T.C.
İSTANBUL ÜNİVERSİTESİ-CERRAHPAŞA
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ

HCE SİSTEMLERDE GÜVENLİ ÇEVRİMDIŞI ÖDEME İŞLEMLERİ

Ahmet Zahit ÖZDEMİR

DANIŞMAN
Doç. Dr. Muhammed Ali AYDIN

Bilgisayar Mühendisliği Anabilim Dalı

Bilgisayar Mühendisliği Programı

İSTANBUL-2020

Bu çalışma, 20.05.2020 tarihinde aşağıdaki jüri tarafından Bilgisayar Mühendisliği Anabilim Dalı, Bilgisayar Mühendisliği Programında Yüksek Lisans tezi olarak kabul edilmiştir.

Tez Jürisi

Doç. Dr. Muhammed Ali AYDIN (Danışman) İstanbul Üniversitesi-Cerrahpaşa Mühendislik Fakültesi

Dr. Öğr. Üyesi Özgür Can TURNA İstanbul Üniversitesi-Cerrahpaşa Mühendislik Fakültesi	Doç. Dr. Akhan AKBULUT İstanbul Kültür Üniversitesi Mühendislik Fakültesi

20.04.2016 tarihli Resmi Gazete'de yayımlanan Lisansüstü Eğitim ve Öğretim Yönetmeliğinin 9/2 ve 22/2 maddeleri gereğince; Bu Lisansüstü teze, İstanbul Üniversitesi-Cerrahpaşa'nın abonesi olduğu intihal yazılım programı kullanılarak Lisansüstü Eğitim Enstitüsü'nün belirlemiş olduğu ölçütlere uygun rapor alınmıştır.

ÖNSÖZ

Yüksek lisans eğitimimde ve tez sürecimde danışmanlığımı üstlenen Doç. Dr. Muhammed Ali Aydın hocama eğitimimde ve tez aşamamdaki yardımları, yol göstermesi, öğrenmemi sağladığı bilgiler ve sunduğu manevi desteği için teşekkürlerimi sunarım.

Eğitim hayatımda ve seçtiğim ihtisas alanda gerekli imkânları sunan ve benden desteklerini esirgemeyen aileme çok teşekkür ederim.

Mayıs 2020 Ahmet Zahit ÖZDEMİR

İÇİNDEKİLER

Sayfa No

ÖNSÖZ ..... **iv**
İÇİNDEKİLER ..... **v**
ŞEKİL LİSTESİ ..... **vii**
TABLO LİSTESİ ..... **ix**
SİMGE VE KISALTMA LİSTESİ ..... **x**
ÖZET ..... **xii**
SUMMARY ..... **xiii**
1. GİRİŞ **1**
2. GENEL KISIMLAR **2**
2. HOST CARD EMULATION TEKNOLOJİSİ VE SİSTEMİ 2
2.1 RFID 2
2.1.2 RFID Eşleme Mekanizması 4
2.2 YAKIN ALAN İLETİŞİMİ 5
2.2.1 NFC Çalışma Modları 6
2.2.1.1 Birebir Çalışma Modu 7
2.2.1.2 Okuma/Yazma Modu 8
2.2.1.3 Kart Emülasyon Modu 9
2.2.2 HCE ve SE Farkları 10
2.3 HCE GERÇEKLEME MİMARİLERİ 13
2.3.1 Bulut Tabanlı HCE Mimarisi 13
2.3.2 Token Tabanlı HCE Mimarisi 14
3. MALZEME VE YÖNTEM **16**
3.1 GÜVENLİK KATMANLARI 16
3.1.1 Uygulama Güvenliği 16
3.1.2 Cihaz Güvenliği 17
3.1.3 İletişim Güvenliği 17
3.1.4 Dinamik Veri 17
3.2 KRİPTOGRAM ALGORİTMALARI 20

3.2.1 DES Şifreleme 20
3.2.2 AES Şifreleme 24
3.2.3 RSA Şifreleme 31
3.2.4 AES MAC Algoritması 35
3.3 WHITEBOX KÜTÜPHANESİ 37
3.4 HARDWARE SECURITY MODULE 38
3.5 MOBİL ÖDEME UYGULAMASI 39
3.5.1 Mobil Ödeme Uygulaması Genel Yapısı 40
3.5.2 Uygulama Senaryosu 42
3.5.3 Güvenlik Mimarisi 44
3.5.3.1 Uygulamanın Güvenliği 44
3.5.3.2 Uzak Sunucu ile İletişimin Güvenliği 48
3.5.4 Uygulama Adımları 49
3.5.4.1 HCE İlklendirilmesi 49
3.5.4.2 Dijital Kart Oluşumu 53
3.5.4.3 Token Oluşumu 55
3.5.4.4 HCE Silme 57
3.5.4.5 Ödeme İşlemi 58
4. BULGULAR 63
4.1 GÜVENLİK ANALİZİ 63
4.1.1 Donanım Saldırı Direnci 63
4.1.2 İletişim Kanalı Saldırı Direnci 63
4.1.2.1 MPA - HCE Bulut İletişimi 63
4.1.2.2 MPA - Validatör İletişimi 64
4.1.2.3 Operatör - HCE Bulut İletişimi 64
4.1.3 Mantıksal Saldırılar 64
4.1.3.1 Güvenilir Olmayan Mobil Ödeme Uygulaması 64
4.1.3.2 Güvenilir Olmayan Validatör 65
4.2 PERFORMANS ANALİZİ 65
5. TARTIŞMA VE SONUÇ 66
KAYNAKLAR 69
ÖZGEÇMİŞ 73

ŞEKİL LİSTESİ

Sayfa No

Şekil 2.1: RFID Sistemin 3 Ana Parçası [1] 3

Şekil 2.2: NFC Çalışma Modları 6

Şekil 2.3: Peer-to-Peer Modunun Protokol Yığını [6] 7

Şekil 2.4: Okuma/Yazma Modunun Protokol Yığını [6] 9

Şekil 2.5: Kart Emülasyon Modunun Protokol Yığını [6] 10

Şekil 2.6: SE Tabanlı ve HCE Tabanlı NFC Ödeme Altyapısı [7] 11

Şekil 2.7: Bulut Tabanlı HCE Mimarisi [8] 14

Şekil 2.8: Token Tabanlı HCE Mimarisi [8] 15

Şekil 3.1: Mobilde Önemli 10 Risk [12] 18

Şekil 3.2: DES Algoritması Çalışma Mantığı [15] 21

Şekil 3.3: DES Algoritmasının Karmaşıklaştırma Fonksiyonu [15] 22

Şekil 3.4: DES Algoritmasının Anahtar Üretme Fonksiyonu [15] 23

Şekil 3.5: DES Algoritmasında Her Döngüde Gerçekleşen Adımlar 24

Şekil 3.6: AES Şifreleme Algoritmasının Genel Görünümü 25

Şekil 3.7: AES Anahtar ile İşleme Adımı 26

Şekil 3.8: AES S-Kutusu Örneği 27

Şekil 3.9: AES Satır Kaydırma İşlemi 27

Şekil 3.10: AES Sütun Karıştırma İşlemi 28

Şekil 3.11: AES Anahtar Türetme İşlemi 29

Şekil 3.12: AES Anahtar Türetme İşlemi Devamı 29

Şekil 3.13: AES Key Türetimi Rcon Sütunları 30

Şekil 3.14: AES Şifre Çözme Algoritmasının Genel Görünümü 30

Şekil 3.15: RSA Algoritması Anahtar Türetimi Adımları 32
Şekil 3.16: RSA Algoritmasında Şifreleme ve Şifre Çözme 33
Şekil 3.17: CRT Dönüşümü 34
Şekil 3.18: CRT Alanında Üssel İşlemler 34
Şekil 3.19: MAC Algoritması Kullanımı 35
Şekil 3.20: MAC Algoritması Akış Diyagramı 36
Şekil 3.21: WBC Üst Düzey Kavramsal Genel Görünüm 38
Şekil 3.22: Sistem Mimarisi 39
Şekil 3.23: Mobil Ödeme Uygulaması Bileşenleri 40
Şekil 3.24: MPA Senaryosu 42
Şekil 3.25: HCE İlklendirilmesi Adımları 52
Şekil 3.26: Dijital Kart Oluşum Adımları 54
Şekil 3.27: Token Oluşum Adımları 56
Şekil 3.28: Token Yapısı 56
Şekil 3.29: HCE Silme Adımları 58
Şekil 3.30: Ödeme İşlemi Adımları 62
Şekil 3.31: Ödeme İşlemi Adımları 62
Şekil 5.1: Dünya Geneli Akıllı Telefon Kullanıcı Sayısı [33] 66
Şekil 5.2: 2016–2019 Yıllarında Dünya Genelinde Mobil Uygulama İndirme Sayısı [34] 67

TABLO LİSTESİ

Sayfa No

Tablo 2.1: HCE ve SE'nin Farklılıkları 12

Tablo 3.1: Mobil Uygulama Güvenlik Analizi 19

Tablo 3.2: Uygulamanın Güvenlik Önemleri 44

SİMGE VE KISALTMA LİSTESİ

Simgeler **Açıklama**

φ(n) : Euler'in Totient Fonksiyonu

Kısaltmalar **Açıklama**

RFID **:** Radio Frequency Identification – Radyo Frekansı Tanımlama
NFC **:** Near Field Communication – Yakın Alan İletişimi
SE **:** Secure Element – Güvenli Eleman
HCE **:** Host Card Emulation
kbps **:** Kilobits per Second
ISO **:** International Organization for Standardization
IEC **:** International Electrotechnical Commission
ECMA **:** European Computer Manufacturers Association
JIS **:** Japanese Endustry Standart
NFC-DEP **:** NFC Data Exchange Protocol
NDEF **:** NFC Data Exchange Format
LLCP **:** Logical Link Control Protocol
SNEP **:** Simple NDEF Exchange Protocol
NFC-SEC **:** NFCIP-1 Security and Services Protocol
TLS **:** Transport Layer Security
SIM **:** Subscriber Identity Module
TEE **:** Trusted Execution Environment
OWASP **:** Open Web Application Security Project
DES **:** Digital Encryption Standart
AES **:** Advanced Encryption Standart
NIST **:** National Institute of Standards and Technology
RSA **:** Rivest–Shamir–Adleman
MAC **:** Message Authentication Code
CRT **:** Chinese Remainder Theorem
CBC **:** Cipher Block Chaining
WBC **:** WhiteBox Cryptography
EMV **:** EuroPay, MasterCard and Visa

HSM **:** Hardware Security Module

SSL **:** Secure Sockets Layer

PIN **:** Personal Identification Number

TLS **:** Transport Layer Security

SHA **:** Secure Hash Algorithm

FCM **:** Firebase Cloud Messaging

LAN **:** Local Area Network

ID **:** Identification

AID **:** Application Identifier

PCI DSS **:** Payment Card Industry Data Security Standard

ÖZET

YÜKSEK LİSANS TEZİ

HCE SİSTEMLERDE GÜVENLİ ÇEVRİMDIŞI ÖDEME İŞLEMLERİ

Ahmet Zahit ÖZDEMİR

İstanbul Üniversitesi-Cerrahpaşa

Lisansüstü Eğitim Enstitüsü

Bilgisayar Mühendisliği Anabilim Dalı

Danışman : Doç. Dr. Muhammed Ali AYDIN

Akıllı telefonlarda elektronik ödeme işlemleri geçtiğimiz yıllarda artış göstermeye başlamasıyla, mobil cihazlarda bulunan yakın alan iletişimi (NFC) donanımı ile mobil ödemeler popülerlik kazandı. Fiziki kartlar yerine, HCE teknolojisi kullanımı önem kazanarak, mobil ödeme uygulamalarında HCE sistemleri aktif rol oynamaya başladı. Bu süreçte, çevrimiçi HCE sistemlerinin güvenliği bulutta sağlandı. İnternet bağlantısı olmama durumunda da ödeme yapılabilmesi için, mobil cihaz tarafında güvenlik kontrolleri yapılarak ve yüksek güvenlik önlemleri olarak cihaz içerisinde güvenilir saklama alanı oluşturup, burada saklanan harcama bilgileri ile güvenli çevrimdışı mobil ödeme sağlanabilir.

Mayıs 2020, 86 sayfa.

Anahtar kelimeler: HCE, NFC, Ulaşım, Güvenlik, Ödeme

SUMMARY

M.Sc. THESIS

OFFLINE SECURE PAYMENT TRANSACTIONS ON HCE SYSTEMS

Ahmet Zahit ÖZDEMİR

Istanbul University-Cerrahpasa

Institute of Graduate Studies

Department of Computer Engineering

Supervisor : Assoc. Prof. Dr. Muhammed Ali AYDIN

Electronic payments on smartphones started to increase in recent years, thus mobile payments gained popularity with the mobile devices embedded Near Field Communication (NFC) hardware. HCE systems started to play an active role in mobile payment applications instead of using physical cards. In this process, the security of online HCE systems was secured in the cloud. In order to be able to make payments even in the absence of an internet connection, secure offline mobile payment can be provided with the transaction information stored here, by making security checks on the mobile device and taking high security precautions, creating a safe storage area within the device.

May 2020, 86 pages.

Keywords: HCE, NFC, Transportation, Security, Payment

1. GİRİŞ

Teknolojinin gelişmesindeki en büyük etkenler, üretici ve geliştirici firmalar arası rekabet ve kullanıcı talebidir. Ulaşım sistemlerindeki teknolojinin gelişmesi ise birçok nedene dayanır.

Başlarda, ulaşım araçları sadece nakit para ile kullanılırken, sonrasında seferler arası aksaklığı önlemek adına bilet kullanımına geçilmiştir. Akbil olarak bilinen, iButton cihazlarının kullanımının dünya çapında birçok alan da kullanımının yaygınlaşmasıyla da, İstanbul'daki bütün ulaşım sistemleri 1998 yılında akbil sistemine geçmiştir. İlerleyen dönemlerde akbil ile beraber jetonlu geçiş sistemleri de kullanılmıştır. RFID (Radio Frequency Identification) kart olan İstanbul Kart kullanımı ise 2010 yılından itibaren İstanbul'un resmi ulaşım kartı olmuştur.

RFID kartların kullanılmasından sonra, ulaşım sistemlerindeki gelişmeler kartlar üzerinden değil, NFC (Near Field Communication) donanımına sahip olan akıllı telefonlar üzerinden yürütülmüştür. Dünya üzerinde birçok büyük şehirde akıllı telefon uygulamaları ile ulaşım sağlanabilmektedir.

Fakat hali hazırda kullanılan HCE (Host Card Emulation) ulaşım uygulamaları, çevrimiçi olarak çalışmaktadır. İstanbul gibi yerin 4-5 kat altında metro durakları bulunan bir şehirde, çevrimiçi bir ulaşım uygulamasının kullanılması mümkün olmadığı için, çevrimdışı çalışabilen bir ulaşım uygulaması gerekmektedir. Çevrimdışı bir uygulama demek, ulaşımın bulutta değil, mobil cihazda güvenli olarak gerçekleşmesi demektir.

Mobil cihazda, eğer ki SE (Secure Element) kullanılmıyor da HCE teknolojisi kullanılıyorsa, çevrimdışı gerçekleşen bir uygulamada kişisel bilgilerin veya bakiyenin değiştirilmesi gibi güvenlik sorunları olabileceğinden ötürü, uygulamada da maksimum güvenlik önlemleri alınmalı ve verinin değiştirilemiyor olması taahhüt edilmedir. Bu tez kapsamında, HCE teknolojisinde çevrimdışı güvenli bir ulaşım uygulamasında gizliliğin ve veri bütünlüğünün nasıl sağlandığı incelenecektir.

2. GENEL KISIMLAR

2. HOST CARD EMULATION TEKNOLOJİSİ VE SİSTEMİ

Host Card Emulation, adından da genel kapsamda anlaşılacağı üzere, SE'ye ihtiyaç duymadan bir akıllı kartı tüm fonksiyonlarıyla emüle edebilen bir teknolojidir.

Akıllı telefonun işletim sistemi ile akıllı kartlar emüle edilir ve NFC donanımı ile de temassız haberleşme sağlanarak, akıllı telefonun bir akıllı kart gibi davranması sağlanır.

Bu sayede akıllı telefonumuzu kredi kartı, banka kartı, ulaşım kartı gibi davranmasını sağlayarak, birçok kartı cüzdanımızda değil akıllı telefonumuzda taşımış oluruz.

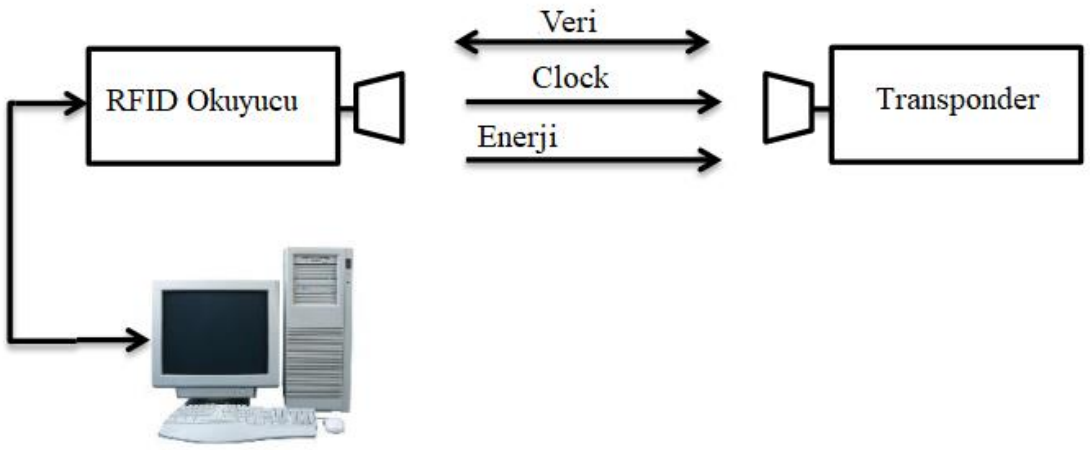
HCE teknolojisini anlayabilmek için öncelikte RFID'yi anlayarak, bundan türeyen NFC iletişimi anlamak gerekmektedir. Tez kapsamında bu konularda alt başlıklarda ele alınmıştır.

2.1 RFID

RFID kullanımı ilk olarak altmış yıl önce İngiliz ordusu tarafından II. Dünya Savaşı'nda başlamıştır. İngiliz ordusu, RFID kullanımını radar sistemini iyileştirerek uçak gibi ordu nesnelerini tanımlayarak gerçekleştirme amacı gütmüştür. RFID ilk olarak 1960'lı yıllarda, ticari amaçlar için bir çözüm olarak kabul edilmiştir. 70'li ve 80'li yıllarda ticari uygulamalar için geliştirilmeler yapılmıştır. Daha sonra, 1998 yılında Massachusetts Institute of Technology (MIT)'de yapılan çalışmalar sonrasında RFID'deki ilk gelişme Tag (Etiket) adı verilen elektronik nesne olmuştur.

RFID sistemleri **Şekil 2.1**'den görüleceği üzere 3 ana parçadan oluşurlar; etiket (tag), okuyucu (interrogator) ve kontrolcü (controller). Etiket, bir nesneye bağlanabilecek şekilde paketlenen ve antene takılan bir mikroçiptir. Etiketler benzeri olmayan bir seri numarası ve kimlik numarasına sahip olmaktadır. Bu da okuyucu ile iletişim kurmasını (sinyal alıp göndermesini) sağlar. Sorgulayıcının (okuyucu) görevi, antenden elektromanyetik dalgalar yaymaktır. Sorgulayıcıdan yayılan bu dalgalar etiket tarafından emilir ve etiketin mikroçipini beslemek

için enerji olarak kullanılır. Dalgalardan sağlanan bu enerji sayesinde, etiketin tanımlamasını (identification) içeren bir sinyalin sorgulayıcıya gönderilmesi sağlanır.



