifikanın gerçek bir sertifika olduğunu ve gerçekten sertifika otoritesi tarafından üretildiğini doğrulayabilir.
- Sadece sertifika otoritesi sertifika üretebilir ve güncelleyebilir.
- Taraflar sertifikanın geçerliliğini doğrulayabilir.

Taraflar ve sertifika makamı arasındaki iletişim şekil 2.13 de görüldüğü gerçekleşmektedir.



Şekil 2-13 Elektronik Sertifika Dağıtımı

Her bir taraf, sertifika makamına kendi açık anahtarı ile başvurarak sertifika isteminde bulunur. Taraflar ve sertifika makamı arasındaki haberleşme birebir olabileceği gibi elektronik posta gibi güvenilir doğrulamalı bir iletişim ile de olabilir.
A tarafının açık anahtarı ile sertifika makamına başvurduğu düşünüldüğünde, sertifika makamı *A* için aşağıdaki gibi bir sertifika bilgisi üretir [2].

$$C_A = E_{KR_{auth}}[T, ID_A, KU_a]$$

Burada KR_{auth} makamın gizli anahtarıdır. *A* tarafı bu sertifika bilgisini diğer taraflara aşağıdaki şekilde gönderebilir.

$$D_{KU_{auth}}[C_A] = D_{KU_{auth}}[E_{KR_{auth}}[T, ID_A, KU_a]] = (T, ID_A, KU_a)$$

Yukarıdaki mesajı sadece sertifika makamının açık anahtarı deşifre edebileceğinden güvenilir alıcılar sertifika makamının açık anahtarını kullanarak bu işlemi gerçekleştirebilirler. Deşifre edilen mesajda yer alan ID_A göndericinin kimlik

bilgisini, KU_a ise açık anahtar bilgisini içermektedir. *T* zaman damgası bilgisi ise sertifikanın geçerli olup olmadığının kontrolü için kullanılır.

2.10 Diffie-Hellman Anahtar Değişimi

1976 yılında geliştirilen ve halen bir çok uygulamada, geliştirilmiş çeşitleri kullanılan Diffie-Hellman anahtar değişimi [6] açık kanal üzerinde anahtar dağıtılması problemine ilk pratik çözümdür.

Sistem iki tarafın açık kanal üzerinden mesajlarını birbirlerine göndererek ortak bir anahtar üzerinde anlaşmaları temeline dayanmaktadır. *p*'nin ayrık logaritma problemi çözülemeyecek kadar büyük bir asal sayı *g*'nin de *p*' nin primitif bir kökü olduğu ve *p* ve *g*'nin herkes tarafından bilindiği kabul edildiğinde Diffie-Hellman anahtar değişimi şu şekilde gerçekleşmektedir;

- *A* tarafı $0 \le a \le p-2$ eşitsizliğini sağlayan rastgele bir *a* sayısı seçer ve $c = g^a \ mod\ p$ yi hesaplayarak *B* tarafına gönderir.
- *B* tarafı $0 \le b \le p-2$ eşitsizliğini sağlayan rastgele bir *b* sayısı seçer ve $d = g^b \ mod\ p$ yi hesaplayarak A tarafına gönderir.
- *A* tarafı $k = d^a = (g^b)^a \ mod\ p$ yi hesaplar.
- *B* tarafı $k = c^b = (g^a)^b \ mod\ p$ yi hesaplar.

Böylece *A* ve *B* tarafları aralarında *k* ortak anahtarı için anlaşmış olurlar. Hattı dinleyen bir saldırgan $p, g, g^a ve g^b$ hakkında bilgi sahibi olabilir ancak *k*'yı elde edebilmek için ayrık logaritma problemini çözmek zorundadır.

Diffie-Hellman Anahtar Değişimi Sayısal Örnek:
Anahtar değişimini gerçekleştirecek iki tarafın paylaştıkları asal sayı değerinin *p=97* ve bunun primitif kökünün *g=5* olduğu kabul edilsin;

İki tarafta $X_A = 36$ ve $X_B = 58$ olmak üzere kendi gizli değerlerini seçerler ve açık anahtarlarını hesaplayarak birbirlerine gönderirler.

Açık Anahtarlar:

$Y_A = 5^{36} = 50 \bmod 97$

$Y_B = 5^{58} = 44 \bmod 97$

Açık anahtar değişiminin gerçekleştirilmesinin ardından iki tarafta ortak gizli anahtar değerini hesaplayarak anahtar değişimini sonlandırırlar.

Ortak Gizli Anahtar:

$K = (Y_B)^X_A \bmod 97 = 44^{36} = 75 \bmod 97$

$K = (Y_A)^X_B \bmod 97 = 50^{58} = 75 \bmod 97$

2.11 RSA Şifreleme Algoritması

Diffie–Hellman anahtar değişimi algoritmasının geliştirilmesi ile simetrik şifreleme algoritmaları için büyük problem olan gizli anahtarın korunması ve dağıtımı sorunu büyük ölçüde ortadan kaldırılmıştır. Diffie-Hellman algoritması sadece ortak gizli anahtarı belirlemekte kullanılmaktadır.

1977 yılında R.Rivest, A.Shamir ve L.Adleman isminde üç bilim adamının oluşturduğu yeni asimetrik şifreleme algoritması RSA, anahtar dağıtımının yanında şifreleme ve deşifre etme işlemlerini de gerçekleştirmektedir.

RSA şifreleme algoritmasının geliştirilmesiyle birlikte asimetrik şifreleme algoritmalarının günümüzdeki kullanım alanı daha da genişlemiştir.

RSA kripto sistemi, hem şifreleme hem de elektronik imza fonksiyonlarını yerine getirmek amacıyla kullanılabilir. Bu sistemin güvenliği tamsayılarda çarpanlara ayrılma zorluğu problemine dayanmaktadır.

RSA kripto sisteminde şifreli mesaj gönderilecek kişilerin açık anahtarlarına ihtiyaç duyulur. Bir kişinin açık anahtarı ile şifrelenmiş bir mesaj sadece o kişinin gizli anahtarı ile deşifre edilebilir.

Her *A* kişisi anahtarını şu şekilde oluşturur:

• İki tane farklı, rastgele *p* ve *q* asal sayılarını seçer.

• $n = pq$ ve $\emptyset = (p-1)(q-1)$ değerlerini hesaplar.

• $1 < e < \emptyset$ ve $gcd(e; \emptyset) = 1$ olacak şekilde rastgele bir *e* sayısı seçer.

Burada *gcd* en büyük ortak bölen(Great Common Divisor) fonksiyonunu ifade etmektedir.

• Öklid algoritmasını kullanarak, $1 < d < \emptyset$ ve $ed = 1 \pmod{\emptyset}$ koşulunu sağlayan *d* sayısını hesaplar.

Sonuç olarak, *A*'nın açık anahtarı $(n; e)$, gizli anahtarı ise *d* olur.

RSA Şifreleme Sayısal Örnek:

Seçilen iki asal sayı $p=7$ ve $q=17$ olsun,

$N = p.q = 7 \times 17 = 119$

$\emptyset(n) = (p-1) \times (q-1) = 96$

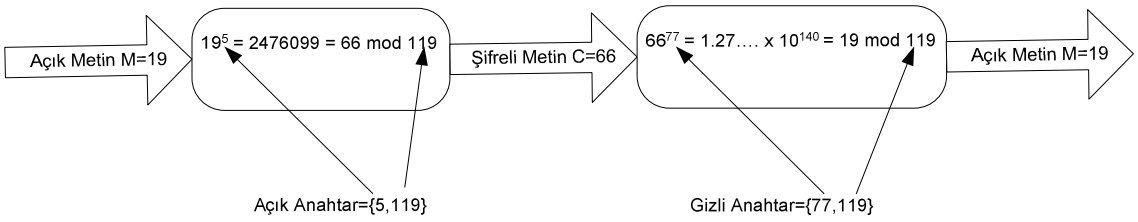
Seçilen rastgele *e* sayısı *5* olmak üzere,

$d.e = 1 \bmod 96$ eşitliğini gerçekleyen *d* sayısı *77* olacaktır,

Sonuç olarak hesaplanan açık anahtar- gizli anahtar çifti:

$\{5,119\} – \{77,119\}$ olacaktır.

Şekil 2-14 de $M=19$ olarak kabul edilen açık bir metin bilgisinin şifrelenerek karşı tarafa iletilmesi ve deşifre edilerek tekrar elde edilmesi gösterilmektedir.



Şekil 2-14 15 RSA Şifreleme

3 MATEMATİKSEL AÇIKLAMALAR

Oldukça yüksek mertebelerdeki sayıların kullanıldığı kriptoloji uygulamalarında algoritmalar sayıların modüler biçimde ifade edilebilmesi ve modüler aritmetik özelliklerine dayanarak oluşturulmaktadır. Kriptoloji uygulamalarında tahmin edilmelerindeki zorluk nedeni ile asal sayılar kullanıldığından, öncelikle bu sayıların üretilmesi ve asallıklarının test edilmesi gerekmektedir. Bu bölümde modüler aritmetik, asal sayılar ve asallık testleri, üstel işlemler, rastsal bit üretimi gibi konularda temel bilgiler verilmektedir.

3.1 Tam Sayıların Modüler Şekilde İfade Edilmesi

Tanım1

a, b ve *n* tam sayıları ve $n \neq 0$ şartı için, eğer *a* ve *b* nin farkı *n*'in *k* katı kadarsa bu durum şu şekilde gösterilebilir:

$$a - b = k.n$$
$$veya$$
$$a \equiv b \bmod n$$

Teorem1

a1, a2, n tamsayıları ve $n \neq 0$ şartı için,

$$(a1 op a2) \bmod n \equiv [(a1 \bmod n) op (a2 \bmod n)] \bmod n$$

denkliği gösterilebilir, burada *op* "+" yada "*" şeklinde bir operatör olabilir.
Bir *a=b mod n* eşitliği *a* ve *b*'nin aynı *n* ile bölündüğünde aynı kalanı verdiklerini ifade eder.

3.2 Modüler Aritmetik Özellikleri

1. Birleşme (Associativity) Özelliği

(a+b) + c = a + (b+c) mod n

2. Geçişlilik (Commutativity) Özelliği

a + b= b + a mod n

3. Dağılma (Distributivity) Özelliği

(a + b).c= a.c + b.c mod n

3.3 Asal Sayılar

Asal sayılar kripto sistemlerinde önemli rol oynamaktadırlar. Yapılan temel işlemler asal bir sayının oluşturulması ve asal olduğu iddia edilen bir sayının asallık testinin yapılmasıdır. Asal sayı oluşturma, verilmiş bir [r1,r2] tam sayılar aralığında asal sayı bulma işlemidir.

$a^{s-1} \equiv 1 \bmod s$ ve 1 $1\langle a\langle s$ şartını sağlayan *s* tam sayısına *a* tabanına göre sanki asal (pseudo-prime) sayı denir.

Kriptoloji biliminde istenilen büyüklükte asal sayılar elde edebilmek için izlenilen yöntem, uygun büyüklükte bir *n* tam sayısının seçilmesi ve bu sayının asallığının test edilmesidir. Bu, seçilen *n* sayısının $\sqrt{n}$ den küçük herhangi bir asal sayı ile bölünüp bölünemeyeceğinin kontrolü ile sağlanabilir [7].

Asallık testi için pratikte etkili birçok yöntem geliştirilmiş olsa da, asallık testi temelde üç ana adım üzerine inşa edilmektedir.

1. Aday olarak istenilen büyüklükte bir *n* tek tam sayısı seçilir.
2. *n* sayısının asal olup olmadığı test edilir.
3. Asal değil ise ilk adıma geri dönülür.

Seçilen *n* sayısından itibaren, sayı *n,n+2,n+4…* şeklindeki gibi bir sıralama ile arttırılarak asal sayı elde etmek için yapılacak tahminlerin sayısı azaltılabilir.

İkinci adım, seçilen sayının asallığını kanıtlayacak (provable prime) şekilde bir yöntem olabileceği gibi, sonucu aday sayının olası asal (probably prime) olduğunu gösteren olasılık temelli bir test yöntemi de olabilir.

Gerçek asallık testleri rastgele *n* sayısının asal olup olmadığını belirlerken, olasılık temelli asallık testleri, üye *n* sayısının asal olup olmadığını gösteren matematiksel nedenlerden oluşturulup oluşturulmadığının araştırılması temeline dayanmaktadır.

Gerçek asallık testleri olasılık temelli testlere oranla daha kesin sonuçlar üretebilse de, kullandıkları bilgisayar kaynakları daha fazla ve test sonucunu üretme süreleri oldukça uzundur.

3.3.1 Rabin ve Miller Asallık Testi

Bu testin algoritmasına geçmeden önce bazı matematiksel çıkarımlar yapmak doğru olacaktır.

- **Fermat Teoremi:** *p* bir asal sayı olsun. Her *a* tam sayısı için $a^p \equiv a \bmod p$ denkliği ve *p* ile bölünmeyen her *a* tamsayısı için ise $a^{p-1} \equiv 1 \bmod p$ denkliği her zaman doğrudur.

- *p* sayısının asal bir tek sayı olduğu kabul edildiğinde;

$x^2 \equiv 1 (\bmod p)$ eşitsizliğinin $x \equiv 1$ ve $x \equiv -1$ olmak üzere iki çözümü mevcuttur.

Fermat Teoremi Kanıt:

$$x^2 - 1 \equiv 0 (\bmod p)$$
$$(x-1)(x+1) \equiv 0 (\bmod p)$$

Modüler aritmetik kuralları gereği *p*, *(x+1)*'e, *(x-1)*'e yada ikisine birden bölünebilir olmalıdır. *p*'nin hem *(x+1)*'e hem de *(x-1)*'e bölünebilir olduğu kabul edildiğinde; *(x+1)=kp* ve *(x-1)=jp* olmak üzere *k* ve *j* tamsayılarının varlığından bahsedilebilir.

İki eşitlik birbirinden çıkartıldığında;

2=(k-j)p eşitliği elde edilecektir.

Bu eşitlik sadece *p*'nin *2* olması durumunda sağlanmaktadır.

Bu teorem sadece tek asal sayılar için geçerlidir. Bu nedenle *x* çözümü sadece *p|x+1* yada *p|x-1* için elde edilebilir.

p|(x-1) çözümünün var olduğu kabul edildiğinde;
bazı *k* sayıları için *x-1 = kp* , ve dolayısıyla $x \equiv 1 \bmod p$ dir.

Diğer çözüm'ün geçerli olduğu kabul edildiğinde ise $x \equiv -1 \bmod p$ denkliğine ulaşılabilir.

Teorem ters yönden işletildiğinde ise şu söylenebilir;

$x^2 \equiv 1 (\bmod n)$ denkliğinde *x*'in *+1* veya *-1*'den farklı bir çözümü mevcut ise *n* sayısı asal sayı değildir.

Fermat Teoremi Örnek:

$1^2 \equiv 1 \bmod 7$

$6^2 \equiv 36 \bmod 7 \equiv 1 \bmod 7; 6 \equiv -1 \bmod 7$

$Çözümler : 1 ve -1$

Dolayısıyla *7* asal sayı.

$1^2 \equiv 1 \bmod 8$

$3^2 \equiv 9 \bmod 8 \equiv 1 \bmod 8$

$5^2 \equiv 25 \bmod 8 \equiv 1 \bmod 8; 5 \equiv -3 \bmod 8$

$7^2 \equiv 49 \bmod 8 \equiv 1 \bmod 8; 7 \equiv -1 \bmod 8$

$Çözümler : 1, -1, 3, -3$

Dolayısıyla 8 sayısının asallığından bahsedilemez.

Yukarıdaki bilgilerin doğrultusunda Rabin ve Miller tarafından geliştirilen ve *WITNESS* algoritması olarak adlandırılmış asallık testi algoritması Tablo 2 de verilmektedir [2].

	WITNESS(a,n)
1	$b_k b_{k-1} \ldots b_0$ (n-1) in ikili düzende gösterimi olmak üzere
2	d = 1
3	i =k'dan 0'a kadar
4	x = d
5	d = (d x d) mod n
6	d = 1 ve x ≠ 1 ve x ≠ n-1 ise
7	TRUE değeri döndür
8	b_i = 1 ise
9	d = (d x a) mod n
10	d ≠ 1 ise
11	TRUE değeri döndür
12	FALSE değeri döndür

Tablo 2 Witness Asallık Testi Algoritması Adımları

WITNESS algoritmasının girdileri asallığı test edilecek olan *n* sayısı ve *n*'den küçük bir *a* sayısıdır. Eğer algoritma TRUE değerini döndürürse *n* sayısı kesinlikle asal değildir, aksi takdirde *n* sayısı asal olabilir.

Algoritmanın 3. satırından 9.satırına kadar *d* sayısı a^{n-1} *mod n* şeklinde hesaplanmaktadır. Fermat teoreminden bilindiği üzere *n* sayısı asal ise $a^{n-1} \equiv 1 \bmod n$ denkliği sağlanır. Dolayısı ile 10. satırda d'nin 1'e eşit olup olmadığı kontrol edilmekte eğer değil ise TRUE değeri döndürülerek sayının kesinlikle asal olmadığı belirtilmektedir.

6. satırda ise $x^2 \equiv 1 \bmod n$ denkliğinin +1 yada -1 den başka kökü olup olmadığı hesaplanmakta, eğer var ise yine TRUE değeri döndürülerek verilen n sayısının asal olmadığı belirtilmektedir.

Algoritmadan FALSE değeri döndürüldüğünde ise sayı olası asal olarak işaretlenir ve diğer *a<n* sayısı olasılıkları için tekrar edilir.

Algoritma *s* değeri kadar işletildiğinde ve bütün bu işletimlerden FALSE değeri dönüyor ise, testi yapılan *n* sayısı $1-2^{-s}$ olasılıkla asaldır denilebilir.

3.4 Grup Teorisi

Verilen herhangi bir *G* grubu için bu gruba ait elemanların sayısına *G* grubunun düzeni denir ve $ord(G) = |G|$ sembolüyle gösterilir. Eğer *H* grubu *G* grubunun bir alt grubu ise |*H*| değeri |*G*| değerini böler. Böylece eğer *G* grubunun düzeni bir asal sayıysa *G*' nin tek alt grubu kendisidir. Bu durumda *G* grubu çarpmalı olarak yazılabilir. *G* grubu çarpmalı olarak yazılabilirse ve $g \in G$ olmak üzere *g* sayısı *G* grubunun düzeni ise, bu *g* sayısı $i \in N \cup \{\infty\} ve g^i = 1$ şartını sağlayan en küçük *i* değeridir.

Burada $\vee j,l \in Z$:

$g^j = g^l \Leftrightarrow j \equiv l \bmod ord(g)$ dir.

G grubunun alt grubu olan tüm gruplar *g* elemanının bir üssüdür ve <*g*> ifadesi ile gösterilir. Eğer <*g*>=*G* ise *g* sayısı *G* grubunun üreteci (jenarötörü) olur. Bir üreteci olan tüm gruplara devirli grup (cyclic) grup adı verilir.

