TOBB EKONOMİ VE TEKNOLOJİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

AÇIK ANAHTAR DOĞRULAMASINDA KULLANILAN PARMAKİZİ VE EMNİYET NUMARASI YÖNTEMLERİNİN GÜVENLİĞİNİN VE KULLANILABİLİRLİĞİNİN KARŞILAŞTIRILMASI

YÜKSEK LİSANS TEZİ

Muhammet Şakir ŞAHKULUBEY

Bilgisayar Mühendisliği Anabilim Dalı
Bilgi Güvenliği

Tez Danışmanı: Prof. Dr. Kemal BIÇAKCI

NİSAN 2018

Fen Bilimleri Enstitüsü Onayı

.....
Prof. Dr. Osman EROĞUL
Müdür

Bu tezin Yüksek Lisans derecesinin tüm gereksinimlerini sağladığını onaylarım.

.....
Prof. Dr. Oğuz ERGİN
Anabilimdalı Başkanı

TOBB ETÜ, Fen Bilimleri Enstitüsü'nün 151111050 numaralı Yüksek Lisans öğrencisi **Muhammet Şakir ŞAHKULUBEY** 'in ilgili yönetmeliklerin belirlediği gerekli tüm şartları yerine getirdikten sonra hazırladığı **"AÇIK ANAHTAR DOĞRULAMASINDA KULLANILAN PARMAKİZİ VE EMNİYET NUMARASI YÖNTEMLERİNİN GÜVENLİĞİNİN VE KULLANILABİLİRLİĞİNİN KARŞILAŞTIRILMASI"** başlıklı tezi **05.04.2018** tarihinde aşağıda imzaları olan jüri tarafından kabul edilmiştir.

Tez Danışmanı:	**Prof. Dr. Kemal BIÇAKCI** TOBB Ekonomi ve Teknoloji Üniversitesi	.....
Jüri Üyeleri:	**Prof. Dr. Ali Aydın SELÇUK (Başkan)** TOBB Ekonomi ve Teknoloji Üniversitesi	.....
	Dr. Öğr. Üyesi Hakan Ezgi KIZILÖZ Türk Hava Kurumu Üniversitesi	.....

TEZ BİLDİRİMİ

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

Muhammet Şakir ŞAHKULUBEY

ÖZET

Yüksek Lisans Tezi

AÇIK ANAHTAR DOĞRULAMASINDA KULLANILAN PARMAKİZİ VE EMNİYET NUMARASI YÖNTEMLERİNİN GÜVENLİĞİNİN VE KULLANILABİLİRLİĞİNİN KARŞILAŞTIRILMASI

Muhammet Şakir ŞAHKULUBEY

TOBB Ekonomi ve Teknoloji Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı
Bilgi Güvenliği

Tez Danışmanı: Prof. Dr. Kemal BIÇAKCI

Tarih: NİSAN 2018

Açık anahtarlı şifreleme, şifreleme ve şifre çözme işlemleri için sırasıyla açık anahtar (public key) ve bu açık anahtardan üretilemeyen bir özel anahtar (private key) olmak üzere iki farklı anahtar kullanılan şifreleme yöntemidir. Bazı sistemlerde, son kullanıcı için açık anahtarlardan türetilen çeşitli arayüzler (örn. SSH fingerprint), bir güvenlik önlemi olarak karşımıza çıkmaktadırlar. Literatürde "açık anahtar doğrulaması" olarak geçen bu probleme farklı çözüm yöntemleri önerilmiştir. Bu yöntemler sayısal sertifika gibi güvenilir üçüncü partilerin kullanımını içeren maliyetli çözümler yerine bu işin kullanıcı tarafından yapılması esasına dayanır. Günlük hayatta kullanılan bazı anlık mesajlaşma uygulamaları da kullanıcılarına açık anahtar doğrulaması yapabilme imkanı tanır. Mobil uygulama geliştiriciler açık anahtar doğrulamayı birbirlerinden farklı arayüzler ile kullanıcılarına sunabilmektedirler. Emniyet numarası ve parmakizi yöntemleri ise, bu uygulamalarda yaygın şekilde kullanılan iki farklı açık anahtar doğrulama şeklidir. Son kullanıcıya yönelik geliştirilmeleri dolayısı ile bu açık anahtar doğrulama yöntemlerinin kullanılabilirliği ve güvenliği temel başarı kriterlerindendir. Uçtan uca şifreleme sağlayan açık kaynaklı bir anlık mesajlaşma uygulaması olan SIGNAL, geçmiş versiyonlarında "parmakizi" ile, güncel versiyonunda ise "emniyet numarası" ile açık anahtar

doğrulama yöntemini kullanıcılarına sunmuştur. Parmakizi ile açık anahtar doğrulama yönteminde sisteme kayıt yaptırmış her kullanıcının, açık anahtarından türetilmiş bir parmakizi bulunmaktadır. Emniyet numarası ile açık anahtar doğrulamada ise kullanıcının ve mesajlaştığı kişinin açık anahtarlarından türetilmiş bir tek emniyet numarası ile doğrulama sağlanmaktadır. Tahmin edileceği üzere üretilen bu emniyet numarası oturuma özeldir; yani bir kullanıcının mesajlaştığı her kişiyle farklı bir emniyet numarası bulunmaktadır. Her iki versiyonda da, daha kolay karşılaştırma yapılabilmesi için QR kod tarama seçeneği de bulunmaktadır.

Yapmış olduğumuz çalışmada, SIGNAL android uygulaması üzerinden bu iki açık anahtar doğrulama yöntemini, laboratuvar ortamında kullanıcı deneyine tabi tuttuk. 21'er kişilik birbirine benzer iki kullanıcı grubu ile yapılan çalışmada, emniyet numarası ile açık anahtar doğrulama yönteminin gerçekte ne kadar güvenlik ve kullanışlılık sunduğunu, parmakizi ile açık anahtar doğrulama yöntemiyle karşılaştırarak test ettik. Elde ettiğimiz sonuçlar, doğrulama süresi ve kolaylığı noktasında "emniyet numarası" yönteminin "parmakizi" yöntemine göre daha avantajlı olduğunu göstermektedir.

Anahtar Kelimeler: Açık anahtar doğrulama, Emniyet numarası, Parmakizi, Kullanışlılık.

ABSTRACT

Master of Science

STUDY ON THE USABILITY AND SECURITY OF THE FINGERPRINT AND SAFETY NUMBER METHODS USED IN PUBLIC KEY VERIFICATION

Muhammet Şakir ŞAHKULUBEY

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

Supervisor: Prof. Dr. Kemal BIÇAKCI

Date: APRIL 2018

Public key cryptography is a cryptographic method that uses two different keys for encryption and decryption, respectively, public key and a private key, which cannot be generated from this public key. In some systems, the various interfaces (e.g. SSH fingerprint) derived from public keys for the end user are being used as a security measure. Different methods of solution have been proposed to this problem called as "public key verification" in the literature. These methods are based on the principle that they are done by the user rather than costly solutions that involve the use of reliable third parties such as digital certificates. Some instant messaging applications that are used in daily life also provide the possibility to perform public key verification. Mobile application developers can present their public key verification to users with different interfaces. The safety number and fingerprint methods are two different public key verification methods that are commonly used in these applications. Since these public key verification methods are developed for end-user, usability and security are key success criteria. SIGNAL, an open source instant messaging application that provides end-to-end encryption, has allowed its users for public key verification with "fingerprint" in past versions and "safety number" in current version. In the public key verification method with fingerprinting, each user registered to the system has a fingerprint derived from the

public key. In the case of public key verification with the safety number, verification is provided by a single safety number derived from the public keys of the user and his/her conversation partner. As predicted, this safety number produced is specific to the session; there is a different safety number for each person that the user is messaging with. In both versions, the QR code scan option is also available for easier comparison.
In our work, we have performed a user study to test these two public key verification methods through the SIGNAL android application, in the laboratory environment. In the study with two similar user groups of 21 users, we tested how much security and usability that safety number method actually offer, by comparing with the fingerprint method. The results show that the "safety number" method is more advantageous than the "fingerprint" method with respect to verification time and simplicity.

Keywords: Public key verification, Safety number, Fingerprint, Usability.

TEŞEKKÜR

Çalışmalarım boyunca değerli yardım ve katkılarıyla beni yönlendiren hocam Prof. Dr. Kemal BIÇAKCI, kıymetli tecrübelerinden faydalandığım TOBB Ekonomi ve Teknoloji Üniversitesi Bilgisayar Mühendisliği Bölümü öğretim üyelerine ve destekleriyle her zaman yanımda olan aileme ve arkadaşlarıma çok teşekkür ederim.

İÇİNDEKİLER

ŞEKİL LİSTESİ

KISALTMALAR

PGP : Oldukça iyi Gizlilik (Pretty Good Privacy)
TLS : Taşıma Katmanı Güvenliği (Transport Layer Security)
SSL : Güvenli Soket Katmanı (Secure Socket Layer)
PKC : Açık Anahtarlı Şifreleme (Public Key Cryptogaphy)
DHAD : Diffie-Hellman Anahtar Değişimi
SSH : Güvenli Kabuk (Secure Shell)
IPSec : İnternet Protokolü Güvenliği (Internet Protocol Security)
KDF : Anahtar Türetme Fonksiyonu (Key Derivation Function)
OTR : Kayıt Dışı (Off The Record)
QR : Hızlı Yanıt (Quick Response)
APN : Erişim Noktası Adı (Access Point Name)
iOS : iPhone İşletim Sistemi (iPhone Operating System)
PC : Kişisel Bilgisayar (Personal Computer)

SEMBOL LİSTESİ

Bu çalışmada kullanılmış olan simgeler açıklamaları ile birlikte aşağıda sunulmuştur.

Simgeler	Açıklama
k_{pubX}	X kişisine ait açık anahtar
k_{prX}	X kişisine ait gizli anahtar
k_{AB}	A ile B kişilerine ait gizli anahtar
x^2	Ki-kare (chi square)
H_0	Başlangıç hipotezi
H_A	Araştırma hipotezi
G	Gözlenen Frekans
B	Beklenen Frekans
sd	Serbestlik derecesi
r	DH anahtar türetme fonksiyonundaki gizli olmayan parametre
p	Seçilen büyük bir asal sayı
a	Seçilen bir tamsayı (alfa)

1. GİRİŞ

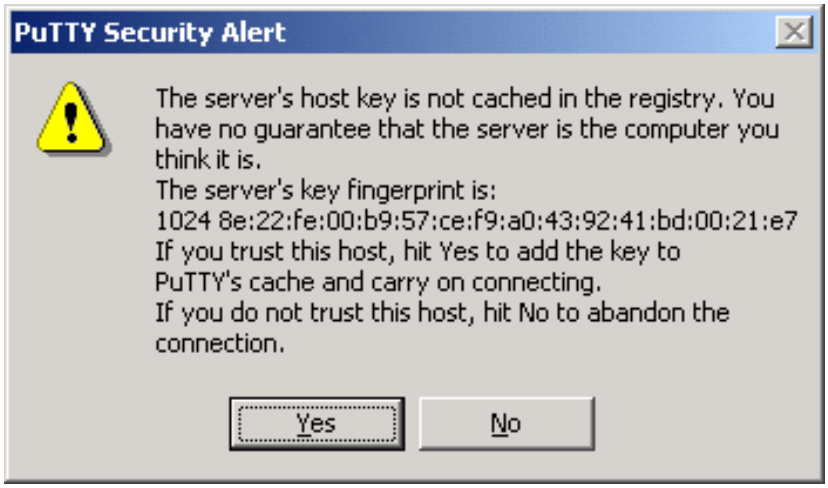
İnternet üzerinden güvenli haberleşme, günümüzde birçoğumuz için vazgeçilmez bir ihtiyaç haline gelmiştir. Günlük hayatta kullandığımız bazı araçlar da, internet üzerinden iletişim sağlarken güvenliği sağlama noktasında, uçtan uca şifreleme yöntemini kullanır. Uçtan uca şifreleme, gizli anahtarların kullanıcıların aygıtlarından dışarıya çıkmadığından, dolayısı ile iletişim sağlayıcıların, mesaj içeriklerini okuyamadığından emin olmamızı sağlayan teknolojinin ismidir. Bununla birlikte, PGP gibi uçtan uca şifreleme araçlarının ilk nesli, kullanışlı olmamaları sebebiyle yaygınlaşmamıştır [1–4].

PGP'nin ilk çıktığı zamandan bu yana, güvenli mesajlaşmanın iki önemli yönü değişmiştir: Akıllı telefonların kişisel bilgisayarların (PC) yerini almasıyla birlikte, mobil cihazlar aracılığıyla iletişim büyümüş ve gizlilik ile güvenlik hakkındaki farkındalık artmıştır.

Yeni nesil mesajlaşma uygulamaları, güvenlik penceresinden iki kategoriye ayrılabilir; kullanıcı ile sunucu arasında şifreleme sağlayan ve uçtan uca şifreleme sağlayan mesajlaşma uygulamaları. Birinci kategorideki uygulamalar servis sağlayıcının kullanıcıların aralarında gönderip aldıkları mesajları okuyabilmesine olanak sağlarken, ikinci kategorideki uygulamalar gönderilen mesajların servis sağlayıcı tarafından okunamayacağından emin olunmasını sağlar [5].

Uçtan uca şifreleme sağlayan mobil mesajlaşma uygulamalarını kullanmak için tek ihtiyaç akıllı bir telefon ve bir cep telefonu hattıdır. Uygulama yüklenir ve cep numarası üzerinden kayıt tamamlanır. Uçtan uca şifreleme için gerekli anahtar setleri arka planda oluşturulur. Kullanıcıların iletişim kuracakları kişilerin kriptografik kimliklerine ulaşabilmeleri için de merkezi bir servis kullanılır. Bu sistem beraberinde bazı riskleri de getirmektedir. Merkezi sunucu, sistem yöneticisi tarafından kasıtlı olarak veya dışarıdan bir saldırı sonucu manipüle edilirse, uygulamayı kullananlar bundan etkilenecektir. Kullanıcılara karşılık gelen açık anahtarların değiştirilmesi, araya girme saldırılarına imkân tanıyacaktır. Açık anahtarlar ile kimlik doğrulamanın önemi, bu gibi bir durumda ortaya çıkmaktadır. Uçtan uca şifreleme hizmeti sunan uygulamalarda açık anahtarlar vasıtasıyla doğrulama imkânı olması, güvenli iletişimin tesis edilebilmesi için büyük önem arz etmektedir.

Açık anahtar doğrulama, çeşitli arayüzlerle sunulabilmektedir. Bir bilgisayara ilk kez SSH bağlantısı yapmak istediğimizde çoğumuzun önemsemediği uyarıdaki parmakizi, doğrulama yapılabilmesi için açık anahtardan türetilmiş bir arayüz örneğidir(Sekil 1.1).



Şekil 1.1: SSH fingerprint.

Çalışmamızda emniyet numarası ile açık anahtar doğrulama yönteminin gerçekte ne kadar güvenlik ve kullanışlılık sunduğunu, parmakizi ile açık anahtar doğrulama yöntemiyle karşılaştırarak test ettik. Bunun için de 21'er kişilik iki kullanıcı grubu ile bir kullanıcı çalışması gerçekleştirdik. Çalışmada akıllı android telefonlar üzerinde SIGNAL anlık mesajlaşma uygulamasının önceki ve güncel versiyonlarını kullandık. Açık anahtar doğrulamasında geçmiş versiyonlarında parmak izi, güncel versiyonlarında ise emniyet numarası yöntemini kullanması, açık kaynaklı olması ve son olarak bir milyardan fazla kullanıcısı olan WhatsApp'ın uçtan uca şifrelemede SIGNAL-Protokolü'nü kullandığını belirtmesi, çalışmamızda SIGNAL android uygulamasını kullanmamızın sebepleri olarak sıralanabilir.

SIGNAL iki farklı mobil uygulamanın birleşmesi sonucu ortaya çıkmıştır: Şifreli mesajlaşma uygulaması olan TextSecure ve şifreli telefon görüşmesi uygulaması olan RedPhone [6]. Kendi güçlü uçtan uca şifreleme protokolünü kullanmaktadır ve açık kaynaklıdır. WhatsApp [7], Nisan 2016'da uçtan uca şifrelemeye geçiş yaptığını ve bu doğrultuda arka planda Signal-Protokolü'nü kullanmaya başladığını duyurmuştur [8]. SIGNAL mobil uygulamasının eski ve yeni versiyonları, açık anahtar doğrulamada iki farklı yöntemi kullanmaktadırlar. Eski versiyon parmakizi yöntemi ile kimlik doğrulama sunarken, halihazırda kullanımda olan güncel versiyon ise Kasım 2016'dan bu yana açık anahtar doğrulama için emniyet numarası doğrulama yöntemini kullanıcılarına sunmaktadır [9].

Uygulama geliştiriciler arasında, açık anahtar doğrulama yöntemi olarak emniyet numarasının gün geçtikçe daha çok tercih edilmesi, bizi bu yöntemin güvenliğini ve kullanışlılığını ölçmek için gerçekleştirdiğimiz çalışmaya yönlendirmiştir. Elimizdeki bilgilere göre emniyet numaraları ile açık anahtar doğrulamanın labaratuvar ortamında ve yüzyüze test edilmesi açısından, çalışmamız bir ilk olma özelliğini taşımaktadır.

Çalışmamızda emniyet numarası ve parmakizi yöntemlerini kullanıcı çalışması ile test ederek birbirlerine göre güçlü ve zayıf yönlerini ortaya koymayı hedefledik. Bu doğrultuda görece birbirinden farkı olmayan 21'er kişilik iki grup ile kullanıcı çalışmamızı gerçekleştirdik.

Çalışmada iki farklı açık anahtar doğrulama yöntemini karşılaştırırken, aynı zamanda SIGNAL uygulamasının eski ve yeni versiyonlarını da kullanışlılık ve güvenlik penceresinden karşılaştırma imkanı bulduk. Yaptığımız çalışmanın sonucunda, emniyet numarası ile açık anahtar doğrulama yönteminin doğrulama süresi ve kolaylığı açısından parmakizi ile açık anahtar doğrulama yöntemine göre daha avantajlı olduğunu gösterdik.

1.1 Literatür Araştırması

Açık anahtar doğrulama, çok yeni bir konsept olmasa da, tek amacı internet üzerinden anlık mesajlaşma gerçekleştirmek olan bir kullanıcıdan bu doğrulamayı gerçekleştirmesini beklemek, yeni bir durum olarak görülebilir. Uçtan uca şifreli e-mail gönderip almaya yarayan PGP'nin kullanılabilirliğini ölçen Whitten ve Tygar'ın kullanıcı çalışmasının sonuçları, PGP'nin neden yaygın bir şekilde kullanılmadığını anlamamıza yardımcı olmaktadır [1].

Steve Sheng ve diğerlerinin çalışmasında güncel PGP'nin kullanışlılığını ölçmek adına gerçekleştirilen kullanıcı çalışması ile PGP 9 versiyonu test edilmiş ve sonuçlar PGP 5 üzerine yapılan çalışmaların sonuçlarıyla kıyaslanmıştır [10]. Gerçekleştirilen kullanıcı çalışması, kullanıcıların anahtar çifti üretmeleri, diğer kullanıcıların açık anahtarlarını edinmeleri, açık anahtarları doğrulamaları, e-posta şifrelemeleri, e-posta imzalamaları, gelen şifreli e-postanın şifresinin çözülmesi, dijital imza doğrulamaları ve açık ve gizli anahtarların yedeğini kaydetme konularını kapsamıştır.

Ruoti ve diğerlerinin çalışmasında "Mailvelope" isimli PGP destekli tarayıcı eklentisinin kullanışlılığı ele alınmıştır [11]. Bu doğrultuda yapılan kullanıcı çalışmasının sonuçları, PGP'nin ilk ortaya çıktığından bu yana geçen zaman sonrasında, PGP ve araçlarının halen kullanışlılık yönünden çoğu insana hitap edemediğini göstermiştir. Kullanıcı çalışmasına katılan 10 çiftten, yalnızca biri adımları başarılı bir şekilde tamamlayıp birbirleri arasında şifreli e-posta gönderip alabilmiştir.