Şekil 2.1: RFID Sistemin 3 Ana Parçası [1]

İki tip etiket vardır; Yüksek Frekanslı (High Frequency) etiketler ve Ultra Yüksek Frekans (Ultra High Frequency) etiketlerdir. Yüksek Frekanslı etiketler, 0.8 metre uzaklıktan 3~30MHz frekanslarında kullanılabilirler. Ultra Yüksek Frekanslı etiketler ise 13 okuyucu ile 15 metreden 300 MHz ~3GHz frekanslarında okunabilmektedirler [1]. Yüksek Frekanslı etiketlerde daha büyük silikon çipler kullanıldığından daha fazla güvenlik işlevselliği sağlamaktalar. Ayrıca etiketlerin aktif veya pasif modlar olmak üzere iki modu mevcuttur. Bir etiket, okuyucudan bilgi döndürmek için verici kullandığında aktif moddadır. Aktif etiketlerin çoğu pille yani ek güç ile çalışmaktadırlar. Pasif etiket, güç kaynağına sahip olmayan etikettir ve enerji kaynağı olarak vericiden gönderilen elektromanyetik dalgaları kullanırlar.

2.1.2 RFID Eşleme Mekanizması

Eşleme Mekanizması (Coupling Mechanism) RFID etiket ve okuyucu arasındaki iletişimi sağlayan tekniktir. İletişimde bulunan pasif ve aktif modu oluşturmak için Eşleme Tekniği (Coupling Technique) kullanılır [1]. Üç ana metot bulunmaktadır. Bunlar:

- RFID Geri Saçılma Kuplaj (RFID Backscatter Coupling)

- RFID Kapasitif Kuplaj (RFID Capacitive Coupling)

- RFID İndüktif Kuplaj (RFID Inductive Coupling)

Bu metotlar, kendilerine has özelliklere sahiptirler. Bu da metotların birbirlerinden farklı olmasını sağlamaktadır. Bu metot tipleri, RFID'nin farklı yönlerini etkilerler, örneğin: iletişim mesafesi, frekans aralığı ve RFID donanımının diğer bileşenleri.

RFID iletişim mesafesi veya bağlantı aralığı 3 bölüme ayrılabilir. Bunlar şu şekildedir; Yakın Mesafeli Bağlantı (Close Range Coupling), Uzaktan Bağlantı (Remote Coupling) ve Uzak Mesafeli Bağlantı (Long Range Coupling).

Yakın Mesafeli Bağlantılarda, iletişim mesafesi 1 santimetre civarında olmaktadır. Buda etiketin, okuyucuya dokundurulacak kadar yakında olması gerektiği anlamına gelmektedir. Bu yakın mesafe de, etiket manyetik alandan büyük miktarda enerji kazanabileceğinden ötürü etiket tarafından enerji emilmesi durumunda bazı faydalar sağlar. Bu bağlantının bir diğer avantajı ise, bu gereksinime ihtiyaç duyan sistemler için yüksek güvenlik sağlar.

Uzaktan Bağlantı, 1 santimetre ile 1 metre arasında çalışır. Bu aralık genellikle pasif etiketlerle uygulanır ve yakın bağlantıya benzemektedir.

Uzun Mesafeli Bağlantı, yakın ve uzak bağlantılardan daha uzun mesafeler için kullanılır. Normalde mesafe aralığı 1 metre ile 10 metre arasındadır ve bu aralık RFID için belirtilenden daha yüksek frekans kullanır. Ayrıca, önceki aralıklardan farklı olarak, bu bağlantı Backscatter Coupling yöntemini uygulamaktadır.

2.2 YAKIN ALAN İLETİŞİMİ

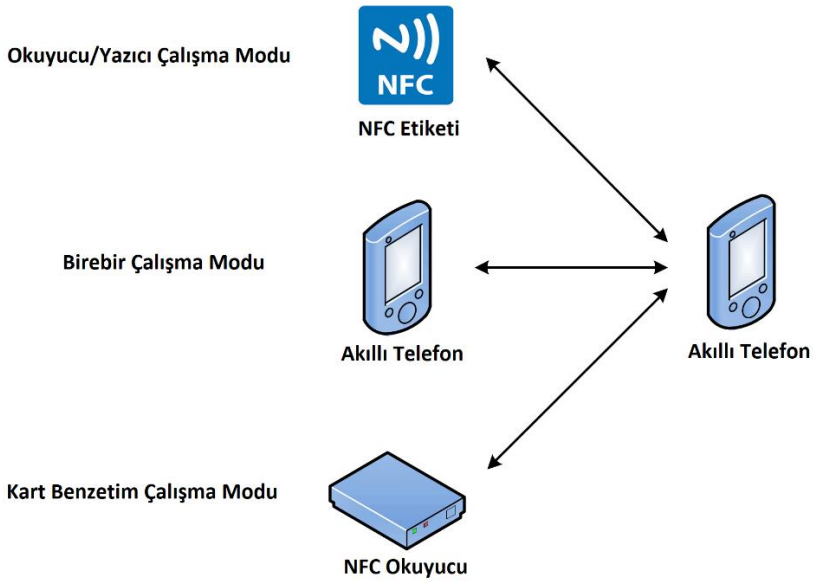
Near Field Communication, Sony ve Philips tarafından 2002 yılında bulunmuş ve geliştirilmiş bir teknolojidir. Çıktığı tarihte ECMA (Avrupa Bilgisayar Üreticileri Birliği) tarafından onaylanmış, 2003 yılında ise ISO/IEC (Uluslararası Standartlar Örgütü / Uluslararası Elektroteknik Komisyonu) bu teknolojiyi, standart bir teknoloji olarak kabul ettiğini açıklamıştır. ECMA–340 ve ISO/IEC 18092 ile NFCIP–1 (Yakın Alan İletişimi Arayüz ve Protokolü-1) ve ECMA–352 ve ISO/IEC 21481 ile NFClP–2 (Yakın Alan İletişimi Arayüz ve Protokolü-2) standardı olmak üzere, ECMA ve ISO/IEC tarafından iki ayrı NFC standardı geliştirilmiştir [2]. 2004 yılında ise NFC Forum; NXP, Sony ve Nokia tarafından bu teknolojinin iyileştirilmesi ve yaygınlaştırılması amacıyla kurulmuştur.

NFC, RFID protokolleri, arayüzleri ve altyapısını kullanarak geliştirilmiştir, bu sayede de mevcut RFID teknolojisine uyumlu bir teknoloji olduğu söylenilebilir. Kablosuz iletişim mesafesi 4 santimetre olan bu teknoloji, birbirine yakın 2 cihaz arasında temassız iletişimi maksimum 424 kbps hızında, 13.56 MHz frekansında sağlayabilmektedir [4]. Bu teknolojisi, 2 aktif, pasif veya iki aktif cihazdaki manyetik modülü eşleyerek iletişimi gerçekleştirir. ISO/IEC 14443 Temassız Akıllı Kart Teknoloji standartlarına da uyum sağlayan bir altyapıya sahip olmaktadır. Bahsedilen uyumlu altyapısı sayesinde, kolay bağlantı, hızlı işlem gücü ve basit veri paylaşımı sağlayabilmektedir. NFC'nin mevcuttaki diğer temassız altyapılar ile uyumlu olması ve kullanıcıların bir cihazı farklı sistemlerde kullanabilmelerini sağlamaktadır. Bahsedildiği üzere, NFC teknolojisi RFID'den türetilerek ortaya çıkmış bir teknolojidir. Ancak onu RFID'den farklı kılan özelliği ise daha fazla iletişim olanakları sağlamasıdır. Bu olanaklar arasında NFC'yi RFID'den ayıran en karakteristik özellik ise NFC cihazlar arasında iki yönlü veri aktarımı yapılabilmesidir [3]. İki NFC cihaz arasındaki iletişimi sağlamak için, cihazları birbirine yaklaştırmak veya dokundurmak yeterli olacaktır. Bu iletişimin kısa mesafede gerçekleşmesinden ötürü kendiliğinden oluşan bir güvenlik (inherent security) vardır. Daha sonra NFC protokolü otomatik olarak cihazların pasif veya aktif modda olabileceği eşler arası peer-to-peer bağlantı kurar. Pasif modda, cihazlardan sadece biri radyo frekansı alanı üretir diğer cihaz veri iletimi için yük modülasyonunu uygular. Kendi RF alanını yaratabilen cihazlar aktif cihazlardır ve tüm öncü cihazlar birer aktif cihazdır. Kendi RF alanını yaratamayan cihazlar ise pasif cihaz olarak nitelendirilir.

NFC'nin temel teknik özelliği, birçok kablosuz teknolojiyi, temel parametreleri ve temassız kart teknolojisi için mevcut standartlardaki unsurları tamamlamasıdır. Akıllı telefonlar ile bütünleşmiş çalışması NFC'nin en güçlü özelliklerinden biridir. Bu özelliğin kullanımı ile beraber kısa zaman içerisinde birçok yenilikçi ve kendine has NFC servisi var olmuştur. Bu servise örnek olarak ulaşım uygulamaları, sadakat uygulamaları, akıllı poster uygulamaları, biletleme, erişim kontrolü, mobil ödemeler verilebilir. Bundan ötürü NFC teknolojisi, insanların günlük hayatında sıklıkla kullanmaya ihtiyaç duyduğu birçok objeyi yanlarında taşıma zorunluluğunu azaltarak, bütün bu servisleri NFC donanımına sahip akıllı telefonlar ile gerçekleştirmeyi sağlamaktadır.

2.2.1 NFC Çalışma Modları

Aktif ve pasif iletişim modu kapsamında NFC iletişiminin üç ayrı çalışma modu bulunmaktadır. **Şekil 2.2**'de görüldüğü üzere, NFC donanımına sahip Akıllı Telefonun, bir NFC okuyucuya veya bir başka NFC uyumlu Akıllı Telefona veya bir NFC etiketine dokundurulması veya yaklaştırılması ile üç farklı çalışma yöntemi ortaya çıkarılmıştır. Bunlar; Card Emulation (Kart Benzetim), Peer-to-Peer (Birebir İletişim) ve Read/Write (Okuma/Yazma) yöntemlerinden oluşmaktadır [4].



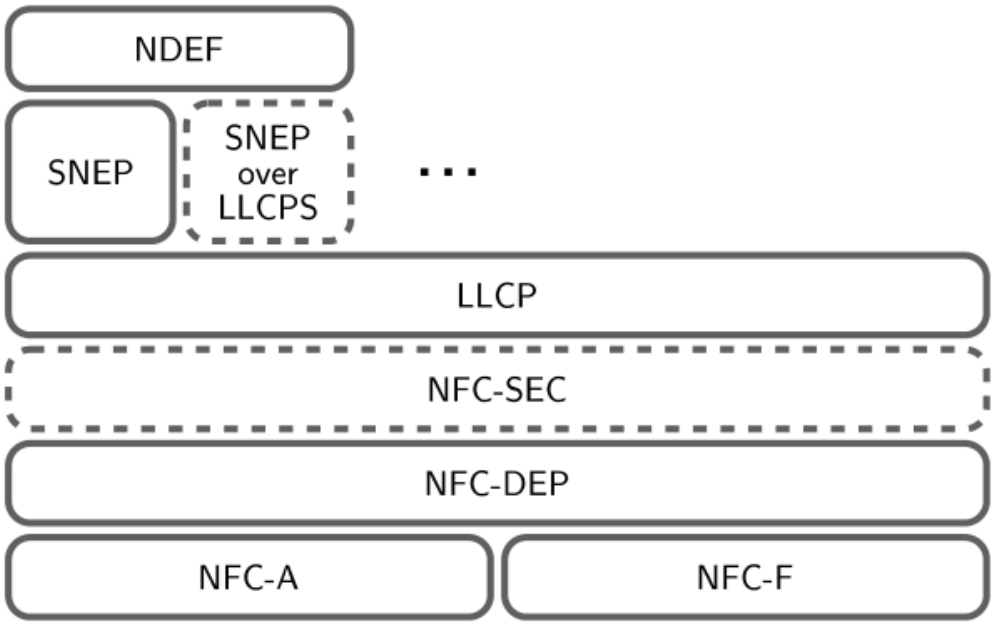
Şekil 2.2: NFC Çalışma Modları

2.2.1.1 Birebir Çalışma Modu

Birebir çalışma modunda, NFC cihazları, temassız akıllı kart okuyucusu veya temassız akıllı kart olmaktan ziyade, birbirleriyle doğrudan iletişim kurabilmeleri için tasarlanmıştır. Bu nedenle, hem okuyucu hem de kart tarafındaki protokol elemanları bu modda birleştirildi ve peer-to-peer moda özel daha yüksek katman protokolleri ile genişletildi.

Şekil 2.3'te görüldüğü üzere, NFC'nin birebir modunun protokol yığını özetlenmiştir. En düşük katman; modülasyon, kodlama, çerçeveleme ve cihaz algılama prosedürünü tanımlayan en düşük seviyeli iletişim teknolojisidir (106 kbps için NFC-A ve diğer veri hızları için NFC-F).

NFC-A ve NFC-F, sırasıyla ISO/IEC 14443 Type A ve JIS X 6319-4 akıllı kart protokollerine dayanmaktadır.



Şekil 2.3: Peer-to-Peer Modunun Protokol Yığını [6]

NFC Veri Değişimi Protokolü (NFC-DEP), NFC-A ve NFC-F'ye cihaz etkinleştirme ve temel veri alışverişi yetenekleri ekler. NFC-DEP'e kadar olan katmanlar ISO/IEC 18092'de

standartlaştırılmıştır. NFC-DEP'de, bir cihaz iletişimi kontrol eden Master'ın (başlatıcı) rolünü alır ve bir cihaz da Slave rolünü (hedef) alır. Ayrıca, iki cihaz aktif modda veya pasif modda iletişim kurabilmektedirler. Aktif modda, hem başlatıcı hem de hedef dönüşümlü olarak kendi Yüksek Frekanslı taşıyıcı sinyallerini üretmektedirler. Pasif modda, başlatıcı kendi Yüksek Frekanslı sinyalini kullanarak veri gönderirken, hedef, başlatıcıdan gönderilen Yüksek Frekanslı taşıyıcı sinyalindeki yükü modüle ederek yanıt vermektedir.

NFC-DEP'in üstünde, Mantıksal Bağlantı Kontrol Protokolü (LLCP) bağlantısına yönelik ve bağlantısız uçtan uca iletişim sağlanmaktadır. Bu iletişim uç noktaları, NFC cihazlarındaki servisler ve uygulamalar tarafından kullanılabilmektedirler ve aynı anda birden fazla iletişim kanalına izin verebilmeyi sağlamaktadırlar. LLCP'nin üzerinde bulunan ise, bilinen ve yaygın olarak desteklenen bir hizmet olan, Basit NDEF Değişim Protokolü'dür (SNEP). Bu protokol uygulamalar arasında NDEF mesajlarının kolayca değiştirilmesine izin verir. NFC Veri Değişimi Biçimi olan NDEF, uygulamaların kullanılan iletişim modundan ve kullanılan iletişim teknolojisinden bağımsız olarak veri alışverişi yapmasına olanak tanır.

Birbirine iletişim modu için güvenlik sağlayan iki protokol vardır: NFCIP-1 Güvenlik ve Hizmet Protokolü (NFC-SEC) ve TLS over NFC LLCP [5].

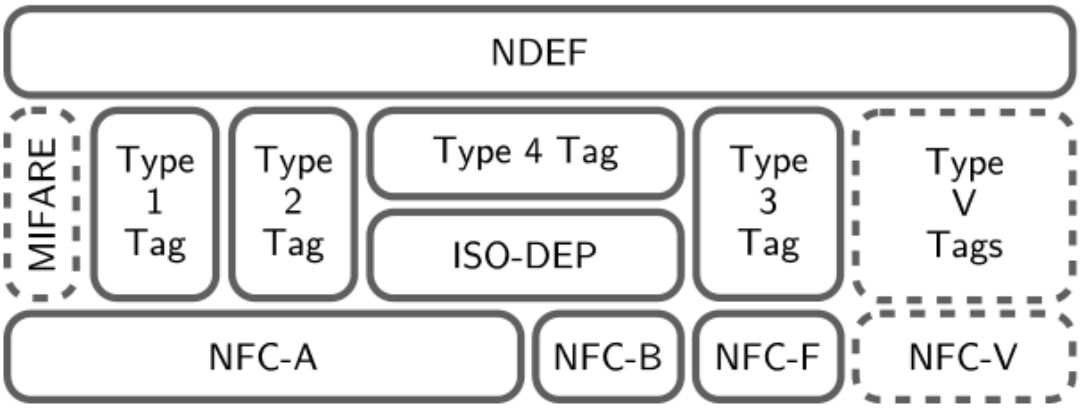
NFC-SEC; NFC-DEP ile LLCP katmanı arasında kanal şifrelemesi ve kimlik doğrulaması sağlarken, LLCPS ise LLCP hizmetleri için HTTPS'e benzer bir TLS (Aktarım Katmanı Güvenliği) protokol katmanı sağlamaktadır. Ancak, bu güvenlik protokollerinin hiçbirinin NFC donanımlı akıllı telefonlarda mevcut olduğu bilinmemektedir.

2.2.1.2 Okuma/Yazma Modu

Okuma/Yazma modunda NFC donanımlı akıllı telefon, etiketleri okuyabilir, etiketlere yazma yapabilir. Temassız iletişim de bu modu destekler.

Şekil 2.4'te, NFC'nin Okuma/Yazma modunun protokol yığını gösterilmektedir. Okuma/Yazma modunda, NFC aygıtları genellikle üç alt düzey iletişim teknolojisini destekler, bunlar: NFC-A (ISO/IEC 14443 Type A tabanlı), NFC-B (ISO/IEC 14443 Type B tabanlı) ve NFC-F (JIS X 6319-4 tabanlı). Bazı NFC aygıtları ayrıca ISO/IEC 15693 tabanlı NFC-V'yi de

desteklemektedirler. ISO Veri Değişimi Protokolü (ISO-DEP) katmanı, ISO/IEC 14443-4'te tanımlanan ortak taşıma protokolüne eşdeğerdir.



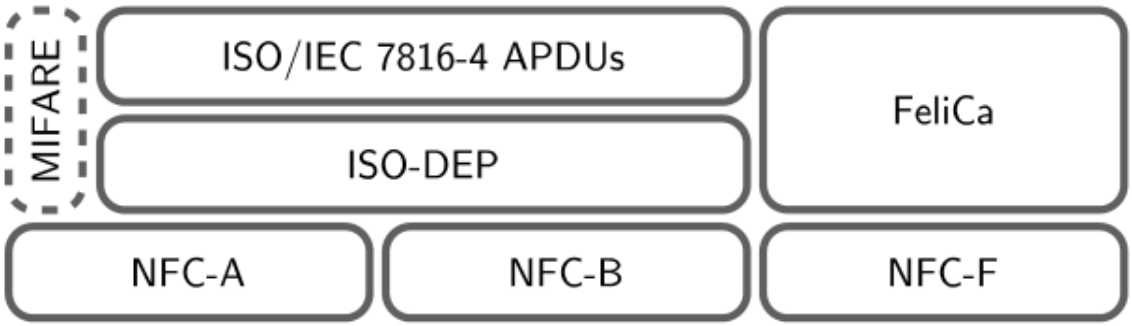
Şekil 2.4: Okuma/Yazma Modunun Protokol Yığını [6]

Bu iletişim teknolojilerinin yanı sıra, dört standart etiket platformu (Tip 1–4) ve birkaç tescilli etiket platformu (örn. MIFARE Classic, Tip V), bu etiketlerde NDEF mesajlarını depolamak için bellek yapılarını tanımlar. Etiket platformları, NFC uygulamalarının gerçek etiket donanımından ve bellek düzeninden bağımsız olarak NDEF mesajlarını okumasına ve yazmasına izin veren bir soyutlama katmanı sağlamaktadırlar.