G grubunun düzeni *p* asal sayısı ise grup içerisinde yer alan 1 dışındaki tüm sayılar *G* grubunun üreteci olur. Diğer bir deyişle <*g*> nin düzeni *p* olur.

3.5 GF(p)'de Üstel İşlem

Bir çok kriptografi algoritması üstelleştirmeyi kullanmaktadır.

$$b = a^e \bmod p$$

Üstelleştirme basit olarak bir *n* sayısı için *O(n)* çarpma olacak şekilde gerçekleşen tekrarlanan çarpmalardan ibarettir.

3.6 GF(p) ' de Ayrık Logaritmalar

Üstelleştirmede ters problem, bir modulo *p* sayısının ayrık logaritmasının bulunmasıdır. Bu işlem:

$a^i = b \bmod p$ 'de i yi hesaplamak olarak ifade edilebilir.

Ayrık logaritmanın hesaplanması, üstelleştirmeye nispeten kolay yolu olmayan zor bir problemdir. Bu problemde eğer *p* asal ise, herhangi bir *b!=0* için her zaman bir ayrık logaritması olan bir *α* olduğu gösterilebilir.

a'nın ardışıl kuvvetleri *mod p* ile grup oluşturur;
α mod p, α^2 *mod p,…..,* α^{p-1} *mod p 1* ile *p-1* arasında değer alır. Hesaplanması oldukça zor olan *α*'ya primitif kök denir.

Herhangi bir *b* tamsayısı ve *p* nin primitif kökü olan *α* için bir *i* üssü bulunabilir ki ;

$b = \alpha^i \bmod p \quad 0 \le i \le p-1$ dir.

Buradaki *i* üssü ayrık logaritma veya indeks olarak gösterilir.

3.7 En Büyük Ortak Bölen (Greatest Common Divisor)

a ve *n* tamsayıları için, ($a \in \{0,1,...n-1\}$) olmak üzere; eğer *a* ve *n* aralarında asal iki sayı ise *a*'nın modül *n*'e göre yalnız bir tane tersi vardır ve a^{-1} sembolü ile gösterilir.

$OBEB(a,n)=1 \Leftrightarrow \exists b \in [a,n-1], 1 = a.b \bmod n$ yani $b=a^{-1}$ dir.

a ve *b*'nin ortak böleni *(a,b), a* ve *b*'nin her ikisini de bölen en büyük sayıdır. *Euclid's* algoritması iki *a* ve *n (a<n)* sayısının en büyük ortak böleninin bulmak için kullanılır.

GCD (a,n)
1 g0=n
2 g1=a
3 gi+1 = gi-1 mod gi
4 gi=0 iken
5 (a,n) = gi-1

Tablo 3 İki Sayının En Büyük Ortak Böleninin Hesaplanması

En Büyük Ortak Bölen Hesaplama Örneği:

GCD(56,98)

g0=98

g1=56

g2 = 98 mod 56 = 42

g3 = 56 mod 42 = 14

g4 = 42 mod 14 = 0

sonuçta *EBOB (56,98)=14*

3.8 Rastsal Bit Üretimi

Birçok kriptografik sistem tahmin edilemez niceliklerin üretilmesi temeline dayanmaktadır. DES şifreleme algoritmasında kullanılan gizli anahtar, Diffie-Hellman [6] anahtar değişiminde kullanılan *p* ve *q* asal sayıları, elektronik imza yöntemleri, DSA algoritmasında kullanılan *a* gizli anahtarı gibi rastsal bit üretiminin kullanıldığı bir çok örnek verilebilir.

Yukarıda geçen örneklerde belirtilen nicelikler herhangi bir saldırganın geliştireceği bir yöntem ile bu nicelikleri elde etmesini veya güvenilir bir şekilde kullanılmasını engelleyebilmesini önleyecek kadar yeterli boyutlarda ve rastsal olarak üretilmelidirler.

Örneğin; DES algoritması için anahtar büyüklüğü 2^{56}'dır. Algoritma işletiminde *k* gizli anahtarı gerçek bir rastsal üreteç tarafından oluşturulduğu takdirde, saldırgan tarafından ortalama 2^{55} kadar olasılık denenmelidir.

Öte yandan, eğer *k* ilk olarak 16 bitlik rastsal gizli değer *s* sayısının seçilmesi ve bunun zor ancak açık olarak bilinen fonksiyonlarla genişletilmesi ile hesaplanırsa saldırganın yapması gereken tahmin sayısı ortalama 2^{15}'e kadar düşmektedir.

Gerçek üreteçler doğal bir rastsallığın uygulanmasını gerektirirler. Donanımsal yada yazılımsal olarak bu rastsallığı oluşturmak oldukça zor bir işlemdir. Bunun yanında birçok kriptografik uygulamada üreteçler, saldırganlar tarafından derinlemesine incelenip kırılmaya çalışılan bileşenler değildir.

3.8.1 Donanım Temelli Üreteçler

Donanım temelli bit üreteçlerinin rastsallığı, çeşitli fiziksel fenomenler ile oluşturup geliştirirler [7]. Bunlara örnek vermek gerekir ise;

- Radyoaktif bozunma esnasında partiküllerin yayılması arasında geçen süre,
- Yarı iletken diyot yada direnç üzerinde oluşan termal gürültü,
- Serbest koşumlu osilatörün frekans düzensizliği,
- Metal yalıtımlı yarı iletken bir kapasitörün belirli bir zaman aralığı sonunda şarj olma kapasitesi,
- Disk sürücüleri okuma zamanları üzerinde rastsal değişimlere neden olan hava türbülansı,
- Mikrofondan yada kameranın video girişinden gelen ses.

İlk iki fenomeni temel alan üreteçler, genellikle cihazın dışına yapılandırılmalıdır. Bu nedenle bu tip üreteçler saldırganlar tarafından daha çok araştırılmaya ve değiştirilmeye yatkın üreteçlerdir.

Osilatör ve kapasitör temelli üreteçler ise VLSI cihazların üzerine konuşlandırılabilir. Bu nedenle aktif saldırılara karşı daha korumalıdır.

3.8.2 Yazılım Temelli Üreteçler

Rastsal bit üreteçlerini yazılımsal olarak tasarlamak, donanım temelli sistemlere oranla daha zordur. Yazılım temelli üreteçlerde temel alınan bazı nitelikler [7];

- Sistem saati,
- Tuş basmaları ve fare hareketleri arasında geçen süre,
- Giriş ve çıkış arabelleklerinin içeriği,
- Kullanıcı girdileri,
- Sistem yükü ve ağ istatistikleri gibi işletim sistemine özgü değerler.

Yukarıda sayılan işlemler bilgisayar platformu gibi bir çok faktöre bağımlı olarak değişiklik gösterebilir. Ayrıca bu işlemler saldırganlar tarafından araştırılmasının ve değiştirilmesinin engellenmesi oldukça zor olan işlemlerdir.

Örneğin; saldırganın sistemde rastsal dizilerin üretildiğine dair kuvvetli düşünceleri ve ipuçları var ise saldırmak istediği sistem saatini yüksek bir olasılıkla tahmin edebilecektir.

İyi tasarlanmış bir yazılım temelli rastsal bit üreteci, bir çok rastsallık kaynağını bir arada kullanabilmelidir. Her bir kaynak örneklenmeli ve değerler örneklerin karıştırma fonksiyonu (mixing function) ile birleştirilmesi ile üretilmelidir.

SHA-1 ve MD5 gibi kriptografik hash fonksiyonları karıştırma fonksiyonlarına örnek olarak verilebilir. Bu fonksiyonların kullanımı ile amaçlanan, üretilen gerçek rastsal bit değerleri ile örneklenen bit dizilerini birbirinden yalıtmaktır.

4 DOĞRULAMA PROTOKOLLERİ ve PAROLA DOĞRULAMALI ANAHTAR DEĞİŞİM YÖNTEMLERİ

Bu bölümde ağ haberleşmesinde kullanılan Doğrulama Protokolleri ve Modern Anahtar Değişim Yöntemleri hakkında bilgiler yer almaktadır.

4.1 Doğrulama Protokolleri

Bu bölümde anlatılmakta olan Doğrulama Protokolleri, Karşılıklı Doğrulama ve Tek Yönlü Doğrulama olmak üzere iki alt madde şeklinde açıklanmıştır. Her bir yöntem için simetrik (klasik) ve asimetrik (modern) şifreleme uygulamaları alt maddeler halinde anlatılmıştır.

4.1.1 Karşılıklı Doğrulama

Doğrulama protokollerinin en kapsamlı kullanım alanı olan çeşididir. Haberleşen tarafların, oturum anahtarını güvenli olarak değiştirebilmeleri için kimliklerini karşılıklı olarak doğrulamaları temeline dayanmaktadır. Doğrulama anahtar değişimi problemi, temel olarak iki ana konuya bağlıdır; gizlilik ve zaman uygunluğu. Taklit etme ve oturum anahtarını ele geçirme saldırılarını engellemek için taraflar arasındaki haberleşme esnasında çeşitli kimlik ve oturum anahtarı bilgileri şifreli bir formda taraflar arasında gönderilir. Bu amaç için taraflar arasında, gizli anahtar, parola yada açık anahtar gibi bilgiler önceden paylaşılmak zorundadır. İkinci konu olan zaman uygunluğu ise taraflar arasında gidip gelen mesajların tekrarlanması durumu açısından büyük önem taşımaktadır. Mesajların tekrarlanması saldırganın oturum anahtarını elde etmesine yada taraflardan birini başarılı bir şekilde taklit etmesine olanak sağlayabilir. Tekrarlama saldırıları olarak tanımlanan bu saldırılar aşağıdaki şekillerde kendini gösterebilir;

- **Basit Tekrarlama:** Saldırgan mesajları kopyalar ve bunları daha sonra tekrarlar.
- **Kayıt Edilebilen Tekrarlamalar:** Saldırgan zaman damgası ile işaretlenmiş mesajları geçerli bir zaman aralığında tekrar edebilir.
- **Fark Edilemeyen Tekrarlamalar:** Orijinal mesaj tamamen ele geçirilerek hedefine gitmesi engellenir. Karşı tarafa sadece tekrar edilen mesaj ulaşır.

- **Değişiklik Yapmadan Geri Tekrar Etme:** Mesajın saldırgan tarafından gönderen tarafa aynen geri iletilmesidir. Bu tip saldırı gönderilen ve geri dönen mesajlar arasındaki farkın kolaylıkla ayırt edilemediği doğrusal şifreleme yöntemlerinin kullanıldığı anahtar değişimlerinde olasıdır.

Bu saldırılardan korunmanın bir yöntemi anahtar değişimi esnasında taraflar arasında gönderilen her bir mesaja sıra numaralarının eklenmesidir. Bu durumda yeni mesaj sadece uygun sıra numarası ile geldiğinde kabul edilecektir. Ancak bu yöntem her iki tarafında her defasında son sıra numarasını takip etmeleri gerekliliği nedeni ile tercih edilmeyen zor bir yöntemdir. En çok kullanılan iki yöntem ise aşağıda anlatılmaktadır.

- **Zaman Damgaları:** Haberleşen taraflar, sadece kendi geçerli zaman bilgilerine yakın zaman damgalarına sahip mesajları kabul ederler. Bu yöntem zaman bilgisinin taraflar arasında senkronize edilmesi gerekliliğini doğurmaktadır.
- **Zorluk/Karşılık:** Mesaj bekleyen taraf ilk olarak bir zorluk değeri üreterek karşı tarafa gönderir ve ondan bu değere uygun bir karşılık mesajının dönmesini bekler. Zorluk (Nonce) değeri, şifreleme işlemlerinde kullanılan başlangıç değerine getirme vektörleri gibi güvenlik amacı ile kullanılan ve rastsal olarak seçilen benzersiz değerlerdir. Bu değerler, güvenli olarak başlatılan bir oturumda sadece bir defaya mahsus kullanılırlar. Üretilmelerinin belirli bir algoritması olmamakla birlikte, üretildikleri yıl gün ve saati birleştirerek değer üreten bir fonksiyon gibi benzeri olmayacak değerler üreten algoritmalar kullanılmaktadır.
-

Zaman damgalarının kullanılması, senkronizasyon zorunluluğu nedeni ile bağlantı temelli uygulamalar için uygun değildir. Protokol muhtemel ağ problemlerine karşı hata toleranslı, çeşitli güvenlik saldırılarına karşı ise güvenli olmak zorundadır. Zaman damgalarının kullanılması durumunda, ağ üzerinde yaşanabilecek problemler nedeni ile senkronizasyon periyodu geniş tutulmak zorundadır. Ancak bu durumda da muhtemel saldırıların başarı oranı artacaktır.

Bağlantı temelli olmayan uygulamalarda ise, zorluk/karşılık yöntemlerinin kullanılması, haberleşme esnasında karşılıklı el sıkışma (handshaking) gerekliliği getirdiği için uygun bir yöntem olmayacaktır. Bu tip uygulamalarda, senkronizasyon gereksinimi karşılayacak güvenli senkronizasyon sunucularına güvenmek en optimum yöntem olmaktadır.

4.1.1.1 Geleneksel Şifreleme Yöntemleri

Yöntem genel olarak güvenli bir anahtar dağıtım merkezinin (KDC) haberleşme ortamına dahil edilmesi temeline dayanmaktadır. Haberleşen her bir taraf KDC ile esas anahtar olarak tanımlanan gizli bir bilgiyi paylaşmaktadır. KDC taraflar arasında oturum anahtarının elde edilmesini sağlamak için esas anahtarı kullanarak kısa süreli anahtarlar üretmek ve bunları haberleşen taraflara dağıtmaktan sorumludur. Modern işletim sistemlerinde kullanılmakta olan *Kerberos Doğrulama*, bu mekanizmaya bir örnek olarak verilir.

Şekil 4.1 de görülen *Needham ve Schroeder* [8] tarafından geliştirilen protokolde, gizli anahtarlar K_a ve K_b , A ile KDC ve B ile KDC arasında paylaşılmaktadır.

Protokolün birinci adımında A tarafı kendisinin ve B tarafının kimlik bilgisi ile ürettiği benzersiz (unique) sözcük değerini KDC'ye göndererek başvuruda bulunur.

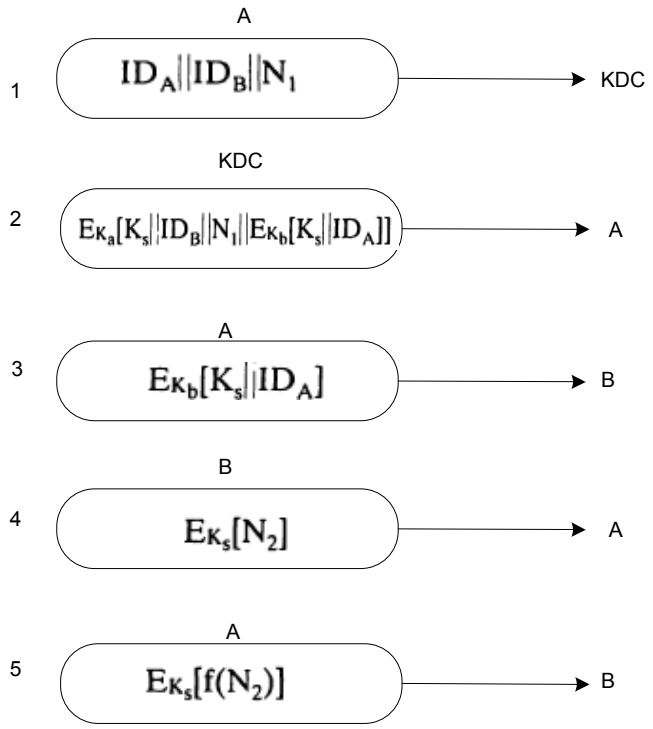
İkinci adımda KDC A tarafına A'nın gizli anahtarı ile şifrelenmiş olarak oturum anahtarını, B tarafının kimlik bilgisini, A tarafının üretip kendisine gönderdiği sözcük bilgisini ve B tarafının gizli anahtarı ile şifrelenmiş bir şekilde oturum anahtarını ve A tarafının kimlik bilgisini gönderir. Mesajı deşifre eden A oturum anahtarını mesajdan elde eder ve mesajın gerçekten KDC'den gönderildiğini kontrol etmek için kimlik ve sözcük bilgilerini doğrular.

Üçüncü adımda A tarafı kendine gelen mesajın B'ye ait gizli anahtar ile şifrelenmiş kısmını B tarafına gönderir B tarafı mesajı deşifre ederek oturum

anahtarını elde eder ve kimlik bilgisini kontrol ederek mesajın gerçekten A tarafından gönderildiğini doğrulayabilir.

Dördüncü adımda B tarafı kendi ürettiği benzersiz sözcük mesajını oturum anahtarı ile şifreleyerek A tarafına gönderir, A tarafı mesajı deşifre eder ve böylece B tarafının oturum anahtarını elde ettiğini doğrulamış olur.

Beşinci ve son adımda ise A tarafı B'nin gönderdiği sözcük değerini bir fonksiyonla işleme sokarak yeni bir değer oluşturur ve bunu oturum anahtarı ile şifreleyerek B tarafına gönderir, B tarafı mesajı deşifre eder ve A'nın oturum anahtarını başarılı bir şekilde elde ettiğini doğrulamış olur.

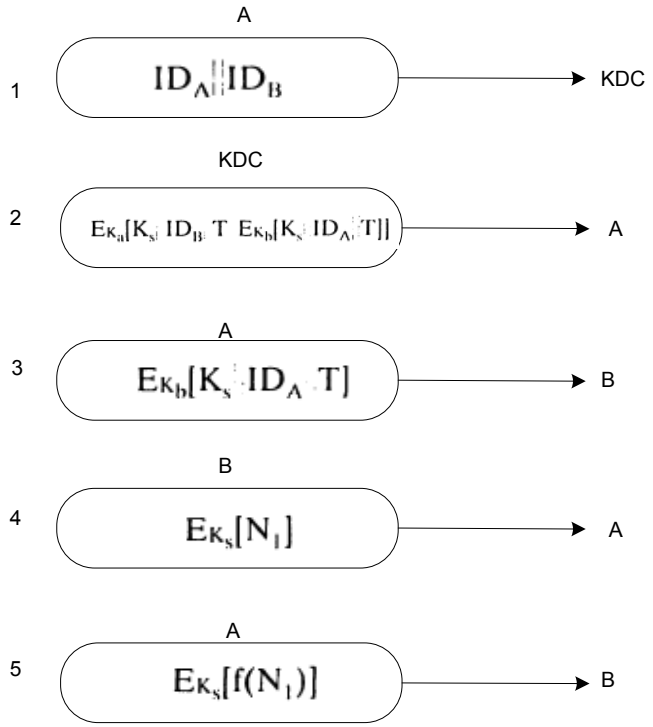


Şekil 4-1 Geleneksel Şifreleme Yöntemli Doğrulama ve Anahtar Değişimi -1

Protokol dördüncü ve beşinci adımlardaki el sıkışma işlemlerine rağmen tekrarlama saldırılarına açıktır. Saldırgan daha önceden taraflar arasında paylaşılmış oturum anahtarı bilgisini ve üçüncü adımda gönderilen mesajı elde edebilir, bu durumda saldırgan A'yı taklit ederek ve eski oturum anahtarını

kullanarak üçüncü adımdaki mesajı tekrarlayıp B tarafına gönderebilir. Eğer B tarafı A ile önceden paylaştıkları oturum anahtarı bilgilerinin kayıtlarını tutmuyor ise mesajın bir tekrarlama atağı olduğunu algılayamayacaktır. Benzer olarak saldırgan dördüncü adımda A tarafına gönderilen el sıkışma mesajını elde ederse A tarafını taklit ederek B tarafına sahte mesajlar gönderebilir.