Ruoti ve diğerlerinin 25 çift yani toplamda 50 katılımcı ile gerçekleştirdikleri bir diğer kullanıcı çalışmasında, kullanıcılardan birbirleri arasında Pwm, Tutanota ve Virtru üzerinden şifreli e-posta gönderip almaları istenmiştir [12]. Bu üç sistem birbiri ile karşılaştırılmış ve kullanıcıların şifreli e-posta kullanımı konusundaki eğilimleri araştırılmıştır. Sonuç olarak güvenlik detaylarının kullanıcılardan gizlenmesinin kullanıcıların sisteme olan güvenini azalttığı, şifreli e-posta gönderip alabilmek için tercih edilen sistemin komple bir sistem olması yerine mevcut e-posta hesaplarına entegre bir sistem olmasının ağırlıklı olarak tercih edildiği ve iyi tasarlanmış eğitim materyallerinin, kullanıcıların sistemi doğru bir şekilde kullanmalarına önemli katkı sağladığı ortaya konulmuştur.

Frosh ve diğerlerinin çalışması, Signal protokolünün detaylı bir analizini ortaya koymaktadır [13]. 2008'de yayınlanan "Security and Usability" isimli kitap, güvenli bir sistem tasarlanırken kullanışlılığın önemini konu edinir [14]. Bu kitabın yazarlarından Cranor, ayrıca makalesinde güvenlik problemlerinin, çoğunlukla bilgisayar sistemlerinin kullanışlılık sorunları dolayısı ile kullanıcıların kasıtsız hatalarından kaynaklanmasını tartışır [15].

Fahl ve Harbach'ın çalışmasında facebook messenger uygulaması ile şifreli mesajlaşma üzerine üç ayrı çalışma gerçekleştirilmiştir [16]. İlk çalışma, 514 kişinin katılımı ile sağlanan bir tarama çalışmasıdır. Bu çalışma sonucunda katılımcıların facebook messenger üzerinden yaptıkları yazışmaların şifreli olmasına yönelik bir ilginin olduğu tespit edilmiştir. Bu çalışmanın sonuçları üzerinden yapılan analizler ışığında ikinci çalışma olarak 96 kullanıcının katılımı ile laboratuvar ortamında bir kullanıcı deneyi tertip edilmiştir. Bu çalışmanın sonuçları, otomatik anahtar yönetimi ve anahtar kurtarma özelliklerinin önemini ortaya koymuştur. Son olarak ilk iki çalışmanın sonuçları göz önünde bulundurularak, facebook messenger ile şifreli yazışma için bir servis geliştirilmiş ve bu servis, kullanışlılığının ölçülebilmesi adına 15 katılımcı ile gerçekleştirilen bir kullanıcı çalışmasına tabi tutulmuştur. Son çalışmada, tüm kullanıcılar sunulan servis üzerinden başarılı bir şekilde şifreli mesajlaşma gerçekleştirmişlerdir.

Sutikno ve diğerlerinin çalışmasında üç ayrı anlık mesajlaşma uygulaması olan WhatsApp, Viber ve Telegram sundukları teknik özellikler, kullanışlılık ve güvenlik pencerelerinden karşılaştırılmıştır [17]. Sonuç olarak her ne kadar Telegram'ın daha iyi bir platform sunduğu vurgulansa da arkasında Facebook gibi bir sosyal medya devi bulunan WhatsApp uygulamasının dünya çapında en yaygın anlık mesajlaşma platformu olduğu belirtilmiştir.

Vaziripour ve Wu'nun çalışmasında anlık mesajlaşma platformları WhatsApp, Viber, ve Facebook Messenger'ın kimlik doğrulama özellikleri 36 çift yani toplamda 72 katılımcı ile gerçekleştirilen kullanıcı çalışmasıyla karşılaştırılmıştır [18]. Çalışmada ayrıca bu üç uygulamanın kullanışlılığı üzerinde de durulmuştur. Sonuçlar kimlik doğrulama başarı oranlarında Viber'ın bir adım önde olduğunu göstermiştir.

Rosler ve diğerlerinin çalışmasında kullanıcılarına uçtan uca şifreleme sağlayan Signal, WhatsApp ve Threema anlık mesajlaşma uygulamalarının grup mesajlaşma için kullandıkları yöntemler incelenmiştir [19]. Bu yöntemlerin halihazırdaki güvenlik seviyeleri, belli kriterleri karşılayıp karşılamadıklarına göre kıyaslanmıştır. Sonuçlar, bire bir mesajlaşmalardaki birçok güvenlik bileşenlerinin, grup mesajlaşmalarında geçersiz olduğunu ortaya koymuştur. Ayrıca bu uygulamaların grup mesajlaşmadaki güvenlik eksikliklerini giderebilmeleri adına bir takım önerilerde bulunulmuştur.

Schröder ve diğerlerinin çalışması, 28 kullanıcı üzerinde signal anlık mesajlaşma uygulamasının genel kullanışlılığı ve güvenliği üzerine, açık anahtar doğrulamasını da içerisine alan bir kullanıcı çalışmasını içermektedir [5]. Bu çalışmaya katılan 28 kullanıcının 21'i parmakizi karşılaştırmasında başarısız olmuştur. Signal uygulamasını temel alması açısından, bizim çalışmamızla oldukça ilgilidir.

Ayrıca TAN ve diğerlerinin çalışması, numara ve heksadesimal ifadeler de dahil olmak üzere çeşitli açık anahtar arayüzlerini birbiri ile karşılaştırmaktadır [20]. Burada 661 katılımcı internet üzerinden çalışmaya iştirak etmiştir. "Emniyet numarası" ve heksadesimal "parmakizi"ni de karşılaştırmış olması açısından bizim çalışmamızla benzer yönleri bulunmaktadır.

2. TEKNİK ARKA PLAN

2.1 Şifreleme

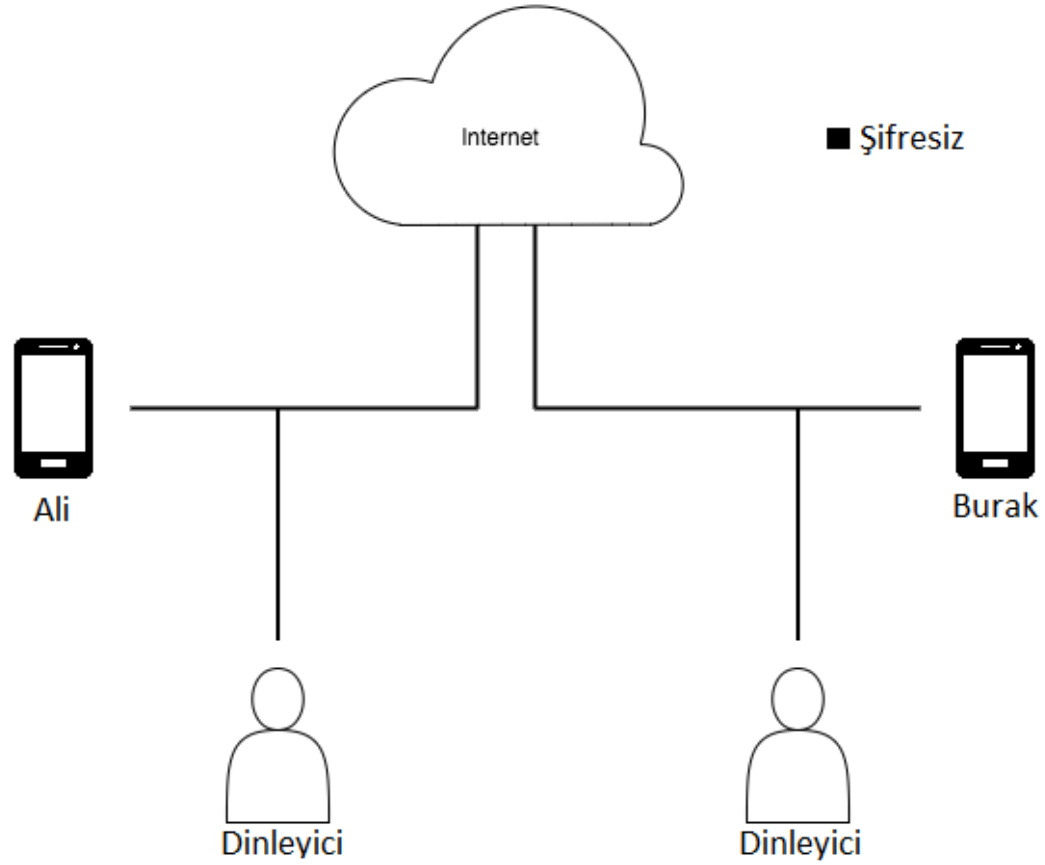
Şifreleme ham verinin bir anahtar vasıtası ile, şifreyi çözecek anahtara sahip olanlar dışındaki ilgisiz kişilerce anlaşılamayacak hale getirilmesi olarak tanımlanabilir. Temel amaç güvensiz ortamda tutulan veya güvensiz bir kanaldan iletilen verilerin içeriğinin gizlenmesidir. Verilerin, tutuldukları sisteme erişim veya iletildikleri hatta pasif veya aktif bir şekilde araya girmek suretiyle elde edilmesini engellemez, ancak içeriklerini ilgisiz kişiler için anlamlandırılması zor veya imkansız hale getirir.

2.2 Şifreleme Yaklaşımları

İnternet ortamı, herhangi bir kişinin veya kurumun kontrolünde değildir. İnternet servis sağlayıcılar son kullanıcılara internet hizmeti sunarken, bu hizmeti uçtan uca yalnızca kendi altyapısı üzerinden sağlamaz. Oluşan trafik çoğu zaman birden çok internet servis sağlayıcınn bağlantı noktalarından, hatta kıtalararası bağlantılar üzerinden sağlanabilir. Dolayısı ile son kullanıcı internet tarayıcısına gitmek istediği sitenin adresini tuşladığında, hedefe hangi altyapı üzerinden ulaştığını bilmez ve önemsemez ancak şifreleme olmaksızın gerçekleştirilen haberleşmenin içeriği, internet servis sağlayıcılara açık durumdadır.

Bunun dışında haberleşme trafiğinin geçtiği başta son kullanıcıya ait bilgisayar, mobil cihaz veya interneti kullanan diğer tüm cihazlar ve bunların bulunduğu yerel ağlar ile birlikte servis sağlayıcıların tüm ağ cihazları da, kötü niyetli kişilerce trafiği dinleyebilmeleri için potansiyel hedef olarak görülebilirler.

Aralarındaki görüşmenin ne kadar güvenli bir şekilde gerçekleştiğine dair hiçbir fikirleri olmayan iki arkadaş olan Ali ve Burak'ın, herhangi bir anlık mesajlaşma uygulaması üzerinden mesajlaştıklarını ele alalım. Şayet kullandıkları uygulama herhangi bir şifreleme sağlamıyorsa, biraz önce saydığımız noktaların herhangi biri üzerinden mesajlarının içerikleri elde edilebilecektir(Şekil 2.1). Dolayısı ile bu senaryonun gizlilik yönünden olumsuz bir durum teşkil ettiği söylenebilir.



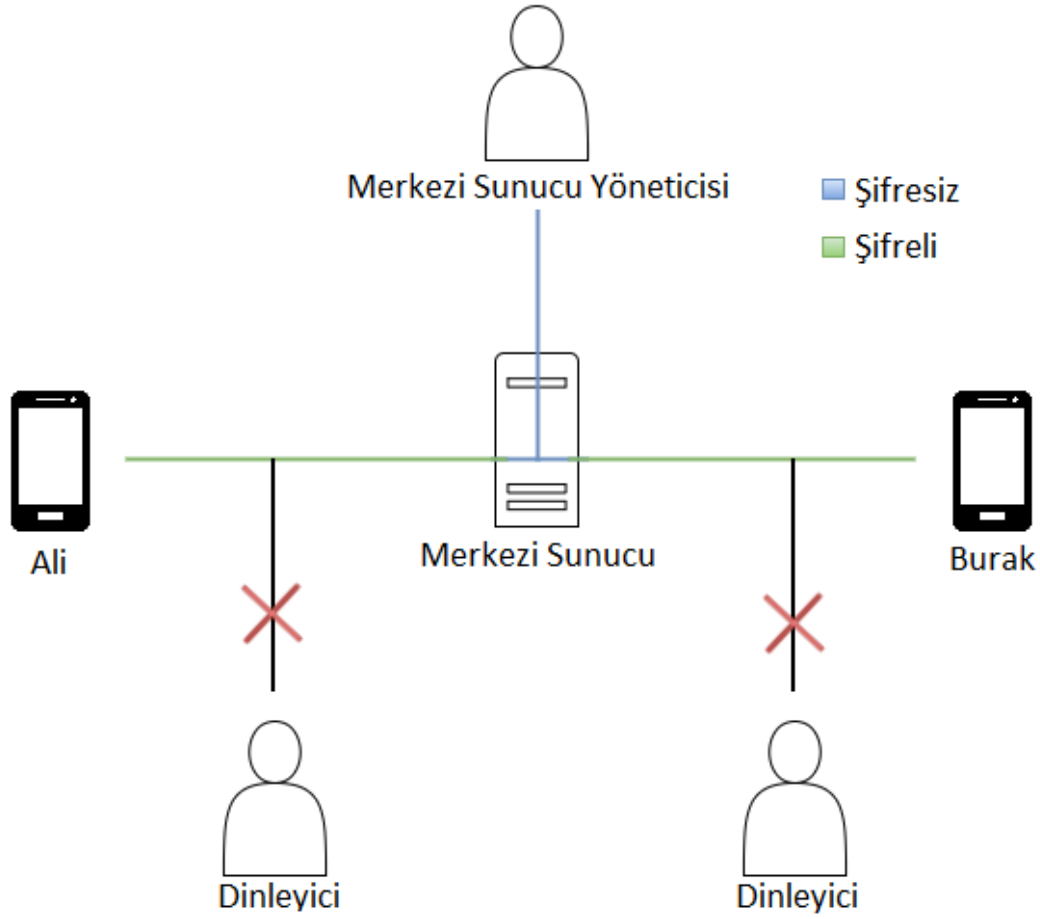
Şekil 2.1: Şifrelemenin olmadığı iletişim [21].

Kullanıcı ile merkezi sunucu arasında şifreleme

Bu senaryoda gizliliği sağlamak adına kullanıcılar ile uygulamanın merkezi sunucusu arasında şifreleme uygulanmıştır. Ali, Burak'a mesaj yollamak istediğinde, bu mesaj Ali'nin mobil cihazında şifrelenip merkezi sunucuya gönderilir. Merkezi sunucu şifreli mesaja şifre çözme işlemi uygular ve tekrar şifreleyip Burak'a gönderir. Son olarak bu şifreli mesaja, Burak'ın telefonunda şifre çözme işlemi uygulanır ve Burak Ali'nin mesajını görür. Bu yöntemin gerçeklenmiş hali, "Taşıma Katmanı Güvenliği"(TLS) [22]'nin uygulanmasına güzel bir örnek teşkil eder.

Merkezi sunucunun mesajlara şifre çözme işlemi uyguluyor olması, merkezi sunucuyu işleten ve yönetenlerin, bu mesajları okuyabiliyor, değiştirebiliyor ve saklayabiliyor olduğunu gösterir(Şekil 2.2). Bunun dışında kötü niyetli kişilerin, gerçekleştirdikleri saldırı ile merkezi sunucunun denetimini elde etmeleri, mesajlar üzerinde de denetimi elde etmeleri anlamına gelecektir. Ayrıca mesajlaşmanın gerçekleştiği ülkenin kanunları, güvenlik kurumlarına merkezi sunucudaki verileri talep etme hakkı tanıyor olabilir. Bu durumda son kullanıcı gönderdiği ve aldığı mesajlara şifreleme uygulandığını ve bunun bir sonucu olarak haberleşmenin

gizli bir şekilde sağlandığını düşünürken, aslında durum farklı olacaktır.

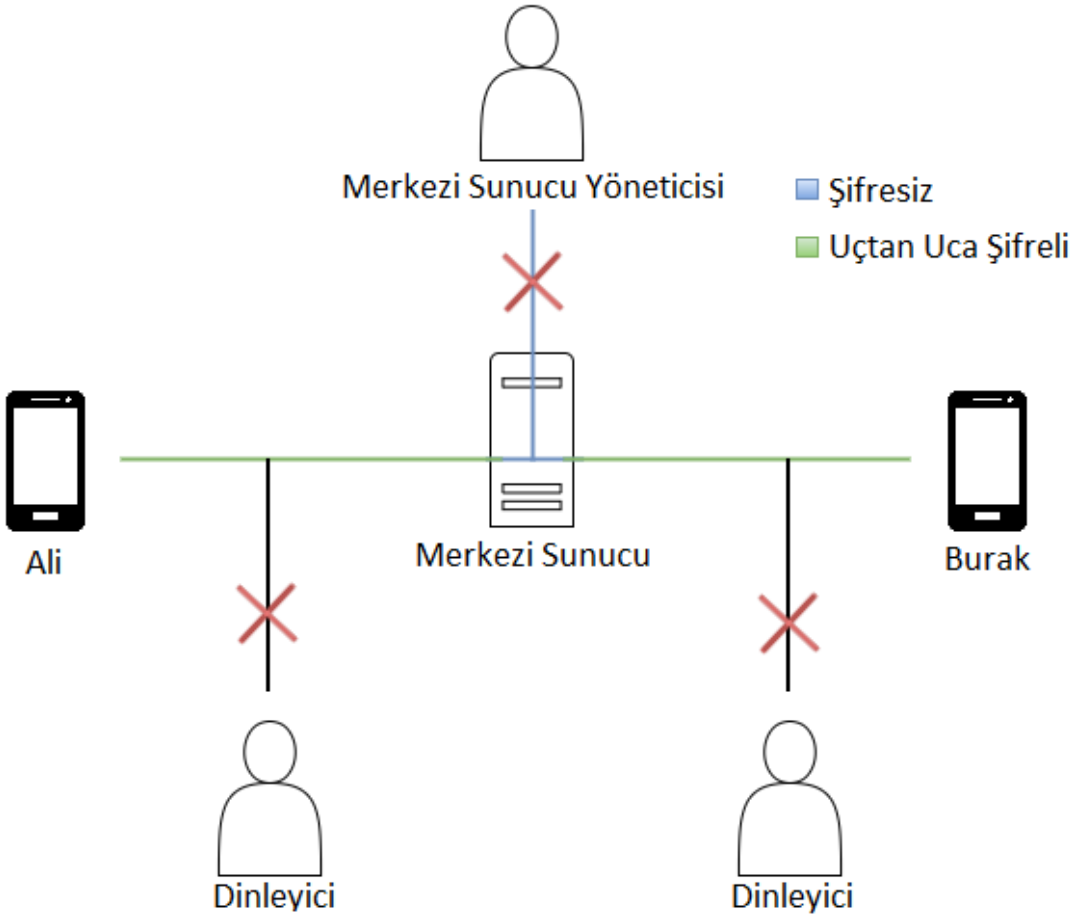


Şekil 2.2: Son kullanıcı ile merkezi sunucu arasında sağlanan şifreleme ile iletişim [21].

Uçtan uca şifreleme

Uçtan uca şifreleme, gizli anahtarların kullanıcıların aygıtlarından dışarıya çıkmadığından, dolayısı ile iletişim sağlayıcıların, mesaj içeriklerini okuyamadığından emin olmamızı sağlayan teknolojinin ismidir.

Bu senaryoda şifreleme ve şifre çözme işlemleri yalnızca son kullanıcıların mobil cihazlarında gerçekleştirilir. Merkezi sunucuda şifreleme veya şifre çözme işlemi gerçekleştirilmez. Dolayısı ile merkezi sunucu üzerinde yetki sahibi kim olursa olsun, iletilen mesajların içeriğini göremez(Şekil 2.3). Sunucunun rolü, mobil cihazlarda şifrelenmiş mesajların ilgili kullanıcılara iletimi ve açık anahtar setlerinin dağıtımının yapılmasından ibarettir. Bu tezde ele alınan genel konu olan açık anahtar doğrulama, uçtan uca şifrelemenin gün geçtikçe popüler hale gelmesi ve yaygınlaşması sonrası önem kazanmıştır.