NDEF üzerinden soyutlamanın yanı sıra, bazı cihaz platformları, ISO-DEP veya düşük seviye iletişim protokolleri kullanarak komut alışverişi yaparak etiketlerle doğrudan iletişimi de desteklemektedirler.

2.2.1.3 Kart Emülasyon Modu

Kart Emülasyon modunda, NFC donanımlı akıllı telefon, bir akıllı kartmış gibi davranır. **Şekil 2.5**'te NFC'nin kart emülasyon modunun protokol yığını özetlenmiştir. Kart emüle özelliklerine sahip mevcut NFC cihazları, NFC-A, NFC-B ve NFC-F alt düzey iletişim protokollerinden birini veya daha fazlasını desteklemektedirler [6].

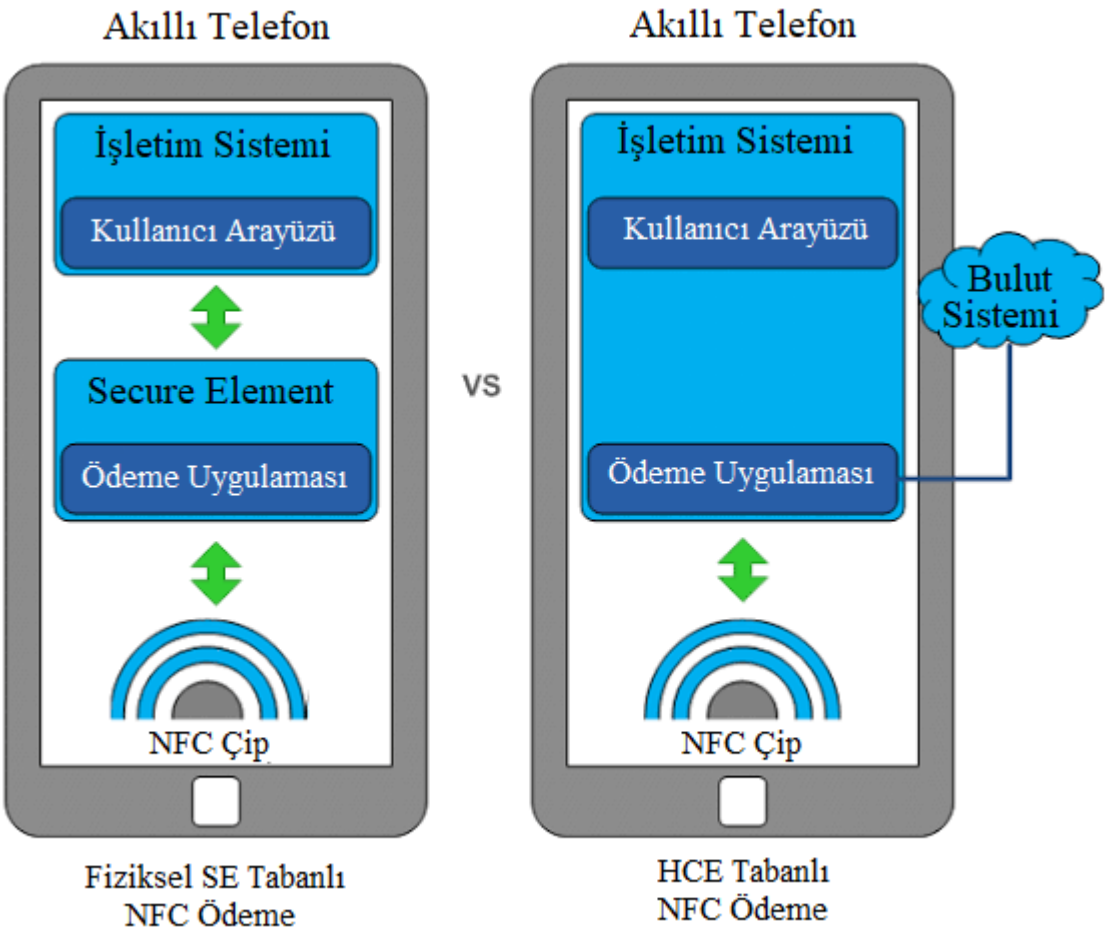


Şekil 2.5: Kart Emülasyon Modunun Protokol Yığını [6]

Sonuç olarak, kart emülasyon modunda, bir NFC cihazı mevcut temassız akıllı kart okuyucularıyla etkileşime girebilir. Kart emülasyonu genellikle, aygıtın NFC denetleyicisine bağlı gerçek emülasyonu gerçekleştiren güvenli bir akıllı kart mikroçipi olan güvenli öğe (SE) ile veya HCE ile ilişkilendirilir. Temassız arayüz üzerinden erişime ek olarak, SE'ye veya HCE'ye NFC cihazının uygulama işlemcisinde çalışan uygulamalar tarafından da erişilebilir.

2.2.2 HCE ve SE Farkları

HCE'de, ödeme uygulaması telefonun işletim sistemi üzerindedir ve doğrudan NFC denetleyicisi ile etkileşime girer. HCE'de bir kart ihraççısının temassız NFC mobil ödemeleri yapmak için SIM kart veya başka bir güvenli elemanı kullanması gerekmez [7]. HCE, uygun bir yazılım tabanlı ödeme çözümü ile NFC ödemelerini uygulamak için başka bir seçenek sunar. Fiziksel SE tabanlı NFC ödeme uygulamaları ile karşılaştırıldığında, HCE'nin olumlu yanları olmakla beraber, kendi sınırlamaları da mevcuttur. Örneğin mobil cihaz kapalı ise, kart emülasyonunun çalışmaması gibi. **Şekil 2.6**'da SE tabanlı ve HCE tabanlı NFC ödeme sistemlerinin farklılıkları gösterilmiştir.



Şekil 2.6: SE Tabanlı ve HCE Tabanlı NFC Ödeme Altyapısı [7]

SE ile HCE arasındaki farklar, **Tablo 2.1**'de incelenmiştir. HCE'nin çıkışı sonrasında, SE kullanımı da gitgide azalmaktadır. Ayrıca çoğu uygulama tabanlı mobil ödeme sağlayıcılarının HCE teknolojisine geçiş yapması ve akıllı telefonların NFC donanımıyla pazara çıkıyor olması da HCE teknolojisinin gelişmesinde büyük rol oynamaktadır.

Tablo 2.1: HCE ve SE'nin Farklılıkları

	Fiziksel SE Çözümü	**HCE Çözümü**
Donanım Sağlama	Ödeme uygulamasının ve kimlik bilgilerinin, telefonda fiziksel bir SE'de (Güvenli Öğe) sağlanması gerekir. Yeni bir SIM karta ihtiyaç duyulabilir.	Ödeme uygulaması, uygulama mağazasından indirilebilir.
Güvenlik	Çok güvenli, çip tabanlı, kurcalamaya dayanıklı donanım.	Ödeme uygulamasını, kötü amaçlı yazılımlara ve virüslere maruz bırakma riskininin uygulama kapsamında yönetilmesi gerekmektedir. Kullanıcıların ödeme bilgilerine erişebilmelerini sağlamak ve güvenli bir uygulama için risk tabanlı kimlik doğrulamaya ihtiyaç vardır. Sınırlı kullanımlı ödeme bilgileri (örn. tokenlar, işlem anahtarları) ve diğer risk yönetimi teknikleri kullanılarak güvenlik sağlanmalıdır.
İşletim Sistemi Desteği	Android, Blackberry, Windows	Android 4.4 KitKat ve üstü, IOS 13, Blackberry OS10
Kullanıcı Tecrübesi	Sorunsuz. Düşük güçle veya kullanıcı etkileşimi olmadan çalışır.	Pil gücü gerekebilir. Token tabanlı işlemler için, uygulama kapsamında kullanılabilir token bulunmalıdır. Bulut tabanlı işlemler için, internet bağlantısı gerekmektedir.
İşlem Desteği	Şuanda sadece fiziksel POS'lar üzerinden sağlanmaktadır.	Fiziksel POS'lar, elektronik ticaret ve mobil ticaret.
İş Modeli	Karmaşık ekosistem ve iş modeli içermektedir. İhraççıların hem SE sahipleri hem de TSM (Güvenilir Hizmet Yöneticisi) tedarikçileri ile anlaşma yapması gerekir.	Hızlı işlem gücü sağlar ve karmaşık ihraççı ve TSM görüşmelerine gerek yoktur. Ancak ihraççılar HCE çözüm sağlayıcılarıyla ortaklık kurmak veya üçüncü taraf token servis sağlayıcılarını kullanmak isteyebilir.

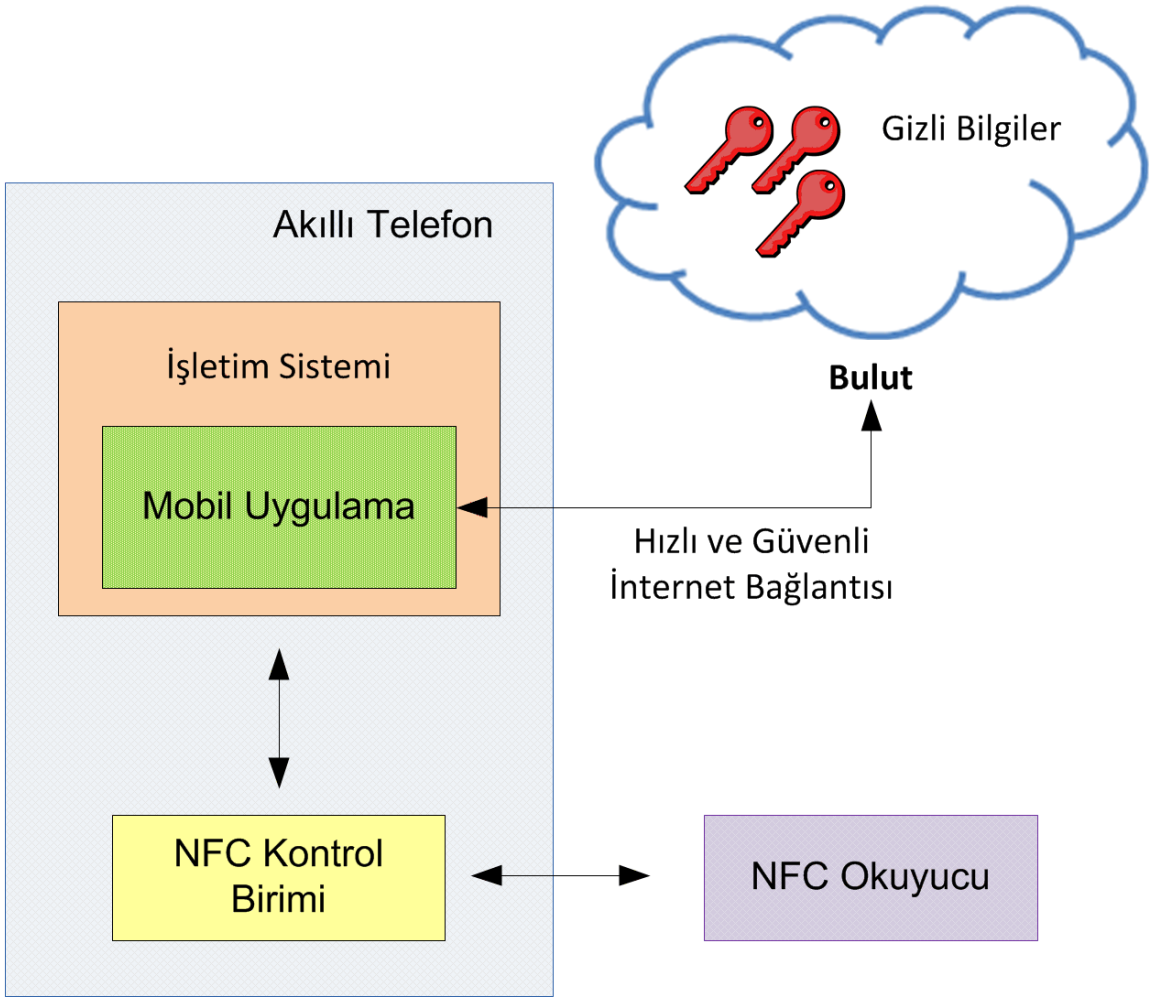
Bu tez kapsamında, HCE teknolojisi üzerinden Kart Emülasyon Modu kullanılmıştır. Çünkü mevcut ulaşım sisteminde, akıllı kart okuyucuları RFID kartları okuyarak ulaşım işlemini gerçekleştirmektedirler. Mevcut sistemde, validatörlere ek bir HCE kernel eklenerek, NFC donanımlı akıllı telefonlar, birer ulaşım kartını emüle ederek sisteme kolayca entegre olabilmektedirler.

2.3 HCE GERÇEKLEME MİMARİLERİ

HCE teknolojisi, mobil cihazlarda donanım tabanlı SE bulunma zorunluluğunu ortadan kaldırırken, SE'nin sağladığı güvenlik altyapısını sağlayamamaktadır. Bu yüzden güvenliğin sağlanması mobil uygulama kısmında olacaktır. Bu bağlamda, uygulama tarafında güvenliği sağlamak adına iki ana HCE gerçekleme mimarisi kullanılmaktadır. Bu modeller uygulama kapsamında kullanılan anahtarların ve kişisel bilgilerin güvenliğini sağlamak için kullanılan mimarilerdir. Güvenliğin ve gizliliğin sağlanması için kullanılan bu iki ana mimarı; Bulut Tabanlı HCE Mimarisi ve Token Tabanlı HCE Mimarisidir [8].

2.3.1 Bulut Tabanlı HCE Mimarisi

Bulut tabanlı mimaride, kart emülasyonu ve işlem tamamen bulutta gerçekleşir. Bu sayede de mobil ödeme uygulamasında kritik bilgilerin tutulmasına gerek kalmaz. Mobil ödeme uygulaması kullanıcı kimliği doğrulaması yapar ve uzak sunucu ile güvenli bağlantıyı oluşturur. Oluşturulan bu bağlantı üzerinden, okuyucudan gönderilen komutlar (CAPDU) uzak sunucuda işlenip, cevap (RAPDU) hazırlanmaktadır. Okuyucuya gönderilmek üzere hazırlan bu RAPDU, güvenli bağlantı üzerinden mobil ödeme uygulamasına gönderilir ve uygulama bu RAPDU'yu NFC üzerinden okuyucuya gönderir. Bu şekilde gerçekleşen bir işlemde bütün gizli bilgiler uzak sunucuda tutulduğu için uygulama kapsamında saklanması gereken gizli bilgi bulunmaz ve çok büyük güvenlik önlemleri almaya gerek kalmadan bir mobil ödeme uygulaması oluşturulabilir. Tüm işlemin uzak sunucuda gerçekleşmesi güvenlik kapsamında faydalı olsa da, bu uzak sunucu ile olan bağlantıyı kesintisiz, hızlı ve güvenle gerçekleştirebilmek için en az hızlı bir 4G veya 5G ağına ihtiyaç duyulmaktadır [9]. **Şekil 2.7**'de bulut tabanlı HCE mimarisinin diyagramı görülmektedir.



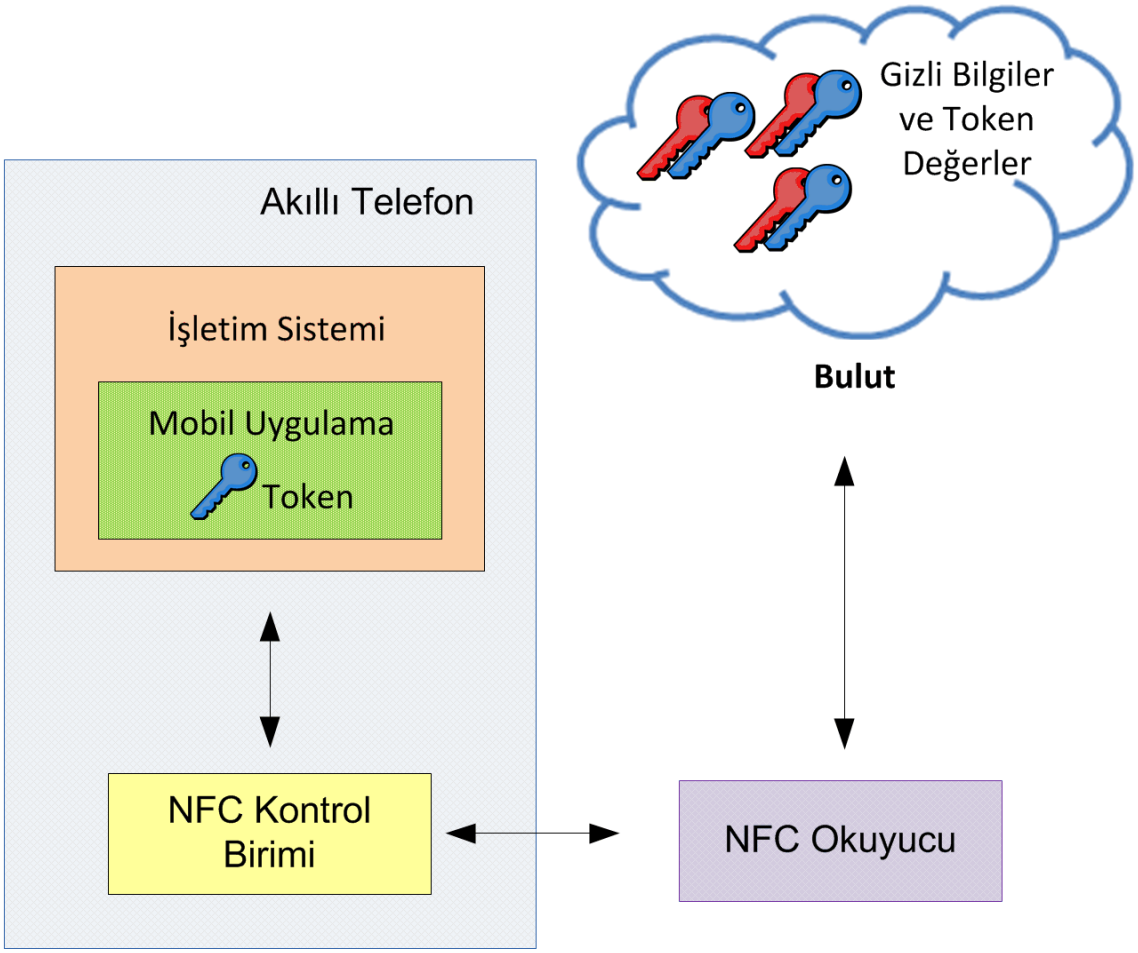
Şekil 2.7: Bulut Tabanlı HCE Mimarisi [8]

2.3.2 Token Tabanlı HCE Mimarisi

Token tabanlı HCE mimarisi, her işlem için uzak sunucuya bağlanmadan, çevrimdışı olarak da işlem yapmaya olanak sağlayan bir mimaridir [10]. Bu yöntem tokenlar sayesinde sağlanmaktadır.