Bahsedilen zayıflıkları engellemek için *Denning* [9] tarafından geliştirilen protokolde ikinci ve üçüncü adımlara zaman damgası bilgileri eklenmiştir. Şekil 4.2 de görülen protokolde üretilen T zaman damgası oturum anahtarının o anda üretildiğinin bir göstergesi olmaktadır.



Şekil 4-2 Geleneksel Şifreleme Yöntemli Doğrulama ve Anahtar Değişimi -2

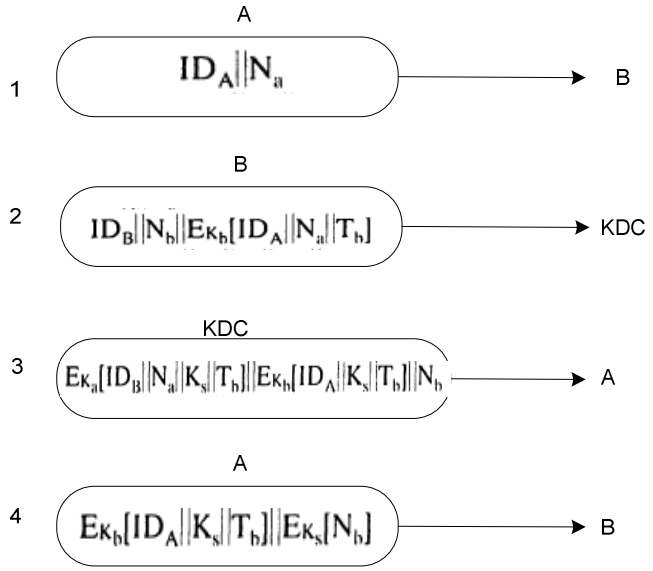
A ve B tarafları Saat $-T < \Delta t_1 + \Delta t_2$ kontrolünü gerçekleştirerek oturum anahtarının üretilme anını kontrol edebilir. Burada Δt_1 tarafların saat değeri ile KDC'nin yerel saati arasındaki tahmin edilen fark, Δt_2 ise tahmin edilen ağ gecikme zamanıdır. Her bir taraf kendi saatini standart bir referans kaynağa göre ayarlayabilir. T zaman damgası tarafların esas anahtarları ile şifreli olarak ağa çıkarıldığı için

saldırgan eski oturum anahtarını elde etse dahi üçüncü adımda gerçekleştireceği tekrarlama atağı başarılı olamayacaktır.

Protokolde zayıflıklar ise senkronizasyon mekanizmasının sabote edilebilme ve saat değerlerinde farklılık hataları oluşma olasılıklarıdır. Örneğin; göndericinin saatinin alıcının saatinden daha ileri olduğu düşünüldüğünde, göndericinin mesajlarını kesen saldırgan, bu mesajları zaman, alıcıda geçerli bir değer olduğunda gönderip oturum anahtarının güvenirliliğini tehlikeye sokabilir. Bunu önlemenin bir yolu, tarafların her mesaj gönderme öncesinde, kendi saatlerini KDC'nin saati ile karşılaştırarak senkronize etmeleri, diğer bir yolu ise dördüncü ve beşinci adımlarda gönderilen mesajlara, tarafların üreteceği benzersiz sözcük değerlerinin eklenmesidir.

Neuman [10] tarafından geliştirilen protokol, yukarıda açıklanan protokoller de yer alan zayıflıkların hepsine çözüm getirmektedir.

Şekil 4.3 de görülen protokolün birinci adımında A tarafı B tarafına kendi kimlik bilgisini ve ürettiği benzersiz sözcük değerini açık bir şekilde gönderir.



Şekil 4-3 Geleneksel Şifreleme Yöntemli Doğrulama ve Anahtar Değişimi -3

İkinci adımda, B tarafı KDC'ye, kendi kimlik bilgisini ve ürettiği benzersiz sözcük değerini açık olarak, A tarafından kendisine gelen mesajın zaman damgası

eklenmiş halini ise kendi anahtarı ile şifrelenmiş olarak gönderir ve böylece KDC'ye oturum anahtarı ihtiyacını bildirmiş olur.

Üçüncü adımda KDC A tarafına, B'nin kimlik bilgisi, A'nın ürettiği benzersiz sözcük bilgisi, oturum anahtarı ve B tarafına ait zaman damgası bilgisini A'nın gizli anahtarı ile şifreli olarak, A'nın kimlik bilgisi, oturum anahtarı ve B'nin zaman damgası bilgisini B'nin gizli anahtarı ile şifreli olarak, B'nin ürettiği benzersiz sözcük değerini ise açık bir şekilde gönderir. A tarafı mesajın kendi anahtarı ile deşifre ettiği bölümündeki bilgileri kullanarak, bilgilerin gerçekten KDC'den gönderildiğini, kendisiyle oturum anahtarı paylaşmak isteyen tarafın B tarafı olduğunu, bilgilerin taze bilgiler olduğunu doğrular ve oturum anahtarını elde eder. Dördüncü adımda, A tarafı bir önceki adımda kendine gönderilmiş olan B'nin gizli anahtarı ile şifrelenmiş mesajı ve oturum anahtarı ile şifrelenmiş, B'ye ait benzersiz sözcük değerini B tarafına gönderir. B tarafı mesajın kendi gizli anahtarı ile deşifre ettiği bölümünden oturum anahtarını elde eder, diğer bilgilerle ise bilgilerin gerçekten A tarafından gönderildiğini ve tazeliğini kontrol eder. Mesajın kalan kısmını ise elde ettiği oturum anahtarı ile deşifre ederek kendisinin ikinci adımda üretip KDC'ye gönderdiği benzersiz sözcük bilgisi ile karşılaştırır ve böylece oturum anahtarını paylaşma işlemlerinin güvenilir KDC tarafından yönlendirildiğini doğrular. Bu adımda kullanılan mesaj bilet (ticket) olarak tanımlanır ve A ile B tarafları arasında gerçekleştirilecek sonraki doğrulama işlemlerinde kullanılabilir.
Geçerli bir zaman aralığında yapılacak olan bir önceki oturuma ait alt doğrulama işleminde ise artık KDC'ye ihtiyaç kalmamaktadır.

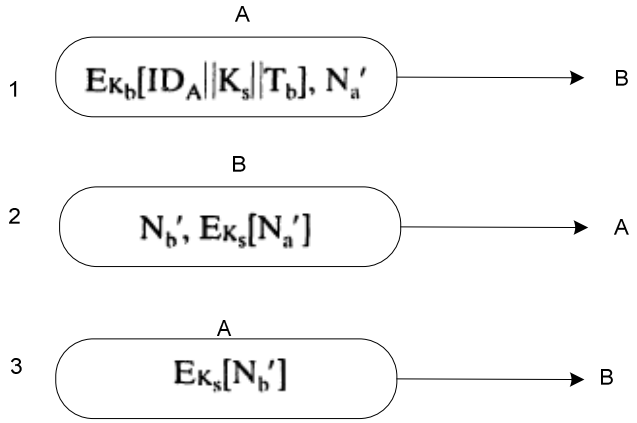
Şekil 4.4 de görülen alt doğrulama işleminin birinci adımında A tarafı B'ye, B'nin anahtarı ile şifrelenmiş bir şekilde kendi kimlik bilgisini, oturum anahtarını ve B'ye ait zaman damgası değerini, açık olarak ise yeni ürettiği benzersiz sözcük değerini gönderir.

İkinci adımda, B tarafı A'ya yeni ürettiği benzersiz sözcük değerini açık olarak, A'nın kendisine gönderdiği benzersiz sözcük değerini ise oturum anahtarı ile şifreli olarak gönderir.

Üçüncü adımda A tarafı B'ye oturum anahtarı ile şifrelenmiş bir şekilde B'nin kendisine gönderdiği benzersiz sözcük değerini gönderir.

Bu adımlarda zaman damgası kontrolü ile biletin dolayısıyla oturum anahtarının geçerliliği doğrulanırken, benzersiz sözcük değerlerinin kontrolü ile gidip gelen mesajların tekrar mesajları olmadığı doğrulanmış olur.

İkinci adımda, B kendi ürettiği zaman damgasının geçerliliğini kontrol ettiğinden herhangi bir ek zaman senkronizasyonu işlemine gerek kalmamaktadır.



Şekil 4-4 Geleneksel Şifreleme Yöntemli Doğrulama ve Anahtar Değişimi -4

4.1.1.2 Açık Anahtar Şifreleme Yöntemleri

Anahtar değişiminde açık anahtar şifreleme yöntemlerinin kullanılması tarafların birbirlerinin açık anahtar bilgilerine sahip olmaları gereksinimini doğurmaktadır.

Bu yöntem için geliştirilen ilk protokol şekil 4.5 de görülmektedir.

Şekil 4-5 Asimetrik Şifreleme Yöntemli Doğrulama ve Anahtar Değişimi -1

Protokolün birinci adımında A tarafı doğrulama sunucusu olarak bilinen güvenli bir sunucuya kendisinin ve B tarafının kimlik bilgileri ile başvuruda bulunur.

İkinci adımda doğrulama sunucusu (AS) A tarafına şifrelenmiş veri blokları biçiminde zaman tarafların açık anahtarlarını ve zaman damgası bilgilerini gönderir.

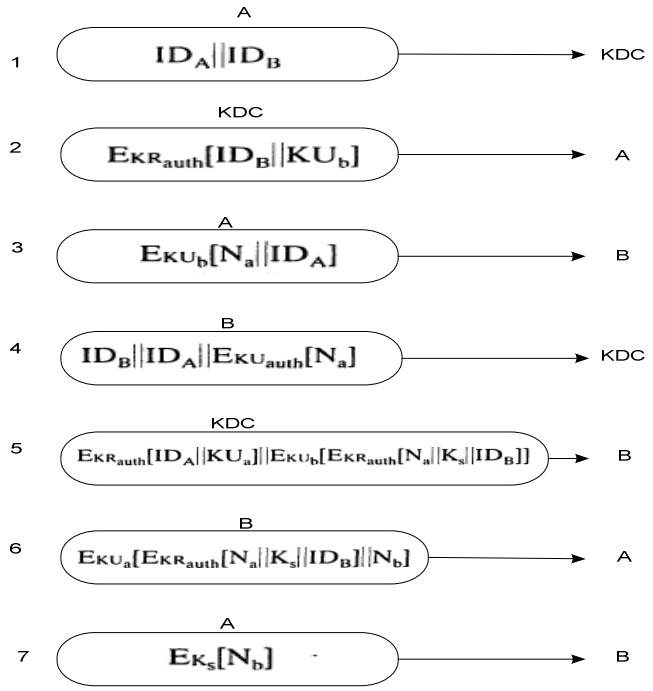
Üçüncü adımda A tarafı B tarafına kendisine AS'den gelen mesaja ek olarak B tarafının açık anahtarı ile şifrelediği oturum anahtarı bilgisini de ekleyerek gönderir. Protokolde AS sadece açık anahtar bilgisini sağlamaktadır, oturum anahtarı ise sadece A tarafından oluşturulup şifrelenerek B tarafına gönderilmektedir. Kullanılan zaman damgaları, muhtemel tekrarlama ataklarına karşı bir önlem olarak düşünülmüştür. Oldukça kullanışlı ve basit olan bu protokolün dezavantajı zaman senkronizasyonu gerektirmesidir.

Şekil 4.6 da görülen, *Woo ve Lam* [11] tarafından geliştirilen protokolde zaman damgaları yerine benzersiz sözcükler kullanılmaktadır.

Protokolün birinci adımında A tarafı KDC'ye kimlik bilgileri ile başvuruda bulunarak B tarafı ile haberleşme isteğini iletmektedir.

İkinci adımda KDC A tarafına B'nin kimlik ve açık anahtar bilgisini şifreli bir biçimde gönderir.

Üçüncü adımda A tarafı B tarafına kendi ürettiği benzersiz sözcük değerini ve kimlik bilgisini B'nin açık anahtarı ile şifreli bir biçimde gönderir.



Şekil 4-6 Asimetrik Şifreleme Yöntemli Doğrulama ve Anahtar Değişimi -2

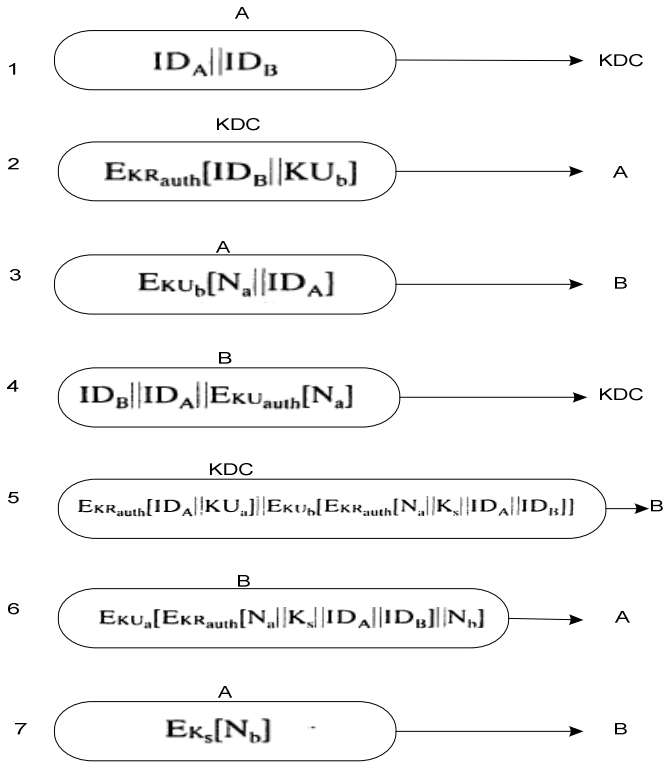
Dördüncü adımda B tarafı KDC'ye gönderdiği mesaj ile oturum anahtarı ve A tarafının açık anahtar bilgilerini ister. KDC gelen mesajdaki benzersiz sözcük bilgisini kontrol ederek mesajın gerçekten güvenilir bir kaynaktan geldiğini doğrular.

Beşinci adımca KDC B tarafında A'nın açık anahtar bilgisi ile beraber benzersiz sözcük, oturum anahtarı ve B'ye ait kimlik bilgilerini kendi gizli anahtarı ile şifrelenmiş bir biçimde gönderir. Bu mesaj sayesinde B tarafı oturum anahtarını elde eder, kimlik ve sözcük bilgilerini kontrol ederek mesajın gerçekten KDC tarafından gönderildiğini doğrular. Mesajın içindeki oturum anahtarının B'nin açık anahtarı ile şifrelenmiş olması, mesajın bu kısmının sadece B'nin gizli anahtarı ile deşifre edilebileceğini garantilemektedir.

Altıncı adımda B tarafı kendine gelen mesajda yer alan KDC'nin gizli anahtarı ile şifrelenmiş biçimdeki A tarafına ait sözcük, oturum anahtarı ve kendisine ait kimlik bilgilerine, kendi ürettiği benzersiz sözcük değerini de ekleyerek A'nın açık anahtarı ile şifreli bir biçimde A tarafına gönderir. A tarafı bu mesajdan oturum

anahtarını elde eder, benzersiz sözcük bilgilerini kontrol ederek mesajın gerçekten B tarafından gönderildiğini doğrular.

Yedinci ve son adımda ise A tarafı B'nin ürettiği benzersiz sözcük değerini oturum anahtarı ile şifreleyerek B'ye gönderir. B tarafı bu mesaj sayesinde A tarafının oturum anahtarını başarılı bir şekilde elde ettiğini doğrulamış olur.
Şekil 4.7 de yukarıdaki protokolün daha da güvenli hale getirilmiş çeşidi gözlenmektedir.



Şekil 4-7 Asimetrik Şifreleme Yöntemli Doğrulama ve Anahtar Değişimi -2

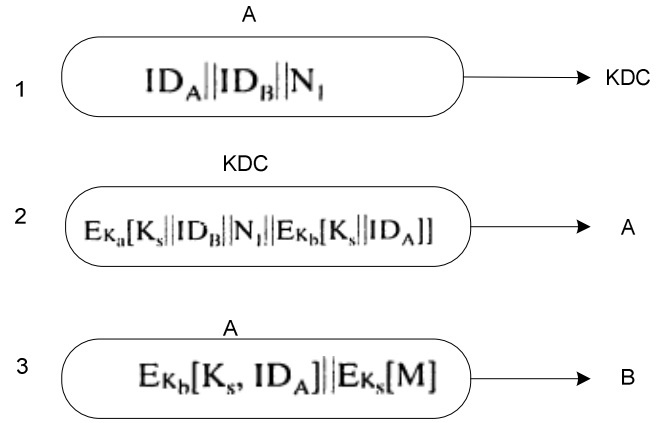
Beşinci adımdaki bütün mesaj bloklarına A tarafının kimlik bilgisi eklenerek oturum anahtarının bütünüyle tarafların kimlikleri ile ilişkilendirilmesi sağlanmış. Altıncı adımdaki mesaja A tarafının kimlik bilgisinin eklenmesiyle sağlanan, A tarafına ait kimlik ve benzersiz sözcük bilgisi ikilisi ile ise, bağlantı isteğinin sadece ve sadece A tarafından geldiği benzersiz olarak garanti altına alınmıştır.

4.1.2 Tek Yönlü Doğrulama

Doğrulama protokolleri arasında oldukça popüler olan tek yönlü doğrulama protokolünün sık kullanılan örneği elektronik postadır. Gönderici ve alıcının hat üzerinde aynı anda bulunmasını gerektirmeyen uygulamada, gönderilen mesaj alıcı mesaja erişinceye kadar saklanmaktadır. Gönderilen mesajın başlığı, SMTP yada X.400 gibi posta protokolleri ile doğru alıcıya yönlendirilebilmesi için belirgin ve açık olmalıdır. Bunun yanında, güvenlik açısından mesaj içeriği şifreli olmalı ve posta protokolünün deşifre edici anahtara sahip olmaması gerekmektedir.
Protokolün ikinci gereksinimi ise doğrulamadır. Alıcı, mesajın gerçekten kaynakta belirtilen gerçek gönderici tarafından gönderildiğine emin olmalıdır.

4.1.2.1 Geleneksel Şifreleme Yöntemi

Şekil 4.8 de gösterilmekte olan yöntemde, gönderici KDC'ye alıcıya mesaj göndermek için oturum anahtarı isteminde bulunmakta ve ancak anahtar oluşturulup kendisine gönderildikten sonra mesaj gönderme işlemine başlayabilmektedir.