Şekil 2.3: Uçtan uca şifreleme ile iletişim [21].

2.3 Uçtan Uca Şifreleme

Önceki başlık altında, uçtan uca şifrelemenin nasıl gerçekleştiğine değinilmişti. Bu bölümde ise uçtan uca şifrelemedeki problemlere ve tehdit modellerine değinilmekte ve bunlar için çözüm önerilmektedir.

Anahtar doğrulama problemi

Uçtan uca şifrelemenin hayata geçirilebilmesi için bir takım yan gereksinimlerin de karşılanmaları gerekmektedir. Bunların başında da anahtar değişiminin güvenli bir şekilde sağlanması gelmektedir.

Şifrelemenin gerçekleştirilebilmesi için, öncelikle iki tarafın birbirlerinin anahtarlarına erişebilmeleri gerekmektedir. Bu işlem iki tarafın direkt olarak internet üzerinden paylaşması şeklinde gerçekleştirilemez, çünkü bu yolla, iki taraf da karşı tarafın anahtarı olarak ellerine

ulaşan anahtarın, gerçekte kime ait olduğundan emin olamaz. Ali, Burak'ın anahtarını talep ettiğinde araya giren birinin, gizli kalması gereken oturum anahtarını ele geçirebileceği söylenebilir.

Ancak "açık anahtarlı şifreleme"(PKC) sayesinde, oturum anahtarları internet üzerinden açık bir şekilde gönderilmez. Buna rağmen tüm problemler tam anlamıyla çözülmüş değildir. Çünkü açık anahtarlı şifreleme için gerekli olan açık anahtarların dağıtımı hala bir problem olarak beklemektedir. Ali, Burak'ın anahtarını talep ettiğinde kendisine ulaşan anahtarın araya giren biri tarafından okunabilmesi artık bir problem olmasa da, aynı kişi tarafından değiştirilmesinin önüne henüz geçilebilmiş değildir.

Bu problemi aşabilmek için de anahtar dağıtımının yapılacağı bir merkezi sunucuya ihtiyaç olacaktır. Ali güvendiği merkezi sunucudan şifreli haberleşme ile Burak'ın anahtarını edinir. Gizli anahtarlar Ali'nin ve Burak'ın mobil cihazlarından dışarıya çıkmaz. Sonrasında ise iletişim Ali ile Burak arasında uçtan uca şifreleme ile sağlanır. Bu durumda merkezi sunucuda şifre çözme işlemi uygulanmadığı için, merkezi sunucuyu yöneten kişi, Ali ile Burak arasındaki mesajlaşmanın içeriğini göremez.

İlk kullanımda koşulsuz güven problemi

Uçtan uca şifrelemede açık anahtarların dağıtımı için bir merkezi sunucu kullanmanımına karar verildiğinde, kullanıcılar güvenlik açısından birtakım zorluklarla karşılaşmaya devam edeceklerdir. Bu zorlukların başında merkezi sunucunun güvenilirliği gelmektedir.

Ali, Burak'a mesaj göndermek istediğinde en başta, merkezi sunucudan Burak'ın açık anahtarını talep eder ve akabinde sunucudan açık anahtarı içeren bir cevap alır. Bu anahtar Ali'nin mobil cihazına Burak'ın açık anahtarı olarak kaydedilir. Ali daha sonra Burak'a yeniden mesaj göndermek istediğinde, daha önce kaydedilmiş olan açık anahtar kullanılır.

Merkezi sunucuda şifre çözme işleminin gerçekleşmiyor olması Ali ile Burak arasındaki mesajlaşmanın içeriğini tüm üçüncü şahıslardan gizli tutacaktır. Ancak Ali, Burak'a ilk mesajı göndermek için Burak'ın açık anahtarını merkezi sunucudan istediğinde, kendisine gönderilen açık anahtarı gözü kapalı bir şekilde kabul edecektir.

Bu aşamada Ali, bu anahtarın gerçekte kime ait olduğunu doğrulayabilecek imkana sahip değildir. Şayet merkezi sunucu bir şekilde Burak'ın açık anahtarı yerine farklı bir açık anahtar gönderirse, Burak ile güvenli bir iletişim tesis edilememiş olacaktır.

Mevcut oturumun güvenliği

Bu senaryoda iki kullanıcı halihazırda bir mesajlaşma oturumuna sahipken, saldırgan merkezi sunucu üzerinden kendi anahtar setini kullanıcılara gönderir. Ali Burak'a mesaj göndermek istediğinde, mobil cihazında halihazırda Burak'a ait olan kayıtlı açık anahtarın artık

geçerli olmadığı anlaşılır ve mesajlaşma, merkezi sunucudan gelen yeni açık anahtar üzerinden devam eder.

Burada problem, aynı durumun Burak'ın anlık mesajlaşma uygulamasını telefonundan silip yeniden kurması veya farklı bir mobil cihaz ile uygulamaya kayıt olmasında da gerçekleşiyor olmasıdır. Bu durumda da aynı araya girme saldırısında olduğu gibi, Burak'ın hesabı için yeni bir anahtar seti oluşturulup sunucuya yüklenir ve Burak'a mesaj göndermek isteyen kişilere bu yeni açık anahtar iletilir. Ali'nin ilk anda bunun bir araya girme saldırısı mı yoksa Burak'ın gerçekten de uygulamaya yeniden kayıt mı yaptırdığını anlaması mümkün değildir.

Bu durumda uçtan uca şifreleme kullanan anlık mesajlaşma uygulamaları, birbirlerinden farklı politikalar izleyebilirler. Örneğin WhatsApp uygulaması güncel versiyonlarında açık anahtar değişimlerinde, varsayılan olarak doğrudan yeni anahtarı eskisinin yerine kaydeder ve kullanıcıya bir uyarı iletmez. Şayet kullanıcı açık anahtar değişimlerinden haberdar olmak istiyorsa, güvenlik ayarlarından bu bildirimlerin gösterimini kendisi aktif hale getirmelidir(Şekil 2.4).

Kullanıcı çalışmamızda kullandığımız Signal uygulaması ise eski ve yeni versiyonlarında farklı politikalar izlemektedir. Çalışmamızda geçmiş versiyon olarak kullandığımız 3.13.0 versiyonunda, mesajlaşılan kişi için sunucudan yeni bir açık anahtar geldiğinde mesajlaşma durdurulur ve mesajlaşmanın devam etmesi için kullanıcının açık anahtar değişimi uyarısını okuyup, uyarıdaki kabul et seçeneğini seçmesi beklenir. Uyarıda kullanıcıya açık anahtar doğrulaması yapması tavsiye edilir. Güncel versiyon olarak kullandığımız 4.11.5 versiyonunda ise anahtar değişimi uyarısı mesajlaşma ekranında gösterilir ve mesajlaşma durdurulmaz("Kullanıcı çalışması" başlığı altında detaylı bir şekilde incelenmiştir).

Yaptığımız kullanıcı çalışmasında ise benzer bir senaryo oluşturulup, SIGNAL uygulamasının bu tehditlere karşı uyguladığı önlemlerin kullanıcı tarafında ne kadar etkili olduğunu araştırdık.

Farklı kanaldan kimlik doğrulama çözümü

Ali'nin anahtar değişiminin hangi sebepten gerçekleştiğini anlaması için, yeni anahtarın gerçekten de Burak'a ait olup olmadığını anlaması gerekmektedir. Bunun için de en kolay yol, anlık mesajlaşma uygulamasının dışında farklı bir kanal üzerinden doğrulama yapmaktır.

Bu işlem genellikle uygulama içerisinde sunulan kimlik doğrulama sayfasındaki kod/kodlar ve bunların görselleştirilmiş hali olan karekodlar ile gerçekleştirilir. Ancak sözkonusu merkezi sunucu üzerinden gelen anahtarın doğru olup olmadığını anlamak olduğundan, bu kodlar uygulama dışında farklı bir kanal üzerinden karşı tarafa gönderilmelidir.

Şekil 2.4: WhatsApp güvenlik bildirimleri ayarları.

Bu kodların karşılaştırılması için en güvenilir yol, mesajlaşılacak kişi ile yüzyüze karşılaştırma yapılmasıdır. Ancak bu her zaman mümkün olmayabilmektedir. Bu gibi durumlarda da kodun elektronik posta, kısa mesaj, sesli görüşme ve görüntülü görüşme gibi diğer kanallar üzerinden iletilip karşılaştırma yapılması mümkündür.

2.4 Temel Kavramlar

Diffie-Hellman anahtar değişimi

1976 yılında Whitfield Diffie ve Martin Hellman tarafından önerilen Diffie-Hellman Anahtar Değişimi (DHAD) [23], açık literatürde yayınlanan ilk asimetrik tasarımdır. Temel anahtar dağıtım problemine pratik bir çözüm, yani güvensiz bir kanal üzerinden iletişim yoluyla iki tarafın ortak bir gizli anahtar elde etmesini sağlar [24]. Akıllıca ve basit yapısı, Güvenli Kabuk (SSH) , Taşıma Katmanı Güvenliği (TLS) ve Internet Protokolü Güvenliği (IPSec) gibi çok çeşitli protokollerin [25] temelini oluşturmuştur. Bu bölümde Diffie-Hellman'ın basit bir açıklamasına ve genel bir zaafiyetine değinilecektir.

Diffie-Hellman, ilki iki uç noktanın kullanacağı genel parametreleri kurmak için kullanılan ve ikincisi anahtar değişimini gerçekleştiren olmak üzere iki protokolden oluşur. Kurulum protokolü üç adımdan oluşur(Şekil 2.5) [24]:

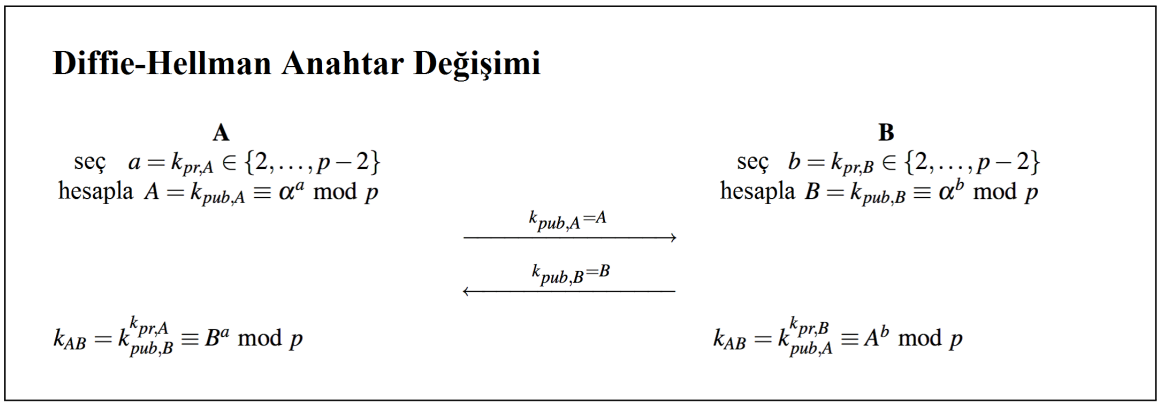
Diffie-Hellman Kurulum

1. Büyük bir asal sayı seç (p).
2. Bir tamsayı seç ($\alpha \in \{2, 3, \ldots, p-2\}$).
3. Seçilen parametreleri gönder (p ve α).

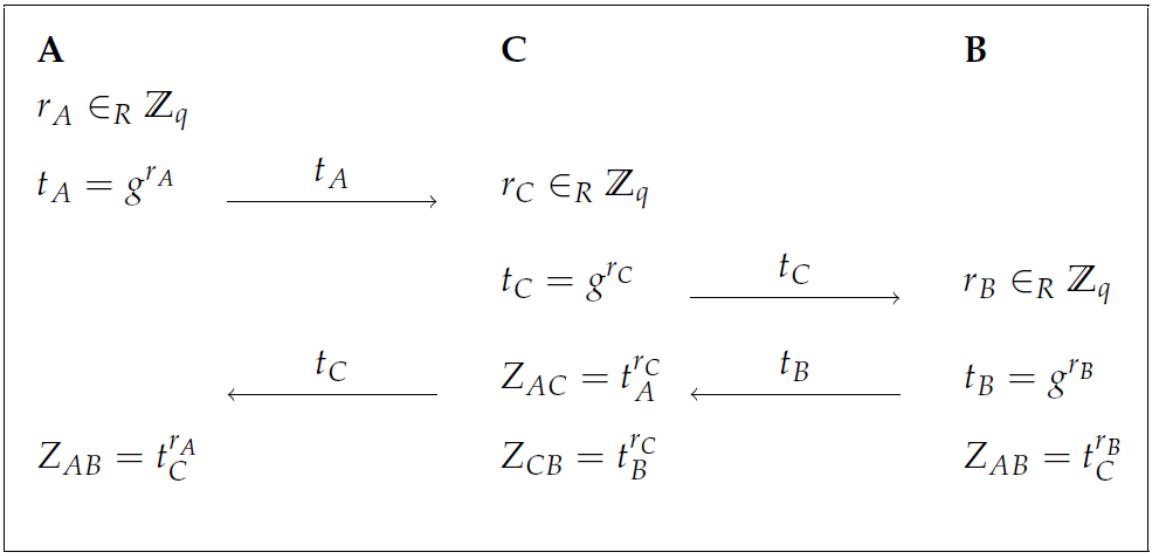
Şekil 2.5: Diffie-Hellman kurulum [24].

Kurulum protokolünden seçilen bu iki değer alan adı parametresi olarak adlandırılır. Ali ve Burak hem p hem de alfa genel parametrelerini bilmeleri durumunda, DH anahtar değişim protokolü ile paylaşılan bir gizli anahtarı k_{AB} oluşturabilirler(Şekil 2.6).

Öte yandan, temel Diffie-Hellman Anahtar Değişimi protokolü gönderilen mesajların herhangi bir doğrulamasını sunmaz. Kimlik doğrulaması olmaması dolayısı ile araya girme saldırısı gerçekleştiren herhangi biri tarafından manipüle edilebilmektedir. Bir saldırgan, kendisini Ali ile Burak'ın iletişim kanalı arasına yerleştirebilir ve hem Ali'yi hem de Burak'ı diğer tarafa taklit edebilir. Bu olduğunda, Ali Burak ile, Burak da Ali ile DHAD yaptığını düşünecek. DHAD tamamlandıktan sonra, birbirleriyle değil saldırgan ile gizli anahtar oluşturmuş olacaklardır. Aşağıdaki şekil(Şekil 2.7), saldırganın A ve B arasındaki DHAD'ye nasıl saldırabileceğini göstermektedir:



Şekil 2.6: Diffie-Hellman anahtar değişimi [24].



Şekil 2.7: Temel Diffie-Hellman anahtar değişimine saldırı [25].

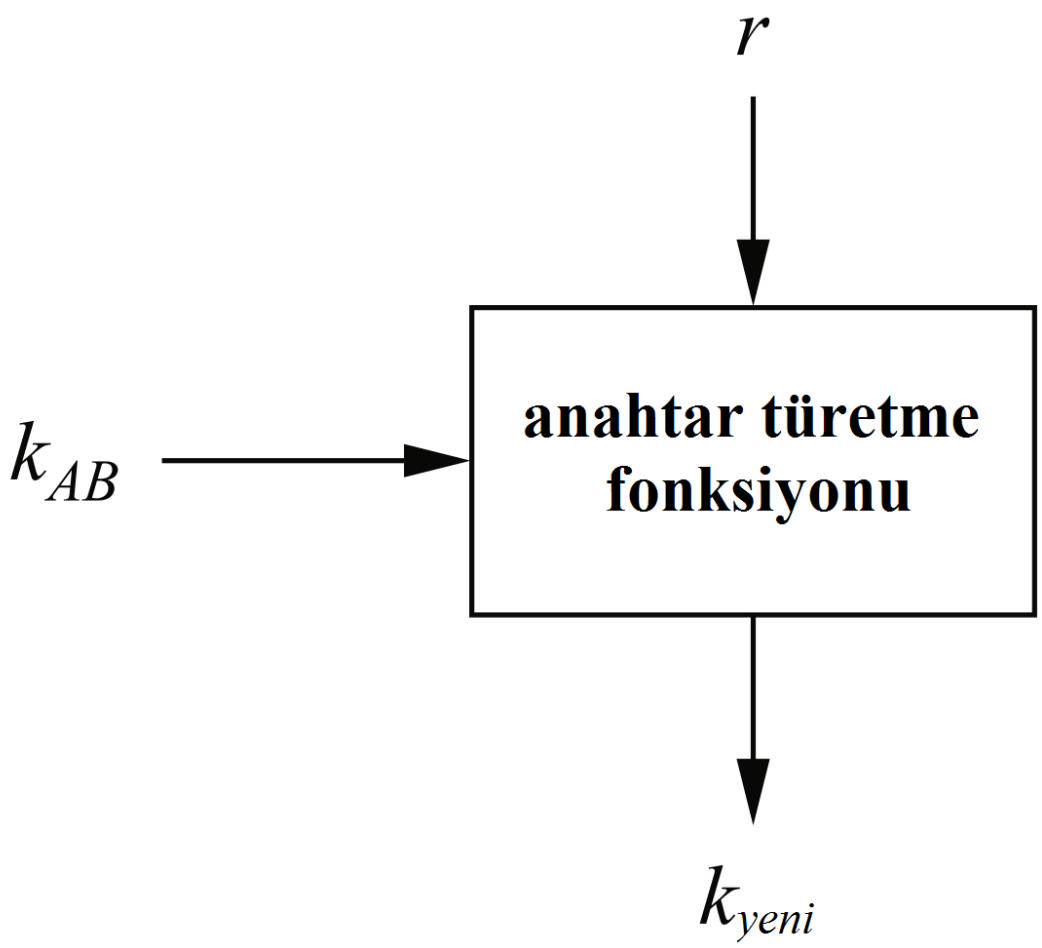
Anahtar türetme fonksiyonu

Anahtar Türetme Fonksiyonu (KDF), iki veya daha fazla taraf arasında gizli bir anahtar oluşturmayı ele alan anahtar kuruluşunun bir parçasıdır. Anahtar kuruluşu, anahtar taşıma ve anahtar uzlaşma yöntemleri olmak üzere iki farklı yönteme ayrılabilir [24]. Signal Protokolü'nün algoritmasının bir parçası olarak kullanıldığı için bu bölümde yalnızca anahtar uzlaşma yönteminden bahsedilecektir.

Çoğu güvenlik sisteminde, sadece sınırlı bir süre için geçerli olan şifreleme anahtarlarının kullanılması istenir. Bu anahtarlara genellikle oturum anahtarları veya geçici anahtarlar denir. Bir kriptografik anahtarın kullanıldığı sürenin sınırlandırılmasının nedeni, anahtarın açığa çıkması durumunda daha az zarar görülmesi gibi bir avantajının olmasıdır [24]. Diğer

bir neden de bir saldırganın elinde, üretilen her anahtarın altında çalışmak için daha düşük miktarda şifrelenmiş trafiğe sahip olmasını sağlamaktır.

Anahtarı sık sık güncellemek önemlidir, böylece saldırganlar şifreli metni deşifre edemez. Anahtar yenileme, anahtar türetme fonksiyonunu ile yeni oturum anahtarları türetmek için halihazırda var olan gizli anahtarı kullanarak yapılabilir. İki kullanıcı arasındaki gizli k_{AB} ile birlikte gizli olmayan r parametreleri anahtar türetme fonksiyonuna sokularak yeni anahtar üretilmiş olur(Şekil 2.8).



Şekil 2.8: Anahtar türetim fonksiyonunun çalışma şekli [24].

Anahtar türetme fonksiyonu, tek yönlü çalışıyor olmalıdır. Fonksiyonun tek yönlü olması, bir saldırganın gelecekteki tüm anahtarları hesaplayabilmesine sebebiyet verecek olan gizli anahtarın açığa çıkmasına engel olmaktadır [24].