Token, uzak sunucudaki Token Service Provider'da (TSP) oluşturularak güvenli ağ bağlantısı üzerinden karta gönderilir [10]. Oluşturulan bu tokenlar, mobil uygulamanın güvenli veritabanında saklanılarak, işlem yapılması gerektiğinde kullanılır. Her işlem için bir token olacak şekilde saklanan bu verilerin kaç tane saklanacağı uygulama mimarisi kapsamında değerlendirilerek karar verilir. Genellikle mobil ödeme uygulamaları beş adet tokenı güvenli veritabanında muhafaza ederek, çevrimdışı olarak beş işlem yapmayı sağlar. Tokenlar bittikçe

veya minimum sayıya ulaştıkça TSP'den yeni tokenlar üretilir ve mobil ödeme uygulamasına gönderilerek, güvenli veritabanında saklanır. **Şekil 2.8**'de token tabanlı HCE mimarisinin diyagramı görülmektedir.



Şekil 2.8: Token Tabanlı HCE Mimarisi [8]

Bu tez kapsamında, çevrimdışı ödeme sağlayabilmek için token tabanlı HCE mimarisi kullanılmıştır.

3. MALZEME VE YÖNTEM

3.1 GÜVENLİK KATMANLARI

Güvenli bir HCE çözümü, birçok güvenlik katmanlarından oluşmaktadır. Katmanlı güvenlik mimarisi sayesinde, saldırganlar tarafından hassas verilere erişim, bu verileri değiştirilmesi veya kopyalanması çok karmaşık ve zor hale geldiği için, güvenlik büyük ölçekte sağlanabilmektedir.

HCE çözümü için dört ana güvenlik katmanı mevcuttur [11]. Bu katmanlardan, mobil cihazdaki hassas verilerin güvenliğini sağlayarak, çalınmayı önlemektedirler.

Mobil ödeme uygulamasında güvenliği sağlamak amacıyla kullanılması gereken güvenlik katmanları alt başlıklardaki gibidir.

3.1.1 Uygulama Güvenliği

Mobil ödeme uygulamasının güvenliğinin ihlal edilmemesini sağlar. Kaynak kod karıştırması (obfuscation) yapılarak, kaynak kodun çözülmesini ve okunabilir hale gelmesini engeller. Token gibi hassas verilerin mobil ödeme uygulamasında saklamak için WhiteBox kriptogramından faydalanılır. Kullanılan mobil işletim sisteminin Trusted Execution Environment (TEE) kapsamında olduğuna dikkat edilir.

3.1.2 Cihaz Güvenliği

Cihaz güvenliği kapsamında amaç, mobil ödeme uygulamasını çalıştıran akıllı telefonun güvenli bir işletim sisteminde ve modda çalıştığını doğrulamaktır. Akıllı telefon root edilmiş mi, yani işletim sistemi üçüncü parti uygulamalar ile kırılmış mı kontrolü yapılır. Akıllı telefon hata ayıklama modunda mı çalışıyor tespiti yapılır. Son olaraksa, kullanılan işletim sistemi bir akıllı telefonda mı çalışıyor yoksa farklı bir cihaz üzerinde işletim sistemi emülasyonu üzerinden mi çalıştırılıyor kontrol edilir. Bu kontroller sonucunda güvenli bir cihaz tespiti yapılabilinmektedir.

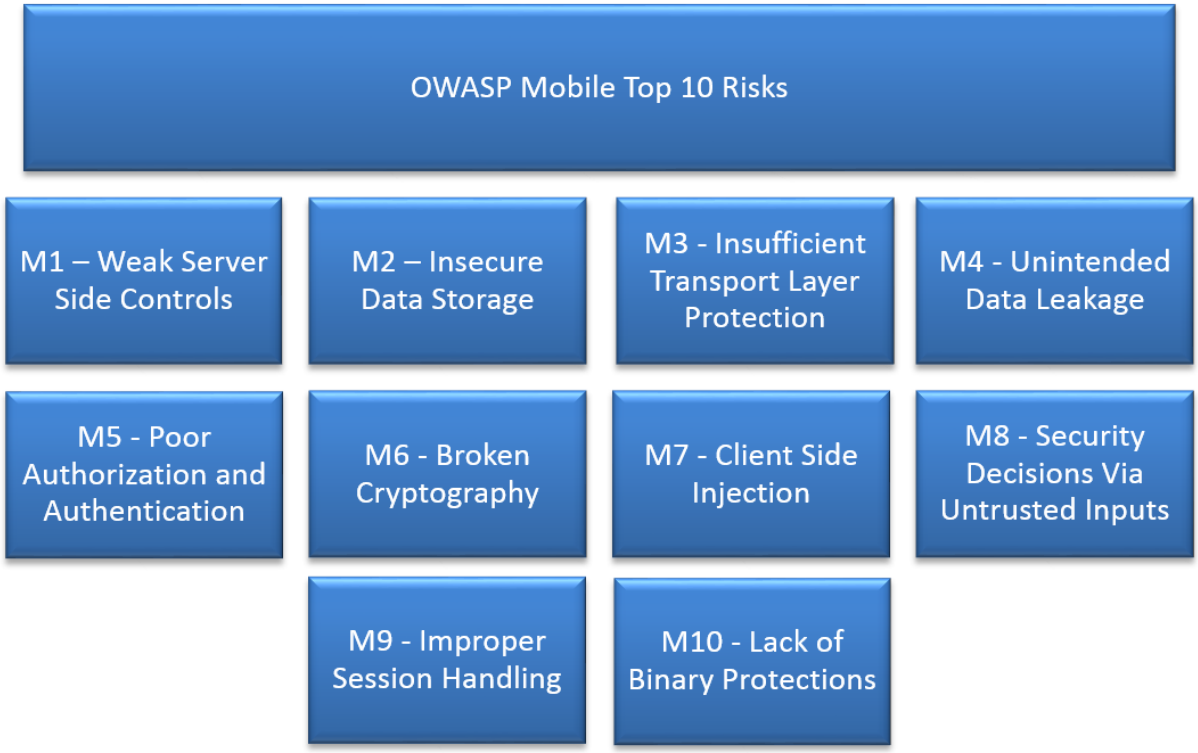
3.1.3 İletişim Güvenliği

Mobil ödeme uygulaması ile uzak sunucudaki ödeme servisleri arasındaki iletişimin güveliğinin ve gizliliğinin sağlanması gerekmektedir. Eğer bu iletişim esnasında araya giren saldırgan olursa, hassas verileri çalabilir veyahut bu verilerde değişiklik yapabilir.

3.1.4 Dinamik Veri

Dinamik veri, olası bir hırsızlıkta veya kişisel bilgi çalınması durumda oluşacak risklerden korunmayı sağlamaktadır. HCE çözümlerinde, her işlemde veyahut belirli işlem sayılarında değişen ödeme verileri kullanılmaktadır. Bu sebepten ötürü, dinamik olan veriler, kullanım süreleri dolmadan önce kullanım sayısı, zaman aralığı veya her ikisi ile sınırlı olabilirler.

Bu güvenlik katmanlarının yanı sıra, **Şekil 3.1**'de görülmekte olan maddeler, *Open Web Application Security Project'te* (*OWASP*) yayınlanan Top 10 Mobile Risks (M1 to M10) çalışması, güvenli bir mobil uygulaması geliştirilmesinde kabul edilmiş güvenlik yönergeleri sağlamaktadır [12].



Şekil 3.1: Mobilde Önemli 10 Risk [12]

Bu mobil ödeme uygulaması kapsamında, HCE güvenlik katmanları ile OWASP'ın güvenlik çalışmaları derlenerek, uygulamanın güvenliği kapsamlı kurallar seti dahilinde sağlanmıştır. Bu kapsamda uygulama tarafında yapılan güvenlik çalışmaları **Tablo 3.1**'de belirtilmiştir.

Tablo 3.1: Mobil Uygulama Güvenlik Analizi

Güvenlik	Güvenlik Kuralları
Asgari	• Android 4.4 veya üstü • NFC kullanım izni • Üçüncü parti olmayan kütüphaneler
Cihaz Güvenliği	• Root işlemi yapılmamış cihaz (M8) • Emülatör kullanımı algılama
Uygulama Güvenliği	• Kaynak kod karıştırması (obfuscate) • Hata ayıklama bayrağı olmaması (M10)
İletişim Güvenliği ve Dinamik Veri	• HTTPS zorunluluğu (M3) • Güçlü kriptografik algoritma kullanımı (M5) • Anahtar yönetim süreci (M1), (M9)
Veri Belleği	• Düz metin uygulama kaydı tutulmaması (M2) • Hassas verilerin şifreli veritabanında tutulması (M2) • WhiteBox Kriptografi kütüphane kullanımı (M4)
Bellek	• Hassas verilerin kullanım sonrası bellekten silinmesi (M4)

3.2 KRİPTOGRAM ALGORİTMALARI

Mobil ödeme uygulamasında güvenli veri saklama ve iletişimi kapsamında kullanılan kriptogram algoritmaları DES, MAC, AES ve RSA'dir. Ayrıca uygulamaya özel hazırlanmış bir WhiteBox şifreleme kütüphanesi de bulunmaktadır. İlerleyen başlıkta detaylı inceleme yapılmıştır.

3.2.1 DES Şifreleme

Data Encryption Standart (DES), verileri şifrelemek veya hali hazırda şifrelenmiş olan verileri çözmek için geliştirilen bir standarttır. Aslında bu şifreleme algoritmasına, Data Encryption Algorithm (DEA) adı verilse de, algoritmanın standartlaşmış olan ismi, yani DES olarak kullanılmaktadır.

International Business Machines (IBM) tarafından geliştirilen Lucifer şifrelemesinin, yine IBM tarafından 1970 yıllarında daha fazla geliştirilerek ve U.S. National Bureau of Standart tarafından da bir standart olarak kabul edilmiş bir şifreleme algoritmasıdır [13].

DES algoritması, simetrik olarak blok şifreleme yapmaktadır. Yani şifrelenecek olan verileri belli bloklar haline getirerek, aynı anahtar ile blok blok şifrelenmesi ve çözmesidir. DES şifreleme üç farklı blok uzunluğu kullanmaktadır. Bunlar 64 bit, 128 bit ve 192 bitlik bloklardır [14].

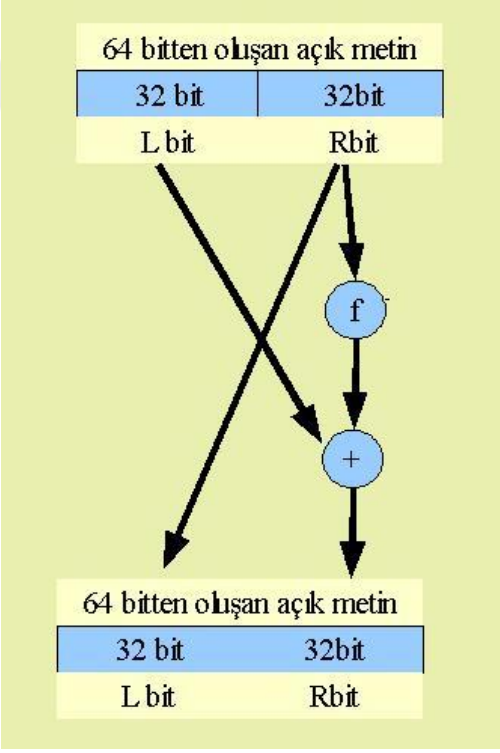
64 bit uzunluğunda blok kullanan DES algoritması, 8 baytlık blok kullanır ve kullandığı anahtar uzunluğu da 8 bayttır.128 bit uzunluğunda blok kullanan algoritma ise 2DES algoritmasıdır, Double DES olarakta bilinir. Bu algoritmada ise 16 baytlık bloklar ve anahtar kullanılmaktadır.192 bit uzunluğundaki blokları kullanan diğer algoritmaya ise 3DES yani Triple DES algoritması olarak adlandırılmıştır. Bu şifreleme algoritmasında 24 baytlık bloklar ve anahtar kullanılmaktadır.

Günümüzde bilgisayarların işlem gücü bakımından çok gelişmesinden ötürü DES algoritması güvenirliğini kaybetmiştir. Bundan dolayı anahtarlarının daha uzun olmasından mütevellit

verinin karmaşıklığının daha çok olacağından ötürü günümüzde 2DES ve 3DES algoritmaları kullanılmaktadır. Fakat bu tez kapsamında, anlatımın daha açık olacağından ötürü DES algoritması incelenmiştir.

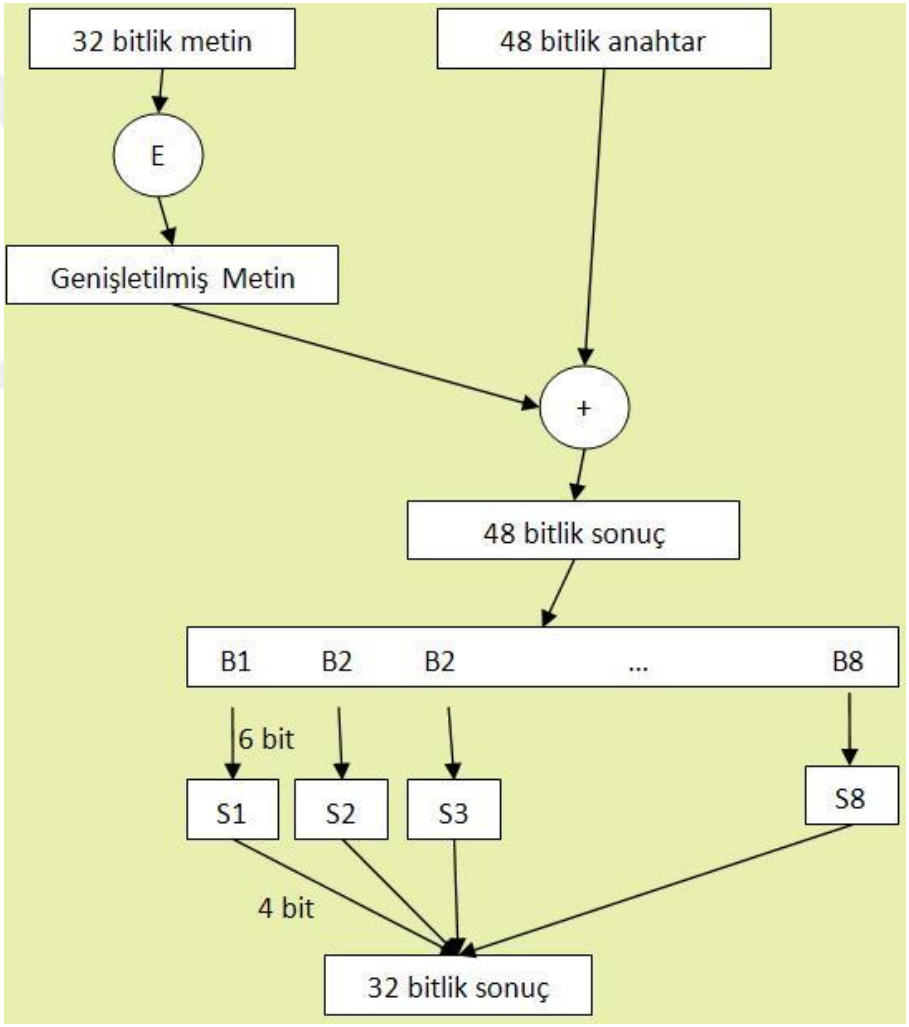
DES algoritması 64 bit uzunluğunda anahtar kullanıyor olsa da, aslında her baytın en anlamsız bitini kontrol (parity) biti olarak kullandığı için, 8 bayttaki her baytın ilk bitlerini saymayarak, 56 bitlik bir anahtara sahip olduğu görülmektedir.

Şekil 3.2'de DES algoritmasının çalışma mantığı verilmiştir. 64 bitten oluşan açık metin, 32 bit uzunluğunda olmak üzere iki parçaya ayrılır, çünkü soldaki L bitleri ve sağdaki R bitleri ayrı ayrı işlenecektirler. İlk olarak R bitleri f fonksiyonundan geçerek veriyi karmaşıklaştırır. Bunun sonucunda oluşan 32 bitlik veri ile L bitleri XOR işleminde geçirilerek elde edilen yeni veri ve bir önceki açık metindeki R bitleri, R bitleri solda olacak şekilde hazırlanır. Bu işlemler 16 defa tekrarlanır ve her döngüde, yeni oluşturulan 64 bitlik açık metin işlenir.



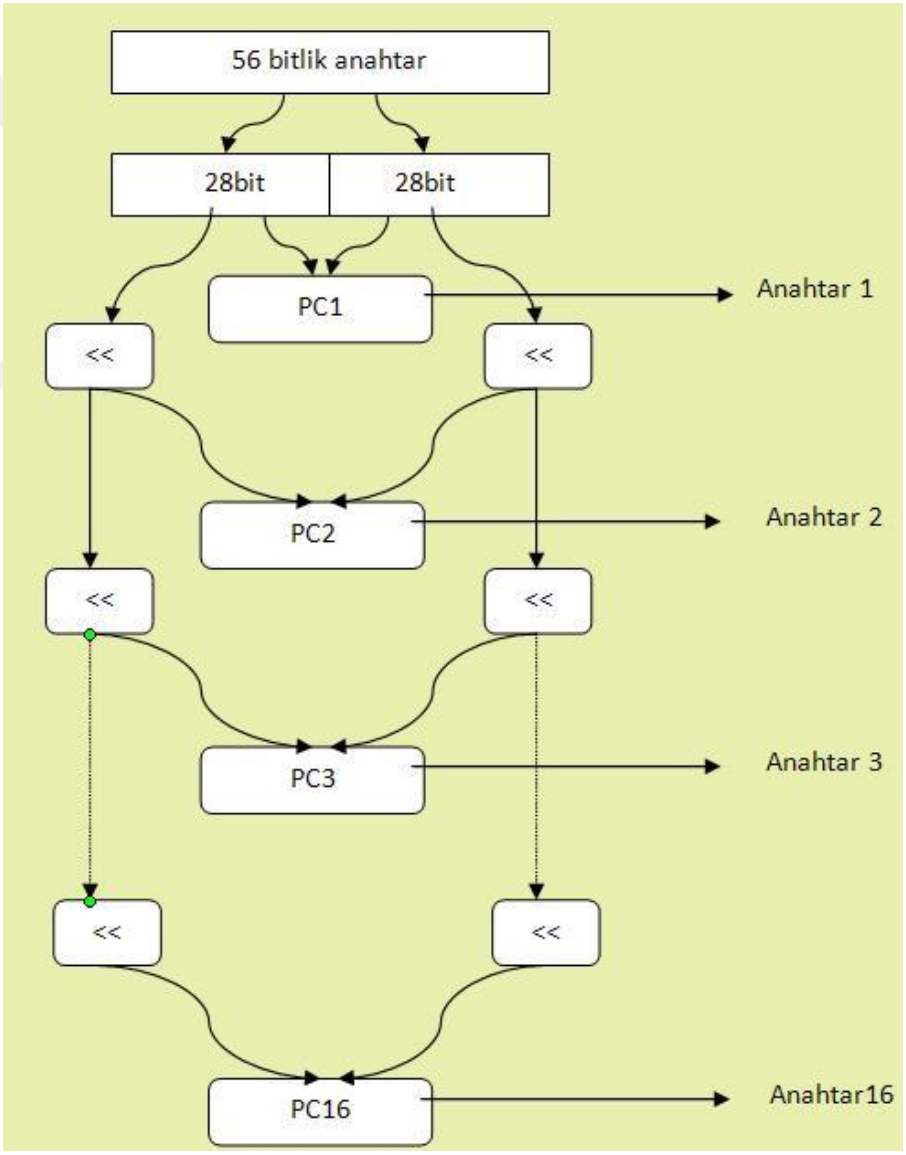
Şekil 3.2: DES Algoritması Çalışma Mantığı [15]

Bir önceki kısımda bahsi geçen f fonksiyonuna değinecek olursak (**Şekil 3.3**'te bu fonksiyon gösterilmiştir) , **Şekil 3.2**'de f fonksiyonuna giren 32 bitlik veri, genişleme permütasyonu ile genişletilerek 48 bitlik veriye dönüştürülür. Bu süreçte, 32 bitin 16 biti tekrardan yazılarak, 48 bite dönüştürülmektedir. Yani girdi verileri haricinde başka bir veri kullanılmamaktadır. Genişleme sonucu oluşan 48 bitlik veri, 48 bitlik anahtar ile XOR işleminden geçirilir. Bu işlem sonucunda oluşan 48 bitlik veri ise 6 bitlik 8 bloğa bölünür. Bu 8 blok teker teker ikame kutularına (Subsitution Box) girer ve genişleme permütasyonunun tersi mantığında çalışarak 6 bitlik veriyi 4 bitlik veriye dönüştürür. Bu şekilde işlenen 8 blok sonucunda toplamda 32 bitlik veri elde edilir.