Şekil 4-8 Geleneksel Şifreleme Yöntemli Tek Yönlü Doğrulama

Tek yönlü doğrulama protokolü için uygun olmayan bu yöntemde, sadece arzu edilen alıcı yada alıcıların mesajı okuyabileceği ve mesajın gerçekten güvenilir bir kaynaktan gönderildiği garanti altına alınmaktadır. Ancak yöntem tekrarlama

ataklarına karşı güvenilir değildir. Yöntemi tekrarlama ataklarına karşı güvenilir kılmak için protokol adımlarına karşılıklı doğrulama protokolü yöntemlerinde olduğu gibi zaman damgası ekleme basamakları eklenebilir ancak, elektronik posta gibi bir uygulamada gönderici ve alıcı arasında gerçekleşebilecek büyük zaman farkları göz önünde bulundurulduğunda zaman damgalarının kullanılması elverişli olmamaktadır.

4.1.2.2 Açık Anahtar Şifreleme Yöntemi

Protokol göndericinin, alıcının açık anahtar bilgisine sahip olmasını (gizlilik), alıcının göndericinin açık anahtar bilgisine sahip olmasını (doğrulama) yada her iki tarafında aynı anda bahsedilen bilgilere sahip olmasını gerektirmektedir.
Eğer gizlilik birinci öncelikli ise A ve B tarafları arasında aşağıdaki gibi bir mesajın gönderilmesi tercih edilmelidir;

$$A \longrightarrow B : E_{KU_b}[K_S] \| E_{K_S}[M]$$

Bu durumda mesaj tek kullanımlık bir gizli anahtar ile şifrelenmektedir. A tarafı aynı zamanda bu tek kullanımlık gizli anahtarı B tarafının açık anahtarı ile de şifreleyebilir. Sadece B tarafı kendi gizli anahtarı kullanarak tek kullanımlık anahtarı içeren mesajı ve bu anahtarı kullanarak asıl posta mesajını deşifre edebilir. Bu yöntem, basitçe bütün mesajı B'nin açık anahtarı ile şifrelemekten çok daha güvenli bir yöntemdir.

Taraflar arasındaki haberleşmede, doğrulama birinci öncelikli ise aşağıdaki mesajda olduğu gibi elektronik imza yöntemi kullanılmalıdır.

$$A \longrightarrow B : M \| E_{KR_a}[H(M)]$$

Bu yöntem A tarafının, sonradan mesajı gönderdiğini inkar etmesini engellemektedir, ancak mesaj sadece imzalı olarak gönderildiğinden hattı dinleyen ve A tarafının açık anahtarına sahip bir saldırganın mesajın imzasını

çözebilir, mesajı değiştirebilir, hatta mesaja kendi imzasını dahil ederek alıcı tarafa mesajı tekrar gönderebilir. Bu durumu engellemek için aşağıda olduğu gibi mesajın ve imzanın alıcının açık anahtarı ile şifrelendiği bir yöntem tercih edilmelidir.

$$A \longrightarrow B : E_{KU_b}[M \| E_{KR_a}[H(M)]]$$

Yukarıda bahsedilen bu iki yöntem B tarafının A'nın, açık anahtar ve bu anahtarın geçerlilik süresi bilgisine sahip olmasını gerektirmektedir. Daha kullanışlı bir yöntem olarak taraflar arasındaki haberleşmede aşağıda görüldüğü gibi elektronik sertifika kullanılabilir.

$$A \rightarrow B : M \| E_{KR_a}[H(M)] \| E_{KR_{as}}[T \| ID_A \| KU_A]$$

Bu yöntemde A tarafı B tarafına mesaja ek olarak kendi gizli anahtarı ile şifreleyerek imza bilgisini, doğrulama sunucusunun gizli anahtarı ile şifreleyerek sertifika bilgisini B tarafına gönderir. B tarafı ilk olarak sertifika bilgisini kullanarak A'nın açık anahtar bilgisini elde eder ve A tarafının güvenilirliğini, daha sonrada elde ettiği açık anahtar bilgisini kullanarak mesajın imzasını doğrular. Alternatif olarak eğer gizlilik de önemli bir faktör ise yukarıdaki mesajın tamamı B tarafının açık anahtarı kullanarak şifrelenebilir.

4.2 Modern Anahtar Değişim Yöntemleri

Güvenli olmayan bir ağ ortamı üzerinden haberleşen taraflar, birbirlerini doğrulamak ve aralarında gerçekleşecek alt iletişim oturumlarını güven altına almak için büyük oturum anahtarları üzerinde anlaşmak isterler.

Parola doğrulamalı anahtar değişim problemi olarak tanımlanan bu işlemlerde taraflardan biri istemci diğeri de sunucu ise problem uzak kullanıcı erişimi alanında değerlendirilir.

Uzaktan kullanıcı erişimi üzerine, kriptografik olarak güvenli anahtarların kullanılması temeline dayanan, anahtar yönetimi, açık anahtar altyapısı yada güvenli donanım gibi çeşitli çözümler sunulmuştur. Bu çözümlerin parola tabanlı

olanlarından telnet, parola bilgilerini taraflar arasında açık olarak iletmesi, Kerberos sistemi ise çevrim dışı sözlük ataklarına karşı zayıf yönleri olması nedeni ile güvenli olmayan yöntemler olarak bilinmektedir.

Modern parola doğrulamalı anahtar değişimi yöntemleri, temel olarak taraflar arasında paylaşılan parola düşük güvenilirlikli bir parola olsa dahi, tarafların büyük oturum anahtarlarını güvenli bir şekilde paylaşabilmelerini sağlamayı amaçlamaktadır.

Bahsedilen bu yöntemler, temel olarak aşağıda sıralanan gereksinimleri karşılamak üzere tasarlanmışlardır [13];

1) Küçük parolalara karşı yapılabilecek çevrim dışı sözlük ataklarını engelleyebilmelidir.
2) Çevrim içi sözlük ataklarına maruz kalsa da doğrulama ve anahtar değişim işlemini başarılı bir şekilde sonlandırabilmelidir.

Parolalar, fırsat verildiğinde sözlük ataklarına karşı hassas olabilmektedirler. Parola doğrulamalı anahtar değişim yöntemlerinde bu fırsatların ortadan kaldırılması amaçlanmaktadır. Erişim hataları oluşturmaları nedeni ile çevrim içi sözlük atakları basit bir şekilde algılanabilir. Çevrim dışı sözlük saldırıları ise algılanması oldukça zor olan saldırılardır ve kendini yetkili bir tarafmış gibi gösteren yada yetkili taraflar arasındaki mesaj trafiğini dinleyen bir saldırgan tarafından gerçekleştirilebilir.

3) Karşılıklı doğrulama sağlayabilmelidir.

Karşılıklı doğrulama tercih edilen bir özelliktir ve haberleşen taraflardan birinin, diğer tarafın parola değerine sahip olduğuna emin olmasını sağlamaktadır.

4) Anahtar değişimi ile entegre olmalıdır.

Parola doğrulamalı anahtar değişimi yöntemlerinde nihai olarak hedeflenen taraflar arasında, gerçekleştirilecek alt haberleşme oturumlarını güven altına alacak, yüksek boyutlu bir oturum anahtarının değiştirilmesidir.

4.2.1 Shamir Üç Geçiş Anahtar Değişimi Protokolü

Adi Shamir tarafından geliştirilen Üç Geçiş Anahtar Değişimi (Three-Pass Key Exchange) protokolü[14] çok sınırlı bir güvenlik anlayışı olması nedeniyle asla akademik olarak yayınlanmamıştır.

Protokol, Değiştirici Şifre (Commutative Cipher) temeline dayanmaktadır. Değiştirici Şifre, şifreleme ve şifre çözme fonksiyonlarının birbirlerinin yerini alabildiği hatta aynı fonksiyon olabildiği şifredir.

Örneğin; $A \times B \times C = A \times C \times B = C \times B \times A$ eşitliğini gerçekleyen XOR fonksiyonu bir Değiştirici Şifre fonksiyonudur.

A ve B tarafları arasındaki anahtar değişim protokolü adımları aşağıda verilmektedir.

- A tarafı B tarafı ile haberleşmesinde kullanacağı değişken oturum anahtarı K_{A-B} yi oluşturur ve diğer bir değişken anahtar K_A ile şifreleyerek B tarafına gönderir.

$$EK_A(K_{A-B})$$

- B tarafı aldığı mesajı kendi değişken anahtarı K_B ile şifreler ve A tarafına gönderir.

$$EK_B(EK_A(K_{A-B}))$$

- A tarafı mesajı kendi anahtarı ile deşifre eder ve B tarafına gönderir.

$$DK_A(EK_B(EK_A(K_{A-B})))$$

- B tarafı mesajı kendi anahtarı ile deşifre eder ve oturum anahtarını elde etmiş olur.

$$DK_B(DK_A(EK_B(EK_A(K_{A-B})))) = K_{A-B}$$

Güvenlik açısından oldukça zayıf olan bu protokolde A ile B tarafı arasında gidip gelen mesajları dinleyen bir saldırgan şifreleme algoritmasını bularak iletişimin güvenliğini rahatlıkla bozabilir.

Örneğin şifreleme fonksiyonunun XOR fonksiyonu olduğu kabul edilirse, birinci ve ikinci mesajın XOR fonksiyonuna tabi tutulması sonucu elde edilen çıktının, üçüncü mesaj ile tekrar XOR işlemine tabi tutulması sonucu ortam anahtarı elde edilmiş olacaktır.

4.2.2 Şifreli Anahtar Değişimi Protokolü (EKE)

Güvensiz bir ağ üzerinden haberleşecek A ve B taraflarının gizli bir P parola değerini paylaştıkları klasik bir anahtar değişimi ortamı düşünelim; güvenli oturum anahtarı değişimini gerçekleştirebilmek için A değişken anahtar R'yi üretir, bunu P anahtarı ile şifreler ve P(R) değerini B'ye gönderir.

Bu andan itibaren A ve B, R değerini paylaşmaktadır ve R değeri oturum anahtarı olarak kullanılabilir. B tarafı A'ya R(Terminal Tipi) şeklinde bir bilgi göndererek anahtar değişiminin başarılı olduğunu açıklayabilir.

Hattı dinleyen saldırgan A ve B arasında gidip gelen mesajları kayıt ederek P değerini elde etmek için gerçekleştireceği sözlük ataklarıyla bir P' parola değeri elde edebilir ve oturum anahtarını $R'=P'^{-1}(P(R))$ olacak şekilde elde edebilir.

Bellovin ve Merrit [15] tarafından geliştirilen EKE protokolü ile, yukarıda anlatılmakta olan, kullanıcı tarafından paylaşılan anahtar ile şifrelenmiş oturum anahtarının güvenli olmayan ağ üzerinden iletildiği klasik kriptografi tekniklerinin yerine, asimetrik ve simetrik kriptografi tekniklerinin bir kombinasyonu olan ve haberleşen iki tarafın ortak olarak paylaştıkları bir şifre sayesinde, güvenli olmayan ağ ortamında bilgi iletimini güvenli bir şekilde gerçekleştirmelerine olanak sağlayan bir yöntem ortaya konmuştur.

Klasik kriptografi tekniklerinin sahip olduğu bazı güvenlik açıklarını ortadan kaldıran ve günümüzde de halen kullanım alanı bulunan EKE protokolü, benzer birçok anahtar değişimi protokolünün geliştirilmesinde de temel ve ortak nokta olarak rol oynamıştır.

A ile B tarafları arasında EKE protokolü kullanılarak gerçekleştirilen anahtar değişimi aşağıda adım adım anlatılmaktadır.

- A değişken R_A sayısını üretir ve $P(\alpha^{R_A} (\mathrm{mod}\, \beta))$ değerini, simetrik şifreleme algoritması ve paylaşılan P parolasını kullanarak hesaplar, {A, $P(\alpha^{R_A} (\mathrm{mod}\, \beta))$} mesajını B tarafına iletir. Burada A anahtar değişimini başlatan tarafın açık ismidir.

- B değişken R_B sayısını üretir ve $\alpha^{R_B} (\mathrm{mod}\, \beta)$ değerini hesaplar. A tarafı ile paylaştıkları P parolasını kullanarak gelen mesajı deşifre eder ve $(\alpha^{R_A R_B})(\mathrm{mod}\, \beta)$ değerini hesaplayarak oturum anahtarı K'yı elde eder. Son olarak değişken zorluk değeri $zorluk_B$'yi hesaplar ve A tarafına $P(\alpha^{R_B} (\mathrm{mod}\, \beta)), K(zorluk_B)$ mesajını iletir. Bahsedilen zorluk değeri hakkında açıklama 4.1.1 bölümündeki Zorluk/Karşılık tanımı altında açıklanmaktadır.

- A paylaşılan P parolasını kullanarak B tarafından gelen mesajın $P(\alpha^{R_B} (\mathrm{mod}\, \beta))$ kısmını deşifre eder, $(\alpha^{R_A R_B})(\mathrm{mod}\, \beta)$ değerini hesaplayarak oturum anahtarı K'yı elde eder ve bunu kullanarak $K(zorluk_B)$ değerini deşifre eder. Son olarak kendi zorluk değeri $zorluk_A$'yı üretir ve B tarafına $K(zorluk_A, zorluk_B)$ mesajını iletir.

- B gelen mesajı deşifre ederek $zorluk_B$ değerinin kendi ürettiği orijinal değer olduğunun doğrular ve A tarafına $K(zorluk_A)$ mesajını iletir.

- A gelen mesajı deşifre ederek $zorluk_A$ değerinin kendi ürettiği orijinal değer olup olmadığını kontrol eder.

Yukarıda açıklanan adımların ilk üçünde gerekli oturum anahtarı değişimi taraflar arasında gerçekleşmektedir, diğer adımlar ise anahtar değişiminin doğru taraflar arasında başarılı bir şekilde gerçekleştiğinin kontrolü için gerçekleştirilmektedir.

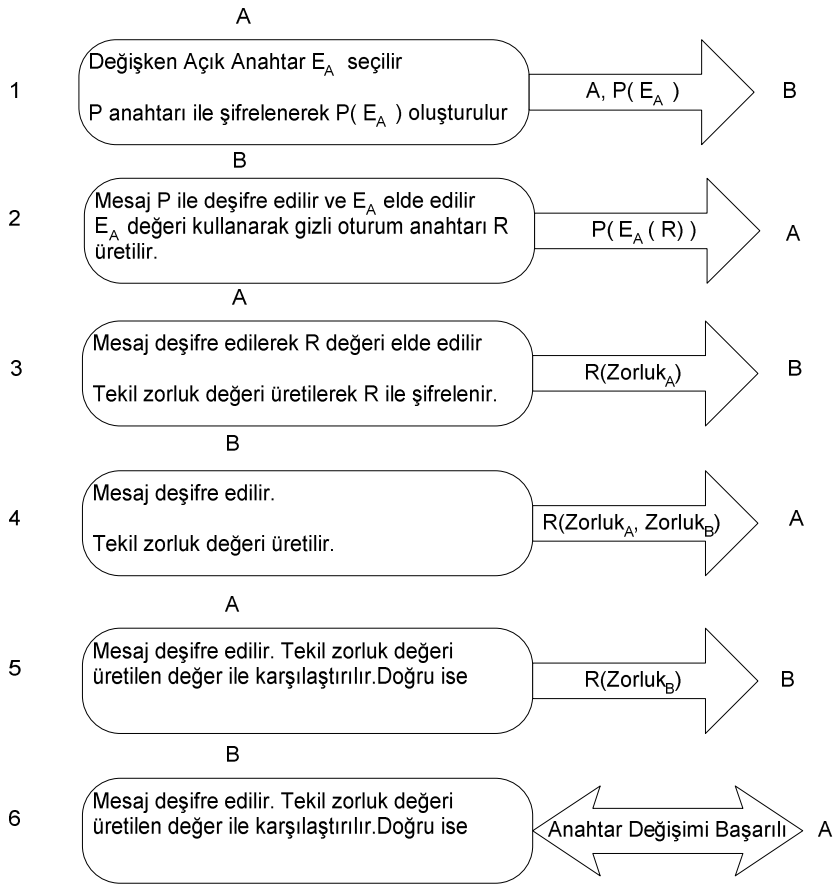
Bu adımlarda bazı simetrik şifreleme bölümleri ihmal edilebilir. Örneğin; birinci adımda B tarafına gönderilen mesaj yerine {A, $\alpha^{R_A} (\mathrm{mod}\, \beta)$} mesajı kullanılabilir. Bu durumda hattı dinleyen saldırganın gerçekleştireceği aktif bir atak başarılı olsa dahi R_A değeri hakkında bir bilgiye sahip olmadan oturum anahtarı K'yı hesaplayamayacak dolayısıyla ikinci adımda A tarafına göndermesi gereken mesajı anlamlı bir şekilde oluşturamayacaktır.

Şekil 4.9 da görülen EKE gösteriminde kripto analist tarafından bazı oturum anahtarlarını(R) elde edildiğini düşünelim, bu durum P parolasına karşı yapılacak bir atak için basamak oluşturacak ve E_A değerine karşı direk olarak bir kripto analitik atağı mümkün kılacaktır.
Alternatif olarak iletilen $E_A(R)$ değerinin $P^{-1}(P(E_A)))(R)$ değeri ile karşılaştırılması ile P olduğu tahmin edilen P' değerlerinin elde edilmesi de mümkün olacaktır. Anahtar değişimi adımlarında yapılacak ufak değişikliklerle bu atakları engellemek mümkün olabilir.

Zorluk değeri üretme ve bununla cevap verme esnasında A ve B taraflarının S_A ve S_B değişken alt anahtarlarını oluşturduklarını ve bunlarında R ile şifrelenerek iletildiğini kabul edelim; bu durumda anahtar değişiminin üçüncü adımında kullanılan mesaj $R(Zorluk_A, S_A)$, dördüncü adımda kullanılan mesaj ise $R(Zorluk_A,Zorluk_B,S_B)$ olacak şekilde değişikliğe uğrayacaktır. Böylece iki tarafta gerçek oturum anahtarını uygun bir f fonksiyonu ile $S=f(S_A,S_B)$ olacak şekilde hesaplayabilir ve bu anahtar diğer bütün alt anahtar değişimlerinde kullanılabilir.

Bu kullanım sayesinde R'nin anahtar değişimindeki rolü azalmaktadır. P hiçbir zaman direk olarak S ile ilgili hiçbir değeri şifrelemekte kullanılmadığından S değeri P ile ilgili hiçbir ipucu vermemektedir. Ayrıca R sadece değişken veriyi şifrelemekte kullanıldığından elde edilmesi için gerçekleştirilecek atakların fazla bir

önemi kalmamaktadır. Diğer bir önemli nokta ise S değerinin hiçbir mesaj içerisinde yer almamasıdır.