2.5 Kullanılan İstatistiksel Testler

Kullanıcı çalışmalarından elde edilen verilerin doğru şekilde yorumlanmaları hayati önem arz etmektedir. Bu veriler çalışmanın amacına ve verinin şekline göre değişkenlik gösteren yöntemlerle incelenmelidirler. Yapılan kullanıcı çalışmasıyla bir sistemdeki hatalar tespit edilmeye çalısılıyorsa karşılaşılan hataların önem düzeyi belirlenmeye çalısılmalı, çalışmamızda olduğu gibi iki sistemin kullanışlılıkları karsılaştırılmaya çalışılıyorsa istatistiksel testler kullanılmalıdır.

Deney sonuçlarının dogru yorumlanabilmesi için hem amaca, hem de veriye uygun istatistiksel testlerin kullanılması çok önemlidir. Çalışmada yapılan degisiklikler, hangi istatistiksel testin uygulanması gerektiğini de etkileyecektir. Örnegin çalışmanın amacı sistemler arasındaki ilişkiyi bulmaksa korelasyon katsayısı hesaplanmalıdır. Çalışmanın amacı sistemlerin farkını bulmaksa karşılaştırma testleri uygulanmalıdır. Benzer sekilde, verilerin sayısal ya da sözel oluşu, normal dagılıma uyup uymamaları, çalışmada yer alan grupların sayısı gibi degişkenler de uygulanacak istatistiksel testin belirlenmesi için kullanılan bilgilerdir. Sayısal veriler normal dagılıma uyuyorlarsa parametrik testler, uymuyorlarsa parametrik olmayan testler uygulanmalıdır. Eger veriler sadece bir grubun verilen görevleri her sistem için yapmasıyla elde edildiyse grup içi testler, her sistemi farklı kullanıcıların bulundugu gruplar test ettiyse gruplar arası testler kullanılmalıdır [26].

Bu tezde, karşılaştırılan sistemlerin kullanışlılığını ölçmek adına görece birbirinden farksız iki kullanıcı grubu üzerinde kullanıcı çalışması yürütülmüştür. Çalışmadan elde edilen verilerin anlamlı olup olmadıklarını ortaya koyabilmek için, bu veriler istatistiksel testlere tabi tutulmuşlardır. Çalışmada kullanılan istatistiksel testler aşağıda kısaca özetlenmişlerdir.

Mann-Whitney U testi

Normal dağılım özelliği göstermeyen bir dağılımda iki bağımsız grup ortalamalarını karşılaştırmak amacıyla kullanılan parametrik olmayan bir istatistiksel test yöntemdir. Diğer bir deyişle, farklı kullanıcıların farklı sistemleri test etmeleri sonucu elde edilen verilerin aynı dağılımı gösterip göstermediğini belirleyebilmek için kullanılır.

Bu testte farklı kullanıcıların farklı sistemlerde elde ettikleri veriler bir araya getirilerek küçükten büyüğe sıralanırlar. Gerçek değerler, sıralarını belirten değerlerle değiştirilir ve bu yeni değerler kullanılarak her bir grubun toplamı ayrı ayrı hesaplanır. Elde edilen toplam değerleri kullanılarak her iki grubun U değerleri hesaplanır ve büyük olan U değeri belli bir eşik değerinden büyükse verilerin aynı dağılımı göstermediğine karar verilir [26].

Ki-kare (x^2) testi

Ki-kare testi, gözlenen frekanslar(G) ile beklenen frekanslar(B) arasındaki farkın istatistiksel olarak anlamlı olup olmadığını anlamak için kullanılan istatistiksel test yöntemidir. Ki-kare

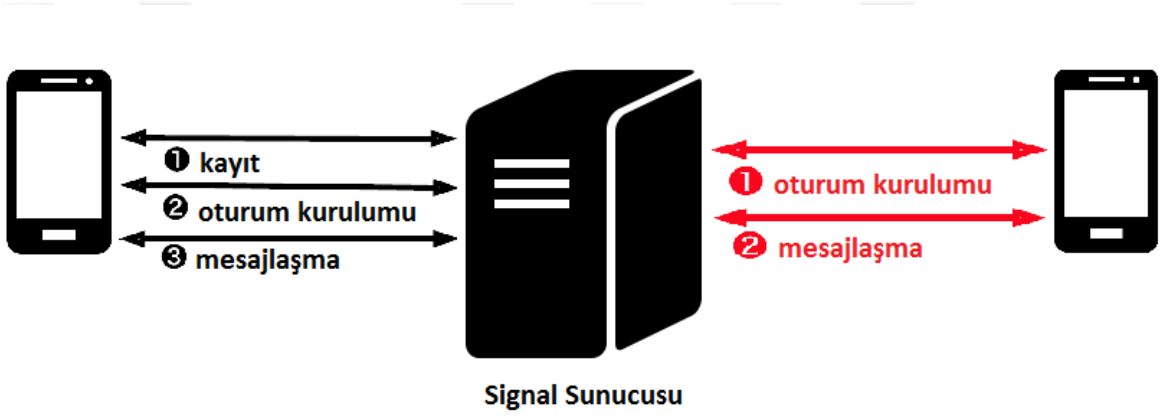
testi, genellikle iki bağımsız kriteri test etmek için kullanılır. Başlangıç hipotezi (H_0), iki kriterin bağımsız olduğunu; araştırma hipotezi(H_A) ise, iki kriterin arasında ilişki olduğunu ifade eder.

Bir ki-kare dağılımı diğer ki-kare dağılımlarından serbestlik derecelerine(sd) göre ayrılır. Kareleri alınıp toplandığında , ki-kare dağılımı gösteren bağımsız standart normal değer sayısına serbestlik derecesi denir. Bir ki-kare dağılımının ortalaması, dağılımın serbestlik derecesine ve varyansı, serbestlik derecesinin iki katına eşittir. Gruplar birbirinden bağımsız olmalıdır. Bağımlı gruplara, normal ki-kare testi uygulanamaz. Bu gruplar için, ki-kare testi ayrı bir yöntemle yapılır.

3. SIGNAL ANLIK MESAJLAŞMA UYGULAMASI

3.1 Signal Protokolü

Signal protokolü en basit şekilde özetlenecek olursa üç kısma ayrılır. İlk kısım merkezi sunucuya kayıt, ikinci kısım mesajlaşma oturumunun kurulması, üçüncü kısım ise mesajlaşma safhasıdır(Şekil 3.1).



Şekil 3.1: Signal protokolünün çalışma şeklinin basit bir gösterimi [5].

Mobil numaranızla SIGNAL sunucusuna kayıt olduğunuzda arka planda gizli ve açık anahtar setleri oluşturulur. Gizli anahtar mobil cihazın dışına çıkmaz. Mesajlaşma isteği gönderdiğinizde veya aldığınızda karşılıklı oturumun kurulabilmesi için gerekli anahtar setleri, ilk kurulum esnasında merkezi sunucuya yüklenir. Uzun dönem anahtar setleri, uygulama cihaz üzerinden kaldırılıp tekrar kurulmadığı sürece değişmez. Mesajlaşma oturumu kurulduktan sonra, mesajlaşılan kişilerin kimlik anahtarları uygulama içerisinde saklanır. Bu anahtarlara uygulama içerisinden erişilebilmektedir. Açık anahtar doğrulama sırasında, işte bu anahtarların doğru kişilere ait olup olmadığı tespit edilmeye çalışılır.

Merkezi signal sunucusunun görevi ise, mesajların iletimi ve açık anahtar setlerinin dağıtımıdır. Şifreleme veya şifre çözme işlemleri bu sunucu üzerinde gerçekleştirilmez. Kullanıcı birisiyle mesajlaşmak istediğinde merkezi sunucudan ilgili kişinin kimlik anahtarını da içeren bir dizi anahtarı talep eder. Elde ettiği bu anahtar seti ile mesajları şifrelemede kullanılacak simetrik bir anahtar oluşturur ve mesajlaşacağı kişinin kimlik anahtarını saklar. Karşı tarafta

da aynı işlem gerçekleşir ve aynı simetrik anahtar üretilmiş olur. Daha sonra bu simetrik anahtar üzerinden mesajlaşma sağlanır.

Signal protokolü, Çift-Mandal (Double Ratchet) algoritması, 3'lü Diffie-Hellman el sıkışması ve ön anahtarlardan meydana gelmiştir. Çift-Mandal ise, KDFye dayalı ve Diffie-Hellman anahtar değişimini temel alan bir mandaldan oluşur. Bu da protokolün, OTR(Kayıt Dışı) protokolü ile aynı güvenlik özelliklerine, aynı zamanda da bazı durumlarda daha güçlü veya yeni özelliklere sahip olmasını sağlar.

Double ratchet algoritması

Signal Protokolü tarafından, iki tarafın kullandığı paylaşılan bir gizli anahtara dayalı şifrelenmiş mesajların gönderilip alınması amacıyla kullanılır. Taraflar genellikle, paylaşılan gizli anahtar üzerinde anlaşmak için çeşitli anlaşma protokollerinden birini kullanırlar [27]. Signal Protokolü ise bunun için, bir sonraki başlığımız da olan X3DH Anahtar Anlaşması [28] Protokolünü kullanmaktadır.

Çift Mandal Algoritması'nın bileşenleri aşağıdaki gibi sıralanabilir:

- Anahtar türetme fonksiyonu zinciri [27],
- Simetrik anahtar mandalı,
- Diffie-Hellman mandalı.

Anahtar türetme fonksiyonu zinciri

Anahtar türetme fonksiyonu zinciri, Çift Mandal algoritmasında temel bir kavramdır [27]. Gizli, rastgele bir KDF anahtarı ve bazı giriş verilerini alır ve bir sonraki zincir için yeni bir KDF anahtarı olarak kullanılan bir anahtarın yanı sıra mesajlar için bir anahtar çıktısı döndürür. Anahtarın bilinmemesi şartıyla çıkış verileri daima rastgele ve ayırt edilemezdirler. Burada KDF anahtarı olarak kullanılan "r"(bölüm 2.4.2) parametresinin gizli veya rastgele bir sayı olması gerekmemektedir. Çünkü her halükarda KDF çıktısı, giriş verilerinin ve anahtarların güvenli bir şifrelenmiş özetini sağlar.

Simetrik anahtar mandalı

Simetrik anahtarlı mandal, gönderme ve alma zincirleri için KDF zincirlerini kullanır. Üretilen anahtarlar, mesajları şifrelemek veya şifre çözmek için kullanılan eşsiz mesaj anahtarlarıdır [27].

Diffie-Hellman mandalı

Daha önce de belirtildiği gibi Çift Mandal Algoritması, simetrik anahtar ve Diffie-Hellman mandallarının birleştirilmesi ile oluşturulur. Şayet Çift Mandal Algoritması, Diffie-Hellman mandalını gönderme ve alma zinciri anahtarları için yeni zincir anahtarlarını hesaplamada kullanmasaydı, bir saldırgan zincir anahtarlarından birini çalabilir ve daha sonra gelecekteki tüm mesaj anahtarlarını hesaplayabilir, dolayısı ile gelecekteki tüm mesajların şifresini de çözebilirdi [27].

X3DH anahtar anlaşma protokolü

Çift Mandal algoritması, iki taraf arasında bir anlaşma protokolüne ihtiyaç duyar. X3DH, açık anahtarlara dayanarak birbirini karşılıklı olarak doğrulayan iki taraf arasında paylaşılan bir gizli anahtar oluşturmak için kullanılır [28] ve aynı zamanda hem ileri yönlü gizlilik hem de kriptografik inkar edilemezliği sağlar.

Bu anahtar anlaşma yöntemi, bir kullanıcının çevrimdışı olduğu ancak bir sunucuya bilgisini yayınladığı eşzamansız senaryolar için tasarlanmıştır. Başka bir kullanıcı, bu bilgiyi çevrimdışı olan kullanıcıya şifreli veri göndermek için kullanır ve gelecekteki iletişimler için de paylaşılan bir gizli anahtar oluşturur [28].

3.2 Uygulama İçerisinden Açık Anahtar Doğrulama

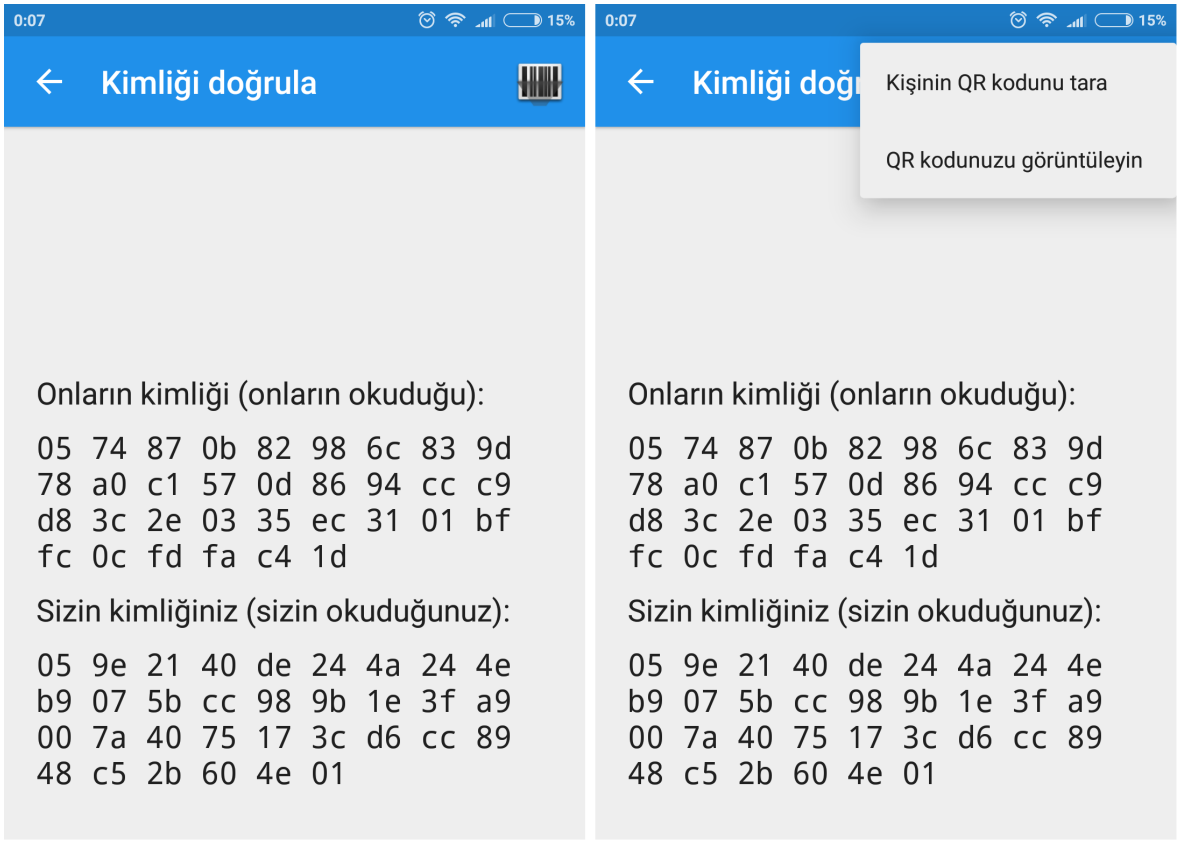
Parmakizi ile açık anahtar doğrulama

Parmakizi ile açık anahtar doğrulama yönteminde sisteme kayıt yaptırmış her kullanıcının, açık anahtarından türetilmiş bir parmakizi bulunmaktadır. Bu parmakizinin gösterimi ise 66 karakterden oluşan bir onaltılık (heksadesimal) ifade şeklindedir(Şekil 3.2).

Kimlik doğrulaması için öncelikle doğrulama sayfasına erişmek gerekmektedir. Buraya ulaşıldığında kullanıcının karşısına kendi parmak izi ve mesajlaştığı kişinin parmak izi çıkmaktadır. Doğrulama işleminin başarılı bir şekilde gerçekleştirilebilmesi için kullanıcının kendi parmakizinin, mesajlaştığı kişinin cihazının ilgili kısmındaki parmakizi ile, aynı şekilde mesajlaştığı kişinin parmakizinin de kullanıcının kendi cihazının ilgili kısmındaki parmak izi ile birebir aynı olduğundan emin olunması gerekmektedir.

QR kodlar aracılığı ile de karşılaştırma yapmak mümkündür. Bunun için her kullanıcının parmakizi'ne karşılık gelen bir QR kod bulunmaktadır. Bu kodu görüntüleyebilmek için, kimlik doğrulama sayfasının sağ üst kısmındaki barkod simgesine dokunulur, ardından açılan menüde(Şekil 3.2) "QR kodunuzu görüntüleyin" sekmesi seçilir. Menüdeki diğer sekme olan "Kişinin QR kodunu tara" seçeneği ile de, mobil cihazda önceden yüklenmiş olması gereken "Barcode Scanner" uygulaması üzerinden, QR kodunu görüntülemiş bir kullanıcının

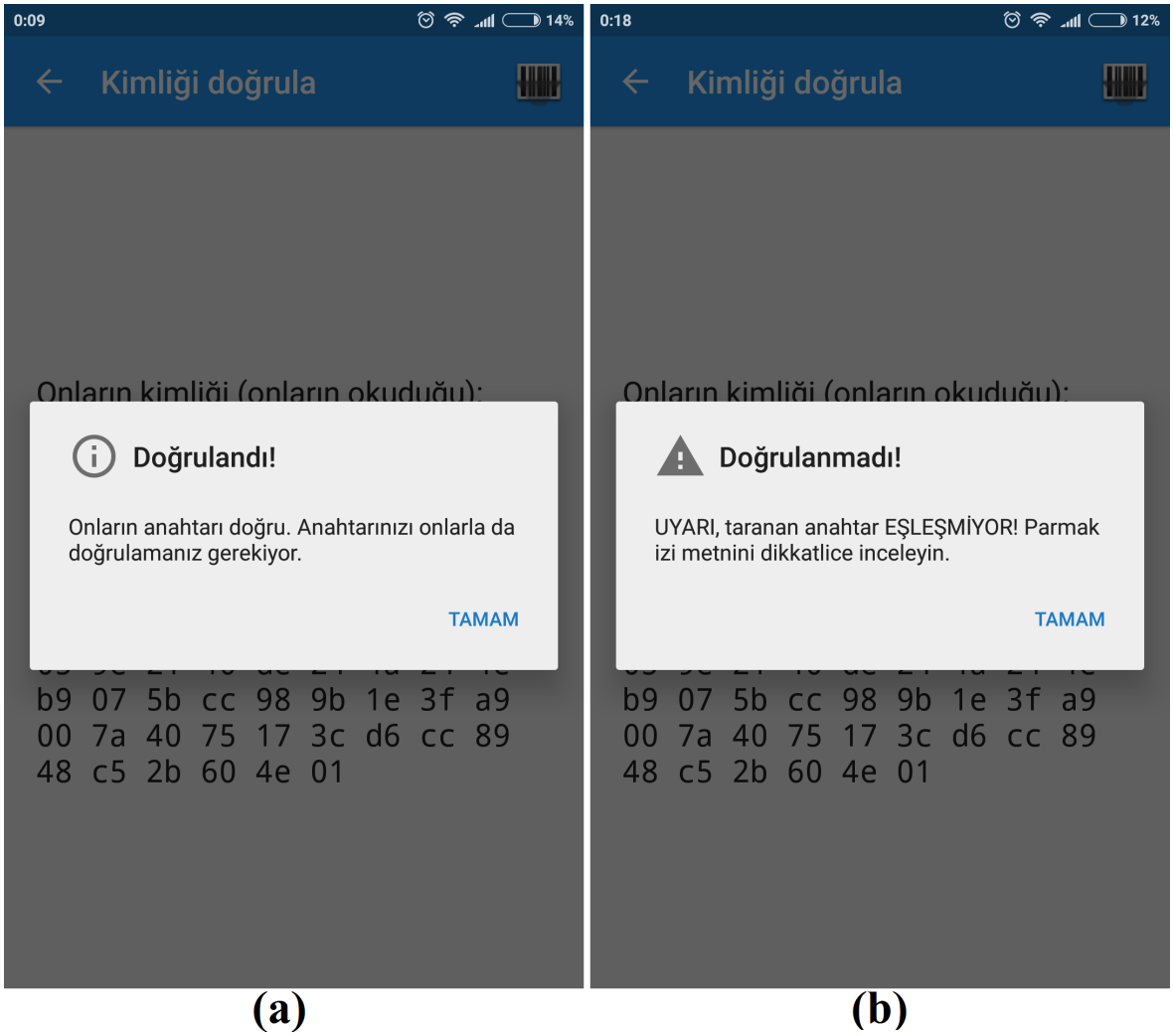
karekodu taratılabilir. QR tarama yolu ile parmakizi karşılaştırma işlemi sonucunda, doğru kişi ile mesajlaşıldığından emin olunması için, mesajlaşmaya taraf olan her iki kullanıcının da mesajlaştığı kişinin cihazındaki QR kodu taraması ve iki taramanın sonucunda da eşleşmenin görüldüğünden emin olunması gerekmektedir(Şekil 3.3).