Şekil 3.3: DES Algoritmasının Karmaşıklaştırma Fonksiyonu [15]

Şekil 3.3'te görülen, f fonksiyonundaki 48 bitlik anahtar, 16 döngünün her bir tekrarında, farklı bir anahtar içermektedir. Bu anahtarın oluşturulma algoritması **Şekil 3.4**'te gösterilmiştir. Bu konuya değinecek olursak, kontrol bitleri haricinde 56 bitlik anahtarımızın mevcut olduğu belirtilmişti. Bu anahtar, 28 bitlik eş parçalara bölünür. Bu soldaki parçanın sağdan ilk 24 biti ile sağdaki parçanın soldan ilk 24 biti alınarak ilk döngünün anahtarı oluşturulur. Sonrasında bu 28 bitlik parçalar, döngüsel olarak birer kez sola kaydırılır, yani sola kayan bit verinin en sağına gelir. Bu işlem ardından yine bir önceki seferde anlatıldığı gibi 48 baytlık veri alınarak ikinci döngünün anahtarı olarak kullanılır. Bu işlem toplamda 16 defa tekrarlanarak, her döngüye özel anahtar türetilir.



Şekil 3.4: DES Algoritmasının Anahtar Üretme Fonksiyonu [15]

DES algoritmasının bir döngüsünde gerçekleşen işlemler **Şekil 3.5**'teki gibi özetlenebilmektedir.



Şekil 3.5: DES Algoritmasında Her Döngüde Gerçekleşen Adımlar

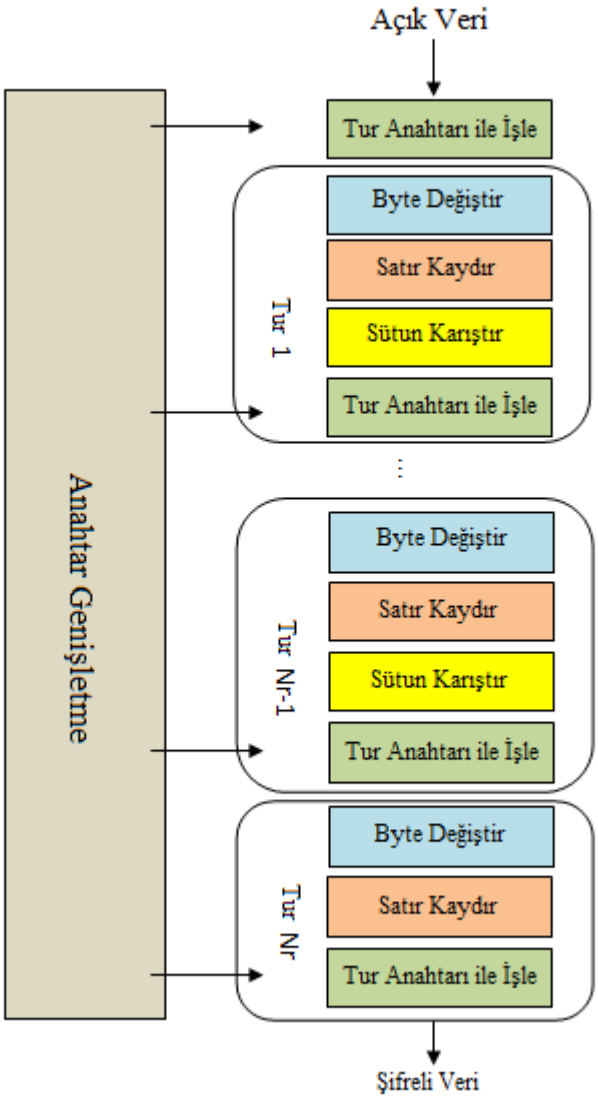
3.2.2 AES Şifreleme

Rijndael şifreleme algoritması, 1997 yılında National Institute of Standards and Technology'nin (NIST) yeni bir şifreleme standartı çağrısıyla, Belçika'lı Vincent Rijmen ve Joan Daemen tarafından geliştirilmiştir. 2000 yılında NIST tarafından kabul gören bu algoritma, Advanced Encryption Standart (AES) olarak bir standart haline gelmiştir. DES algoritması, teknolojinin gücüne yenik düşüp, güvenirliğini kaybettiğinden ötürü NIST tarafından böyle bir çağrı gerçekleşmiştir. Günümüzde halen AES algoritması birçok alanda, güvenilir bir şekilde kullanılmaktadır.

AES algoritması, 128 bitlik bloklar üzerinden simetrik yapıda şifreleme yapan bir kriptogram algoritmasıdır [14]. Bu bloklar 4x4 durum matrislerinden oluşmaktadır ve bu matrislerin her bölmesine 8 bit (1 bayt) düşmektedir.

AES algoritması, üç farklı anahtar uzunluğu desteklemektedir. Bu uzunluklar 128 bit (16 bayt), 192 bit (24 bayt) ve 256 bittir (32 bayt). Anahtar uzunlukları farklı olsa da, şifreleme yapılan bloklar hep 128 bittir. Fakat anahtar uzunluğuna bağlı olarak değişen ise AES'in şifreleme

yaptığı döngü sayısıdır. 128 bitlik anahtar uzunluğunda 10 defa, 192 bitlik anahtar uzunluğunda 12 defa, 256 bitlik anahtar uzunluğunda ise 14 defa yapılmaktadır. Bu tez kapsamında, 128 bit anahtar uzunluğuna göre anlatım yapılmıştır [16].



Şekil 3.6: AES Şifreleme Algoritmasının Genel Görünümü

Şekil 3.6'da AES algoritmasının genel yapısı görülmektedir. Son tur haricinde, her turda 4 farklı işlem yapılmaktadır. Bu yapılan işlemler sırasıyla şöyledir; bayt değiştirmek, satır kaydırmak, sütun karıştırmak ve tur anahtarı ile işlemektir. Bu turlar sonucunda, açık olarak giren veri, şifrelenmiş bir biçimde çıkmaktadır. Sadece ilk turda, anahtar orijinal haliyle girer, ileriki turlarda ise bir önceki anahtardan türetilerek yeni bir anahtar kullanılır.

Şifrelenmek istenen açık metin, 128 bitlik parçalara bölünerek durum matrislerine yerleştirilir ve aynı şekilde 128 bitlik anahtarda bir durum matrisine yerleştirilir. **Şekil 3.7**'de görüldüğü gibi, döngünün başında, veri matrisi ile anahtar matrisi XOR işleminden geçerek oluşan sonuç matrisi döngüye girer. Bu şekilde anahtar ile işleme adımına da değinmiş olduk.

47	40	A3	4C
37	D4	70	9F
94	E4	3A	42
ED	A5	A6	BC

$\oplus$

AC	19	28	57
77	FA	D1	5C
66	DC	29	00
F3	21	41	6A

$=$

EB	59	8B	1B
40	2E	A1	C3
F2	38	13	42
1E	84	E7	D6

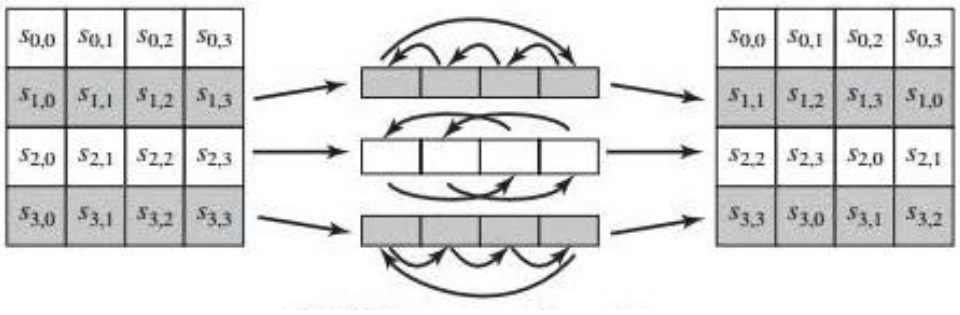
Şekil 3.7: AES Anahtar ile İşleme Adımı

Döngünün işlemi olan bayt değiştirme adımı, algoritmanın doğrusal olmayan tek işlemidir. Durum matrisindeki her bir veri, **Şekil 3.8**'de bir örneği gösterilen S-Kutusundaki (S-Box) veriler ile eşleşerek, yeni bir durum matrisi oluşturmaktadır. Bu değiştirme işlemi şöyle gerçekleşir, durum matrisinden değiştirilecek olan eleman, mesela "D6" örneğinden anlatılacak olursa, S-Kutusundaki karşılığı bulunur. Burada ise değiştireceğimiz bayt, nibble olarak incelenir. İlk nibble olan "D" bakılması gereken satırı, ikinci nibble olan "6" ise bakılması gereken sütunu belirtir. Bu durumda D. satır, 6. sütuna bakıldığında "F6" baytını görülür. Bu da demek oluyor ki, durum matrisimizde bulunan "D6" baytı, S-Kutusundan geçerek "F6" olarak değiştirilir. Durum matrisindeki her bir eleman aynı işlemden geçirilerek yeni bir durum matrisi bu şekilde elde edilmektedir

	0	1	2	3	4	5	6	7	8	9	0a	0b	0c	0d	0e	0f
0	63	7c	77	7b	f2	6b	6f	c5	30	1	67	2b	fe	d7	ab	76
10	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
20	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
30	4	c7	23	c3	18	96	5	9a	7	12	80	e2	eb	27	b2	75
40	9	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
50	53	d1	0	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
60	d0	ef	aa	fb	43	4d	33	85	45	f9	2	7f	50	3c	9f	a8
70	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
80	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
90	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a0	e0	32	3a	0a	49	6	24	5c	c2	d3	ac	62	91	95	e4	79
b0	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	8
c0	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d0	70	3e	b5	66	48	3	f6	0e	61	35	57	b9	86	c1	1d	9e
e0	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f0	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Şekil 3.8: AES S-Kutusu Örneği

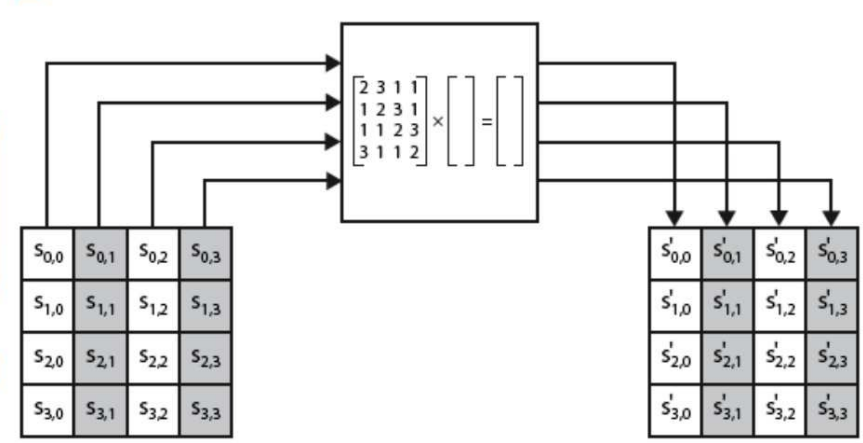
Satır kaydırma işleminde, bayt değiştirme ile oluşan yeni durum matrisi, **Şekil 3.9**'da görüldüğü üzere, belli aralıklarla döngüsel olarak kaydırılmaktadırlar. Durumun matrisinin ilk satırında kaydırma işlemi yapılmaz. İkinci satırda, bütün elemanlar sola doğru birer kez ötelenir. Üçüncü satırda, bütün elemanlar sola doğru ikişer kez ötelenirler, üçüncü satırda ise tüm elemanlar sağa doğru birer kez ötelenerek, satır kaydırma işlemi tamamlanır. Döngüsel kaydırma yapıldığından ötürü, sütundan taşan elemanlar kaydırmanın başına eklenmektedirler. Satır kaydırma sonucunda, yeni bir durum matrisi oluşmaktadır.



Şekil 3.9: AES Satır Kaydırma İşlemi

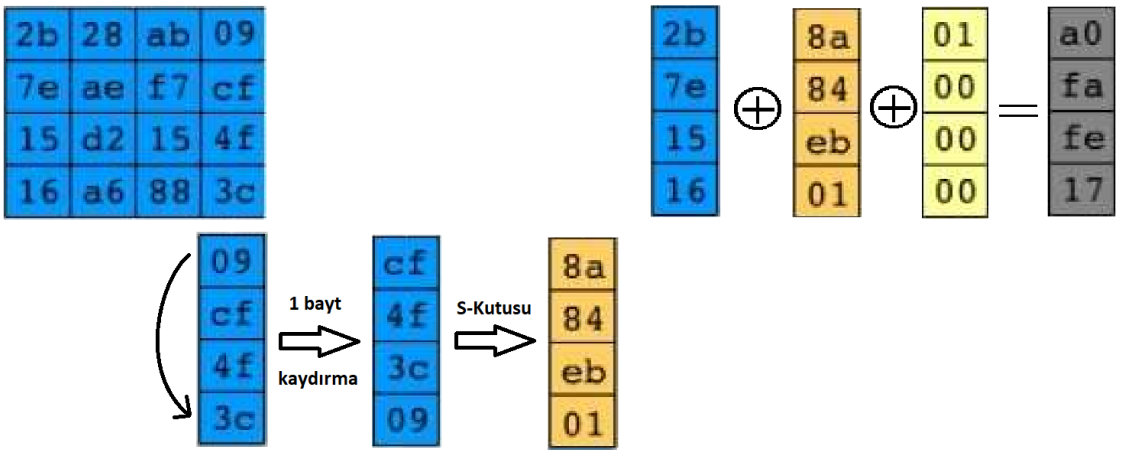
Bir önceki adımda oluşan durum matrisi, sütun karıştırma işleminden geçer. Bu işlem, satır kayma işleminde de olduğu gibi matrisi karıştırarak difüzyon sağlamaktadır. Bu aşamada **Şekil 3.10**'da de görüldüğü üzere, durum matrisindeki her bir sütun, Rijndael tarafından belirlenen

4x4'lük bir matris ile çarpılarak, tüm değerleri değişen sütunlar işlem sonucunda yeni bir durum matrisi olarak kullanılabilir duruma gelmektedir.



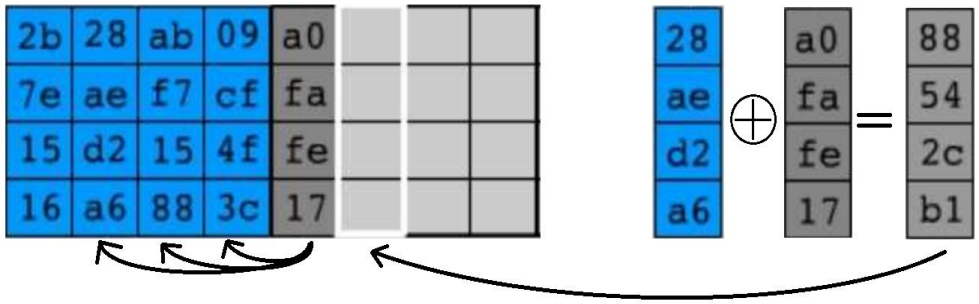
Şekil 3.10: AES Sütun Karıştırma İşlemi

Bir döngüdeki gerçekleşen tüm aşamalara değinildi. Her döngüye giren anahtarın farklı olduğundan bahsedilmişti. Anahtar türetiminin ilk aşaması **Şekil 3.11**'de görülmektedir. İlk olarak 128 bitlik şifremiz bir durum matrisine dönüştürülür. Matrisin son sütunu yukarıya doğru bir bayt kaydırılır ve oluşan yeni sütun, S-Kutusundan geçerek, sütundaki bütün değerler S-Kutusundaki değerler ile önceki adımlarda anlatıldığı gibi değiştirilir. Sütunun ilk hali, S-kutusundan çıkan hali ve Rcon sütunları birbirleri arasında XOR işleminden geçerek, bir döngünde kullanılacak olan anahtarın ilk sütununu oluştururlar. Her döngü anahtarının ilk sütunu için bu işlemler bir önceki anahtar üzerinden yapılmaktadır. Rcon sütunları ise **Şekil 3.13**'te belirtilmiş olup, Rijndael tarafından belirlenmişlerdir.



Şekil 3.11: AES Anahtar Türetme İşlemi

İlk sütunu oluşturulan yeni döngü anahtarının, kalan sütunları ise, yeni döngü anahtarının ilk sütunu ile bir önceki döngü anahtarının kalan sütunları ile XOR işleminden geçirilerek oluşturulur. Bu adımlar **Şekil 3.12'**de görsek olarak açıklanmıştır.

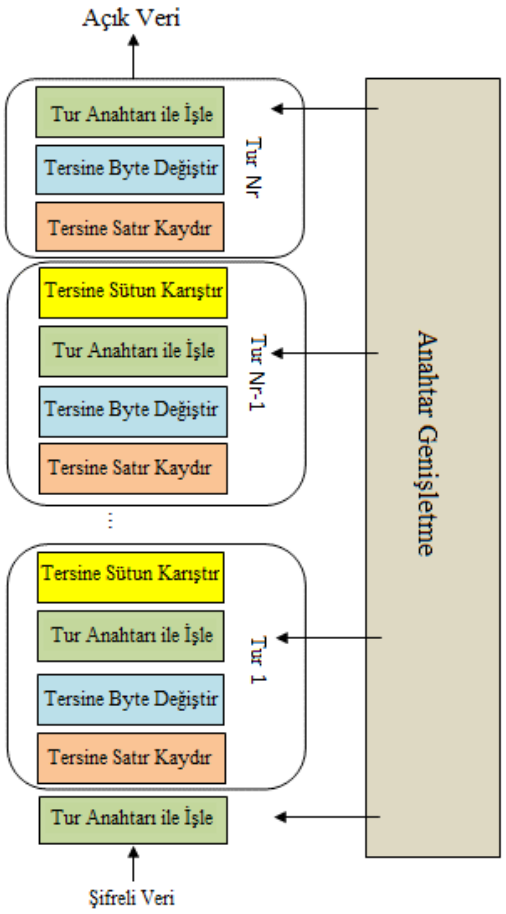


Şekil 3.12: AES Anahtar Türetme İşlemi Devamı

01	02	04	08	10	20	40	80	1b	36
00	00	00	00	00	00	00	00	00	00
00	00	00	00	00	00	00	00	00	00
00	00	00	00	00	00	00	00	00	00

Şekil 3.13: AES Key Türetimi Rcon Sütunları

Rijndael algoritması ile şifrelenen veri ise kolay bir şekilde geri çözülebilmektedir. Şifreli veri tersine işlemlerden geçerek, açık metine ulaşılabilir. Bu süreçte yapılan işlemler ise, tersine satır kaydırmak, tersine S-Kutusu, tersine sütun karıştırma ve tur anahtarı ile işlemektir. Tabi bu süreçte, tersine anahtar türetimi de yapılmaktadır. Şifre çözme aşamaları **Şekil 3.14**'te gösterilmiştir.