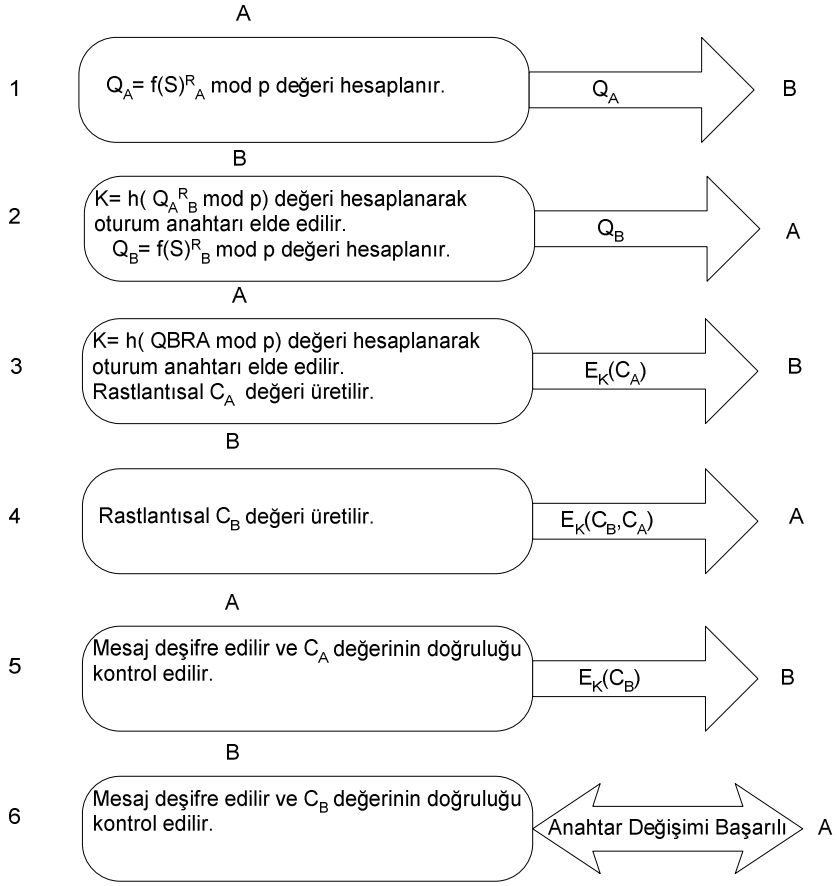
Şekil 4-9 EKE Anahtar Değişimi

4.2.3 Güçlü Sadece Parola Doğrulamalı Anahtar Değişimi(SPEKE)

SPEKE anahtar değişimi protokolü [13] ile akılda kalabilecek kadar küçük boyutlu parolalar kullanılarak güvenli bir anahtar değişimi gerçekleştirmek amaçlanmıştır. Kullanılan parola, anahtar değişiminden bağımsız bir faktör olarak kabul edilmiştir. Diğer anahtar değişim protokollerinde yer alan ortak paylaşılan parola kullanılarak simetrik, asimetrik şifreleme ve ek mesajların gönderilmesi gibi işlemler SPEKE protokolü adımlarında yer almamaktadır.

Şekil 4.10 da görülen SPEKE şematik gösteriminde, diğer anahtar değişim protokollerinde yer alan DH tabanlı asal sayılarla gerçekleştirilen işlemler yerine bir f fonksiyonu ve taraflar arasında paylaşılan küçük boyutlu parola kullanılarak DH tabanlı büyük asal sayılar oluşturulmaktadır.

Pratik de f(S) fonksiyonu için $f(S) = g_q^{\ S} \bmod p$ ve $f(s) = S^{(p-1)/q} \bmod p$ fonksiyonları kullanılmaktadır.



Şekil 4-10 SPEKE Anahtar Değişimi

4.2.4 Geliştirilmiş Şifreli Anahtar Değişimi (A-EKE)

EKE protokolü temel alınarak Bellovin ve Merrit [16] tarafından geliştirilen bu protokol ile taraflar arasında paylaşılan ortak parolanın taraflar arasında açık bir

şekilde (clear text) kullanılmasının engellenmesi amaçlanmıştır. Haberleşen taraflar bir *H(P)* tek yönlü hash fonksiyonu ile, kullandıkları parolayı şifreleyerek kullanmakta, ayrıca anahtar değişimi sırasında farklı bir tek yönlü fonksiyon ile parolaya ve oturum anahtarına bağımlı ek bir mesaj göndermektedir. *H(P)*, gönderilen ek mesaj ve oturum anahtarının hep birlikte doğrulanması ile taraflar arasındaki anahtar değişimi oturumu geçerli sayılmaktadır.

Başlangıçta da bahsedildiği gibi A-EKE protokolünde parola hash fonksiyonu ile işleme tabi tutularak sunucular üzerinde gizli tutulmaktadır. İletişim hattını dinleyen saldırgan *H(P)* değeri hakkında bilgi sahibi olmadan P değerlerini tahmin edebilmek için sözlük saldırıları gerçekleştiremeyecektir.

Protokolde yer alan ikinci tek yönlü fonksiyon ise P parolası ve bir önceki oturumda taraflar arasında başarılı bir şekilde paylaşılmış olan K oturum anahtarına bağımlı *F(P, K)* fonksiyonudur. Kullanıcı bu değeri üreterek K oturum anahtarı ile şifreler ve sunucuya *K[F(P,K)]* değerini gönderir. Sunucu *H(P)*, *F(P,K)* ve K değerlerini doğruladığı takdirde anahtar değişimi başarılı bir şekilde sonlanmış demektir. Sunucu bu değerleri doğrulamak için *T(H(P), F(P,K),K)* fonksiyonunun doğru sonucunu üretmesini beklemektedir. (*T(X,Y,Z)* fonksiyonu sadece *X=H(P') Y=F(P',Z)* değerlerini gerçekleyen P' değerleri için doğru sonucunu üretir.)

Anahtar değişimi sonucunda taraflar arasında bilinir hale gelen oturum anahtarı K'yı elde etmek için öncelikle *H(P)* bilgisine sahip olunması gereklidir. Çok düşük bir ihtimalde olsa *H(P)* değerini elde eden saldırgan kendini, sunucuya istemci tarafı yada istemciye sunucu tarafı olarak tanıtabilir ancak karşı tarafı buna inandırabilmek için aynı zamanda *F(P,K)* değerini oluşturarak göndermesi gerekmektedir.

A ile B tarafları arasında A-EKE protokolü kullanılarak gerçekleştirilen anahtar değişimi aşağıda adım adım anlatılmaktadır.

- A tarafı R_A değişken sayı değerini seçer ve A, $H(P)\left[\alpha^{R_A} (\mathrm{mod}\, \beta)\right]$ değerini B tarafına gönderir. B tarafı P parola değerine sahip olduğundan *H(P)* değerini kolayca hesaplayabilir.

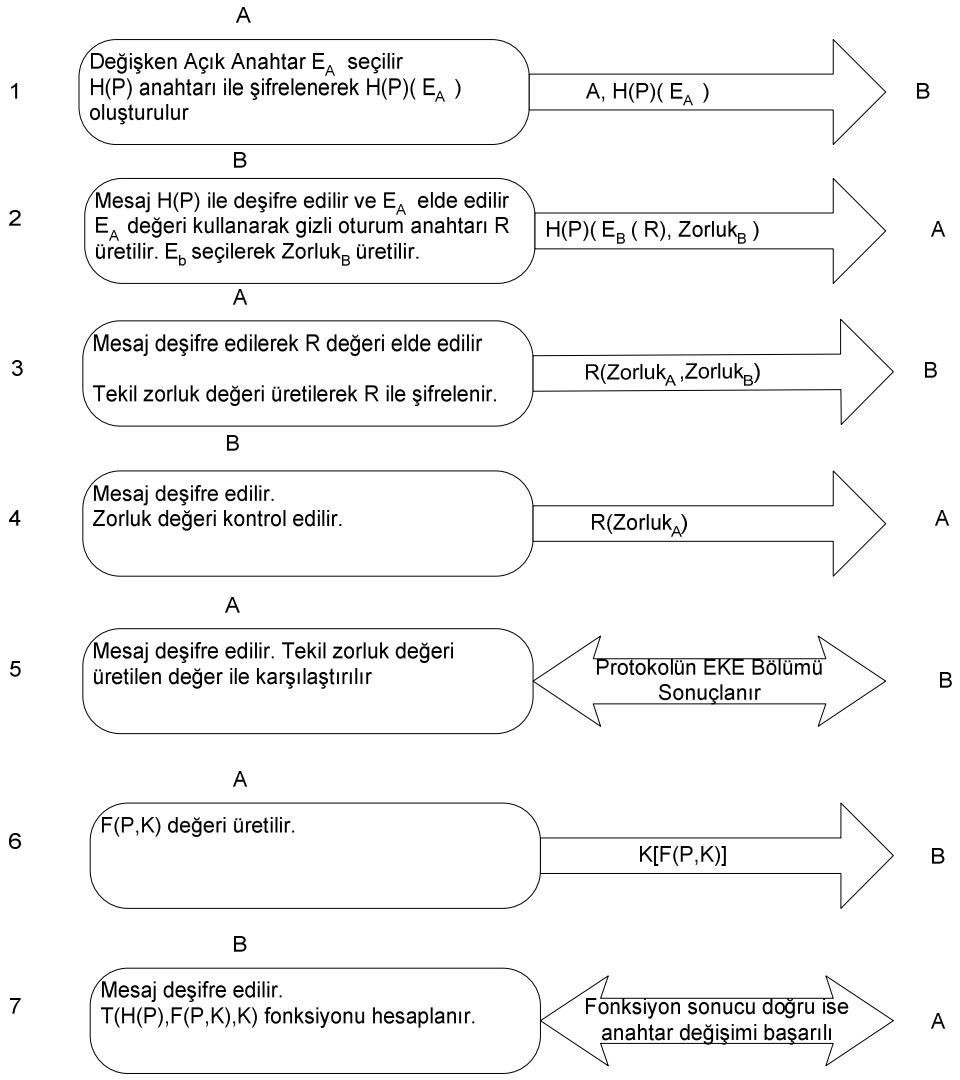
- B tarafı R_B değişken sayı değerini seçer ve gizli *H(P)* değerini kullanarak gelen mesajı deşifre eder. $(\alpha^{R_A R_B}) \bmod \beta$ değerini hesaplayarak K oturum anahtarını elde eder ve $Zorluk_B$ değerini hesaplayarak $H(P)\left[\alpha^{R_B} (\mathrm{mod}\, \beta)\right], K[Zorluk_B]$ mesajını A tarafına gönderir.

- A tarafı *H(P)* gizli değerini kullanarak mesajın $H(P)\left[\alpha^{R_B} (\mathrm{mod}\, \beta)\right]$ kısmını deşifre ederek K oturum anahtarını hesaplar. K anahtarını kullanarak $K[Zorluk_B]$ değerini deşifre eder. Son olarak $Zorluk_A$ değerini üreterek B tarafına $K[Zorluk_A, Zorluk_B]$ mesajını gönderir.

- B tarafı gelen mesajı deşifre ederek $Zorluk_B$ değerinin doğruluğunu kontrol eder. A tarafına $K[Zorluk_A]$ mesajını iletir.
- A tarafı gelen mesajı deşifre ederek $Zorluk_A$ değerinin doğruluğunu kontrol eder.
- A tarafı *K[F(P,K)]* mesajını üreterek B tarafına gönderir.
- B tarafı gelen mesajı deşifre ederek *F(P,K)* değerini elde eder. *T(H(P), F(P,K) , K)* fonksiyonunun doğru sonucunu ürettiğini kontrol eder.

Şekil 4.11 de blok diyagramı verilen A-EKE protokolünde ilk beş adım P parolasının açık olarak kullanılmasının yerine *H(P)* değerinin kullanılması dışında tamamen EKE protokolü ile aynıdır. Altıncı ve yedinci adımların eklenmesi ile A-EKE protokolü tanımlanmış olmaktadır.

A-EKE anahtar değişimi protokolünde *H(P)* hakkında bilgi edinebilen saldırgan kendini A tarafına B tarafı gibi gösterebilir ancak P parolasına sahip olmadan B tarafına kendini A tarafı gibi göstermesi imkansızdır.



Şekil 4-11 A-EKE Anahtar Değişimi

A-EKE protokolü gereksinimlerini karşılayacak farklı H(), F() ve T() fonksiyonu seçenekleri mevcuttur. Elektronik imza ve değiştirici tek yön fonksiyonları örnek olarak verilebilecek kullanışlı iki yöntemdir.

Elektronik imza yönteminde, protokolde simetrik şifreleme için kullanılan *H(P)* tek yönlü fonksiyonu yerine, P parola değerinden üretilmiş V_P açık anahtar değeri kullanılır. Altıncı adımda hesaplanan *F(P,K)* değeri yerine, K oturum anahtarı değeri A tarafına ait P parola değerinden üretilmiş S_P gizli anahtarı ile imzalanarak B tarafına gönderilir. B tarafı açık anahtarını kullanarak imzayı ve oturum anahtarını doğrular.

A ile B tarafları arasında A-EKE protokolü ve elektronik imza yöntemi kullanılarak gerçekleştirilen anahtar değişimi aşağıda adım adım anlatılmaktadır.

- A tarafı R_A değişken sayı değerini seçer ve A, $V_P\left[\alpha^{R_A} (\mathrm{mod}\, \beta)\right]$ değerini B tarafına gönderir.

- B tarafı R_B değişken sayı değerini seçer ve paylaşılan açık anahtar V_P değerini kullanarak gelen mesajı deşifre eder. $(\alpha^{R_A R_B}) \bmod \beta$ değerini hesaplayarak K oturum anahtarını elde eder ve $Zorluk_B$ değerini hesaplayarak $V_P\left[\alpha^{R_B} (\mathrm{mod}\, \beta)\right], K[Zorluk_B]$ mesajını A tarafına gönderir.

- A tarafı V_P açık anahtar değerini kullanarak mesajın $V_P\left[\alpha^{R_B} (\mathrm{mod}\, \beta)\right]$ kısmını deşifre ederek K oturum anahtarını hesaplar. K anahtarını kullanarak $K[Zorluk_B]$ değerini deşifre eder. Son olarak $Zorluk_A$ değerini üreterek B tarafına $K[Zorluk_A, Zorluk_B]$ mesajını gönderir.

- B tarafı gelen mesajı deşifre ederek $Zorluk_B$ değerinin doğruluğunu kontrol eder. A tarafına $K[Zorluk_A]$ mesajını iletir.
- A tarafı gelen mesajı deşifre ederek $Zorluk_A$ değerinin doğruluğunu kontrol eder.
- A tarafı gizli anahtarı S_P ile K oturum anahtarı değerini imzalar ve K ile simetrik şifreleme işlemine tabi tutarak $K[S_P(K)]$ değerini B tarafına iletir.
- B tarafı gelen mesajı deşifre ederek $S_P(K)$ değerini elde eder ve bu değeri açık anahtar V_P değeri ile asimetrik şifreleme işlemine tabi tutarak imzayı doğrular. $(V_P (K^{-1}[K[S_P(K)]],K)$ sonucu doğru değerin üretmelidir.)

Diğer kullanışlı bir yöntem olan Değişmeli Hash Fonksiyonları (Commutative Hash Functions) yönteminde ise A-EKE protokolünün ilk beş adımı (EKE) aynen gerçekleştirilir. $H(P)$ değerinin, Değişmeli Hash Fonksiyonları Kümesinin $(\{H_0,H_1,\ldots\})$ bir üyesi olan $H_0(P)$ olarak tanımlandığı kabul edilirse;

- $H(P)=H_0(P)$ olacak şekilde EKE anahtar değişimi gerçekleştirilmesinin ardından A ve B tarafları $H_K()$ hash fonksiyonunu üretirler.

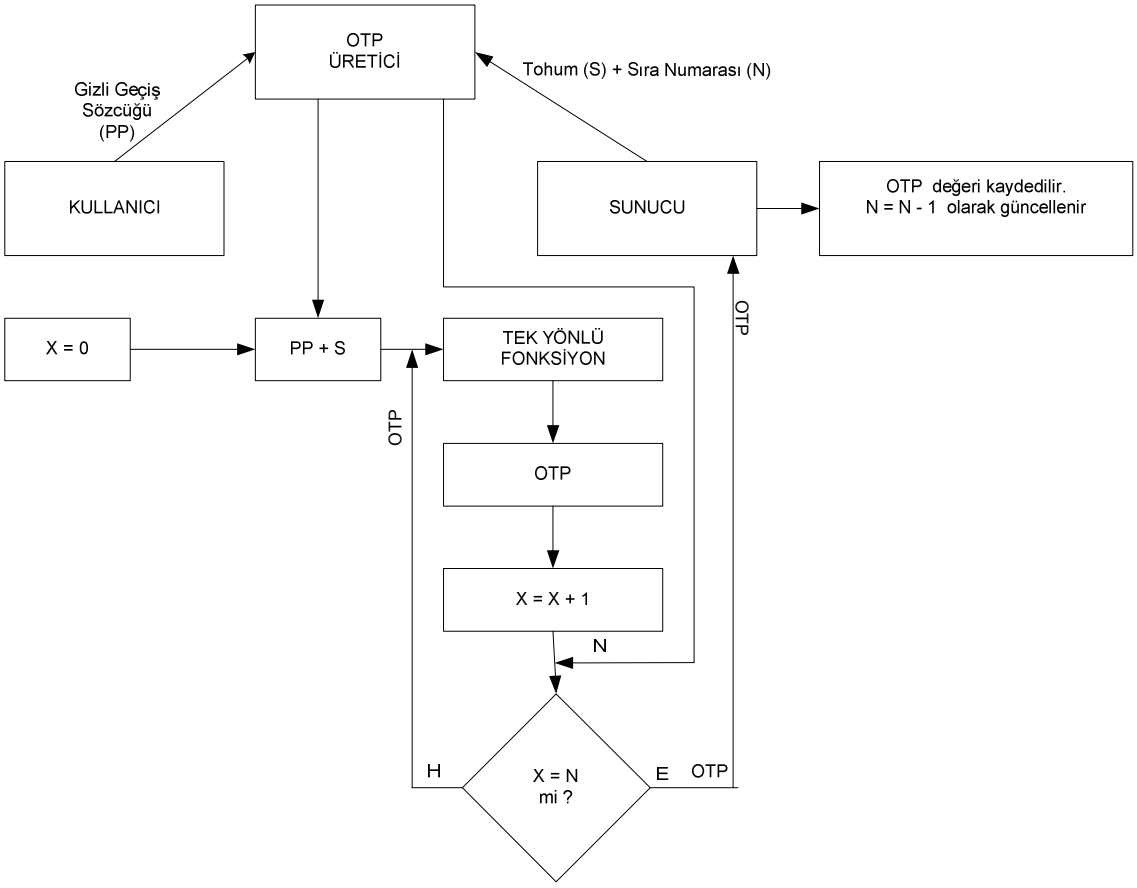
- A tarafı $F(H,K)= H_K(P)$ değerini üreterek B tarafına gönderir.