Şekil 3.2: a) Kimlik doğrulama sayfasındaki parmakizi anahtarları(Alttaki kullanıcının, üstteki karşı tarafın parmakizi) ve barkod simgesi. b) Barkod simgesine dokunulduğunda açılan menü (parmakizi yöntemi).

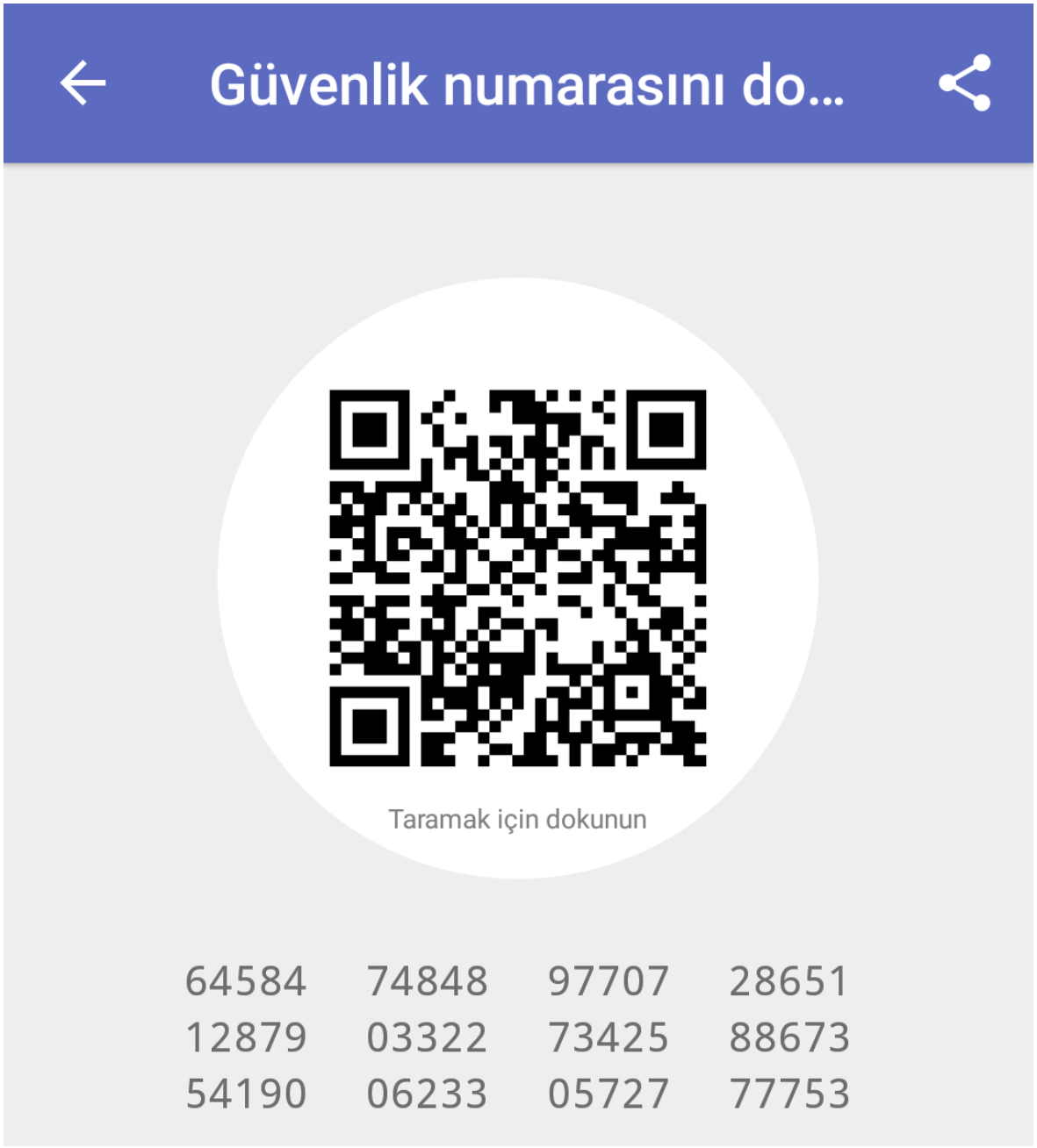
Emniyet numarası ile açık anahtar doğrulama

Emniyet numarası ile açık anahtar doğrulamada ise kullanıcının ve mesajlaştığı kişinin açık anahtarlarından türetilmiş bir tek emniyet numarası ile doğrulama sağlanmaktadır. Tahmin edileceği üzere üretilen bu emniyet numarası oturuma özeldir; yani bir kullanıcının mesajlaştığı her kişiyle farklı bir emniyet numarası bulunmaktadır. Bu emniyet numarası ise 12 adet 5 hanelik tamsayı, yani toplamda 60 rakam şeklinde ifade edilmiştir(Şekil 3.4).



Şekil 3.3: QR tarama sonuçları. a) Eşleşme durumu. b) Eşleşmeme durumu (parmakizi yöntemi).

Bu yöntemde doğrulama sayfasına ulaşıldıktan sonra kullanıcının karşısına mesajlaşma oturumuna ait emniyet numarası çıkartılmaktadır. Kullanıcının araya girme saldırısına(Man in the Middle Attack) maruz kalmadığından emin olabilmesi için mesajlaştığı kişinin cihazında da aynı emniyet numarasının bulunduğundan emin olması gerekmektedir. Burada da yine QR kodu taraması seçeneği mevcuttur. Bu yöntemde parmakizi doğrulamasına kıyasla tarama sayısı yarı yarıyadır. Mesajlaşma yapılan iki telefondan herhangi biri ile diğerinin QR kodunun taranması sonucu eşleşme görülürse, doğrulama başarılı bir şekilde tamamlanmış olacaktır(Şekil 3.5).



Şekil 3.4: Emniyet numarası.

3.3 Signal Sunucusu Kurulumu

OWS(Open Whisper Systems) Signal, arka planda iki ayrı bileşen üzerine kurulmuş bir ekosistemdir:

Son kullanıcılar tarafından kullanılan uygulamayı yürüten TextSecure-Server(TSS) ve APN

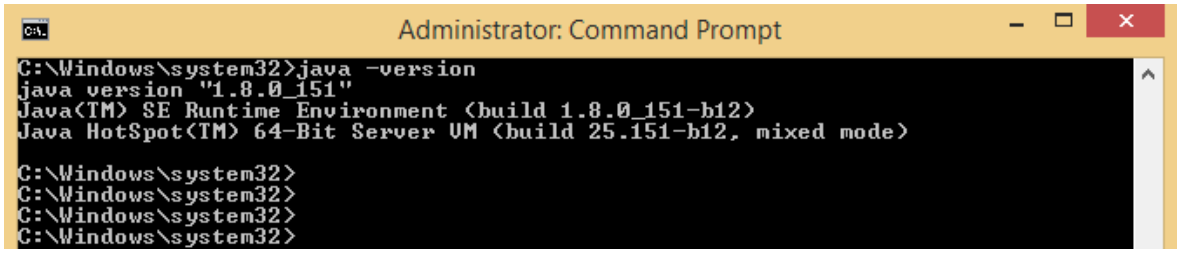
(Erişim Noktası Adı)-GCM(Google Bulut Mesajlaşma) bildirimlerini yürüten PushServer. Gerçekte şifreli sesli görüşmeden sorumlu üçüncü bir bileşen olan RedPhoneServer da bulunmaktadır, ancak an itibari ile bu bileşenin kaynakları erişilebilir durumda olmadığından, bununla ilgili tüm fonksiyonlar devre dışı bırakılmalıdır.

Şekil 3.5: QR tarama sonuçları. a) Eşleşme durumu. b) Eşleşmeme durumu (emniyet numarası yöntemi).

Ön gereksinimler

- Java 1.7 veya üstü(Şekil 3.6)
- Git bileşenleri
- Windows 8.1 veya üstü

- Redis(önbellekleme için)
- PostgreSQL Veritabanı
- Twillio hesabı(Kısa mesaj ile doğrulama işlemi için)
- Amazon AWS S3 bucket servisi(mesaj ekleri için)
- GCM(Google Bulut Mesajlaşma) hesabı(Google geliştirici konsolundan edinilen) ile alınan bir gönderici kimliği(senderID) ve Uygulama Anahtarı(ApiKey):
  - ✓ Yeni bir proje oluşturun. Oluşturduğunuz projenin proje numarası(projectNumber) gönderici kimliğidir.
  - ✓ Uygulama anahtarı için sırasıyla API management -> credentials -> create credentials -> API Keys -> Server Key sekmelerine ulaşın ve bu anahtarı kaydedin.



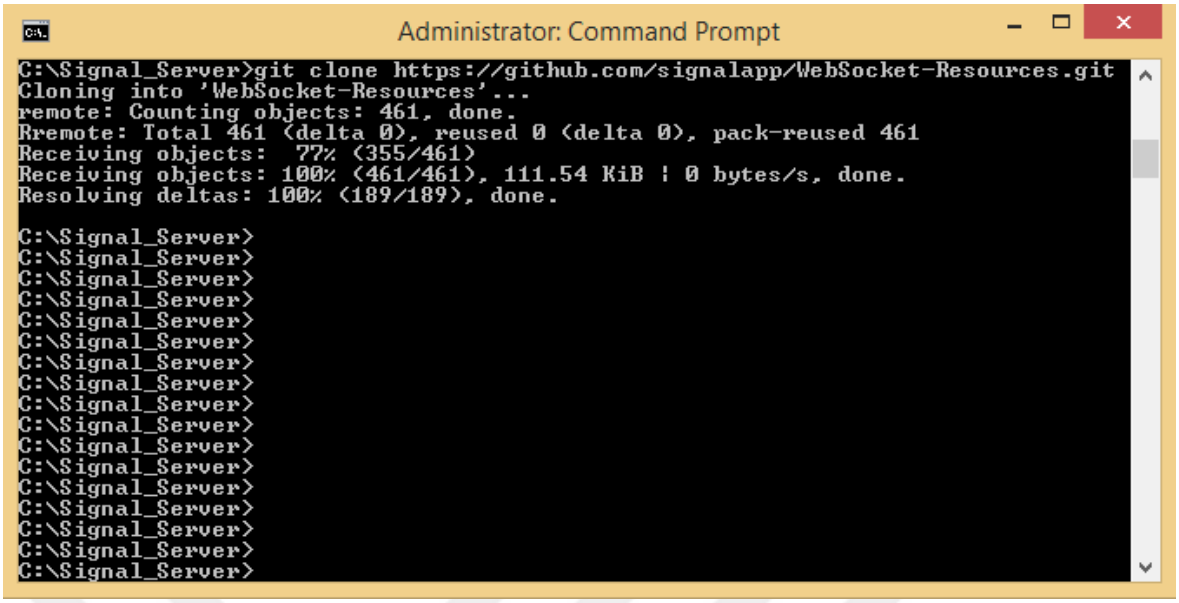
Şekil 3.6: Java versiyonunun görüntülenmesi.

Projeleri dosyalarını indirme ve derleme

WebSocket

İndirme:

git clone https://github.com/signalapp/WebSocket-Resources.git(Şekil 3.7)



Şekil 3.7: Websocket projesinin kaynaktan indirilmesi.

Derleme:

cd ../WebSocket-Resources
mvn clean install

JavaDoc ile ilgili aşağıdaki gibi bir hata mesajı ile karşılaşılabilir:

[ERROR]
Failed to execute goal org.apache.maven.plugins:maven-javadoc-plugin:2.8.1:jar (attach-javadocs) on project websocket-resources: MavenReportException: Error while creating archive: [ERROR] Exit code: 1 - javadoc: error - invalid flag: -Xdoclint:none

Ancak yine de işlem sonucunda ihtiyacımız olan kısım derlenmiş durumdadır : ./library/target/websocket-resources-0.3.2.jar

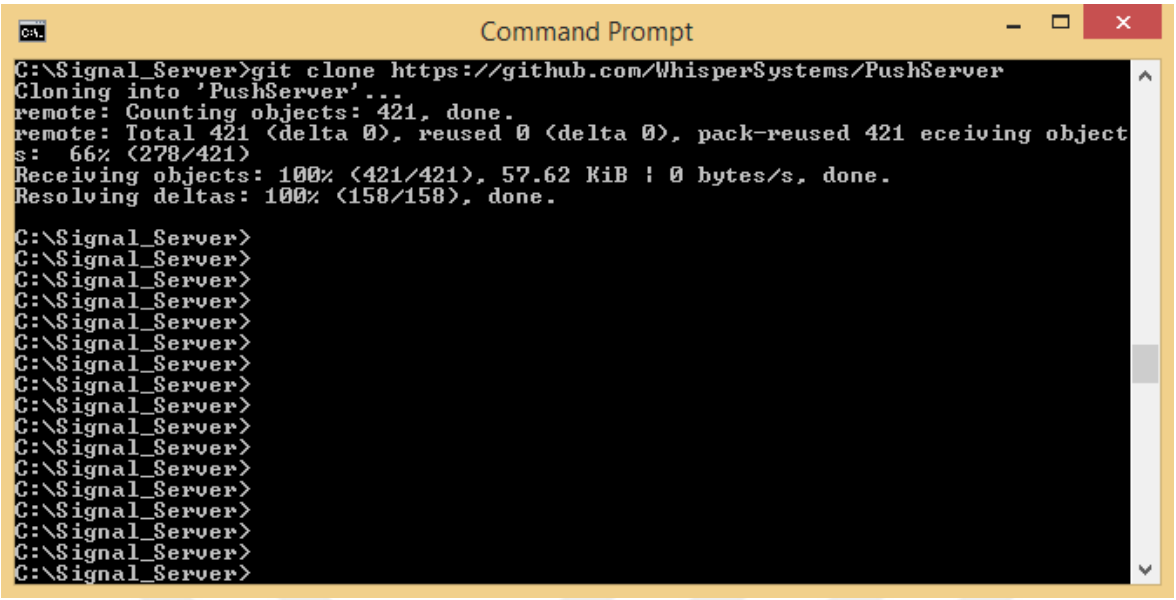
Şimdi bunu doğru isimlerle yerel Maven reposu içine aşağıdaki gibi yüklemek gerekmektedir:

mvn install:install-file -Dfile=./library/target/websocket-resources-0.3.2.jar -DgroupId=org.whispersystems -DartifactId=websocket-resources -Dversion=0.3.2 -Dpackaging=jar

PushServer

İndirme:

git clone https://github.com/WhisperSystems/PushServer(Şekil 3.8)



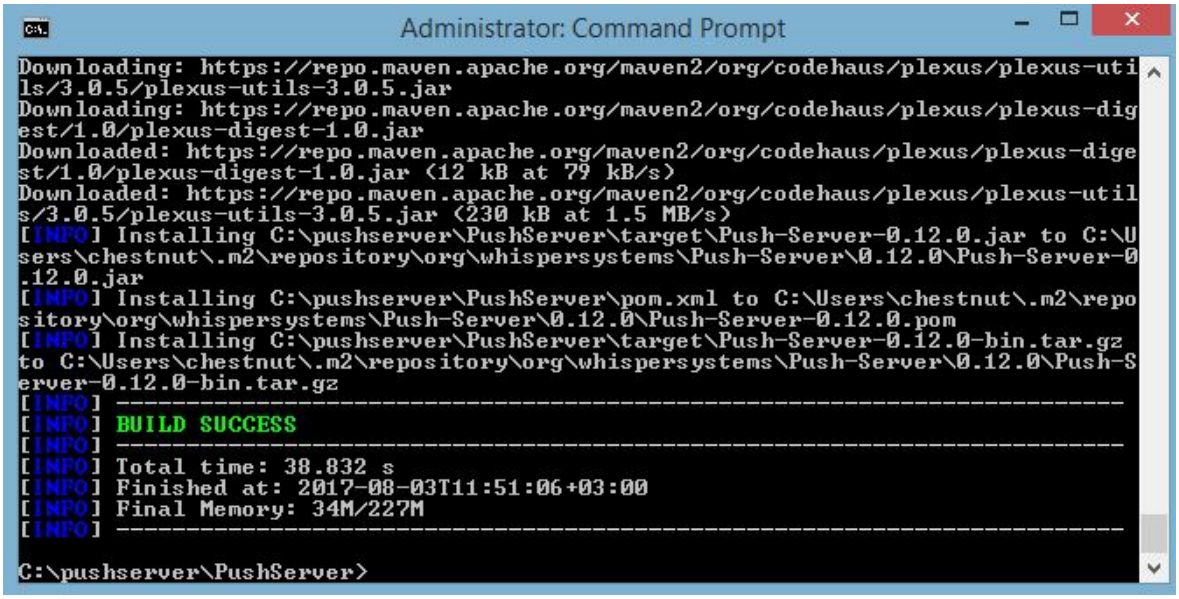
Şekil 3.8: PushServer projesinin kaynaktan indirilmesi.

Derleme:

Derlemeye geçmeden önce, Push Server altındaki pom.xml dosyasındaki, capsule.maven.plugin. version değeri 1.0.1 olarak değiştirilmelidir:

cd PushServer
mvn clean install

Derleme sonrası ekran görüntüsü aşağıdaki gibidir(Şekil 3.9)

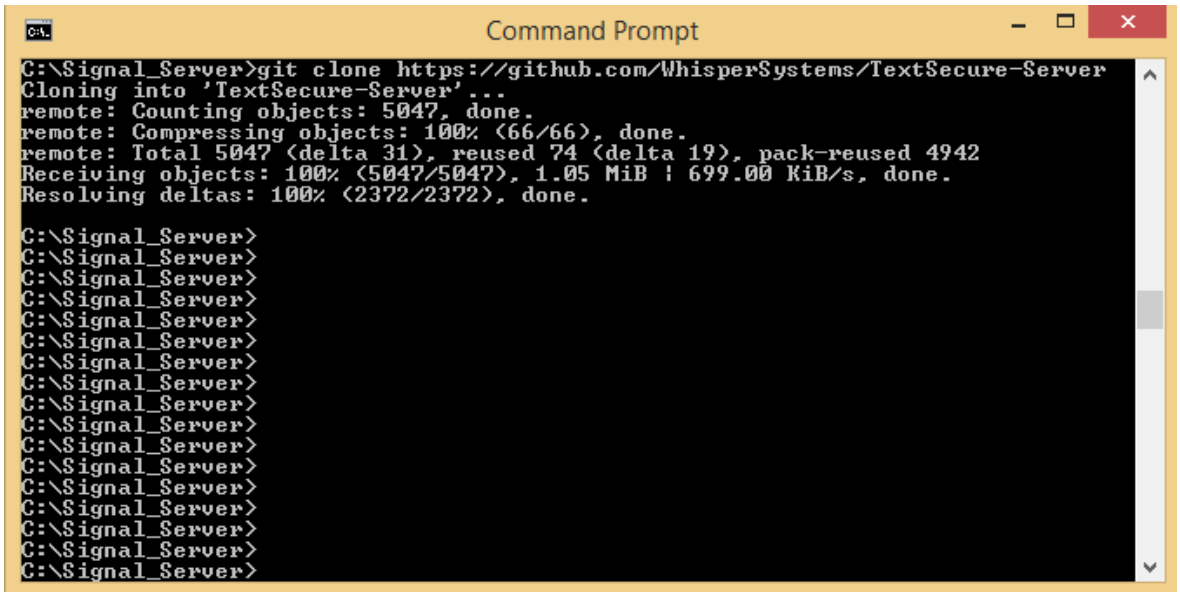


Şekil 3.9: PushServer derlenmesinin başarıyla sonuçlanması.