Şekil 3.14: AES Şifre Çözme Algoritmasının Genel Görünümü

3.2.3 RSA Şifreleme

RSA algoritması, Ron Rivest, Adi Shamir ve Len Adleman tarafından 1977 senesinde bulunmuştur. Algoritmanın adı da bulucuların soy isimlerinin ilk harflerinden isimlendirilmiştir. Bu algoritma, asimetrik şifreleme yapmaktadır. Asimetrik olması, bir açık anahtar (public key) ve bir de özel anahtar (private key) kullanılmasından ötürü, yani anahtar çiftinden oluşmasından kaynaklıdır. Açık anahtar ile şifrelenen mesaj, sadece özel anahtar ile çözülebilir. Bu bağlamda, açık anahtar halka açık bir şekilde paylaşılır ve bu anahtarla şifrelenen veriler, sadece özel anahtara sahip olan kişi tarafından çözülebilir.

RSA algoritması, DES ve AES gibi veri karmaşıklaştırılmasından ziyade, matematiksel formüllerden faydalanır. Bunun temelinde ise iki adet asal sayı yatmaktadır. Büyük asal sayılarda çarpma yapmak kolay olsada, sonucunu çarpanlarına ayırmak çok zordur. Bu kriptogram algoritması, başarısını büyük asal sayılarda, çarpanlarına ayırmanın zorluğundan kazanmaktadır.

Açık anahtarlı şifreleme algoritması olan RSA, en çok tercih edilen algoritmalardandır[17]. Fakat diğer şifreleme algoritmalarına göre daha yavaş olduğundan ötürü, genellikle küçük verilerin şifrelenmesinde ve dijital sertifikalarda kullanılmaktadır. Küçük veriler kapsamında ise, daha çok simetrik şifreleme algoritma anahtarlarının paylaşımı için tercih edilmektedir. Genellikle 1024 bit ve 4096 bit arası uzunluklardaki anahtar çiftleri kullanılmaktadır.

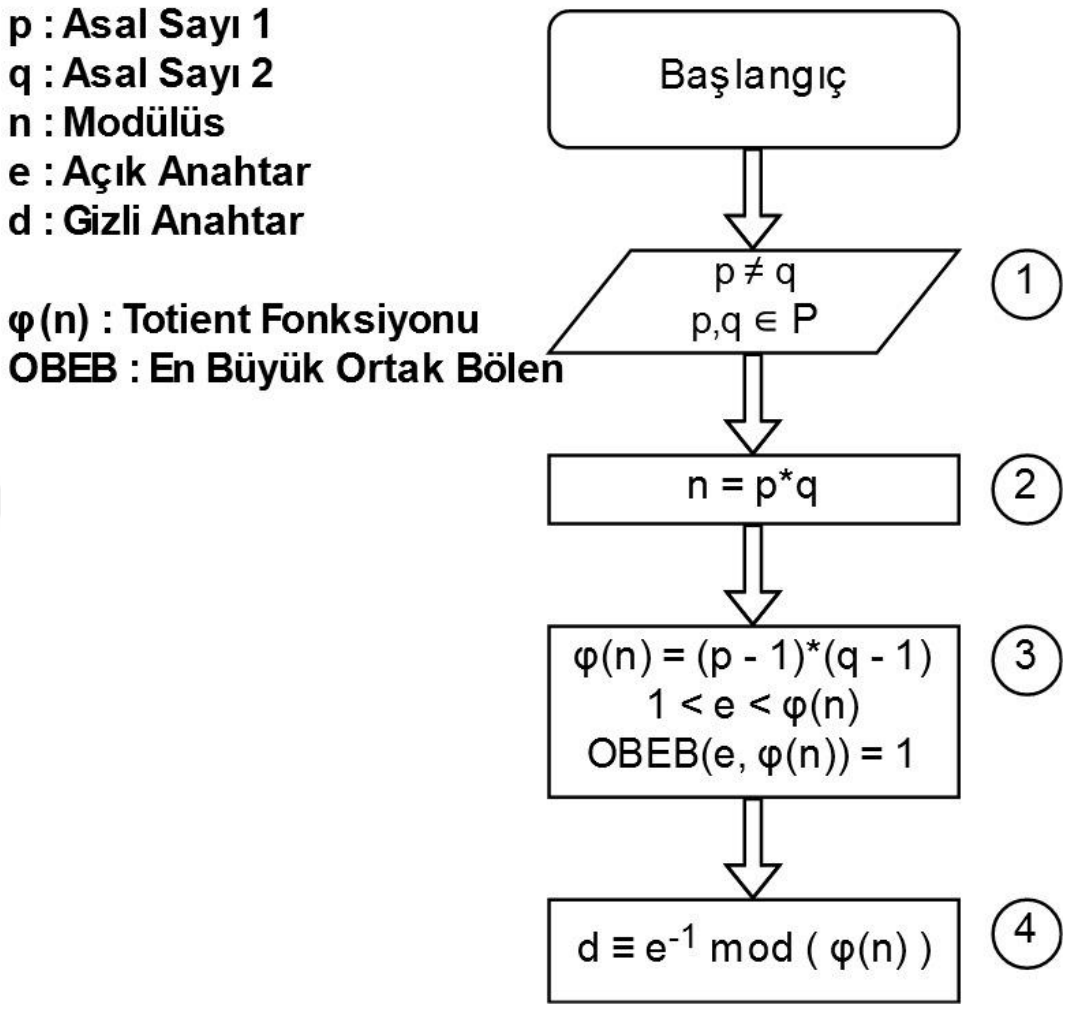
Şekil 3.15'de, RSA algoritmasının anahtar çifti türetim adımları gösterilmiştir. Bu şekildeki diyagramlara göre adımlar aşağıdaki açıklandığı gibidir [18].

1. p ve q olmak üzere, birbirlerinden farklı iki büyük asal sayı seçilir.

2. p ve q asal sayılarının çarpımından elde edilen modulüs değeri, özel ve açık anahtar üretiminde kullanılmak üzere elde edilir ve n olarak adlandırılır.

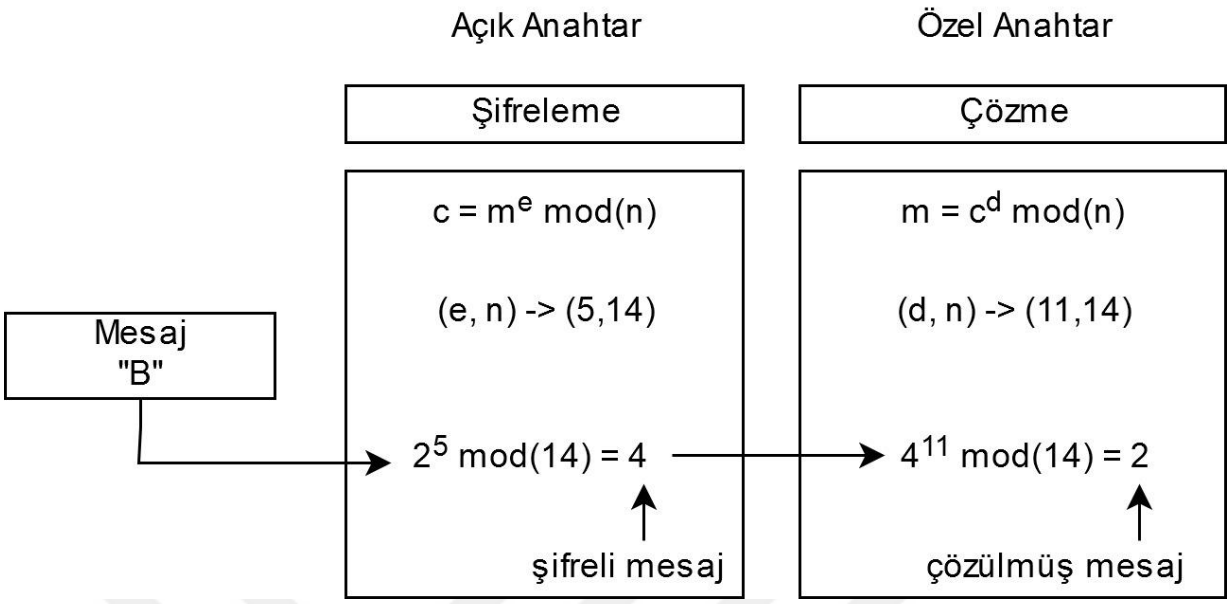
3. Modülüsün totient değeri hesaplanır. Bu hesaplanan sayı ile aralarında asal, 1'den büyük ve totient değerinden küçük olan bir e sayısı seçilir. Seçilen bu e değeri açık anahtar olarak kullanılmak üzere saklanır.

4. Bu adımdaki denkliği sağlayan d değeri ise özel anahtar olarak kullanılmak için saklanır.



Şekil 3.15: RSA Algoritması Anahtar Türetimi Adımları

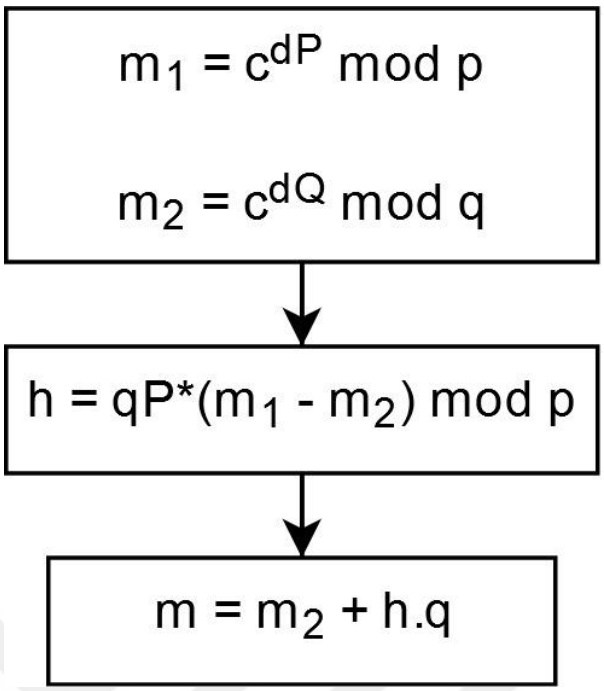
RSA şifreleme algoritması kullanılarak gerçekleştirilen bir şifreli haberleşmede yalnızca özel anahtar sahibini şifreli veriyi çözebilsin amacıyla gönderilmesi gereken veri, herkese açık olan (n, e) değerleri ile yani açık anahtar kullanılarak şifrelenir [19]. **Şekil 3.16**'da şifreleme bölümünde görünen m^e (mod(n)) formülü ile şifrelenecek verinin e değerindeki kuvvetinin mod n karşılığı şifreli veri olarak hesaplanmaktadır. Açık anahtar ile şifrelenen veri, sadece o anahtarın çifti olan özel anahtar ile çözülebilmektedir, bundan ötürü sadece (n, d) özel anahtarı ile şifreli veri çözülebilinir. **Şekil 3.16**'da çözme bölümünde görünen c^d (mod(n)) formülü ile çözülecek verinin d değerindeki kuvvetinin mod n karşılığı açık veriyi hesaplamaktadır.



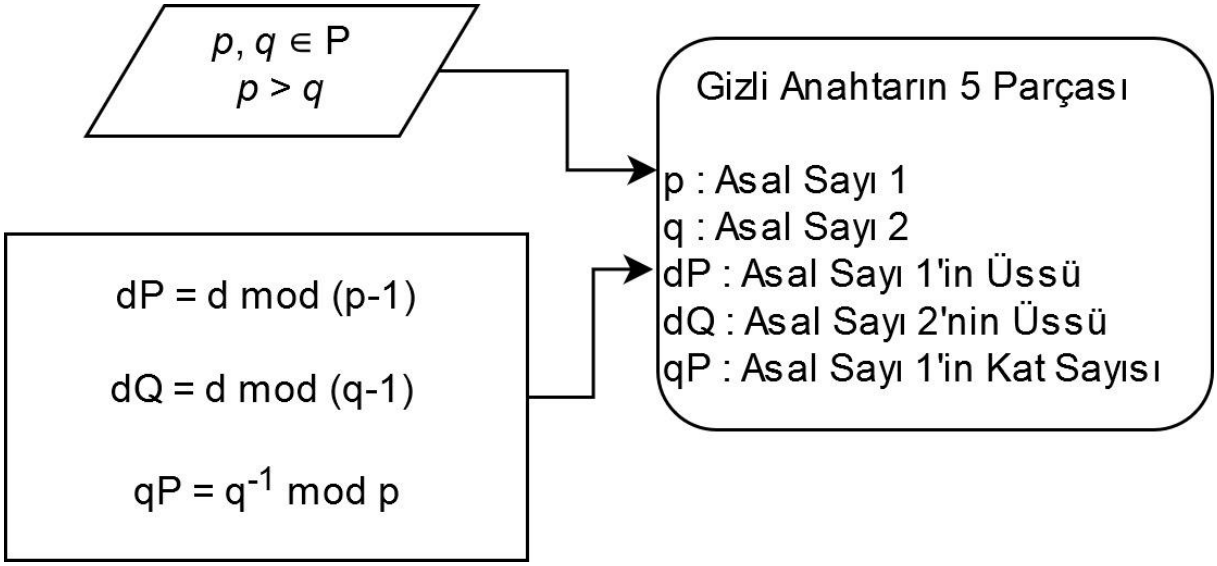
Şekil 3.16: RSA Algoritmasında Şifreleme ve Şifre Çözme

RSA'de işlem hızını yükseltmek için, açık anahtarın üssü olan e değeri, daha küçük seçilerek yapılacak çarpımların sayısı azaltılır [17]. Bu e sayısı seçerken ise, küçük Hamming ağırlığına sahip olan bir değer seçilmelidir. Örneğin e = 3 olarak seçilirse, bit bazında 0011 olarak belirtilir ve yapacağı işlem sayısı 3'tür. e = 217 seçilirse, bit bazında 1 0000 0000 0000 0001 olarak belirtilir ve yapacağı işlem sayısı 17'dir [20].

Açık anahtar üssünün küçük değerlerde kullanılmasında sorun olmasa da gizli anahtar üssünde küçük değerler seçilememektedir. Çünkü küçük değerler seçilirse, sistem brute force (kaba kuvvet) ataklarına karşı zayıf kalacaktır. Bu yüzden gizli anahtar üsleri için oldukça büyük değerler seçilmektedir. Fakat bu değerin çok büyük olması ise şifreleme ve imzalama işlemlerinin yavaş gerçekleşmesini sağlamıştır. Durum böyle olunca, Chinese Remainder Theorem (CRT) geliştirilerek, gizli anahtar üssü ile gerçekleşen işlemler hızlandırılmıştır [21]. **Şekil 3.17**'de görüldüğü üzere, verilen p ve q asal sayıları ile iki farklı gizli anahtar üssü ve bir katsayı elde edilir. Bunlar gizli anahtarın 5 temel parçası olarak kullanılır. **Şekil 3.18**'de ise şifrelenmiş bir c verisi, m_1 ve m_2 olmak üzere iki parçaya ayrılarak daha küçük üs değerleri ile hesaplanır ve açık veriye daha hızlı bir şekilde ulaşır.



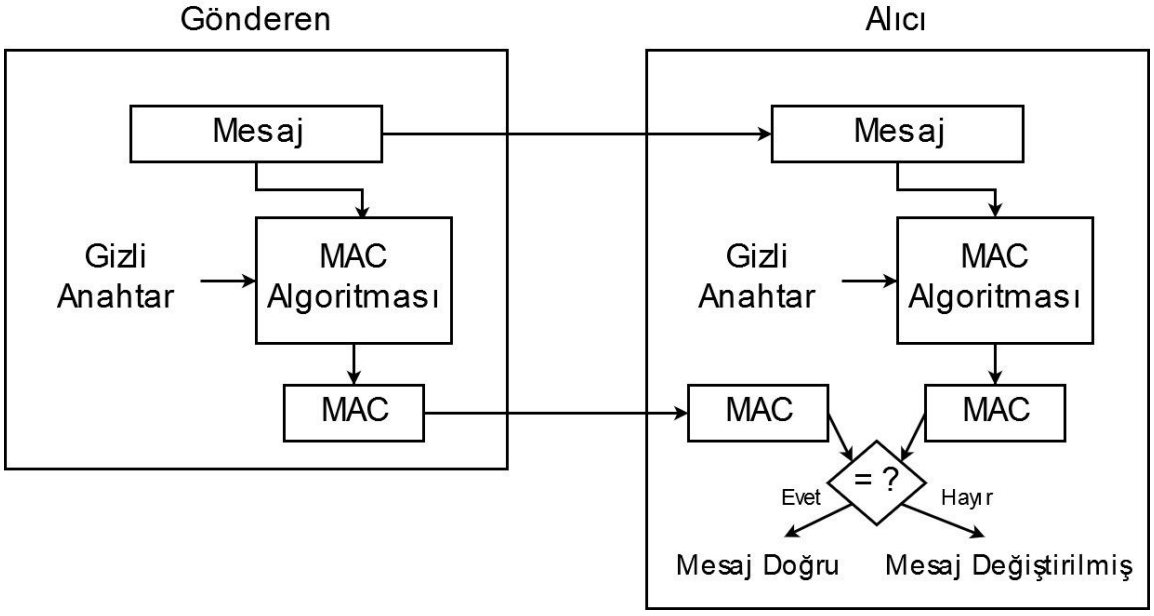
Şekil 3.17: CRT Dönüşümü



Şekil 3.18: CRT Alanında Üssel İşlemler

3.2.4 AES MAC Algoritması

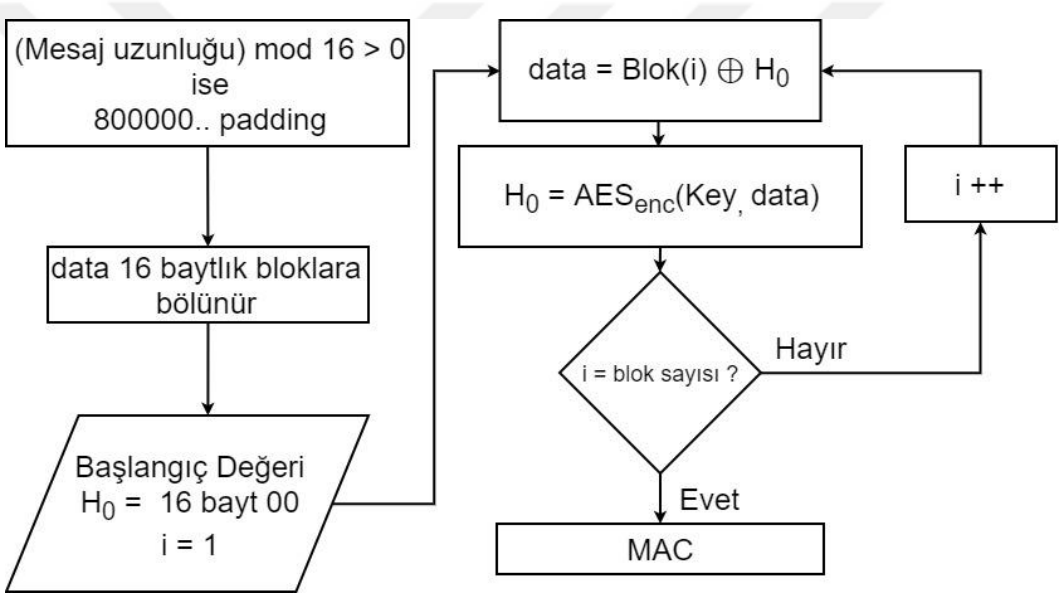
Message Authentication Code (MAC) algoritması, mesaj bütünlüğünü doğrulama amaçlı kullanılan bir algoritmadır. Bu algoritma sayesinde mesajın yetkin bir gönderici tarafından gönderildiği ve mesajın değişikliğe uğramadığı doğrulanmaktadır. **Şekil 3.19**'a bakıldığında, mesaj göndericisi, göndermek istediği mesajı MAC algoritmasında gizli anahtar ile işleyerek bir MAC değeri elde etmektedir. Mesaj ve MAC değeri alıcıya gönderilir. Alıcı tarafa ulaşan mesaj, gönderici tarafta bulunan aynı gizli anahtar ile bu mesajı MAC algoritmasında işleyerek bir MAC değeri oluşturur. Oluşturulan bu MAC değeri ile göndericiden gelen MAC değerleri karşılaştırılır ve değerler eşit ise veride bir değişme olmadığı ve göndericinin doğru bir gizli anahtar kullanmış olduğu teyit edilir.