- B tarafı $H_0(P)$ değerine sahip olduğundan ilk olarak gelen mesajı H_0 ile hash işlemine tabi tutarak $H_0(H_K(P))$ değerini ve daha sonrada sakladığı $H_0(P)$ değerini $H_K()$ ile hash işlemine tabi tutarak $H_K(H_0(P))$ değerini hesaplar. Bu iki değer birbirine eşitse anahtar değişimi başarılı bir şekilde sonlandırılmış demektir.

4.2.5 Tek Kullanımlık Parola ve Tek Kullanımlık Parola Doğrulamalı Anahtar Değişimi

Güvenli olmayan ağ ortamı üzerinde, taraflar arasında gerçekleşen haberleşmeye yapılabilecek saldırılardan biri hattın saldırgan tarafından dinlenerek yetkili kullanıcılara ait kullanıcı adı ve parola bilgilerinin elde edilmeye çalışılmasıdır.
Bu bilgiler elde edildiği takdirde, daha sonra sistemlere erişim sağlama amaçlı kullanılabilir. Tek kullanımlık parola sistemleri, yukarıda bahsedilen tekrarlama ataklarına karşı koymak için tasarlanmışlardır.
Tek kullanımlık parola sisteminde etkileşimde bulunan iki taraf yer alır [17]. Üretici, kullanıcıdan gelen geçiş sözcüğü (pass-phrase) ve sunucudan gelen zorluk bilgilerini kullanarak uygun tek kullanımlık parola değerini üretir. Sunucu uygun parametre bilgilerini içeren zorluk değerini parola üretecine göndermekten, kendisine gönderilen tek kullanımlık parola değerini doğrulamaktan ve başarılı olarak doğruladığı son tek kullanımlık parola değeri ile ilgili parolanın sıra numarasını saklamaktan sorumludur. Sunucu sistemi aynı zamanda gerekli olduğu zamanlarda, kullanıcıya ait gizli geçiş sözcüğü değerini güvenilir bir yol ile değiştirmekten de sorumludur.
Şekil 4.12 de görülen tek kullanımlık parola değerinin ilklendirme işleminde üretici, kullanıcı ve sunucudan elde ettiği bilgileri birden çok kere tek yönlü bir fonksiyon ile işleme sokarak tek kullanımlık parola değerini üretir. Üretilen tek kullanımlık parola değeri, ilk parola değeri olması nedeni ile ilerleyen zamanlarda

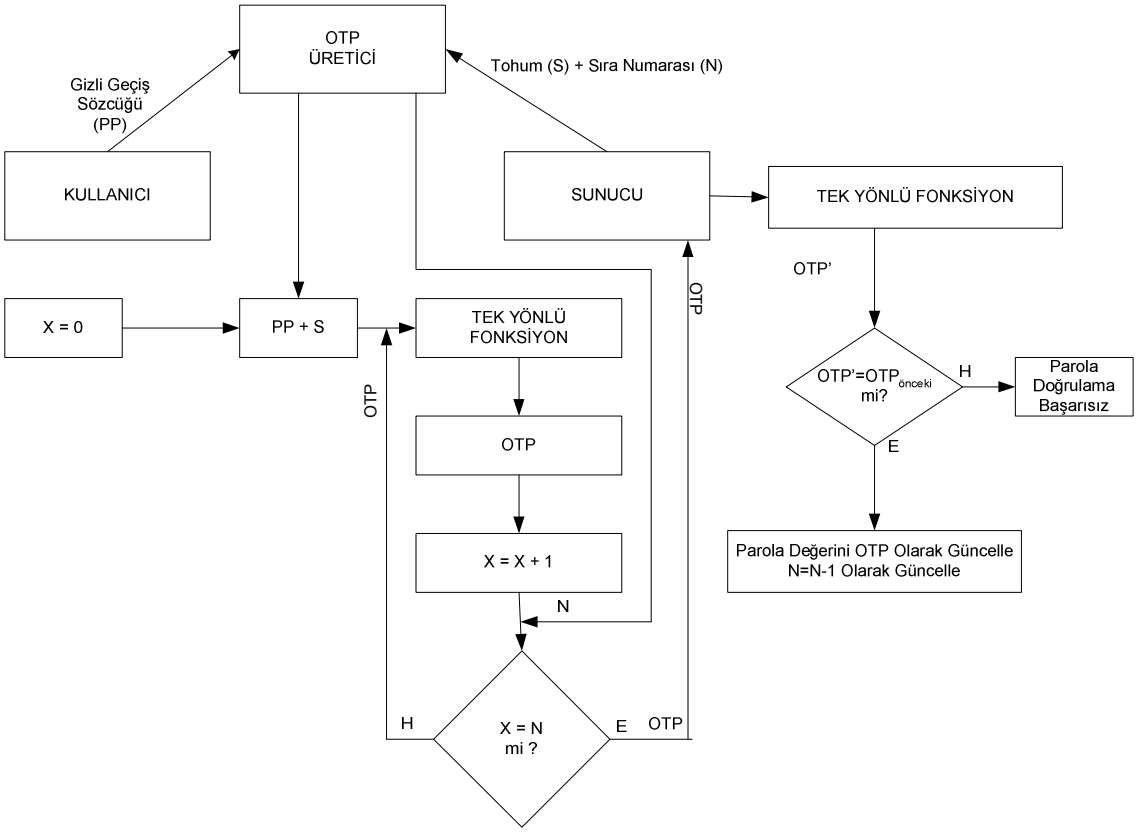
doğabilecek, parola değerini tekrar ilklendirme ihtiyaçlarına karşılık sunucu veri tabanında saklanır.



Şekil 4-12 Tek Kullanımlık Parola Senkronizasyonu

Şekil 4.13 de görülen tek kullanımlık parola değerinin sonraki oluşturulma ve kullanılma işlemlerinde ise, üretici ilklendirme işlemindekine benzer biçim parola değerini oluşturur ve sunucuya gönderir.

Sunucu, üreticiden gönderilen tek kullanımlık parola değerini sadece bir defa üreticideki ile aynı tek yönlü fonksiyonla işleme sokarak elde ettiği değeri, bir önceki doğrulama sonrası saklamış olduğu parola değeri ile karşılaştırır. Eğer değerler aynı ise doğrulama başarılıdır ve yeni tek kullanımlık parola ve parolaya ait sıra numarası değeri sonradan kullanılmak üzere sunucu veri tabanında güncellenir.



Şekil 4-13 Tek Kullanımlık Parola Kullanımı

Tek kullanımlık parola doğrulamalı anahtar değişimi işleminde ise, önceden bahsedilen anahtar değişim protokollerinde kullanılan ve haberleşen taraflar arasında paylaşılan, sabit parola değerinin yerine, tek kullanımlık parola değeri kullanılır.

4.2.6 Diffie-Hellman Temelli Tek kullanımlık Kimlik Doğrulamalı Anahtar Değişimi

Buraya kadar bahsedilen çeşitli anahtar değişim protokollerinin ortak ve nihai amacı, taraflar arasında gerçekleşen haberleşme ortamını güvenli kılmak için güvenli olmayan haberleşme kanalının şifrelenmesidir. Bir çoğu Diffie-Hellman anahtar değişim protokolünü temel alan bu yöntemler, taklit etme, tekrarlama gibi çeşitli saldırıları önleyebilseler de, günümüz gelişen internet ortamında en çok başvurulan saldırı tipi olan, servis yalanlama (DOS) ataklarını engelleyememektedirler.

Ayrıca, bahsedilen bütün protokollerde, haberleşmeye başlamadan önce tarafların birbirlerini tanıyabilmeleri için kimlik bilgileri ağ ortamında açık olarak karşı tarafa iletilmektedir. Bu durum günümüz internet uygulamalarında büyük önem kazanmaya başlayan, kullanıcı gizliliği ve izlenemezliği ile kaynak güvenliği gereksinimleri açısından problem teşkil etmektedir.

Protokoller de yer alan anahtar değişimi ve doğrulama için gerçekleştirilen karşılıklı işlem adımı (round) sayılarının oldukça fazla oluşu, internet uygulamalarındaki hız ve performans ihtiyacının önem kazanması ile birlikte bahsedilen anahtar değişim protokollerinin olumsuz taraflarını arttırmıştır.

Tek kullanımlık Kimlik Doğrulamalı Anahtar Değişimi protokolü, yukarıda bahsedilen problemleri ortadan kaldırmak için geliştirilmiş, Diffie-Hellman anahtar değişimi yöntemini temel alan, günümüz internet ortamına son derece uygun bir anahtar değişim yöntemidir.

Tek kullanımlık Kimlik kavramına dayanan anahtar değişim yöntemleri üzerine günümüze kadar birçok çalışma yapılmıştır. Geliştirilen yöntemler de, Tek kullanımlık Kimlik Üretmek için uzun süreli bir ortak gizli değerin (long-term secret) kullanılması ve dolayısıyla bu değerin saldırgan tarafından elde edilmesi sonucu, saldırganın bütün haberleşme oturumlarını ele geçirebilmesi ile farklı kullanıcıların aynı anda, aynı tek kullanımlık kimlik değerini üretmelerine karşılık bir kontrol mekanizmasının gerçekleştirilmemesi problemlerinin çözümü, DH temelli Tek kullanımlık Kimlik Doğrulamalı Anahtar Değişimi yöntemi ile gerçekleştirilmiştir.

Sistem de kullanılan kimlik kavramının temel özellikleri; saldırganın, hattı dinleyerek Tek kullanımlık Kimlik değerlerini elde etse de kimlerin haberleştiğini belirleyememesi ve Tek kullanımlık Kimlik değeri ile bu değerin hesaplanmasında kullanılan kök değerin sadece bir defaya mahsus olarak kullanılmasıdır.

Yöntemde doğrulama bilgileri, taraflar arasında gerçekleşen haberleşmenin ilk turunda gönderilmekte, ikinci turun sonunda ise, anahtar değişimi ve doğrulama işlemlerinin tamamı sonlandırılmaktadır.

Kullanıcılar ile sunucunun haberleşmesi esnasında, küçük bir ihtimal olsa da farklı taraflar aynı Tek kullanımlık Kimlik değerini oluştururlarsa, sunucu, doğrulama bilgileri içerisinde yer alan kullanıcıya ait sabit kimlik değerini kontrol ederek hangi Tek Kullanımlık Kimlik değerinin hangi kullanıcıya ait olduğunu belirleyebilir.

Diffie-Hellman temelli Tek kullanımlık Kimlik Doğrulamalı Anahtar değişimi [18] güvenli ve verimli bir iletişim ortamı için aşağıda sıralanan gereksinimleri karşılamayı hedefleyen bir anahtar değişim protokolüdür:

- **Bütünlük:** Saldırgan, taraflar arasındaki haberleşme devam ederken mesaj içeriğini değiştirememelidir.
- **Güvenilirlilik:** Saldırgan, şifrelenmiş bir mesajı deşifre edememelidir.
- **Doğrulama:** Kullanıcı, haberleştiği tarafın gerçekten haberleşmek istediği kişi olup olmadığını doğrulayabilmelidir.
- **Anahtar Doğrulama:** Kullanıcı, anahtar değişimi sonucunda, haberleştiği tarafın anahtar değerine sorunsuz bir şekilde sahip olduğunu doğrulayabilmelidir.
- **İleriye Dönük Mükemmel Güvenlik:** Uzun süreli gizli değer çalınsa bile, sona eren oturumların güvenliği garanti altına alınmalıdır.
- **Gizlilik:** Saldırgan, taraflar arasında transfer edilen mesajları dinlese de kimlerin haberleştiğini belirleyememelidir.
- **Verimlilik:** Protokol, mümkün olduğunca az sayıda işlem ve haberleşme turu ile yürütülebilmelidir.

Protokolde, diğer tek kullanımlık kimlik kullanan anahtar değişim yöntemlerindeki, tek kullanımlık kimliğin uzun süreli bir ortak gizli değerden üretilmesi işlemi yerine sinyal olarak adlandırılan ve hesaplandığı kök değer her başarılı haberleşme oturumu sonrasında değiştirilen bir tek kullanımlık kimlik üretme işlemi gerçekleştirilmektedir.

Sinyal değeri aşağıdaki gibi hesaplanmaktadır:

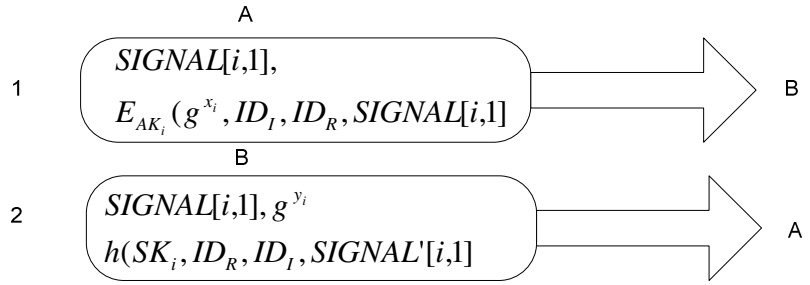
$$SIGNAL[i, j] = h(SS_i, j)$$

$$SS_i = h_1(K_{i-1})$$

Burada, *SIGNAL[i,,j]* *i.* oturumun *j.* turundaki sinyali, deiğer bir ifadeyle tek kullanımlık kimlik değerini belirtmektedir. SS_i, *i.* oturumdaki sinyal değerini hesaplamak için kullanılan kök değer, K_{i-1} ise *(i-1).* oturumdaki Diffie-Hellman anahtar değişimi sonrasında elde edilen gizli anahtar değeridir.

Yukarıda açıklanan hesaplama modeli sayesinde, ileriye dönük mükemmel bir güvenlik sistemi sağlanmış olmaktadır. Saldırgan herhangi bir oturumda Sinyal değerinin hesaplandığı kök değeri elde etse de, sadece o oturuma ait tek kullanımlık kimlik hakkında tahmin üretebilir.

Şekil 4.14 de görülen protokol adımları aşağıda maddeler halinde açıklanmaktadır:



Şekil 4-14 DH temelli Tek kullanımlık Kimlik Doğrulamalı Anahtar Değişimi

1) Kullanıcı rastsal bir x_i değeri üreterek g^{x_i} değerini hesaplar ve sunucu tarafına $SIGNAL[i,1]$ ve $E_{AK_i}\left(g^{x_i}, ID_K, ID_S, SIGNAL[i,1]\right)$ değerini gönderir.

2) Sunucu, gelen mesajdaki $SIGNAL[i,1]$ değerini kontrol ederek mesajın yetkili bir kullanıcıdan gelip gelmediğini öğrenir. Eğer mesaj yetkili bir kullanıcıdan gelmiyor ise anahtar değişimi oturumu iptal edilir.

Aksi takdirde, $E_{AK_i}\left(g^{x_i}, ID_K, ID_S, SIGNAL[i,1]\right)$ mesaj değeri deşifre edilir ve deşifre edilen mesajın içerisinde yer alan $SIGNAL[i,1]$ değerinin aynı kullanıcıya ait olup olmadığı kontrol edilir. Eğer değer aynı kullanıcıya ait ise sunucu öncelikle rastsal bir y_i değeri üreterek g^{y_i} değerini, ardından da $g^{x_i y_i} = K_i$ değerini hesaplar.

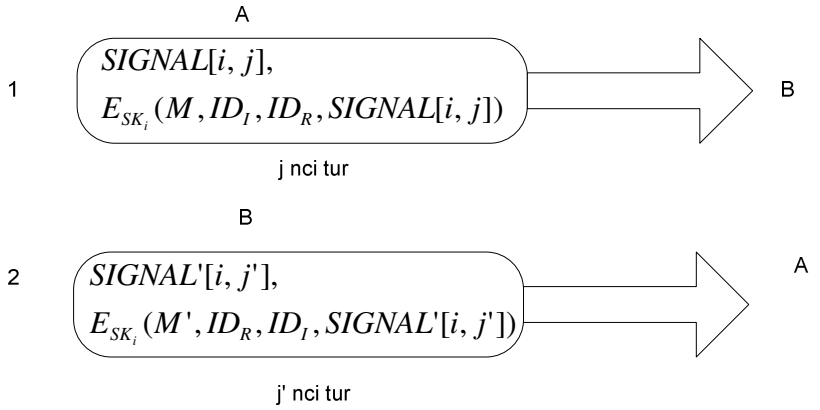
Son olarak sunucu istemciye, $SIGNAL'[i,1]$, g^{y_i} ve $h(SK_i, ID_S, ID_K, SIGNAL'[i,1])$ değerlerini içeren mesajı gönderir. Burada $SIGNAL'[i,1]$ değerinin $SIGNAL[i,1]$ değerinden tek farkı farklı bir kriptografik hash fonksiyonu kullanılarak hesaplanmasıdır.

3) Kullanıcı $SIGNAL'[i,1]$ değerini kontrol ederek mesajın gerçekten sunucu tarafından üretilip gönderildiğini doğrular ve $g^{x_i y_i} = K_i$ oturum anahtarı değerini

hesaplar. $h(SK_i, ID_S, ID_K, SIGNAL'[i,1])$ değerini üreterek, bu değeri gelen mesajda yer alan aynı değer ile karşılaştırır. Eğer karşılaştırma sonucu eşit ise anahtar değişimi işlemi başarılı bir şekilde sonlandırılır.

Yukarıdaki protokol adımlarında yer alan g^{x_i} ve g^{y_i} *i.* oturumda üretilen Diffie-Hellman açık anahtar değerleridir. *AK_i* doğrulama anahtarı $AK_i = h_2(K_{i-1})$ şeklinde hesaplanabilir. $SIGNAL'[i,1]$ değeri ise $SIGNAL[i,1]$ değerine benzer bir şekilde $SIGNAL'[i,j] = h'(SS_i, j)$ şeklinde hesaplanabilir.
Hesaplamalarda kullanılan *h, h', h₂* gibi gösterimler farklı kriptografik hash fonksiyonlarını temsil etmektedir.

Protokol aynı zamanda, anahtar değişimi sonrasında şekil 4.15 de görüldüğü gibi üretilen sinyallerin kullanılabildiği bir şifreli haberleşme ortamının geliştirilmesini de desteklemektedir. Mesaj ve diğer doğrulama bilgilerinin şifrelenmesinde kullanılan anahtar değeri $SK_i = h_3(K_i)$ şeklinde hesaplanabilir. Şifreli haberleşmenin her bir turunda sinyal değeri, tek yönlü hash fonksiyonu işlemine tabi tutularak değiştirilir.



Şekil 4-15 Şifreli Haberleşme

Şifreli haberleşme esnasında karşılaşılabilecek farklı kullanıcılar tarafında üretilecek aynı tek kullanımlık kimlik değerlerinin çakışması sorunu ve tekrarlama saldırıları, sinyal değerleri ve mesaj içerindeki sabit kimlik değerlerinin kontrol edilmesiyle önlenebilir.

5 GELİŞTİRME ORTAMI

Tez yazılımında, yazılım geliştirme ortamı olarak Microsoft Visual Studio.Net ürün ailesinin bir üyesi olan C# yazılım geliştirme dili kullanılmıştır. Bu bölümde .NET mimarisi ile tez yazılımında kullanılan yazılım teknolojilerinden olan, Uzak Nesne Erişimi(.NET Remoting) ve web servisleri hakkında bazı temel bilgiler anlatılmaktadır.