TextSecure sunucusu

İndirme:

git clone https://github.com/WhisperSystems/TextSecure-Server
(Şekil 3.10)



Şekil 3.10: TextSecure-Server projesinin kaynaktan indirilmesi.

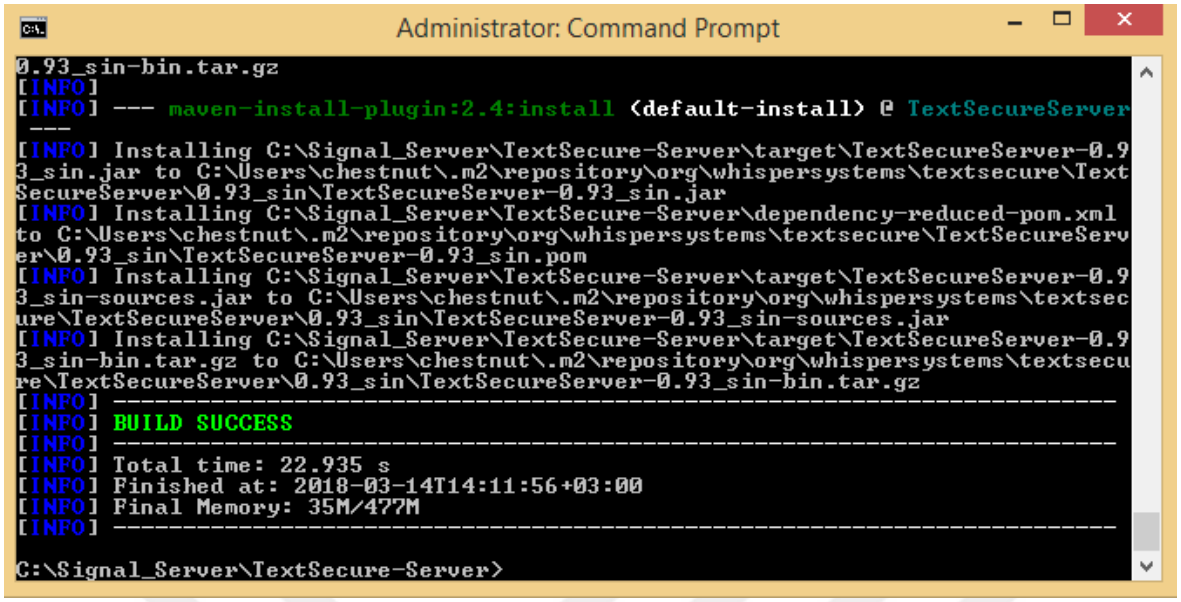
Derleme:

cd ../TextSecure-Server
mvn install

Anahtar bulunamadığına dair test kısmında karşılaşılacak hataları pas geçebilmek için komutumuzun sonuna -DskipTests parametresi eklenmelidir:

mvn clean install -DskipTests

Derleme sonrası ekran görüntüsü aşağıdaki gibidir(Şekil 3.11)



Şekil 3.11: TextSecure-Server derlenmesinin başarıyla sonuçlanması.

Konfigürasyon dosyalarının hazırlanması

APN deaktivasyonu

Kullanıcı çalışmamızda iOS mobil cihazlara ihtiyaç duymadığımızdan, IOS kullanan cihazlar için gerekli olan APN(Erişim Noktası Adı)'e ihtiyaç duyulmamıştır. Bu sebeple APN deaktivasyonu gerçekleştirilmiştir. Erişim noktası adı ile ilgili kısımları konfigürasyonumuzdan çıkartmak için aşağıdaki adımlar uygulanır:

- org.whispersystems.pushserver.PushServerConfiguration dosyasındaki "private ApnConfiguration apn;" alanının üstündeki @NotEmpty ifadesini kaldırılır.
- Ayrıca kodun içerisindeki "run" fonksiyonununda, "apnSender" değişkeni geçen her satır yorum satırı olarak değiştirilir ve PushController içerisinde geçen "apnSender" yerine "null" ifadesi yerleştirilir.

  ✓ environment.jersey().register(new PushController(null, gcmSender));

  yerine

  ✓ environment.jersey().register(new PushController(apnSender, gcmSender));

  (Satırlar : 55, 58 ve 62)

"pushserver.yml" ve "textsecure.yml" konfigürasyon dosyaları

EK-1 ve EK-2'de, çalışan "pushserver.yml" ve "textsecure.yml" dosyalarının örnek değerler ile doldurulmuş hali gösterilmiştir. Kurulumu yapan kişi, ilgili alanlara kendi verilerini girmeli ve hazırlanan pushserver.yml dosyası, PushServer klasörünün içerisine, textsecure.yml dosyası da TextSexure/config klasörünün içerisine atılmalıdır:

Veritabanı oluşturma ve düzenleme

Postgresql

- Postgres'i indirin ve kurun:
  https://www.postgresql.org/download/windows/

- Veritabanı oluşturun (Öncesinde postgres ile ilgili tüm komut satırlarından çıkış yapın)
  createdb -U postgres accountsdb
  createdb -U postgres messagedb

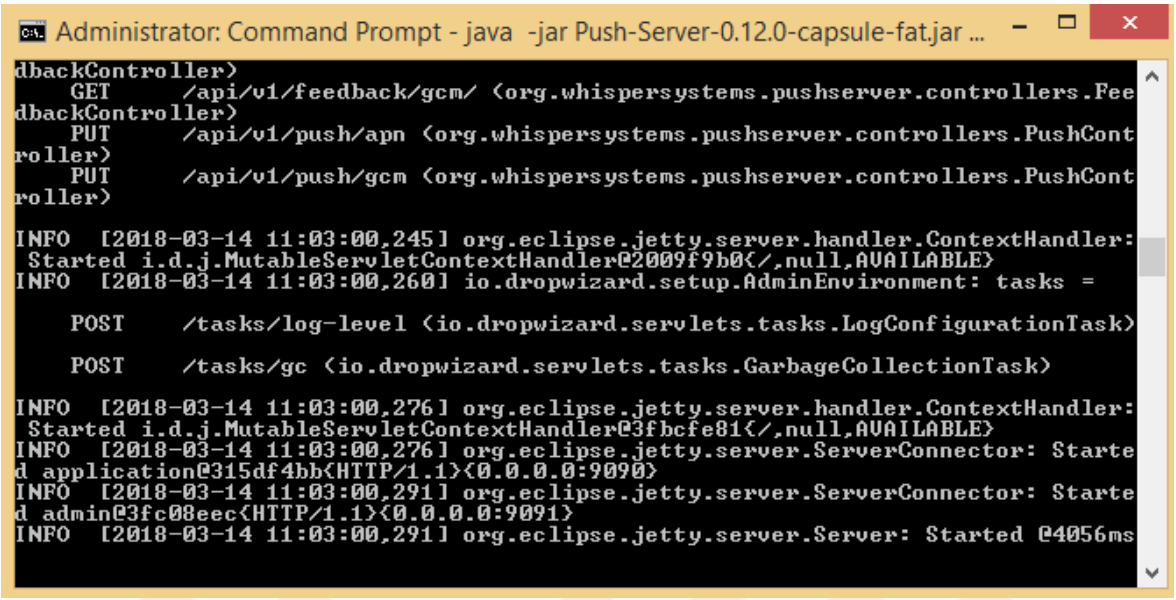
- Tablo şemalarını içeriye aktarın: (Signal-Server klasörünün içerisindeyken)
  java -jar target/TextSecureServer-<VERSION>.jar accountdb migrate config/textsecure.yml
  java -jar target/TextSecureServer-<VERSION>.jar messagedb migrate config/textsecure.yml

Sunucuları çalıştırma

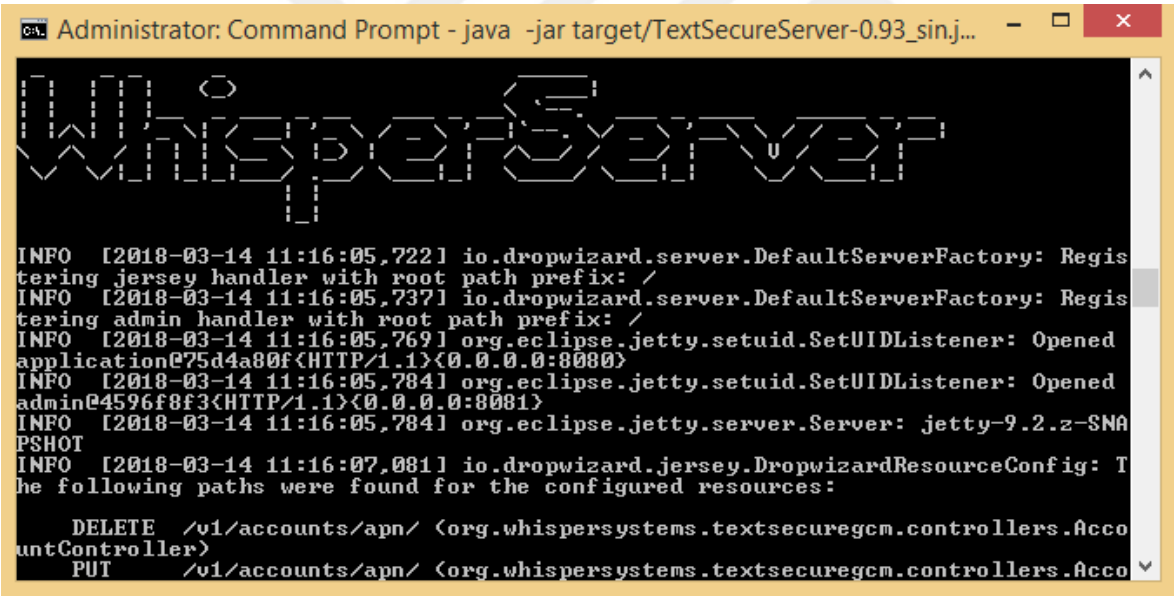
- Redis'i çalıştırın:
  redis-server

- Push sunucusunu çalıştırın
  java -jar Push-Server-<VERSION>-capsule-fat.jar server pushserver.yml(Şekil 3.12)

- Text Secure (Signal) sunucusunu çalıştırın
  java -jar target/TextSecureServer-<VERSION>.jar accountdb migrate config/textsecure.yml(Şekil 3.13)



Şekil 3.12: PushServer'ın çalıştırılması.



Şekil 3.13: TextSecure sunucusunun çalıştırılması.

Signal android uygulamasını indirme ve derleme

Kullanıcı çalışmamızda iOS mobil cihazlara ihtiyaç duymadığımız için, signal sunucusunun kurulumu yapılırken, parametreler "Android" işletim sistemli mobil cihazlara göre girilmiştir. Ayrıca Android mobil cihazlarla çalışabilmemiz için APN ve SSL(HTTPS) kullanmak zorunluluğu bulunmamaktadır ve bu da işimizi kolaylaştırmıştır.

- Signal-Android reposu aşağıdaki gibi indirilebilir:

  git clone –depth 1 https://github.com/signalapp/Signal-Android.git

- Sonrasında indirilen klasörün içerisindeki build.gradle dosyasını açıp "TEXTSECURE_URL" parametresinin olduğu satıra gelinir. Burada varsayılan adres yerine kendi sunucumuzun bulunduğu adres girilir. Adres girilirken çalışacak protokole de dikkat edilmelidir. Bizim sunucumuz yerel ağda çalışacağı için aşağıdaki gibi bir parametre girilmiştir(Şekil 3.14):

```
162     defaultConfig {
163         minSdkVersion 9
164         targetSdkVersion 22
165
166         buildConfigField "long", "BUILD_TIMESTAMP", System.currentTimeMillis() + "L"
167         buildConfigField "String", "TEXTSECURE_URL", "\"http://192.168.1.100:8080\""
168         buildConfigField "String", "USER_AGENT", "\"OWA\""
169         buildConfigField "String", "REDPHONE_MASTER_URL", "\"https://redphone-master.whispersystems.org\""
170         buildConfigField "String", "REDPHONE_RELAY_HOST", "\"relay.whispersystems.org\""
171         buildConfigField "String", "REDPHONE_PREFIX_NAME", "\".whispersystems.org\""
172         buildConfigField "boolean", "DEV_BUILD", "false"
173     }
174
```

Şekil 3.14: build.gradle konfigürasyonu.

- org.thoughtcrime.securesms.jobs.GcmRefreshJob dosyası açılır. Buradaki REGISTRATION_ID parametresinin değeri, Push-Server ayar dosyamızdaki gönderici kimliği ile değiştirilmelidir. Bizim ayarlarımızda bu değer önceki adımlarda "38750274690" olarak girilmiştir.

- Son olarak signal android uygulamamız Android Studio kullanılarak derlenir ve çalışmaya hazır hale getirilir.

4. KULLANICI ÇALIŞMASI

Labaratuvar ortamında SIGNAL android uygulamasının iki farklı versiyonunu kapsayan, benzer iki ayrı kullanıcı grubu üzerinde kullanıcı çalışması yürütülmüştür. Çalışmada mevcut mesajlaşma oturumuna yönelik bir saldırı simüle edilmiş ve kullanıcılardan mesajlaşma oturumunun güvenli olup olup olmadığını anlayabilmeleri için uygulama içerisinden açık anahtar doğrulama işlemi gerçekleştirmeleri istenmiştir. İki versiyon da denenirken harici tüm etkenler eşit tutulmuştur. Her iki gruba da aynı talepler yöneltilmiştir. Katılımcılardan, her adımda ne düşündüklerini anlayabilmemiz adına sesli düşünmeleri ve yorumda bulunmaları istenmiştir. Çalışma sırasında ve sonrasında katılımcıların genel demografik yapısını ve çalışma hakkında düşündükleri hakkında kendilerine birtakım sorular yöneltilmiştir. Çalışma TOBB Ekonomi ve Teknoloji Üniversitesi'nin Teknoloji Merkezindeki "Bilgi Güvenliği Labaratuvarı"nda gerçekleştirilmiştir.

Kullanıcı hiçbir aşamada bilgilerini operatöre göndermeye zorlanmamış ve ihtiyacı olduğu her anda operatörü yanına çağırabileceği konusunda bilgilendirilmiştir. Çalışma sırasında operatör, laboratuvarın uzak köşesinde oturmuş olup, tüm bu aşamalarda aktif bir rol oynamamış ve kimlik doğrulama hakkında kullanıcıya bilgi vermemiştir. Kimlik doğrulama işleminden sonra katılımcıya, çalışmanın geneli hakkında ne düşündüğü ve simüle edilen araya girme saldırısının kendisine ne ifade etmiş olabileceğini anlamak adına kısa bir anket uygulanmıştır.

4.1 Çalışma Senaryosu

Çalışmanın başında kullanıcılara, SIGNAL yüklü android bir telefon kullanarak SIGNAL uygulaması ile operatöre yollamasını istediğimiz birtakım bilgiler verilmiştir. Bu bilgilerin kendilerine ait şahsi bilgiler olduğunu kabul etmeleri ve her adımda bir bölümünü operatöre yollamaları istenmiştir. Kullanıcıların operatöre yollayacağı mesajlar:

• Ön aşamada adres bilgilerini,
• Birinci aşamada banka hesap bilgilerini,
• İkinci aşamada ise kredi kartı bilgilerini
içermektedir.

Ayrıca çalışma hakkında kısa bir bilgilendirme yapılmıştır. Kullanıcılardan, operatörü günlük hayatta bu bilgileri vermekte sakınca duymayacakları, güvendikleri biri (örn. bir aile ferdi) olarak varsaymaları istenmiştir.

Ön Aşama: Bu aşamada kullanıcılardan, kendilerine verilen bilgiler içerisinden, adres kısmını operatöre iletmeleri istenmiştir.

Araya Girme Saldırısı: Kullanıcılar ön aşamada adres bilgilerini operatöre yolladıktan sonra, birinci aşamaya geçmeden önce kendilerine birtakım sorular yöneltilmiştir. Katılımcılar bu soruları yanıtlarken araya girme saldırısını (MITM) simüle edecek şekilde bir işlem gerçekleştirilmiştir. Bu işlem operatörün kullandığı telefondaki sim kartın başka bir telefona takılarak, operatörün halihazırda kullandığı numaranın yeni telefon üzerinden SIGNAL uygulamasına kayıt yaptırmasını kapsamaktadır. Farklı bir telefondan, operatörün numarası ile SIGNAL uygulamasına kayıt yaptırmak, merkezi sunucudaki bu numaraya ait açık anahtarın değişmesine yol açacaktır. Bu sayede merkezi sunucudaki açık anahtarın kasıtlı olarak değiştirilmesi ile gerçekleştirilecek bir araya girme saldırısı simüle edilmiş olur. Böylelikle, kullanıcı operatöre yeni bir mesaj yollamak istediğinde, telefonuna otomatik olarak yeni bir anahtar seti ulaşacaktır.

Aşama 1: Birinci aşamada kullanıcıdan banka hesap bilgilerini operatöre yollaması istenmiş ve kullanıcı bu mesajı yollamak istediğinde SIGNAL uygulamasının versiyonuna göre değişkenlik gösteren bir durumla karşılaşmıştır. Eski versiyonda bu, anahtar uyuşmazlığı hakkında bir hata mesajı iken(Şekil 4.1), yeni versiyonda ise emniyet numarasının değiştiğine dair bir uyarı olarak karşımıza çıkmaktadır(Şekil 4.4). Bu aşamanın sonunda, SIGNAL uygulamasının ilettiği uyarılar sonucu kendi kendilerine kimlik doğrulamaya geçmeyen kullanıcılardan, SIGNAL uygulaması üzerinden mesajlaştıkları kişinin kimliğini doğrulamaları istenmiştir.

Aşama 2: İkinci aşamada ise kullanıcılardan kredi kartı bilgilerini operatöre iletmeleri istenmiştir. Ancak bu bilgiyi mesajla karşı tarafa gönderdiklerinde, kredi kartı bilgileri düşündükleri gibi operatöre değil, araya giren kişiye ulaşmış olacaktır. Bu yüzden kimlik doğrulama işlemini gerçekleştirip, anahtarların uyuşmadığını anlayan bir kullanıcıdan, mesajlaşmaya devam etmemesi, dolayısı ile kredi kartı bilgilerini karşı tarafa iletmemeleri beklenmektedir.

4.2 Teknik Altyapı

Çalışan bir sistem, üç adet akıllı telefon (Android 6.0) ve bir adet bilgisayardan oluşturulmuştur. SIGNAL'in eski versiyonunu çalıştırabilmek adına bilgisayar SIGNAL sunucusu olarak kullanılmış ve aynı zamanda akıllı telefonlara internet sağlayan bir kablosuz erişim noktası görevi görmüştür. SIGNAL'in yeni versiyonu üzerinde çalışılırken ise SIGNAL'in varsayılan merkezi sunucusu kullanılmıştır. Akıllı telefonların ikisi mesajlaşma, üçüncüsü ise araya girme saldırısı için kullanılmıştır. Parmak izi ile açık anahtar doğrulama sağlayan SIGNAL versiyonu 3.13.0, emniyet numarası ile açık anahtar doğrulama sağlayan SIGNAL versiyonu ise, güncel versiyon olan 4.11.5 olarak seçilmiş ve uygulanmıştır. Her kullanıcı için cihazlar yeniden başlangıçtaki haline döndürülmüştür.

4.3 Katılımcı Profili

Tüm katılımcılar TOBB Ekonomi ve Teknoloji Üniversitesi, bilgisayar mühendisliği bölümünde "Bilgisayar Bilimlerine Giriş" dersini alan öğrencilerdir. Çalışmaya katılan tüm öğrenciler ilgili derste ekstra puan almaya hak kazanmışlardır. Test edilecek SIGNAL uygulamasının iki versiyonu için rastgele bir biçimde iki grup oluşturulmuştur. 15 ila 25 dakika arası süren çalışmada, her iki versiyon için 21 kişi olmak üzere toplamda 18-25 yaş aralığında 42 kişi yer almıştır. İlk grupla (10 kadın, 11 erkek), açık anahtar doğrulamada emniyet numarası yöntemini kullanan yeni versiyon, ikinci grupla (8 kadın, 13 erkek) ise parmakizi yöntemini kullanan eski versiyon test edilmiştir.