Şekil 3.19: MAC Algoritması Kullanımı

MAC algoritması tek yönlü işleyen bir algoritmadır. Bu sebepten ötürü hesaplanan bir MAC değeri geri döndürülemez. Bir MAC değerini doğrulayabilmek için, algoritmaya giren mesaj ve gizli anahtar, doğrulama yapılan tarafta da bulunmalıdır. Her iki tarafta da aynı gizli anahtar bulunması gerektiğinden, doğrulama öncesinde bu gizli anahtarın taraflar arasında güvenli bir şekilde paylaşılması gerekmektedir.

MAC algoritması, **Şekil 3.20**'deki akış diyagramı ile gösterilmiştir. İlk olarak, 16 baytlık bloklar halinde çalışılacağı için giren mesajın uzunluğunun 16'nın katı olup olmadığına bakılır ve eksik kısımlar bloğu doldurana kadar metot 2 olan 8000 paddingi ile doldurulur. CBC modda AES şifreleme yapılacağından ötürü, mesaj ve gizli anahtar 16 baytlık bloklara bölünür. Her blok, bir H_0 değeri ile XOR işleminden geçerek oluşan yeni data gizli anahtar ile AES algoritması altında şifrelenir. Buradan çıkan değer ise yeni H_0 değeri olarak tutulur. Mesajın bütün blokları bu şekilde işlendikten sonra, en son kalan H_0 değeri mesaj onaylama kodu (MAC) olarak kullanılır[22].



Şekil 3.20: MAC Algoritması Akış Diyagramı

3.3 WHITEBOX KÜTÜPHANESİ

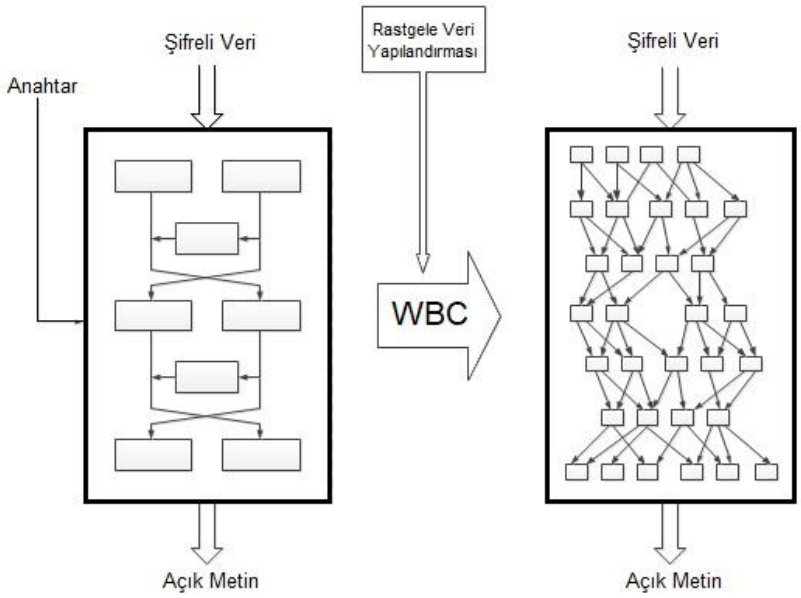
Whitebox kriptografi, kriptogramın yeni bir alanıdır denebilir. Bu alandaki ilk çalışmalar, Chow tarafından AES[23] ve DES[24]'in ilk WhiteBox geliştirmesi ile başlatılmıştır.

Şifreleme algoritmaları; akıllı telefon, bilgisayar gibi kişisel cihazlarda çalıştırılan uygulamalarda giderek daha fazla kullanılmaktadır. Bu cihazların açık yapısının bilinmesinden ötürü, saldırganların sistemlerde ve yazılım süreçlerinde tam denetime sahip olmasından dolayı, şifreleme algoritmasına sahip yazılımlar saldırganlara karşı savunmasız hale gelmektedir. Bu da demek oluyor ki, saldırganlar uygulamanın binary kodunu ve çalışma esnasında karşılık gelen bellek değerlerini kolayca analiz edebilmektedirler. Bu sayede saldırganlar; hata ayıklayıcılar, emülatörler, IDA Pro gibi saldırı araçlarını kullanarak sistem çağrılarına, binary dosyalara ve uygulamaya müdahalede bulunabilirler.

WhiteBox Cryptography (WBC) amacı, saldırılara maruz kalsa bile güvende olacak bir şifreleme algoritması uygulamaktır. Bu yüzden, WBC, yazılım koruma stratejilerinde çok önemli bir teknolojidir. Bu teknoloji, kullanılacak gizli anahtar gibi hassas bilgilerin herhangi bir kısmını açığa çıkarmadan kriptografik işlemlerin gerçekleştirilmesini sağlamaktadır. Bu teknoloji olmadan, saldırganlar binary uygulamalardan, bellekten veya yürütme zamanından saldırılar sonucu açığa çıkacak hassas bilgileri kolayca alabilmekteler.

WBC, verilen şifreyi güçlü bir sunum haline dönüştüren özel amaçlı bir kod üreticisi olarak da görülebilmektedir. Bu öyle bir sunum ki, gizli anahtardaki operasyonların rastgele veriler ve kodlarla birleştirilmesinden ötürü, gizli anahtar ayırt edilemeyecek bir biçime gelmektedir.

Şekil 3.21'de sabit bir anahtar uygulaması için WBC'nin üst düzey kavramsal genel görünümü gösterilmektedir. Verilen şekilde, anahtarın sabit olarak kodlandığı varsayılmıştır. Sol taraftaki kutuda, anahtar ve şifreleme adımları bulunmaktadır. Sağ tarafta ise, WhiteBox transformasyonları sonucunda oluşturulan, semantik olarak aynı kabiliyette olan fakat koda gömülü anahtarı çıkarması çok daha zor olan bir uygulamanın kodu gösterilmektedir.



Şekil 3.21: WBC Üst Düzey Kavramsal Genel Görünüm

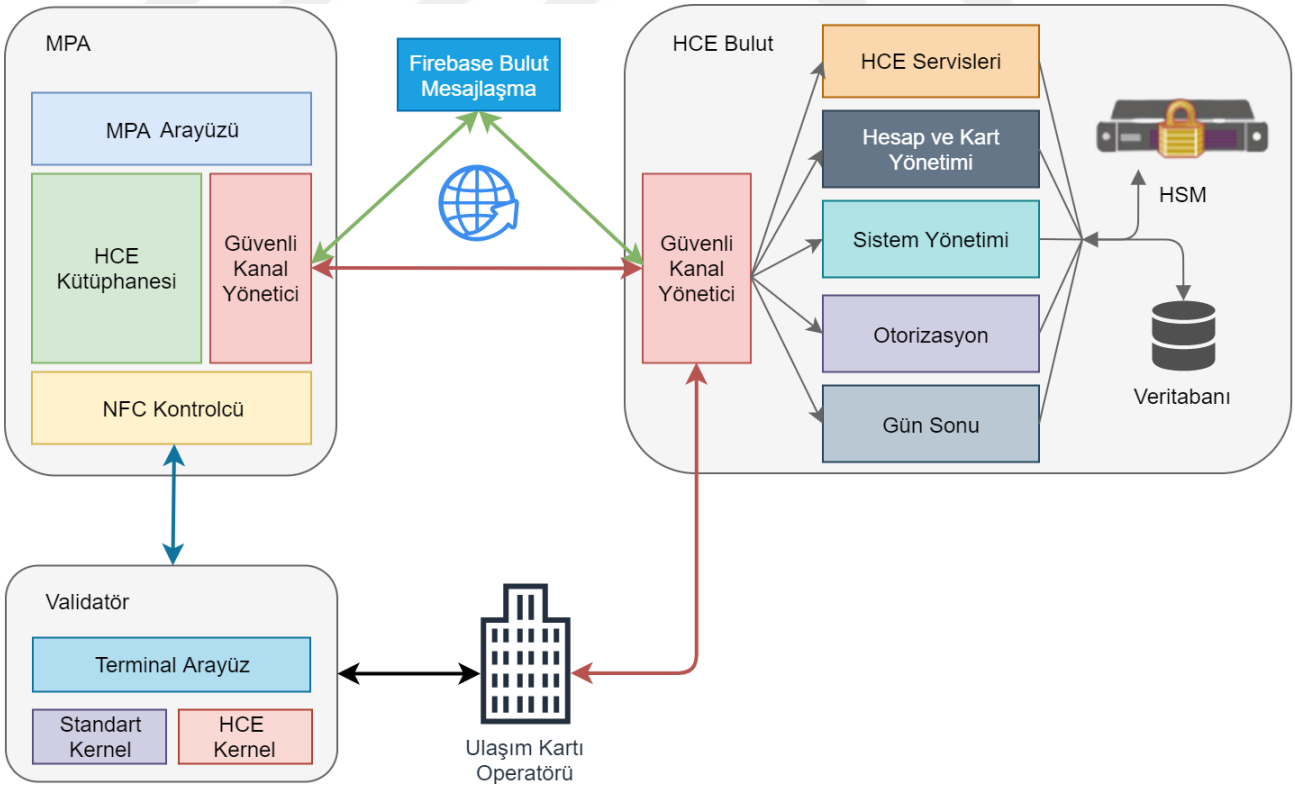
3.4 HARDWARE SECURITY MODULE

Hardware Security Module (HSM), fiziksel ortamlarda kriptografik anahtarları saklayabilmek ve kriptografik işlemleri en güvenli şekilde gerçekleştirebilmek amacıyla üretilmiş donanım güvenlik modülleridir. Bu modüller; veritabanı şifreleme, çevrimiçi bankacılık, anahtar korunumu, doküman haklarının korunumu, doküman ve kod imzalama, SSL Web, XML web servisleri, sertifika geçerliliği, DNS güvenliği, zaman damgası gibi uygulamaların güvenli bir şekilde çalışmasını sağlamaktadır.

HSM'ler talep edilen genel güvenlik standartlarını NITS veya Federal Information Processing Standard (FIPS) sertifikasyonlarına uyarak sağlamaktadırlar. HSM'ler olası saldırılara karşı kendisini savunabilecek sistemler ile donatılmak zorundadırlar. Herhangi bir fiziksel müdahale olması durumunda ise kendi kendilerini sıfırlama yani içeride saklanan bütün hassas verilen yok edilmesi özelliğine sahiptirler [25].

3.5 MOBİL ÖDEME UYGULAMASI

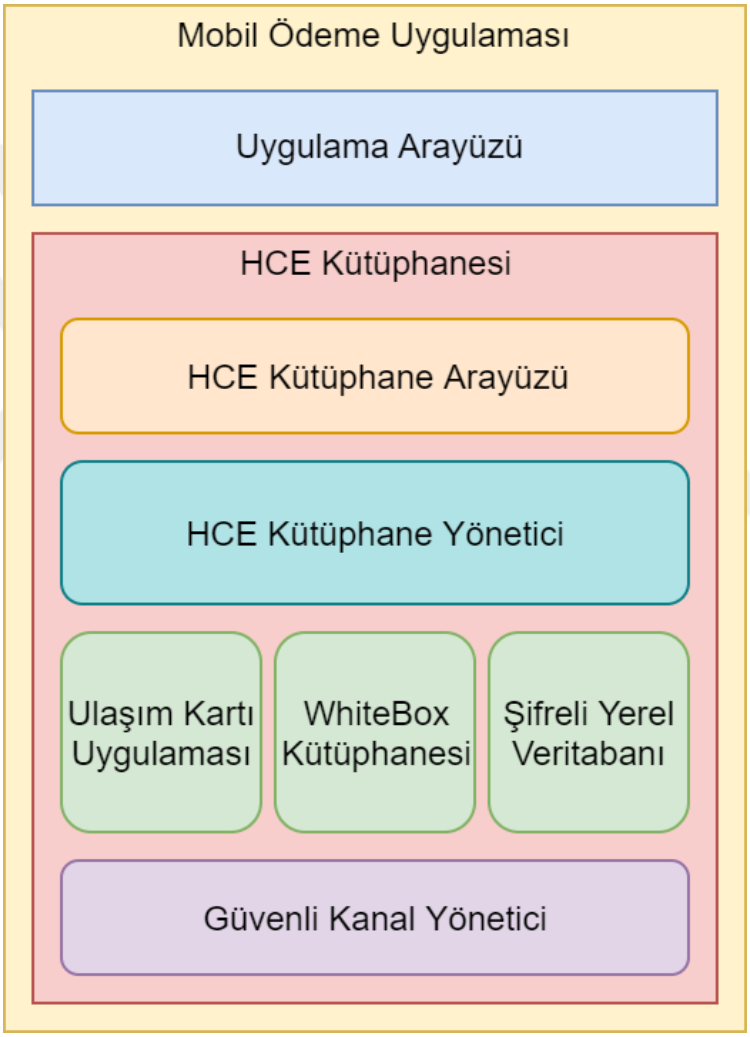
HCE kütüphanesi, birçok mobil ödeme sistemine entegre olabilir şekilde tasarlanarak, ulaşımda çevrimdışı mobil ödeme yapabilmeyi sağlamaktadır. **Şekil 3.22**'de görülen sistem mimarisine bakıldığında, Mobile Payment Application (MPA) olarak adlandırılan mobil cihazlarda çalışan uygulama, ara katmanda Firebase bulut mesajlaşma (FCM) ile veya güvenli kanal ile uzak sunucuyla iletişim kurmaktadır. Sunucu tarafında ise HCE bulut olarak adlandırılan, HCE yönetimi, dijital kart üretimi, işlem doğrulama, rapor oluşturma gibi işlemleri yapan ve güvenlik altyapısı ile anahtar yönetimi için HSM cihazı kullanan bir sunucu bulunmaktadır. MPA'nın işlem gerçekleştirmek için NFC kontrolcüsü üzerinden eriştiği validatörlerde ise, HCE kütüphanesindeki ulaşım kartı uygulamasıyla haberleşebilecek olan HCE kernel terminalde bulunur ve yaptığı işlemleri ulaşım kartı operatörüne ileterek, operatör tarafından HCE buluta erişilir ve işlemler doğrulanır.



Şekil 3.22: Sistem Mimarisi

3.5.1 Mobil Ödeme Uygulaması Genel Yapısı

MPA, Android işletim sistemli akıllı telefonlarda çalıştırılmak üzere geliştirilmiş bir mobil uygulamadır. **Şekil 3.23**'te uygulamanın modülleri ve katmanları belirtilmiştir.



Şekil 3.23: Mobil Ödeme Uygulaması Bileşenleri

MPA, uygulama arayüzü ve HCE Kütüphanesi olarak iki ayrı katmandan oluşmakta. Uygulama arayüzü, bu tez kapsamında incelenmeyecektir. HCE kütüphanesi, 6 modülden oluşmaktadır.

HCE kütüphane arayüzü, uygulama arayüzünün kullanması gereken metotları bulundurmaktadır. Bu arayüz ile HCE'nin ilklendirilmesi, djital kartın oluşturulması, validatör ile ulaşım işlemi, HCE verilerinin silinmesi ve arayüzde kullanıcı ile paylaşılması gereken verilerin alınmasını sağlayan metotlar bulunmaktadır.

HCE kütüphane yöneticisi, arayüzden, sunucudan ve NFC arayüzden gelen istekler doğrultusunda diğer alt modülleri yöneten ve tüm işlemleri gerçekleştiren ana modüldür.

Ulaşım kartı uygulaması, EMV altyapısıyla hazırlanmış bir fiziksel temassız kartı emüle eden modüldür. Ulaşım araçlarında veya durak girişlerinde bulunan validatörlerle iletişim kurarak, bir ulaşım kartı gibi davranmasını sağlar.

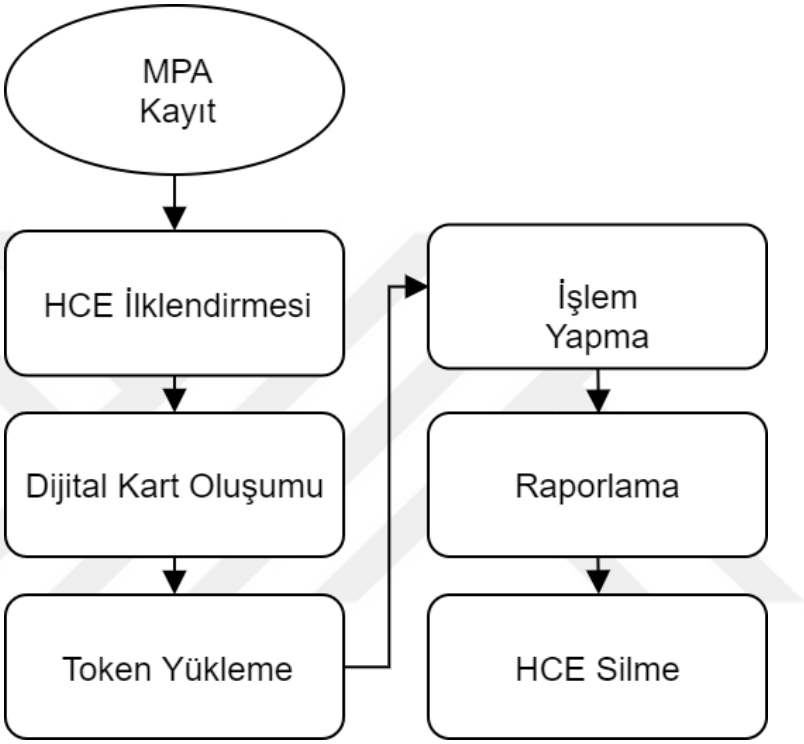
WhiteBox kütüphanesi, cihazın parmak izi ile ilklendirilerek, o cihazdan başka cihazlarda çalıştırıldığında doğru sonucu vermeyi önleyen, C dilinde geliştirilmiş bir kripto kütüphanedir. Saldırganın belleğe erişimi olsa bile, anahtarın açığa çıkmasını önlenmektedir.