5.1 .Net Framework

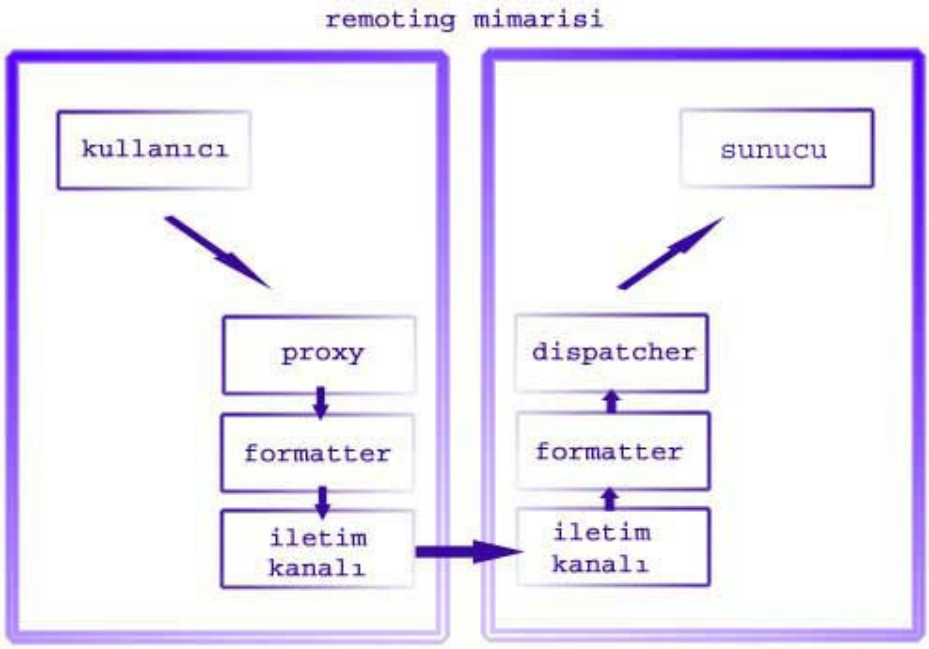
Microsoft .NET Framework [19], uygulamaların ve web servislerinin inşa edilebildiği(build), yayımının yapılabildiği (deploy) ve çalıştırılabildiği (run) bir platformdur.
Verimliliği yüksek, standartlara uygun ve çoklu dil desteği bulunan bir platformdur. İnternet ölçekli uygulamaların operasyonu ve yayımlanması için karşımıza çıkan zorluklar .NET Framework'ün sağladığı servisler sayesinde rahatlıkla aşılabilmektedir. .NET Framework yazılım geliştirme ortamı sahip olduğu dillere sağladığı geniş yazılım kütüphaneleri sayesinde farklı ortam ve dillere oranla geliştirilecek bir uygulamada yazılacak kod satırı sayısını, geliştirilen uygulamaya bağlı olarak, ortalama %10-%20'ler seviyesinde azaltmıştır. Platform, kriptoloji uygulamalarında kullanılabilecek işlemler için de oldukça geniş bir altyapı hazırlamaktadır. Örneğin; tez yazılımında kullanılan işlemlerden olan 3-DES şifreleme, tek yönlü fonksiyon kullanma gibi uygulamalar diğer yazılım geliştirme dillerine nazaran çok basit bir şekilde gerçekleştirilebilmektedir. Yine ortamın sağladığı gelişmiş matematiksel sınıflar sayesinde anahtar değişimi gibi oldukça yüksek basamaklı asal sayılarla işlemlerin yapıldığı bir uygulama için oldukça büyük bir kolaylık sağlanmaktadır.

.Net Framework ortamında, yazılım geliştirme işlemi için hangi dil kullanılırsa kullanılsın, kodlar makine koduna çevrilmeden önce, Microsoft Ara Dili (MSIL) olarak tanımlanan bir dile dönüştürülmekte, dolayısıyla bütün .NET dilleri arasında bir bütünleşme noktası sağlanmaktadır. Bahsedilen MSIL kod, çalıştırılmadan önce doğal makine koduna (native code) dönüştürülür ancak bu esnada kısaca,

kaynak kodun her bir deyiminin bir sonraki deyime geçmeden önce tek tek incelenmesi olarak tanımlan, yorumlama (interpretation) işlemi gerçekleştirilmez.

5.2 Uzak Nesne Erişimi (.Net Remoting)

Tez yazılımında, sunucu istemci arası haberleşme adımları *.NET Remoting* mimarisi kullanılarak geliştirilmiştir. Bu mimaride istemci ve sunucu tarafları belirli portlar üzerinden tcp yada http protokollerinden birini kullanarak haberleşir. Remoting mimarisinin web servislerine göre en önemli farkı, sunucu üzerinde çeşitli sorguların çalıştırılıp sonuçlarının görüntülenebilmesinin yanında, sunucu üzerinde yer alan referanslara erişilebilmesi ve sorgu sonuçlarının istemci tarafına nesne olarak döndürülebilmesidir.



Şekil 5-1 Uzak Nesne Erişimi Mimarisi

.Net Remoting mimarisinde İstemcilerin tek amacı sunucu üzerinde yer alan uzak nesne referanslarını kullanabilmektir. Bu açıdan bakıldığında, istemci uygulamanın uzak sunucu üzerindeki nesne referanslarının yapısını bilmesi gerekmektedir [20]. Kullanılabilecek yollardan ilki uzak nesne sınıfının bulunduğu paylaşımlı derlenmiş

uygulama dosyalarını (assembly) tüm istemcilere dağıtmaktır. Bu, istemci uygulamalar için ekstra kod yazmadan kolayca gerçekleştirilebilecek bir işlemdir. Ancak istemci uygulamalarda, uzak sınıfın tüm içeriğinin yer aldığı uygulama dosyaları yer alır. Bu da ILDASM (Intermediate Dis-Assembler Tool) ve başka üçüncü parti araçlar yardımıyla iş mantığının (business logic) istemci tarafından kolayca okunabileceği anlamına gelir. Bu teknik, güvenli bir anahtar değişimi ve ağ üzerinden haberleşme ortamı yaratmayı hedefleyen tez yazılımında, yaratabileceği güvenlik açığı nedeni ile tercih edilmemiştir.

İkinci yöntem uzak sınıf modelinin türetildiği bir ara yüz (interface) tipini istemci ve sunucu arasında paylaşıma sunmaktır. İstemci tarafında sadece ve sadece uzak nesne modelinin bilgisini tutan bir ara yüz tipi yer alacaktır. Bu modele göre uzak sunucu üzerindeki uzak nesne referansına, nesne yönelimli programlamanın bir özelliği olan çoklu kalıtım'ın (polimorfizm) bir sonucu olarak ara yüz tipi üzerinden erişilebilir. Daha da önemlisi istemci tarafında, uzak nesne içerisindeki kodların kesinlikle görünmemektedir. Bu da doğal olarak, iş mantığını istemciden gizleyen etkili bir tekniktir ve geliştirilen tez yazılımında bu tekniğin kullanılması tercih edilmiştir.

5.3 Web Servisleri

Günümüzde program geliştiriciler için en büyük imkan programların birbirleriyle entegre bir şekilde çalışabilmeleridir. Farklı programlama dilleriyle yaratılmış, farklı obje modelleri kullanan ve hatta farklı işletim sistemlerinde çalışan bir grup uygulamanın bir araya getirilip kullanımı kolay Web uygulamaları haline dönüştürülmesi mümkündür.

COM nesnelerinde olduğu gibi Web Servisleri de kara kutu gibidir. İçerisindeki kodun ne olduğu görülmez fakat sağladığı imkanlardan kolayca yararlanılabilir [19]. Web Servisleri, kullanıcılarına nasıl kullanılacaklarını ve sağladıkları servisleri tanımlayan çok iyi bir ara yüz sağlarlar.

Microsoft'un Distrubuted Component Object Model (DCOM), Sun'ın Remote Method Invocation (RMI), Object Managament Group'ın Common Object Request Broker Architecture (CORBA) ve benzeri teknolojilerden farklı olarak Web

Servisleri belirli obje modellerine özgü protokolleri kullanmaz. Web Servisleri iletişimde, standart Web protokollerini ve veri formatlarını kullanır. Örneğin Hypertext Transfer Protocol (HTTP), Extensible Markup Language (XML) ve Simple Object Access Protocol (SOAP) gibi. Bu Web standartlarını destekleyen sistemlerin tümü Web Servislerini destekler.

Bir Web Servisi içsel olarak bir uygulama içerisinde veya dışsal olarak Internet üzerinden birçok uygulama tarafından ortak olarak kullanılabilir. Web Servislerine erişim standart bir ara yüz aracılığıyla gerçekleştiği için birbirinden tamamen farklı sistemlerin birlikte çalışabilmesine olanak verir. Web Servislerinin modeli platformdan, dilden ve obje modelinden bağımsızdır.

Web Servisi modeli ASP.NET tarafından desteklenir. ASP.NET bütünleşik bir Web geliştirme platformudur ve Active Server Pages (ASP)' den geliştirilmiştir. ASP.NET Web Servisleri modeli durumsuz (stateless) bir mimari sunar. Durumsuz mimariler genelde daha ölçeklenebilirdir. Gelen her servis isteğine karşılık yeni bir nesne yaratılır.

Web servisleri teknolojisi özellikle ortak kullanılabilirlik, yüksek performans özellikleri nedeni ile tez yazılımında, istemci bir uygulamanın sunucu web servisi ile etkileşimde bulunarak tek kullanımlık parola üretmesi işleminin gerçekleştirilmesi işleminde tercih edilmiştir.

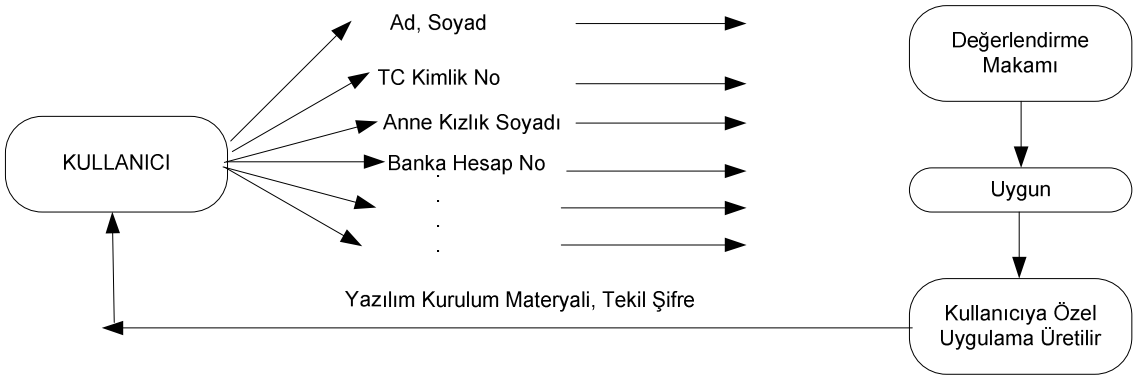
6 GELİŞTİRİLEN YAZILIM

Bu bölümde, tek kullanımlıklı parola ve kimlik doğrulamalı anahtar değişimi ortamını kurgulamak için geliştirilmiş olan tez yazılımı, kullanıcının başvuru işleminden başlanarak maddeler halinde anlatılmaktadır. Bu bölümde güvenilir ortam olarak bahsedilecek olan kavram tek kullanımlık parola ve kimlik doğrulamalı anahtar değişiminin kullanılacağı geniş alan haberleşme ortamıdır.

6.1 Kullanıcı Başvurusu ve İlklendirme

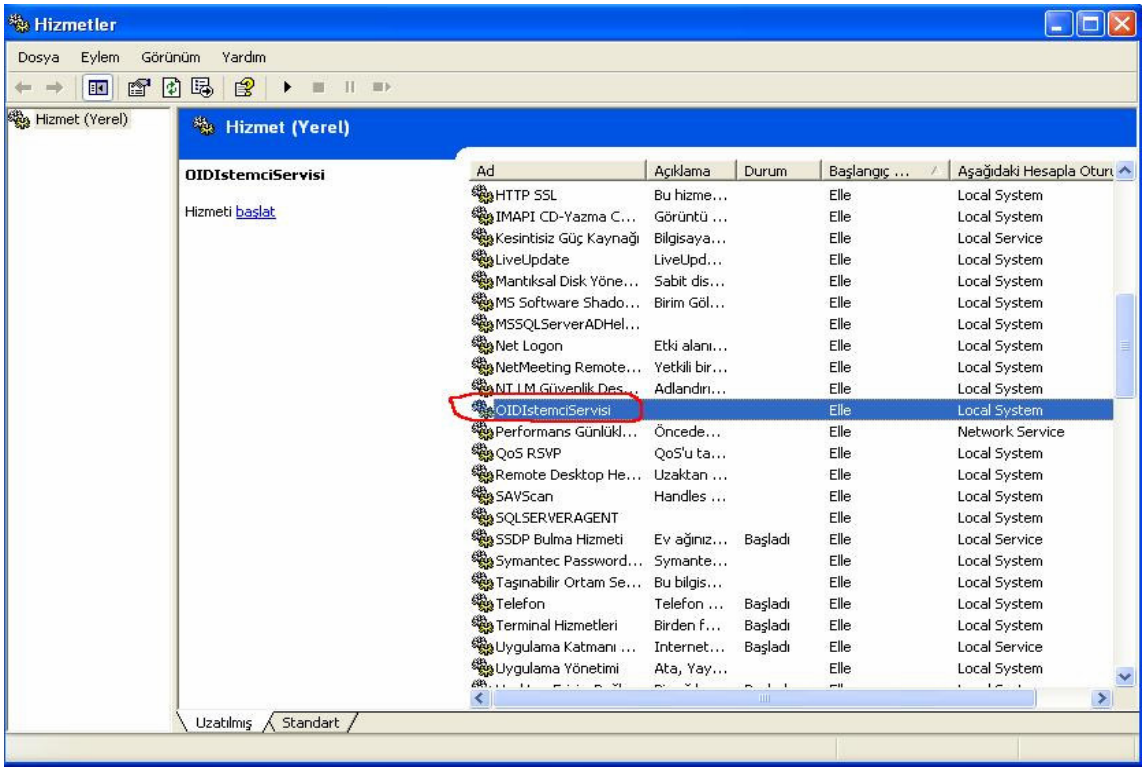
Sistemden hizmet almak isteyen kullanıcı ilk olarak çeşitli bilgiler ile internet, telefon yada kağıt form doldurarak Senkronizasyon Makamına başvuruda bulunmalıdır. Başvuru bilgilerinin değerlendirilmesi sonucu kullanıcı için özel ve benzersiz bir şifre değeri oluşturulur.

Şifre değeri, istemci tarafında çalışacak program içerisindeki bazı işlemlerde kod içerisinden sürekli olarak kullanılacağından, kod içerisine gömülerek kullanıcıya özel bir yazılım sürümü oluşturulur. Ayrıca bu şifre bir defaya mahsus gerçekleştirilecek senkronizasyon işlemi içinde kullanılacağından, posta yada kısa mesaj şeklinde kullanıcıya gönderilmelidir.



Şekil 6-1 Başvuru İşlemi

İstemci tarafında tek kullanımlık kimlik doğrulamalı anahtar değişimi işlemlerini gerçekleştirecek olan uygulama kullanıcı ile herhangi bir etkileşime ihtiyaç duymaması amacı ile bir Windows servisi olarak tasarlanmış olup, yazılımın kurulması sonucunda otomatik olarak aktif hale gelir. Uygulama, çalıştığı bilgisayar üzerindeki ara yüz dosyalarını kullanarak güvenli sunucu üzerindeki metodlara erişir ve bu sayede işlemlerini gerçekleştirir.

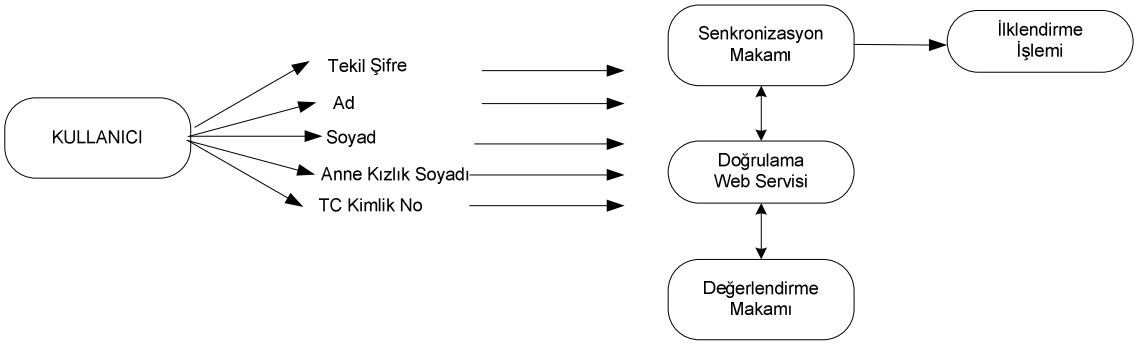


Şekil 6-2 OID İstemci Servisi

Kullanıcı, güvenilir ortama dahil olan internet sitelerine erişirken kullanacağı tek kullanımlık parola değerini kendisi oluşturup, ilgili sitenin şifre bölümüne kendisi gireceğinden, parola değerini üreten yazılım, kullanıcı ile etkileşimli (interaktif) olarak çalışan bir form uygulaması şeklinde tasarlanmıştır.

Servis uygulaması aktif olduğu andan itibaren sunucu uygulama ile etkileşim halindedir ve senkronizasyon işleminin yapılması için başvuruda bulunmaktadır. Sunucu uygulamasının ilklendirme işlemini başlatması için kullanıcının bir defaya mahsus senkronizasyon web sitesini ziyaret ederek şekil 6.3 de görüldüğü gibi

gerekli bilgileri girmesi ve böylece başvurusu sonucu kendisine iletilen kullanım bilgilerini başarılı bir şekilde edindiğini bildirmesi gereklidir.