Katılımcıların tamamı aktif şekilde anlık mesajlaşma uygulaması olarak WhatsApp (42) kullanmakla birlikte ilk grupta diğer uygulamalar SnapChat (16), Skype (14), Facebook Messenger (7), Telegram (2), Viber (1), Bip (1), KakaoTalk (1), Signal (1) ikinci grupta ise SnapChat (12), Skype (8), Facebook Messenger (7), Telegram (3), Discord (2) şeklinde dağılmıştır. Bilgi güvenliği alanındaki bilgi seviyelerini ifade etmelerini istediğimizde ilk grupta 16 kişi bu alanda yabancı veya oldukça düşük seviye , 3 kişi orta seviye, 2 kişi ise oldukça bilgi sahibi olduğunu, ikinci grupta ise 15 kişi bu alanda yabancı veya oldukça düşük seviye , 4 kişi orta seviye, 2 kişi ise oldukça bilgi sahibi olduğunu belirtmiştir. İlk grupta 2, ikinci grupta ise 3 kişi araya girme saldırıları hakkında detaylı bilgiye sahip olduğunu ifade etmiştir.

4.4 Çalışma Sonuçları ve Analiz

Her iki gruptaki tüm katılımcılar, ön aşamadaki açık adres bilgilerini operatöre gönderme görevini zorlanmadan tamamlamışlardır.

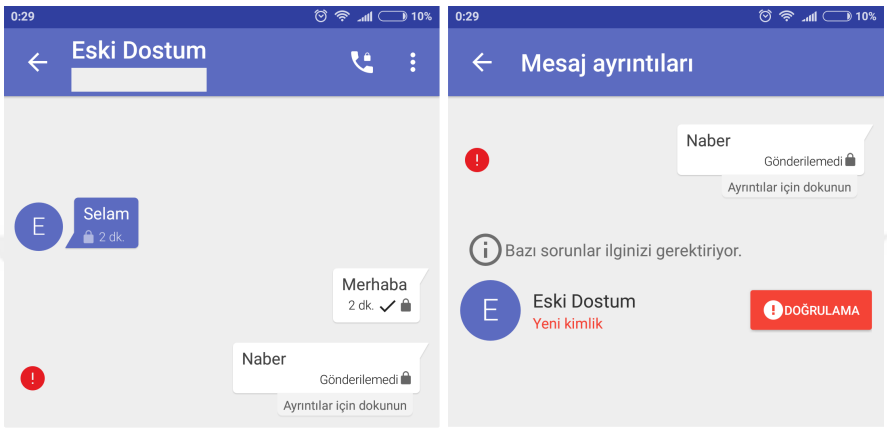
Ön aşamada kullanıcı, adres bilgisini operatöre gönderdikten sonra kendisine yöneltilen anket sorularını yanıtlarken, operatörün sim kartı başka bir telefona takılıp SIGNAL sunucusuna kayıt olunarak araya girme saldırısı simüle edilmiştir. Bunun sonucunda kullanıcı, birinci aşamada kendisinden beklenen banka hesap bilgilerini operatöre yollamak istediğinde artık operatör için bilinen açık anahtar değişmiş durumdadır.

SIGNAL uygulamasının açık anahtar doğrulamaya yönlendirebilmesi

Parmakizi ve emniyet numarası yöntemlerini karşılaştırmak için çalışmada kullanılan SIGNAL uygulamasının geçmiş ve güncel versiyonlarının(parmakizi için v3.13.0 ve emniyet numarası için v4.11.5), açık anahtar değişimi sonrası kullanıcılarını bilgilendirme şekilleri aynı değildir. Bu bölümde, bahsedilen versiyonlar arası farkın, kullanıcıların açık anahtar doğrulamaya yönlendirilmesinde olumlu veya olumsuz bir etkisi olup olmadığı incelenmiştir.

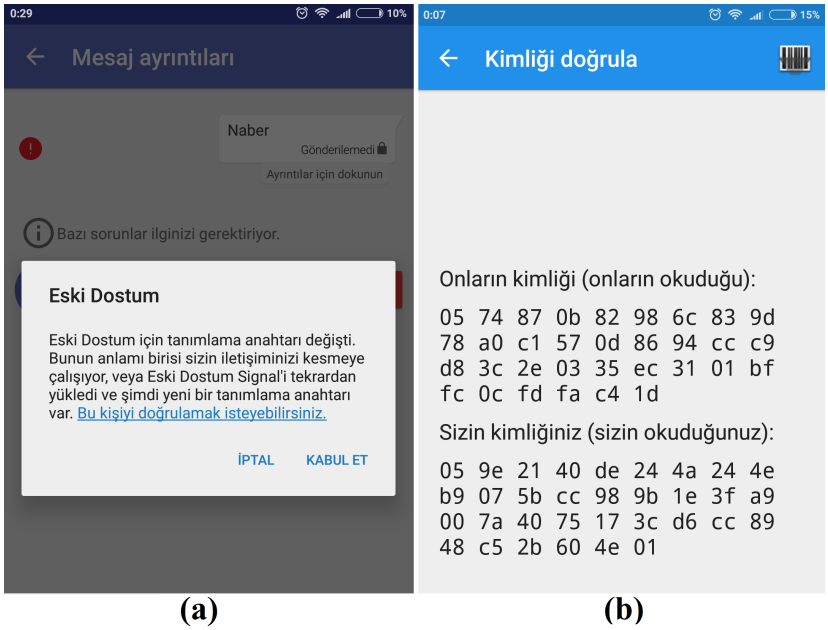
Parmakizi doğrulama kullanan versiyonda, yeni açık anahtar kabul edilene dek, gönderilmek

istenen mesajlar karşı tarafa iletilmez, mesaj gönderilemedi uyarısı ile karşılaşılır ve ayrıntılı bilgi için kullanıcının uyarıya dokunması beklenir (Şekil 4.1). Uyarıya dokunulduğunda mesajlaşma gerçekleştirilen kontak için "Yeni Kimlik" ibaresi ve kırmızı renkli "DOĞRULAMA" butonu görülür. Bu butona tıklandığında da kullanıcının karşısına bir diyalog penceresi çıkartılır(Şekil 4.2). Bu diyalogdaki linke tıklanarak veya sohbet ayarları menüsünden, kimlik doğrulama sayfasına erişmek mümkündür(Şekil 4.3). Bu kısımda 21 kullanıcının 4'ü uyarıdaki link vasıtası ile kimlik doğrulama sayfasına ulaşmıştır(Şekil 4.5).



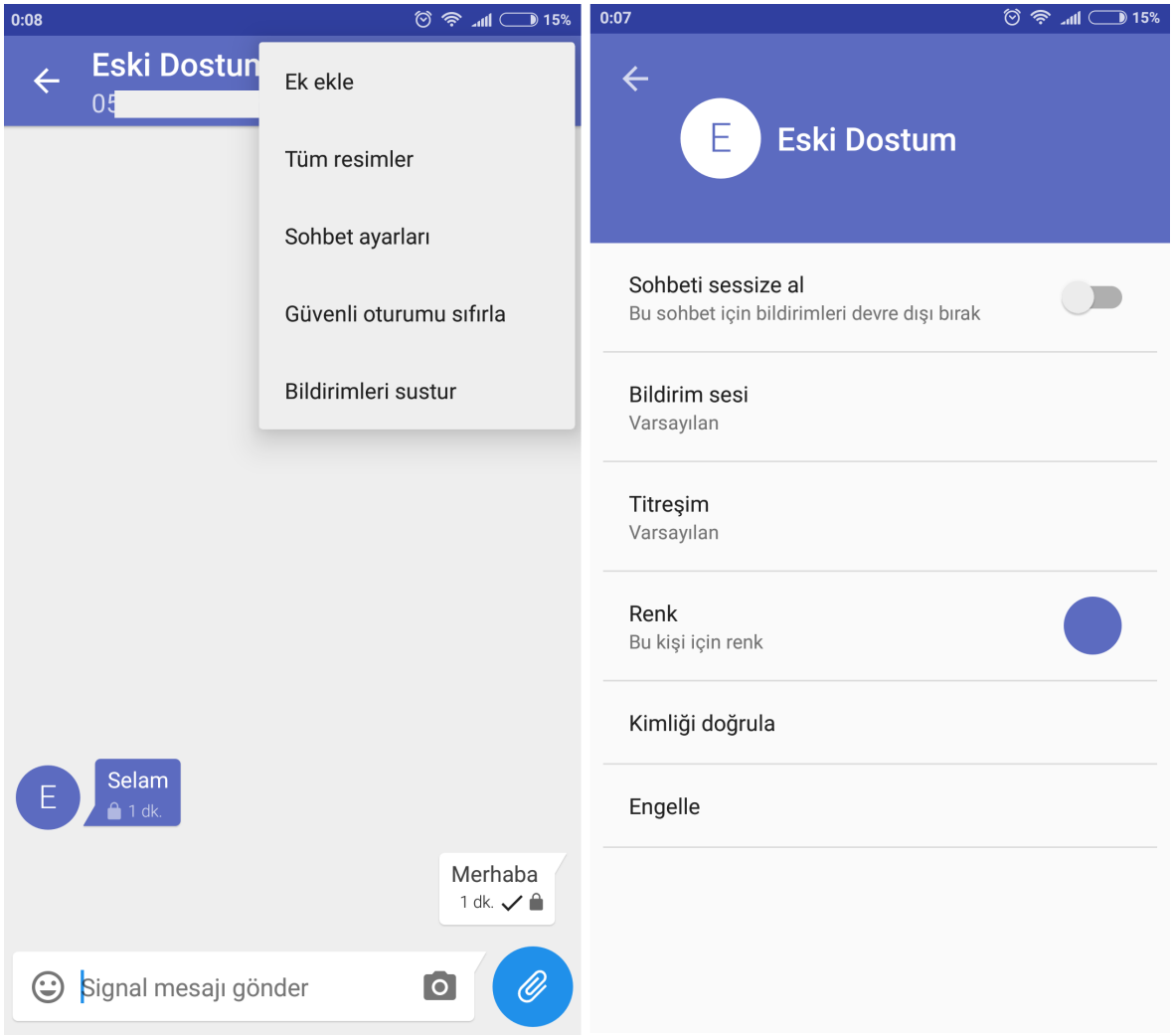
(a) **(b)**

Şekil 4.1: a) Gönderilemedi hatası. b) Kimlik bilgilerinin değiştiği uyarısı (parmakizi yöntemi).



(a) **(b)**

Şekil 4.2: a) Yeni kimlik diyaloğu. b) Kimlik doğrulama sayfasındaki anahtarlar(Alttaki kullanıcının, üstteki karşı tarafın parmakizi) (parmakizi yöntemi).

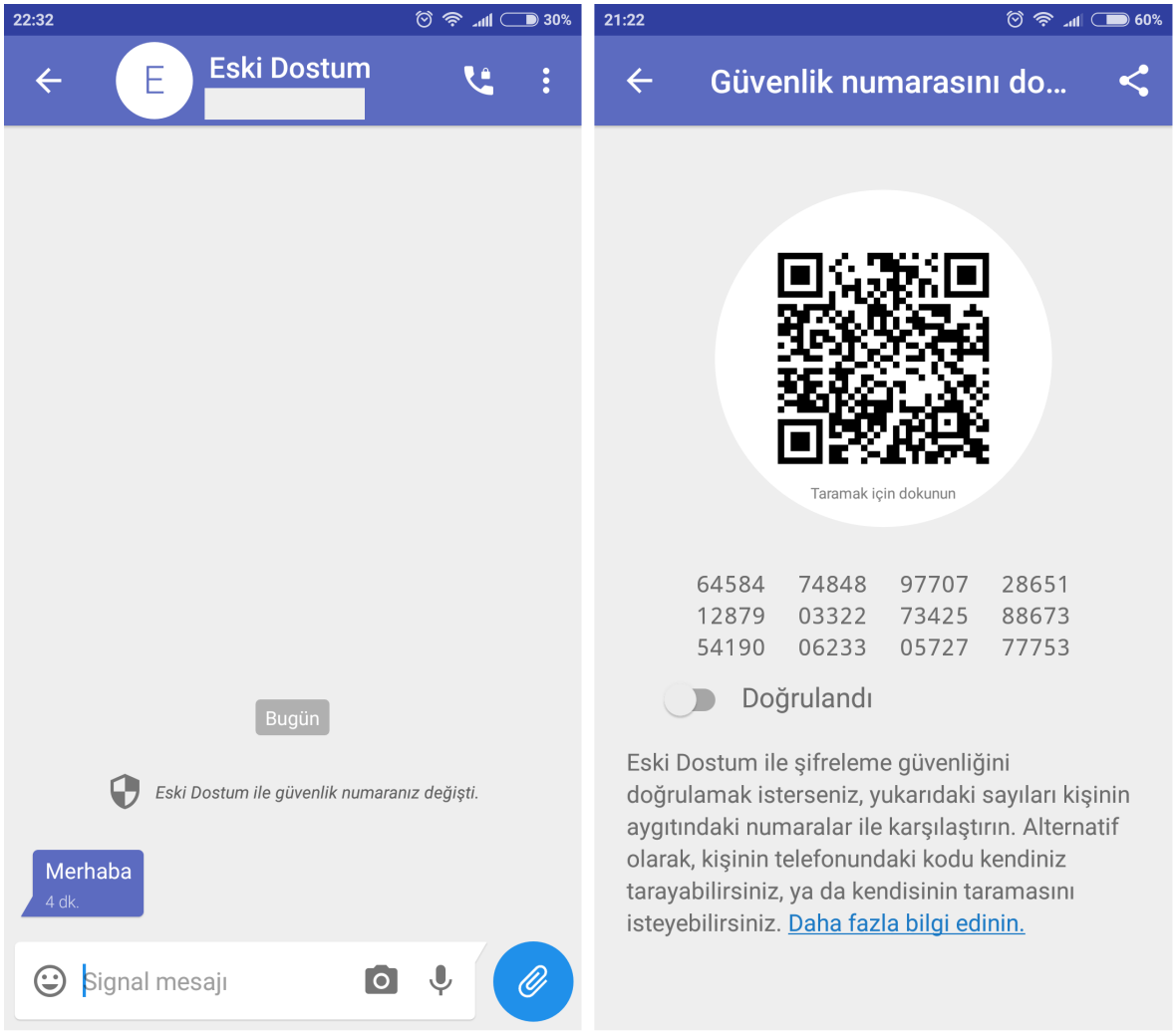


Şekil 4.3: a) Sohbet sayfasındaki ayarlar sekmesi. b) Kimlik doğrulama sayfasına erişilen sohbet ayarları menüsü(parmakizi yöntemi).

Emniyet numarası kullanan versiyonda ise mesajlaşma kesilmez ve kullanıcının ekranında görebileceği şekilde, emniyet numarasın değiştiğine dair bir uyarı çıkartılır (Şekil 4.4). Kullanıcı bu uyarıya dokunduğunda direkt olarak kimlik doğrulama sayfasına yönlendirilir. Ancak bu gruptaki 21 katılımcının içerisinden ise yalnızca bir kişi bu uyarıya dokunarak kimlik doğrulama sayfasına ulaşmıştır(Şekil 4.5).

H_0**:** Kullanıcıları açık anahtar doğrulama işlemine yönlendirme açısından, mesajlaşılan kişinin açık anahtarının değişmesi sonrası yeni anahtarın kullanıcı tarafından kabul edilene dek mesajlaşmanın bloklanması ile bloklanmaması arasında bir fark yoktur.

Elimizdeki verileri x^2 testi'ne tabi tuttuğumuzda bu farkın anlamlı olmadığı görülmüştür(p $<$ 0,05 için). $[x^2(1) = 2.04, p = 0,15]$.



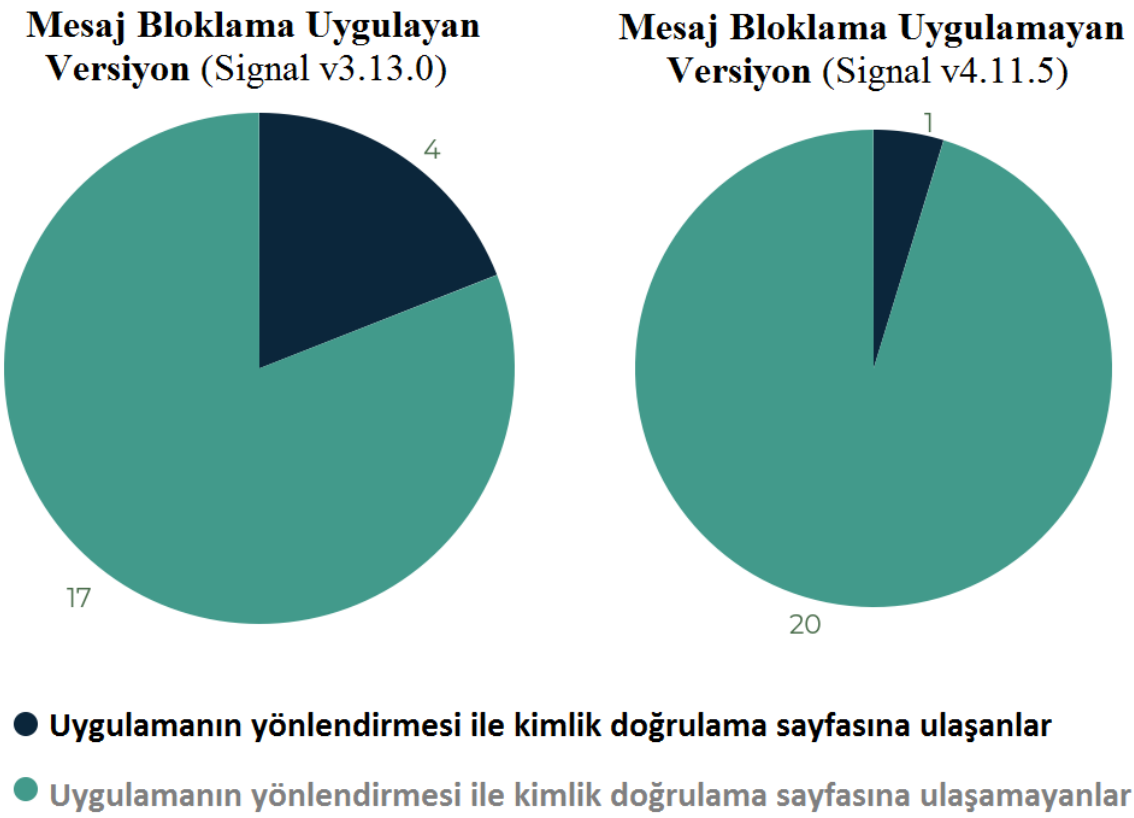
(a) **(b)**

Şekil 4.4: a) Emniyet/Güvenlik numarası değişimi uyarısı. b) Emniyet numarası doğrulama sayfası(emniyet numarası yöntemi).

21'er kişilik iki grup için bir grupta 4 kişi, diğerinde ise 1 kişinin uygulamanın yönlendirmesi sonucu doğrulama adımına geçmiş olması, istatistiksel açıdan çarpıcı bir sonuç olarak görünmese de, gözlemlerimiz bize çalışma gruplarının sayısının artırılarak tekrarlanması sonucunda aradaki farkın anlamlı bir hale gelebileceğini göstermiştir.

Açık anahtar doğrulama başarısı

Açık anahtar değişimi sonrası kendiliğinden kimlik doğrulama işlemine geçmemiş olan tüm kullanıcılardan, uygulama içerisinden kimlik doğrulama adımını gerçekleştirmeleri istenmiştir. Bu işlem daha önce de değinildiği gibi, anahtar dizilerinin karşılaştırılması veya QR kodların taratılması yoluyla yapılmaktadır.