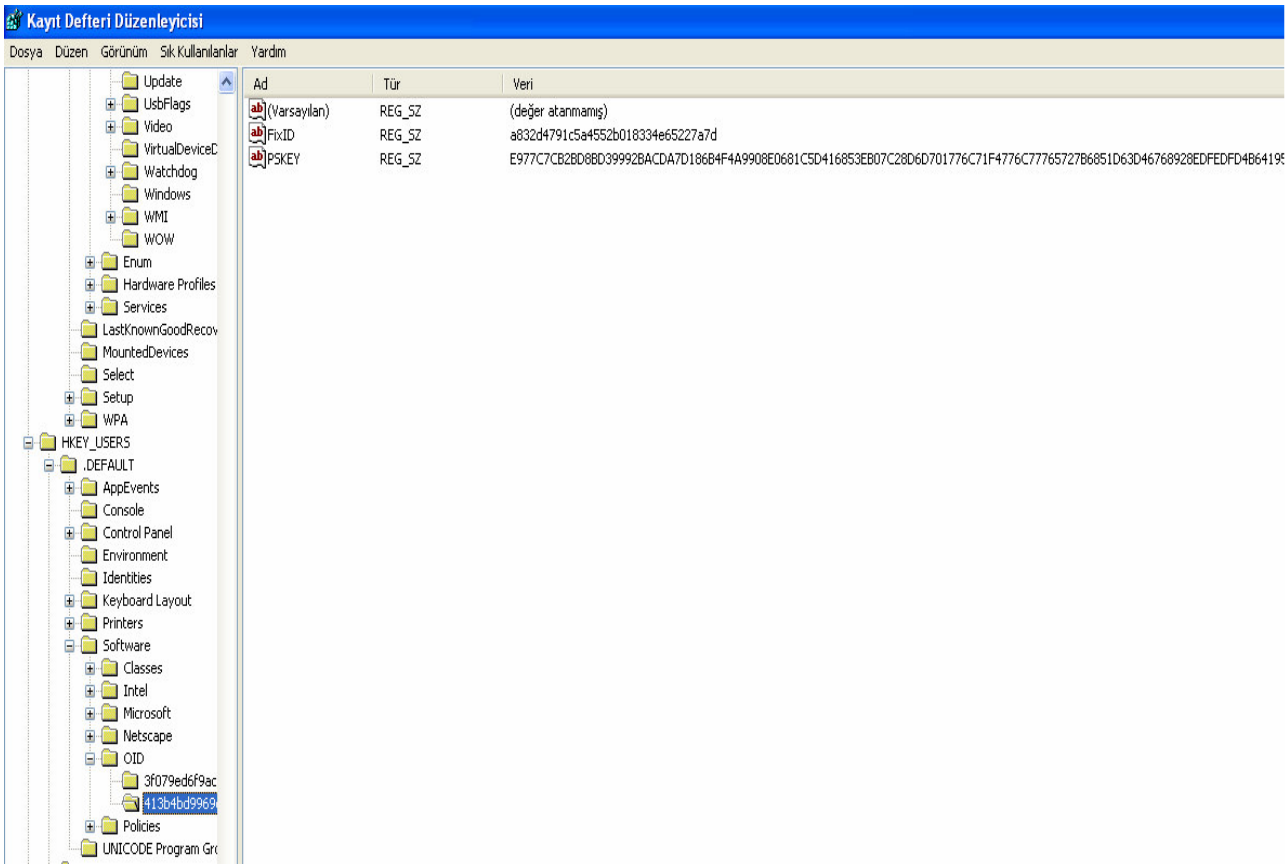
Şekil 6-3 İlklendirme İşlemi

Senkronizasyon Makamı web uygulaması, kullanıcının girdiği bilgileri web servis uygulaması aracılığı ile Doğrulama Makamına iletir. Eğer kontrol sonucu olumlu ise kullanıcıya ait veritabanı kaydı senkronizasyon işlemini başlatmaya uygun şekilde günceller. İlklendirme işlemi senkronizasyon makamı ve doğrulama makamı arasındaki güvenli hat üzerinden gerçekleştirilmektedir dolayısıyla güvensiz internet ağı üzerinden, sunucular üzerine yapılacak saldırılarla ilklendirme işleminin başlatılabilmesi engellenmiştir.

6.2 Senkronizasyon İşlemi

İlklendirme işleminin sonlandırılması ile birlikte, sunucu uygulaması kendisine senkronizasyon işlemi için başvuran istemci uygulamasının işlemlerini başlatılır. Bu işlemler kapsamında ilk olarak sunucu ürettiği rastsal değerler ile, bölüm 4.2.6 da açıklanan anahtar değişimi işlemini kendi kendine gerçekleştirilir ve 1024 bit uzunluğundaki paylaşılan anahtar değerini (pre-sharedkey) hesaplar, veritabanı üzerinden kullanıcının ilklendirme işleminde belirlediği, internet üzerinden işlem yapmak istediği güvenli ortama dahil sunuculara ait benzersiz kimlik bilgilerini (GUID) okur. Son olarak istemciye ait benzersiz kimlik değerini de hesaplayarak, bütün bu hesapladığı verileri paketleyip istemci tarafına gönderir ve ilgili verileri veritabanı üzerinde günceller.

İstemci sunucudan gelen paket içinden okuduğu verileri yorumlayarak Şekil 6.4 deki gibi bilgileri Windows kayıt defteri üzerinde günceller ve senkronizasyon işlemini başarılı bir şekilde sonlandırır.



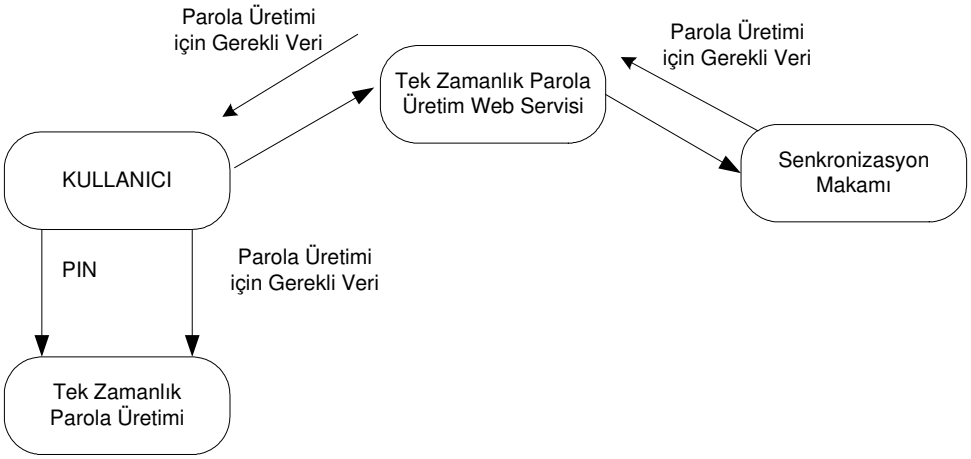
Şekil 6-4 OID Kayıt (Registry) Bilgileri

Dosya içerisinde, bir sonraki haberleşme işleminde kullanılacak ortak anahtar bilgisi, sunucuya ait sabit kimlik ve istemciye ait sabit kimlik bilgileri yer almaktadır.

6.3 Haberleşme Adımları

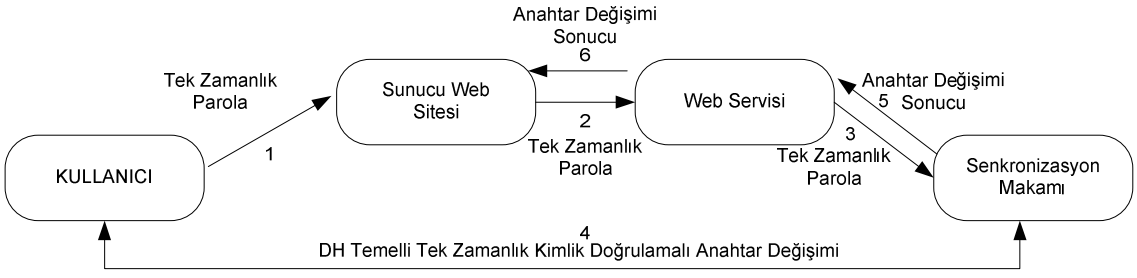
İstemci ve sunucu, ilklendirme ve senkronizasyon işlemlerinin ardından güvenli bir şekilde haberleşmeye hazır hale gelirler. Kullanıcı, güvenli ortama üye olan web sitesinde işlemlerini gerçekleştirmeden önce tek kullanımlık parolasını oluşturmak zorundadır. Tek kullanımlık parola üretimi kullanıcı ile etkileşim halinde bulunan windows form uygulamasının web servisi aracılığı ile sunucu ile haberleşmesi sonucu gerçekleştirilir. Bölüm 4.2.5 de detaylı bir şekilde anlatılmakta olan tek kullanımlık parola üretiminde uygulama kullanıcıdan, parola değerinin üretilmesi

için bir PIN değerini girmesini ister. Üretilen tek kullanımlık parola değerinin geçerlilik süresi ayarlanabilir bir özelliktir ve bu değer aşıldığı takdirde eski parola geçerli olmayacak, kullanıcıdan tekrar yeni bir tek kullanımlık parola değeri oluşturması istenecektir.



Şekil 6-5 Tek Kullanımlık Parola Üretimi

Şekil 6.5 de görüldüğü gibi üretilen tek kullanımlık parola değeri, kullanıcı tarafından güvenli ortama dahil sunucuya ait web sitesine girişte parola alanına girilir. Sunucu girilen parola değeri ile web servisi üzerinden tek kullanımlık kimlik doğrulamalı anahtar değişimini başlatması için senkronizasyon makamına başvurur. Senkronizasyon makamı, parola geçerlilik süresi güncel ise istemci ile anahtar değişimi oturumunu başlatır ve anahtar değişimi sonucunu ve sunucuya başvuran istemcinin sabit kimlik değeri bilgisini, kendisine başvuran sunucuya iletir. Anahtar değişimi işleminin başarılı olması ile sunucu üzerinde istemcinin doğrulama işlemi gerçekleştirilmiş olur.



Şekil 6-6 OTP ve OID Kullanarak Haberleşme

Şekil 6.6 da görülen anahtar değişimi sonucunda üretilen, yeni paylaşılan parola değeri senkronizasyon makamı üzerindeki veritabanında ve istemci tarafında yer alan dosya üzerinde bir sonraki haberleşme işlemi için güncellenir.

Doğrulama işlemlerinin ardından istemci sunucu web sitesi üzerinde istediği işlemleri gerçekleştirebilir. Sunucu istediği takdirde kritik işlemler öncesinde, yine bir web servisi aracılığı ile senkronizasyon makamına başvurarak istemciye "hala orada mısın?" gibi şifreli bir yenileme mesajı çektirebilir. Senkronizasyon makamı ve istemci arasında gerçekleşecek olan bu haberleşme, bölüm 4.2.6 da açıklanan anahtar değişimi sonrası şifreli haberleşme ortamında olduğu gibi gerçekleşir.

Yukarıda anlatılan doğrulama ve şifreli haberleşme işlemlerini tetikleyen haberleşme trafiği, sadece sunucu ve senkronizasyon makamı arasındaki güvenli ağ ortamı üzerinden gerçekleşmektedir. Açık internet ağı üzerinden gönderilecek mesajlar ile anahtar doğrulama ve şifreli haberleşme işlemlerinin başlatılması mümkün değildir. Ayrıca işlemlerin tetiklenmesi ve gerçekleştirilmesinin farklı hatlar üzerinden yürütülmesi, işlemlerin tutarlılığını ve doğruluğunu kuvvetlendirmektedir.

7 SONUÇLAR

Tez çalışması sonucunda geliştirilen yazılım ile aşağıda sıralanmakta olan niteliklere sahip bir ağ haberleşme ortamı sağlanmıştır:

- Haberleşme ortamına dahil olmak isteyen kullanıcı, gerekli başvuru bilgilerini ilettikten sonra kendisine uygulama yazılımı kurulmakta ve benzersiz bir kayıt numarası bildirilmektedir. Kullanıcı sistemi kullanmadan önce bu benzersiz numarayı ve bazı özel bilgilerini kullanarak sadece bir defalığa mahsus olmak üzere senkronizasyon makamı web sunucusu üzerinden senkronizasyon işlemlerini yerine getirmektedir. Bu işlemin ardından kullanıcı, senkronizasyon işlemine ait herhangi bir bilgiyi saklamak ve daha sonraki işlemlerinde kullanmak zorunda değildir.
- Kullanıcı, sistemin içinde yer alan tek kullanımlık parola üretim yazılımı ile web sunucularından hizmet almadan önce kendine özel PIN numarasını kullanarak tek kullanımlık parola değerini üretebilmektedir. Web sunucularına girişte sadece üretilen bu tek kullanımlık parola bilgisi istenmekte başka herhangi bir bilgiye ihtiyaç olmamaktadır. Bu sayede kullanıcı, çok farklı şifre değerlerini saklamak yada aklında tutmak zorunda kalmamaktadır.
- Sistemin kullanılabilmesi için, sisteme başarılı şekilde dahil olmuş yetkili kullanıcının bilgisayarı üzerinde kullanıcıya özel üretilmiş, tek kullanımlık kimlik istemci uygulamasının çalışır durumda olması, çalışma esnasında ihtiyaç duyulan kayıt defteri (registry) üzerindeki senkronizasyon bilgilerinin güncel ve geçerli olması gereklidir. Bu açıdan bakıldığında saldırgan zorda olsa o an geçerli olan bir tek kullanımlık parola değeri elde ederek ilgili sunucuya giriş yapsa dahi, istemci uygulamaya ve geçerli kayıt bilgilerine sahip olmadığı için sunucular üzerinde hiç bir işlem gerçekleştiremeyecektir.
- Senkronizasyon makamının sorumluluklarından biri olan, tek kullanımlık kimlik doğrulamalı anahtar değişimi işleminin tetiklenmesi, sadece ortama dahil güvenilir sunuculardan gelen isteklerle başlatılabilmektedir. Bu sayede saldırganın internet üzerinden geliştireceği ataklarla anahtar değişimini başlatması gibi bir durum söz konusu olmamaktadır.

- Sistem, istenildiği takdirde kullanıcının uygulamayı kendi bilgisayarı dışındaki bilgisayarlarda da kullanabilmesi için uygulamanın ve senkronizasyon bilgilerinin taşınabilir bir disk ünitesine aktarılmasına olanak sağlayabilmektedir.
- Sistemde kullanıcının hizmet almak istediği web sunucusuna tek kullanımlık parolasını girmesi, sunucunun parolayı kontrol ederek, anahtar değişimini başlatma isteğini senkronizasyon makamına iletmesi, makamın anahtar değişimini gerçekleştirerek sonucunu tekrar web sunucusuna döndürmesi işlemleri aynı zamanda fakat ayrı iletişim hatları üzerinden gerçekleştirilmektedir. Bu durum ortamın güvenlik ve doğruluğunu güçlendirmektedir.
- Sistemde yer alan web sunucuları kendilerinden hizmet alan kullanıcılara ait hiçbir kişisel bilgiye sahip değildir. Hizmet sunucuları kullanıcıdan gelen bilgileri senkronizasyon makamı ile koordine kurarak doğrulatmakta ve anahtar değişim işlemini başlatarak işlem sonucuna göre geçerli olan kullanıcı oturumuna, gerekli işlem yetkilerini vermektedirler. Sistemde gerekli olduğu takdirde kullanıcının kişisel bilgilerine ulaşabilecek tek sunucu senkronizasyon makamıdır. Bu sayede kullanıcı kimliği haberleşme esnasında ağ ortamına iletilmemekte kullanıcı gizliliği gereksinimi yerine getirilebilmektedir.
- Sistem, doğrulama ve anahtar değişim işlemlerinin ortak bir nokta (senkronizasyon makamı) üzerinden yürütülmesi sayesinde gerekli olduğu takdirde makamın sunucular ile etkileşime geçerek sunucular arası koordinasyon ve ortak çalışılabilirliğin yürütülmesine olanak sağlamaktadır.
- Sistemde kullanılan anahtar değişim yöntemi sayesinde anahtar değişimi için gerçekleştirilmesi gereken haberleşme turu sayısı minimize edilmiş performans artışı sağlanmıştır.
- Sistemde kullanılan anahtar değim yöntemi ve anahtar değişim işlemlerinin sadece güvenilir sunucular üzerinden gelen istekler ile başlatılması sayesinde hizmeti yalanlama (DOS) ve tekrarlama gibi ataklar engellenmiştir.
- Sistemde yer alan anahtar değişim ve şifreli haberleşme işlemlerinde simetrik şifreleme yönteminin kullanılması ile haberleşme performansı arttırılmış, sertifika ve sil listesi üretim, dağıtım ve kontrolü gibi karmaşık uygulamalar kullanılmamıştır.

- **KAYNAKLAR**

[1] "www.kamusm.gov.tr" web sayfası, Nisan 2006.

[2] William Stallings, Cryptography and Network Security Principles and Practice Second Edition, 2000, Prentice Hall.

[3] Yunus Emre ALPÖZEN, Microsoft.Net ve Güvenlik Birinci Baskı, 2005, Seçkin.

[4] Albert Levi, Mahmut ÖZCAN, Açık Anahtar Tabanlı Şifreleme Neden Zordur?, Sabancı Üniversitesi, Mühendislik ve Doğa Bilimleri Fakültesi.

[5] "www.mathworld.com" web sayfası, Nisan 2006.

[6] W. Diffie, M.E. Hellman, 1976, New Directions in Cryptography, IEEE Transactions on Information Theory, IT-22:644-645.

[7] A. Menezes, P. Van Oorschot and S. Vanstone, Handbook of Applied Cryptography, 1996, CRC Press.

[8] Needham,R., and Schroeder, M. , "Using Encryption for Authentication in Large Networks of Computers." Communications of the ACM, December 1978.

[9] Denning, D. "Timestamps in Key Distribution Protocols", Communications of the ACM, August 1981.

[10] Neuman, B., and Stubblebine, S. , "A Note on the Use of Timestamps as Nonces.", Operating Systems Review, April 1993.

[11] Woo, T., and Lam, S. , "Authentication for Distributed Systems.", Computer, January 1992.

[12] Woo, T., and Lam, S. , "Authentication Revisited.", Computer, April 1992.

[13] David P. Jablon, "Strong Password-Only Authenticated Key Exchange", ACM Computer Communications Review, October 1996.

[14] "http://www.afn.org/~afn21533/keyexchg.htm" web sayfası, Haziran 2006.

[15] S. M. Bellovin and M. Merritt, Encrypted Key Exchange:Password-Based Protocols Secure against Dictionary Attacks, Proceedings of the IEEE Symposium on Research in Security and Privacy, Oakland, May1992.

[16] S. M. Bellovin and M. Merritt, Augmented Encrypted Key Exchange:A Password-Based Protocol Secure Against Dictionary Attacks and Password File Compromise.

[17] Neil Haller, Craig Metz, Philip J. Nesser II, and Mike Straw, RFC 2289: A One-Time Password System.

[18] Kenji IMAMOTO, Kouichi SAKURAI, "A Design of Diffie-Hellman Based Key Exchange Using One-time ID in Pre-shared Key Model", Proceedings of the18th International Conference on Advanced Information Networking and Application, 2004.

[19] “www.msakademik.net” Web Sayfası, Temmuz 2006.

[20] “www.csharpnedir.com” Web Sayfası, Temmuz 2006.

ÖZGEÇMİŞ

Adı Soyadı: Ali Umut YÜKSEL

Doğum Yeri: Ankara

Doğum Yılı: 16 Eylül 1979

Medeni Hali: Evli

Eğitim ve Akademik Durumu:

Lise: Keçiören Fatih Sultan Mehmet Lisesi 1993-1996

Lisans: Ankara Üniversitesi Elektronik Mühendisliği Bölümü 1997-2001

Yabancı Dil: İngilizce

İş Tecrübesi:

2003-2003 Antek Elektronik

2003-2005 T.C Maliye Bakanlığı Bilgi İşlem Daire Başkanlığı

2005-…. Havelsan