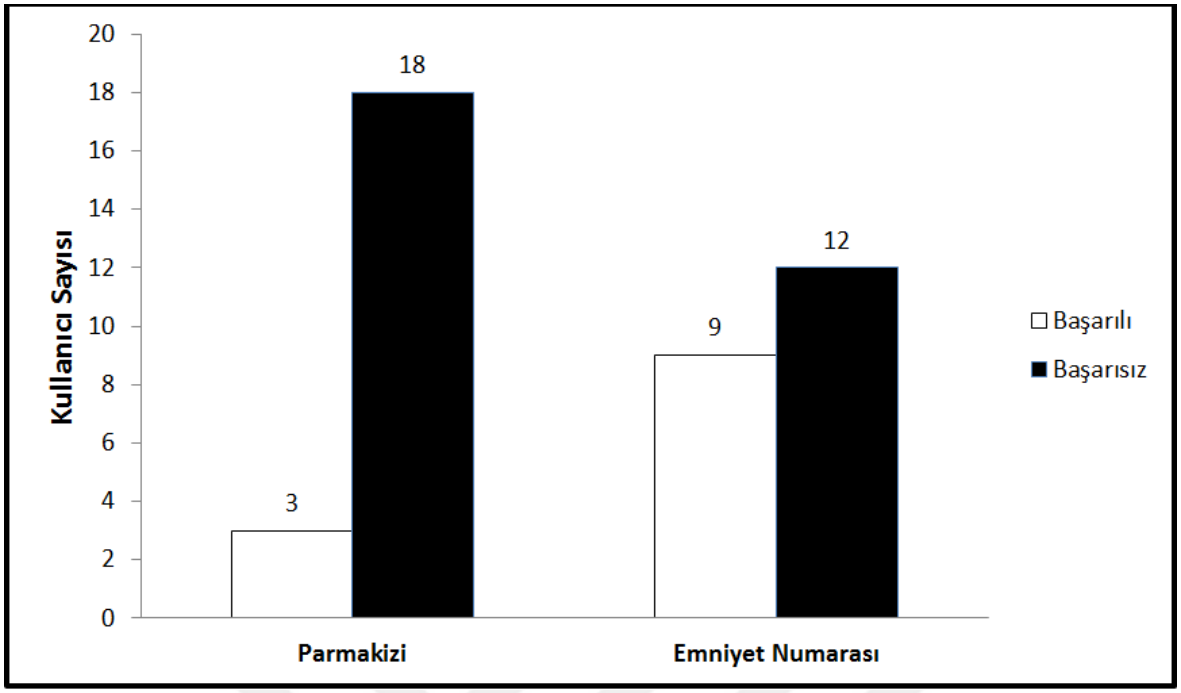
Şekil 4.5: Çalışmada kullanılan SIGNAL uygulamasının açık anahtar değişimi sonrası mesajlaşmayı bloklayan(3.13.0) ve bloklamayan(4.11.5) versiyonlarının kimlik doğrulamaya yönlendirebilme performansları.

Çalışmamızda simüle etmiş olduğumuz araya girme saldırısı gereği, kullanıcıların parmakizleri ve mesajlaşma oturumlarına ait emniyet numaraları birbirleriyle eşleşmezler. Dolayısı ile ideal senaryoda doğru adımları izleyen bir kullanıcı:

- Anahtarları doğru bir şekilde karşılaştırmalı,
- Karşılaştırdığı anahtarların uyuşmadığını görmeli,
- Anahtar uyuşmazlığının, mesajlaşma oturumunun güvenli olmadığını ifade ettiğini anlayıp mesajlaşmaya devam etmemelidir.

Şayet ikinci aşamaya geçip kredi kartı bilgilerini uygulama üzerinden karşı tarafa gönderirlerse, bu bilgiler düşündükleri gibi operatöre değil, saldırı sırasında açık anahtarını kullanıcıya gönderen kişiye gitmiş olacaktır.

Buna göre emniyet numarası yöntemi ile doğrulama yapmasını beklediğimiz 21 katılımcının 9'u başarılı olabilmiştir. Buna karşın parmakizi yöntemi ile doğrulama yapmasını beklediğimiz 21 katılımcının ise yalnızca 3'ü başarı ile çalışmayı tamamlamıştır(Şekil 4.6).



Şekil 4.6: Parmakizi ve emniyet numarası yöntemlerinin uygulandığı kullanıcı gruplarının kimlik doğrulama başarısı.

H_0**:** Anahtar doğrulama başarısı açısından emniyet numarası yöntemi ile parmakizi yöntemi arasında bir fark yoktur.

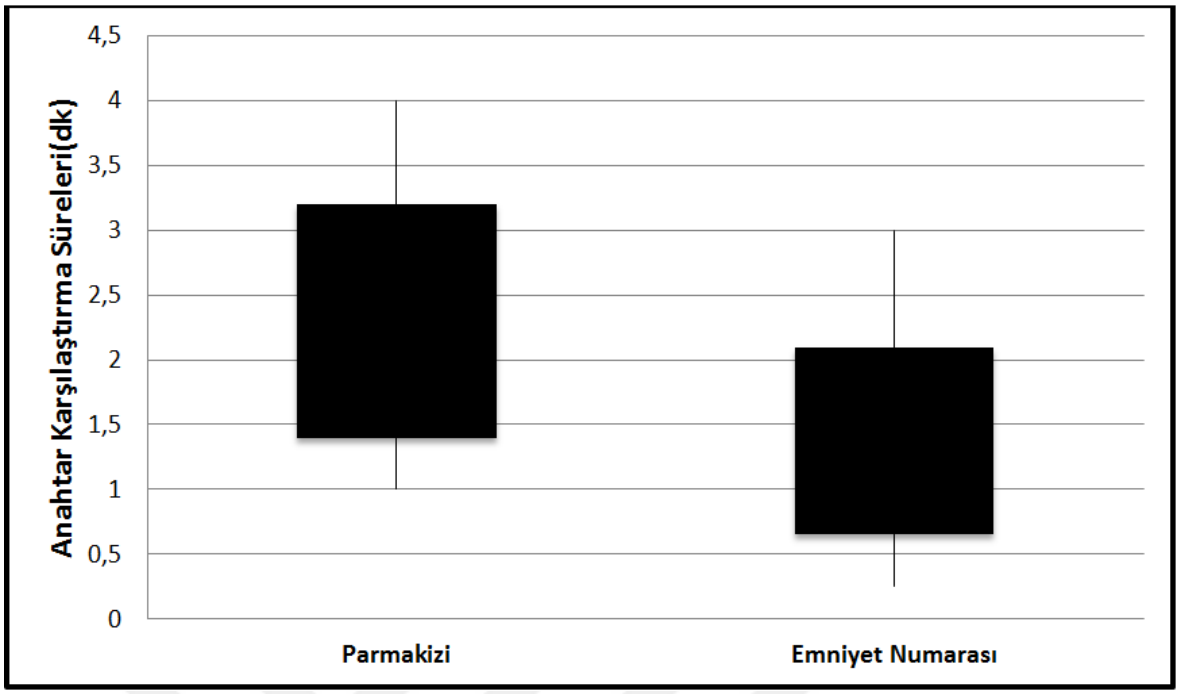
Elimizdeki verileri x^2 testi'ne tabi tuttuğumuzda bu farkın anlamlı olduğu ortaya çıkmıştır(p < 0,05 için). $[x^2(1) = 4.20, p = 0,04]$. Dolayısı ile H_0 başlangıç hipotezimiz reddedilmiştir.

Açık anahtar karşılaştırma süresi

Açık anahtar doğrulama süreleri ölçüldüğünde, emniyet numarası ile açık anahtar doğrulama yapmasını istediğimiz 21 kişiden, operatörü yanına çağırıp emniyet numaralarını karşılaştıran 17 kişinin ortalama karşılaştırma süresi yaklaşık 1,3 dk olarak ölçülmüş, buna karşın parmakizi ile açık anahtar doğrulama yapmasını beklediğimiz 21 kişiden operatörü yanına çağırıp parmakizlerini karşılaştıran12 kişinin ortalama karşılaştırma süresi ise yaklaşık 2,1 dk olarak ölçülmüştür(Şekil 4.7).

H_0**:** Açık anahtar karşılaştırma süresi açısından emniyet numarası yöntemi ile parmakizi yöntemi arasında bir fark yoktur.

Elimizdeki verileri Mann-Whitney-U testine tabi tuttuğumuzda bu farkın anlamlı olduğu görülmüştür (p < 0,05 için). [W : 58; p=0,04]. Dolayısı ile H_0 başlangıç hipotezimiz reddedilmiştir.



Şekil 4.7: Parmakizi ve emniyet numarası yöntemleri ile kimlik doğrulama gerçekleştiren kullanıcıların ortalama anahtar karşılaştırma süreleri.

5. SONUÇ VE ÖNERİLER

Yapmış olduğumuz çalışmada, uçtan uca şifreleme sağlayan android ortamındaki açık kaynak SIGNAL anlık mesajlaşma uygulaması üzerinden açık anahtar doğrulamada kullanılan "parmakizi" ve "emniyet numarası" yöntemlerini kullanışlılık ve güvenlik perspektifinden inceledik. Toplamda 42 katılımcı ile gerçekleştirdiğimiz kullanıcı çalışması ile bu yöntemlerin gerçekte hangi ölçüde kullanılabilir olduklarını test ettik.

Benzer kullanıcı çalışmalarında katılımcıların büyük çoğunluğunun açık anahtar doğrulama konusunda başarısız olduğu gerçeği, çalışmamızda da görülmüştür. Bunun yanında emniyet numarası yönteminin parmakizi yöntemine göre açık anahtar karşılaştırma süreleri ve doğrulama işleminin kolaylığı noktalarında daha başarılı olduğu ortaya konulmuştur.

Çalışmamız sonrası elde edilen veriler, emniyet numarası ile açık anahtar doğrulama yönteminin numaralardan oluşması ve tek karşılaştırma ile tamamlanması sebebiyle, kullanıcılar tarafından daha kolay gerçekleştirilebildiğini görülmüştür. Burdan hareketle uygulama geliştiricilerin parmakizi yöntemini kullanmak yerine emniyet numarası yöntemini kullanmaları kullanışlılık açısından daha avantajlı olacaktır.

Ayrıca Signal uygulaması içerisinde, eski versiyonda varsayılan olarak anahtar değişimi sonrası mesajların bloklanması seçeneği, opsiyonel bir şekilde yeni versiyonda kullanıcılara sunulabilir. Önceliği güvenlik olan kullanıcılar için bu seçeneğin, uygulamayı tercih etme noktasında dahi etkili olabileceği düşünülmektedir.

KAYNAKLAR

[1] **A. Whitten ve J. D. Tygar.** Why johnny can't encrypt: A usability evaluation of pgp 5.0. *Usenix Security, vol. 1999.*, (1999)

[2] **S. L. Garfinkel, D. Margrave, J. I. Schiller, E. Nordlander, ve R. C. Miller.** How to make secure email easier to use. *Proceedings of the SIGCHI conference on human factors in computing systems. ACM.*, (2005), s.701-710.

[3] **K. Renaud, M. Volkamer, ve A. Renkema-Padmos.** Why doesn't jane protect her privacy? *Privacy Enhancing Technologies. Springer.*, (2014), s.244-262.

[4] **A. Fry, S. Chiasson, ve A. Somayaji.** Not sealed but delivered: The (un) usability of s/mime today. *Annual Symposium on Information Assurance and Secure Knowledge Management (ASIA'12), Albany, NY.*, (2012)

[5] **Schröder, Svenja, vd.** When SIGNAL hits the Fan: On the Usability and Security of State-of-the-Art Secure Mobile Messaging. *European Workshop on Usable Security. IEEE.*, (2016)

[6] **O. W. Systems.** Open whisper systems blog: Just signal. *https://signal.org/blog/just-signal/* (2015) son ziyaret: 1 Nisan 2018

[7] **WhatsApp Inc.** Whatsapp *https://whatsapp.com* (2016) son ziyaret: 1 Nisan 2018

[8] **EFF.** Whatsapp rolls out end-to-end encryption to its over one billion users. *https://www.eff.org/deeplinks/2016/04/whatsapp-rolls-out-end-end-encryption-its-1bn-users* (2016) son ziyaret: 1 Nisan 2018

[9] **O. W. Systems.** Open whisper systems blog: Safety Number Updates. *https://signal.org/blog/safety-number-updates/* (2016) son ziyaret: 1 Nisan 2018

[10] **Sheng, S., Broderick, L., Koranda, C. A., & Hyland, J. J.** Why johnny still can't encrypt: evaluating the usability of email encryption software *Symposium On Usable Privacy and Security*, (2006), s.3-4

[11] **Ruoti, S., Andersen, J., Zappala, D., & Seamons, K.** Why Johnny still, still can't encrypt: Evaluating the usability of a modern PGP client. *arXiv preprint arXiv:1510.08555*, (2015)

[12] **Ruoti, S., Andersen, J., Heidbrink, S., O'Neill, M., Vaziripour, E., Wu, J., ... & Seamons, K.** we're on the same page: A usability study of secure email using pairs of novice users. *In Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, (2016), s.4298-4308

[13] **T. Frosch, C. Mainka, C. Bader, F. Bergsma, ve T. Holz.** How secure is textsecure?, (2014)

[14] **L. F. Cranor ve S. Garfinkel.** Security and usability: designing secure systems that people can use. *O'Reilly Media. Inc.*, (2005)

[15] **L. F. Cranor.** A framework for reasoning about the human in the loop. *UPSEC, vol. 8.*, (2008), s.1-15.

[16] **W. Fahl, S., Harbach, M., Muders, T., Smith, M., & Sander, U.** Helping Johnny 2.0 to encrypt his Facebook conversations *In Proceedings of the Eighth Symposium on Usable Privacy and Security*, (2012), s.11

[17] **Sutikno, T., Handayani, L., Stiawan, D., Riyadi, M. A., & Subroto, I. M. I.** WhatsApp, Viber and Telegram which is Best for Instant Messaging? *International Journal of Electrical and Computer Engineering.*, (2016), 6(3), s.909.

[18] **Vaziripour, E., Wu, J., O'Neill, M., Clinton, R., Whitehead, J., Heidbrink, S., ... & Zappala, D.** Is that you, Alice? A Usability Study of the Authentication Ceremony of Secure Messaging Applications. *In Symposium on Usable Privacy and Security (SOUPS).*, (2017)

[19] **Rösler, P., Mainka, C., & Schwenk, J.** More is Less: On the End-to-End Security of Group Chats in Signal, WhatsApp, and Threema. *In Proceedings of 3rd IEEE European Symposium on Security and Privacy (EuroS&P 2018).*, (2018)

[20] **Tan, Joshua, vd.** Can Unicorns Help Users Compare Crypto Key Fingerprints? *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems. ACM.*, (2017)

[21] **Justin Engler ve Cara Marie.** Secure Messaging for Normal People. *Tech. rep. NCC Group, URL: https://www.nccgroup.trust/uk/our-research/secure-messaging-for-normal-people/.* (2015) son ziyaret: 1 Nisan 2018

[22] **Tim Dierks.** The Transport Layer Security (TLS) Protocol Version 1.2. *RFC*

[23] **W. Diffie ve M. Hellman.** *New Directions in Cryptography.* IEEE Trans. Inf. Theor. 22.6*, (2006), s.644-654*

[24] **C. Paar ve J. Pelzl.** *Understanding Cryptography: A Textbook for Students and Practitioners.* Springer Berlin Heidelberg, ISBN: 9783642041013. URL: https://books.google.no/books?id=f24wFELSzkoC *(2009)*

[25] **C. Boyd ve A. Mathuria.** *Protocols for Authentication and Key Establishment.* Information Security and Cryptography. Springer, ISBN: 9783540431077. *(2003)*

[26] **Kızılöz H.E.** *Metin Tabanlı İnsan Etkileşim İspatı Sistemleri İçin İnsan Hesaplama Kullanımı.* TOBB Ekonomi ve Teknoloji Üniversitesi, Fen Bilimleri Enstitüsü*, Doktora Tezi, Ankara, (2016), s.16-18*

[27] **Moxie Marlinspike ve Trevor Perrin.** *The Double Ratchet Algorithm.* URL: https://whispersystems.org/docs/specifications/doubleratchet/. *(2016) son ziyaret: 1 Nisan 2018*

[28] **Moxie Marlinspike ve Trevor Perrin.** *The X3DH Key Agreement Protocol.* URL:https://whispersystems.org/docs/specifications/x3dh/. *(2016) son ziyaret: 1 Nisan 2018*

EKLER

EK 1 : Örnek Çalışan "pushserver.yml" Konfigürasyonu

EK 2 : Örnek Çalışan "textsecure.yml" Konfigürasyonu

EK 1 - Örnek Çalışan "pushserver.yml" Konfigürasyonu

redis:
url: redis://localhost:6379/2
authentication:
servers:
-
name: 123
password: 123

gcm:
xmpp: false
apiKey: AIdrPdHGcf-4lsuNbfo94Acueplvh3nKrJCahrp #değiştir
senderId: 38750274690 #değiştir
redphoneApiKey: AIddSyAsviyMy8jKe8chCEfr8NbeqGghy7oOCi4 #değiştir

server:
applicationConnectors:
- type: http
port: 9090
adminConnectors:
- type: http
port: 9091
gzip:
enabled: true

logging:
level: INFO
appenders:
- type: file
currentLogFilename: /tmp/pushserver.log
archivedLogFilenamePattern: /tmp/pushserver-%d.log.gz
archivedFileCount: 5
- type: console

EK 2 - Örnek Çalışan "textsecure.yml" Konfigürasyonu

```
# This is the sample config/textsecure.yml file for the TextSecure Server
# Pay attention! To start TextSecure server you will need to install and start PushServer

twilio:
accountId: AF48shn50alcb49sha82bakd94lf3456ad #değiştir
accountToken: 2bdu4065hsgb5038dnsls04mavdj23sd #değiştir
numbers:
-
+14354687424 #değiştir
localDomain: foo.org

push:
host: localhost
port: 9090
username: 123
password: 123

s3:
accessKey: AKTNDHUIOW5NSJRFLACA #değiştir
accessSecret: Hjtu4osnnfhY5wbkYgfjDdsadBfkslDdstdsrsmK #değiştir
attachmentsBucket: newbucketqwerty #değiştir

directory:
url: "redis://localhost:6379/0"

cache:
url: "redis://localhost:6379/1"

server:
applicationConnectors:
- type: http
port: 8080
#keyStorePath: config/example.keystore
#keyStorePassword: example
#validateCerts: true
adminConnectors:
- type: http
port: 8081
#keyStorePath: config/example.keystore
#keyStorePassword: example
#validateCerts: true

websocket:
enabled: true
```

```
messageStore: # Mesajları saklamak için Postgres veritabanı ayarları
driverClass: org.postgresql.Driver
user: "postgres"
password: "postgres"
url: "jdbc:postgresql://localhost:5432/messagedb"

database:
driverClass: org.postgresql.Driver
user: "postgres"
password: "postgres"
url: "jdbc:postgresql://localhost:5432/accountsdb"
properties:
charSet: UTF-8

#federation: # pasif durumda

logging:
level: INFO
appenders:
- type: file
currentLogFilename: /tmp/textsecureshserver.log
archivedLogFilenamePattern: /temp/textsecureserver-%d.log.gz
archivedFileCount: 5
- type: console

redphone:
authKey: 1234567890
```

ÖZGEÇMİŞ

Ad-Soyad : Muhammet Şakir ŞAHKULUBEY

Uyruğu : TC

Doğum Tarihi ve Yeri : 1992 İstanbul

E-posta : msakirsahkulubey@gmail.com

ÖĞRENİM DURUMU:

- **Lisans** : 2014, Sakarya Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği

MESLEKİ DENEYİM VE ÖDÜLLER:

Yıl	Yer	Görev
2014-	DESTEL Bilişim Çözümleri	Bilgisayar Mühendisi

YABANCI DİL: İngilizce

TEZDEN TÜRETİLEN YAYINLAR, SUNUMLAR VE PATENTLER:

- **ŞAHKULUBEY M.Ş.**, ALTUNCU E., BIÇAKCI K., "Emniyet Numaraları Açık Anahtar Doğrulamada Ne kadar Emniyetli? Parmakizi ve Emniyet Numarası Yöntemlerinin Kullanılabilirliği ve Güvenliğinin İncelenmesi", Akademik Bilişim Konferansı, 2018, Karabük