Şifreli yerel veritabanı, SQLCipher kütüphanesiyle şifrelenmiş bir SQLite veritabanıdır. Dijital kart, token, işlem kayıtları ve diğer veriler bu veritabanında, şifreli olarak saklanmaktadır.

Güvenli kanal yöneticisi, MPA ile sunucu arasında güvenli kanal kurar ve iletişimi buradan gerçekleştirir.

3.5.2 Uygulama Senaryosu

Şekil 3.24'te görülen senaryo, MPA'da çevrimdışı şekilde, verilerin bütünlüğünü ve güvenliği sağlayarak HCE teknolojisiyle gerçekleştirilecek olan bir ulaşım işleminin senaryosudur.



Şekil 3.24: MPA Senaryosu

Kullanıcı tarafından, mobil cihaza yüklenen MPA'dan, ulaşım sağlayıcı operatör sistemine kayıt olunur. Operatör tarafından, gerekli bilgiler HCE sunucusuna iletilerek, HCE sisteminde kullanıcı kaydı açılır. Kullanıcı, bilgilerini ve bakiyesini MPA arayüzünden görebilmekte ve güncelleyebilmektedir. MPA'da 2 çeşit bakiye bulunmaktadır: sunucuda bulunan merkez bakiye ve mobil cihazda HCE ile kullanmak üzere karta token aracılığıyla yüklenen token bakiyedir.

Kullanıcı tarafından mobil ödeme aktifleştirilmek istendiğinde, HCE ilklendirmesi gerçekleşir. Bu adımda, gerekli güvenlik ve cihazın uygunluğu kontrolleri sonrasında, mobil cihazda şifreli kütüphane oluşturulup ve WhiteBox kütüphanesi ilkendirilmesi yapılarak, güvenli veri saklama

ve kriptografik hesaplamalar yapmak için gerekli ortam oluşturulur. Sunucuyla güvenli kanal üzerinden haberleşmek için gerekli anahtarlar da alınarak, şifreli veritabanında saklanır.

MPA'da güvenli ortam başarıyla oluşturulunca, dijital kart oluşturmak için HCE sunucusu ile haberleşilir. MPA'da hazırlanan mobil cihazın parmak izi ile sunucu tarafında MPA'ya özel WhiteBox kütüphanesinde kullanmak için anahtarlar türetilir. HCE sunucusu ile operatör sunucusu arasında kişisel veriler alınır ve dijital kart için gerekli veriler oluşturulur. Oluşturulan dijital kart, güvenli kanal üzerinden MPA'ya iletilir ve şifreli veritabanında saklanır.

Dijital kart ile işlem yapılabilmesi için, HCE sunucu tarafından üretilecek tokena ihtiyaç vardır. Kullanıcı tarafından dijital kart için bakiye yüklenir ve bu bakiye merkez bakiye olarak saklanır. Kullanıcının MPA'ya yüklemek istediği tutar ile HCE sunucuda belli kullanım limitleri olan token hazırlanır. Oluşturulan token, güvenli kanal üzerinden MPA'ya iletilir ve şifreli veritabanında saklanır.

Tokenın kullanıcı talimatıyla yenilenmesi gerektiğinde, internet erişimi sağladığında otomatik olarak yenilenir.

Ulaşım için geçiş işlemi yapmaya hazır olan MPA ile ulaşım sağlayıcı operatörün validatörüne okutulan mobil cihazda, validatördeki HCE kütüphanesiyle iletişim kurmak için hazırlanmış olan HCE kernel ile MPA'da HCE kütüphanesinde bulunan ulaşım kartı uygulaması arasında gerçekleşen ISO/IEC 14443 standartında iletişim ile HCE kernel tarafından işlem onaylanır ve geçiş sağlanır.

MPA tarafında yapılan geçiş işleminin kaydı tutularak, internet bağlantısı varsa HCE sunucusuna bildirilir, erişim yoksa şifreli veritabanında saklanarak, erişim sağlandığında HCE sunucusuna kaydı iletir. Validatör tarafında ise, internet erişimi işlem başına değilde, belli zaman aralıklarıyla gerçekleşir. İnternet erişimi sağladığında, validatörde tutulan işlem kayıtları operatör sunucusuna iletilir ve oradan da HCE sunucusuna iletilerek yapılan geçiş işlemi doğrulanır.

Kullanıcı isteğiyle veyahut güvenlik kontrolleriyle tespit edilen saldırılar durumunda, HCE kütüphanesinde şifreli veritabanı ve WhiteBox kütüphanesinde bulunan dijital karta ait veriler silinerek, gerekli güvenli ortam sağlanıp, aynı kurulum tekrar yapılana kadar MPA ile geçiş işlemi yapılamaz.

3.5.3 Güvenlik Mimarisi

Mobil ödeme uygulamasında alınan güvenlik önlemleri ve uzak sunucu bağlantısındaki önlemler alt başlıklardaki gibidir.

3.5.3.1 Uygulamanın Güvenliği

MPA'da alınan güvenlik önlemleri **Tablo 3.2**'de belirtilmiştir ve devamında maddeler halinde incelenmiştir.

Tablo 3.2: Uygulamanın Güvenlik Önemleri

Şifreli Yerel Veritabanı	Cihaz Parmak İzi	Root Kontrolü	Hata Ayıklama Modu ve Emülatör Kontrolü
Dış Müdahale Kontrolü	Şifreleme Algoritmaları	SSL Pinning Koruması	Hassas Kişisel Veriler
Hassas Kart Verileri	Şifrelenmemiş Veriler	Versiyon Kontrolü	Hata Ayıklama Fonksiyonları ve Logları
Loglama ve İzleme	Kayıtlı Veriler	Kod Karmaşıklığı	WhiteBox Crypto Kütüphanesi

- **Şifreli Yerel Veritabanı**

  HCE kütüphanesi, uygulamayla alakalı tüm verileri, SQLCipher kütüphanesiyle şifrelenmiş olan SQLite veritabanında depolar. SQLCipher anahtarı, cihazın parmak izi ve bir rastgele sayı ile oluşturulur. Veritabanı cihaz bağımlıdır, farklı bir cihazda okunmaya çalıştığında doğru sonuç vermeyecektir.

- **Cihaz Parmak İzi**

  Uygulamanın kullanıldığı akıllı telefonun donanım ve işletim sisteminin benzersiz parametreleri ile cihazın parmak izi oluşturulur. Bu değer cihaza özel ve farklı cihazlarda oluşmayacak eşsiz bir değerdir. Bu değer, cihazda saklanmaz, gerektiği her zaman yeniden üretilir. MPA'nın sunucu ile kimlik doğrulama adımında kullanılır.

- **Root Kontrolü**

  Mobil cihazın, işletim sisteminin kırılıp kırılmadığı kontrol edilir. Android işletim sistemlerinin kırılmasına root denmektedir. Eğer cihaz root edilmiş ise, uygulama başlamayacak ve sunucuya bu bilgi verilecek ardından şifreli veritabanı ve WhiteBox kütüphanesi silinecek. Bu kontrol, uygulama her açıldığında gerçekleşir.

- **Hata Ayıklama Modu ve Emülator Kontrolü**

  İşletim sisteminin özellikleri, parametreleri kontrol edilir. Üçüncü parti kütüphaneler ile farklı kontroller gerçekleştirilir. Eğer cihaz hata ayıklama modunda veya emülatör üzerinden çalıştırılmışsa, sunucuya bilgi verilir ve şifreli veritabanı ve WhiteBox kütüphanesi silinir.

- **Dış Müdahale Kontrolü**

  MPA'da yetkisiz yama veya kaynak değişikliği ile uygulama bütünlüğünün bozulduğunun tespit edilmesidir. Kriptogram hesaplamalarıyla alakalı tüm kritik operasyonlar özel yerel kütüphane ile gerçekleşmektedir. Her işlem esnasında ve sunucu ile doğrulamada, yerel kütüphanenin hash değeri alınır ve kontrol edilir. Eğer yerel kütüphanede bir değişiklik yapıldığı tespit edilirse şifreli veritabanı ve WhiteBox kütüphanesi silinir.

  Kaynak kodda değişiklik yapılmasını önlemek için AndHook kütüphanesi kullanılmaktadır. Ayrıca kodda flow check ve double check mekanizmaları ile de kontroller sağlanmaktadır.

- **Şifreleme Algoritmaları**

  MPA'da, NIST tarafından onaylanan güçlü şifreleme algoritmaları kullanılmıştır, çünkü güçsüz algoritmalar saldırganlar tarafından verilen çözülme riskini oluşturur. Anahtar üretim aşamalarında, tamamen kriptografik olarak güçlü rastgelelik kaynakları kullanılmıştır. Her bağlam için farklı anahtar kullanılması, örneğin; kimlik doğrulama, oturum başlatma isteği gibi. Bu sayede, tek bir anahtarın çalınması durumunda sistemin kullanılabilir olmaması sağlanmaktadır.

- **SSL Pinning Koruması**

  SSL Pinning'i önlemek için, SSL sertifikasının doğru PIN'lerinin SHA256 hash'i, sunucudaki veritabanında saklanır. MPA, TLS güvenlik kanalı ile bu hash'i, sunucu ile her iletişime girdiğinde şifreli olarak gönderir.

  İki iletişim tipi bulunmaktadır, biri kayıt için, diğeri ise oturum başlatmak içindir. Kayıt adımında, veriler sunucunun public anahtarı ile şifrelenerek gönderilir. Oturum başlatma adımında ise, oturumun güvenli kanal gizlilik anahtarı ile şifrelenerek gönderilir.

  Bu hash değeri sunucu tarafında kontrol edilir ve değer uyuşmazsa iletişim kesilir.

- **Hassas Kişisel Veriler**

  MPA; şifre, PIN, oturum bilgisi veya şifreleme anahtarı gibi hassas verileri şifreli olmadıkları sürece kayıt etmez veya bellekte bulundurmaz.

- **Hassas Kart Verileri**

  MPA; kart numarası, token veya uygulama kriptogramı gibi hassas verileri şifreli olmadıkları sürece kayıt etmez veya bellekte bulundurmaz.

- **Şifrelenmemiş Veriler**

  Bellekte veri açık olarak tutulacak ise; sadece veriye ihtiyaç duyulan süre boyunca bellekte tutulur, hafızadan da silinmesi isin çöp toplayıcı (garbage collector) çalıştırılır, bellekteki adresler 0x00 baytı ile doldurulur.

- **Versiyon Kontrolü**

  Sunucuyla iletişime geçildiğinde, MPA'nın versiyonu gönderilir ve uygun olmayan bir versiyon kullanılıyorsa, güncellenmeye zorlanır.

- **Hata Ayıklama Fonksiyonları ve Logları**

  Uygulama yayınlanmadan önce, kaynak kodda geliştirme aşaması için kullanılan hata ayıklama fonksiyonlar ve loglar kapatılmalıdır. Uygulama hata ayıklama modunda değil, yayınlamak için derlenmelidir.

- **Loglama ve İzleme**

  MPA'da kullanıcının yaptığı işlemler, gerçekleştirdiği değişiklikler kayıt altında tutularak, olağan dışı bir durumda tespit edilebilmektedir.

- **Kayıtlı Veriler**

  Kriptografi anahtarları ve token verileri, veritabanına açık olarak kaydedilmez, veritabanı şifreli olsa da bu veriler şifrelenerek veritabanına kaydedilir. Çünkü bu veriler, cihaza özel oldukları için, sadece o cihazda okunabilir olmalılar ki farklı cihazlara veya emülatörler kopyalanarak kullanılamamalılar.

- **Kod Karmaşıklığı**

  Uygulama kaynak koda dönüştürüldüğünde, anlaşılabilirliği bozmak için kod karmaşıklığı (obfuscation) yapılır. Böylece uygulamanın akışı ve kritik kısımlar, statik kod analizi ve tersine mühendislik ile anlaşılamayacak biçimde olur.

- **WhiteBox Kripto Kütüphanesi**

  WhiteBox kript kütüphanesi, sunucudan gelen token verilerini çözmek için kullanılmaktadır. WBC kütüphanesi sayesinde, saldırganın belleğe erişimi olsa dahi anahtarların çalınması önlenmektedir. Cihazın parmak izi ile kullanıldığından ötürü, farklı cihazlarda çalıştırılması sonucunda, doğru çıktıyı vermeyecektir.

  Sunucu tarafında, HSM ile şifrelenen verileri mobil cihazda çözmek için kullanılan, C dilinde yazılmış bir kütüphanedir. MPA tarafından, kütüphanenin binary hash'i kontrol edilerek, kütüphaneye müdahele edilip edilmediği tespit edilir. MPA'yı kullanan her mobil cihaz için, eşsiz bir WB anahtarı oluşturulur. Cihazın parmak izi ile türetilerek üretilen eşsiz anahtar, her zaman şifreli olarak saklanır.

3.5.3.2 Uzak Sunucu ile İletişimin Güvenliği

En hassas veriler olan tokenlar ve anahtarlar, sunucudan MPA'ya iletilirken, sadece MPA'da çözülebilecek şekilde şifrelenmelidir ve cihazdan dışarı çıkarılamamalıdır. Anahtarlar, ek anahtarlarla korunarak aktarılırsa, bu anahtarlar saldırganlar tarafından çalınamaz. Bu yüzden, NIST tarafından onaylanmış bir standart olan Transport Layer Security (TLS) kullanılarak uçtan uca şifreli bir güvenli kanal kullanılmalıdır. Hem MPA hem de sunucu birbirlerini doğrulamalılardır. MPA tüm iletişim için güvenli kanal kanallar kullanacak ve bağlantı oturumları kullanılmadığı zaman sonlandırılacaktır. MPA'da TLS 1.1 ve TLS 1.2 kullanılmıştır.

- **Ön Doğrulama**
  Sunucuda oturum başlatmak isteyen her MPA, ilk olarak ön doğrulamadan geçmesi gerekmektedir. Bu yüzden, MPA ilklendirilirken cihazın parmak izi ve oluşturulan bir kriptogram ile sunucu da doğrulanır.

- **İki Kanallı İletişim**
  Oturum başlatma istekleri, MPA'dan sunucuya iletilir. Sunucuda oluşturulan oturum ID'si, MPA'ya farklı bir kanal üzerinden gönderilir. Farklı kanal olarak Google'ın bir servisi olan Firebase Cloud Messaging (FCM) kullanılmıştır.

- **Üç Katmanlı Şifreleme ve Güvenlik**
  Tüm mesajlaşma işlemleri TLS kanalı üzerinden gerçekleştirilir. Güvenlik, güvenli kanal anahtarları MPA'ya iletilene kadar birkaç asimetrik RSA anahtarı ile sağlanır. MPA'dan sunucuya her iletişim isteği için benzersiz iki AES oturum anahtarı oluşturulur. Bu anahtarlardan birisi kimlik doğrulamak için, diğeri ise mesajı şifrelemek için kullanılır. İlk olarak, MPA'dan sunucuya gelen doğrulama kriptogramı doğrulanır. Doğrulama aşaması onaylanınca, şifreleme anahtarı ile mesaj çözülür.

  Güvenlik düzeyini arttırmak için, şifrelenmiş mesajı şifrelemek için WBC anahtarı da kullanılır. Ayrıca tüm şifreli mesajlar, mesaj indeksi içerir. Eğer bu mesaj indeksi MPA'dan önceden alınmış ise, istek reddedilir.

3.5.4 Uygulama Adımları

Bu başlık altında mobil ödeme uygulamasının akışları detaylı olarak incelenmiştir. Bu akışlarda uygulamanın ve uzak sunucu iletişimindeki güvenliğe değinilmiştir.

3.5.4.1 HCE İlklendirilmesi

MPA ilk yüklendiğinde, HCE kütüphanesi tarafından kontroller yapılır, cihaz kullanıma uygunsa, gerekli veriler üretilir ve uzak sunucuya gönderilerek cihaz kaydedilir. Ardından da şifreli veritabanı ve WhiteBox tablosu oluşturulur. **Şekil 3.25**'te HCE ilklendirilmesinin aşamaları gösterilmiştir ve alt kısımlarda açıklanmıştır.

1. adımda, arayüzden gelen istekle işlem başlatılıyor. İlk olarak cihazda NFC donanımı kontrolü, root kontrolü, hata ayıklama modu kontrolü ve emülatör kontrolü ve diğer genel kontroller yapılarak mobil cihazın uygun olup olmadığı kontrol edilir. Cihazın uygun olma durumunda, ilklendirilmeye başlanır ve ilk olarak cihazdan alınan (ro.product.brand, ro.product.cpu.abi gibi) parametrelerle bir değer oluşturulur ve WhiteBox kütüphanesine iletilerek bu verinin SHA256 hash değeri hesaplanır ve oluşturulacak olan şifreli veritabanı anahtarı olarak kullanılır. Bu anahtar hiçbir yerde saklanmaz, ihtiyaç olduğunda üretilerek kullanılır ve ardından silinir. Şifreli veritabanı gerekli tablolar ile oluşturulur.

İlerleyen süreçlerde de kullanılmak üzere uzun bir rastgele sayı oluşturulur. Sonrasında, cihazın yapı bilgileri (ro.build.flavor, ro.build.host gibi) ile cihaza özel bir değer üretilir. Bu üretilen rastgele sayı ve özel değer uç uca eklenerek WhiteBox kütüphanesine iletilerek SHA256 hash değeri hesaplanır, bu değer eşsiz MPA numarası olarak adlandırılmaktadır.

2. adımda, eşsiz MPA numarası sunucuya iletilerek, sunucudan bulut açık anahtarı beklenir. Sunucu tarafında, gelen bu istek doğrultusunda eşsiz MPA numarası yeni bir kullanıcı olarak kayıt edilir ve güvenli iletişimi kurmak için bulut açık anahtarını 3. adımda döner. MPA, bu gelen bulut açık anahtarı WhiteBox kütüphanesine gönderiyor ve içerde MPA için bir asimetrik RSA key çifti üreterek, MPA açık anahtarını, bulut açık anahtarı ile şifreleyerek sunucuya gönderilmek üzere hazırlıyor. Bulut açık anahtarı, sadece MPA açık anahtarını sunucuya güvenli şekilde iletme amacıyla kullanılmaktadır.

4. adımda, MPA açık anahtarı sunucu ile paylaşılacaktır. Ayrıca, sunucunun doğruluğunu tespit etmek amacıyla, MPA'da gömülü olarak bulunan bulut açık anahtarının hash değerini de WhiteBox kütüphanesinde MPA gizli anahtarı ile şifreleyerek sunucu ile paylaşılır. Eşsiz MPA numarası, bulut açık anahtarı hash değeri ve MPA açık anahtarı sunucuya gönderilir. Sunucu tarafında, gizli bulut anahtarı ile bulut açık anahtarının hash değeri ve MPA açık anahtar HSM'de çözülür, hash değeri kontrol edilir ve doğrulanırsa eşsiz MPA numarasına göre, MPA açık anahtarı kaydedilir. İlerleyen süreçlerde, kullanıcıya özel MPA açık anahtarı ile veriler şifrelenerek gönderilecektir, MPA tarafına ulaşılan veriler ise WhiteBox kütüphanesinde, MPA gizli anahtarı ile çözülerek kullanılacaktır. 5. adımda ise onay veya iptal mesajı sunucudan MPA'ya dönmektedir. Sonuca göre işlem devam eder veya iptal edilir.

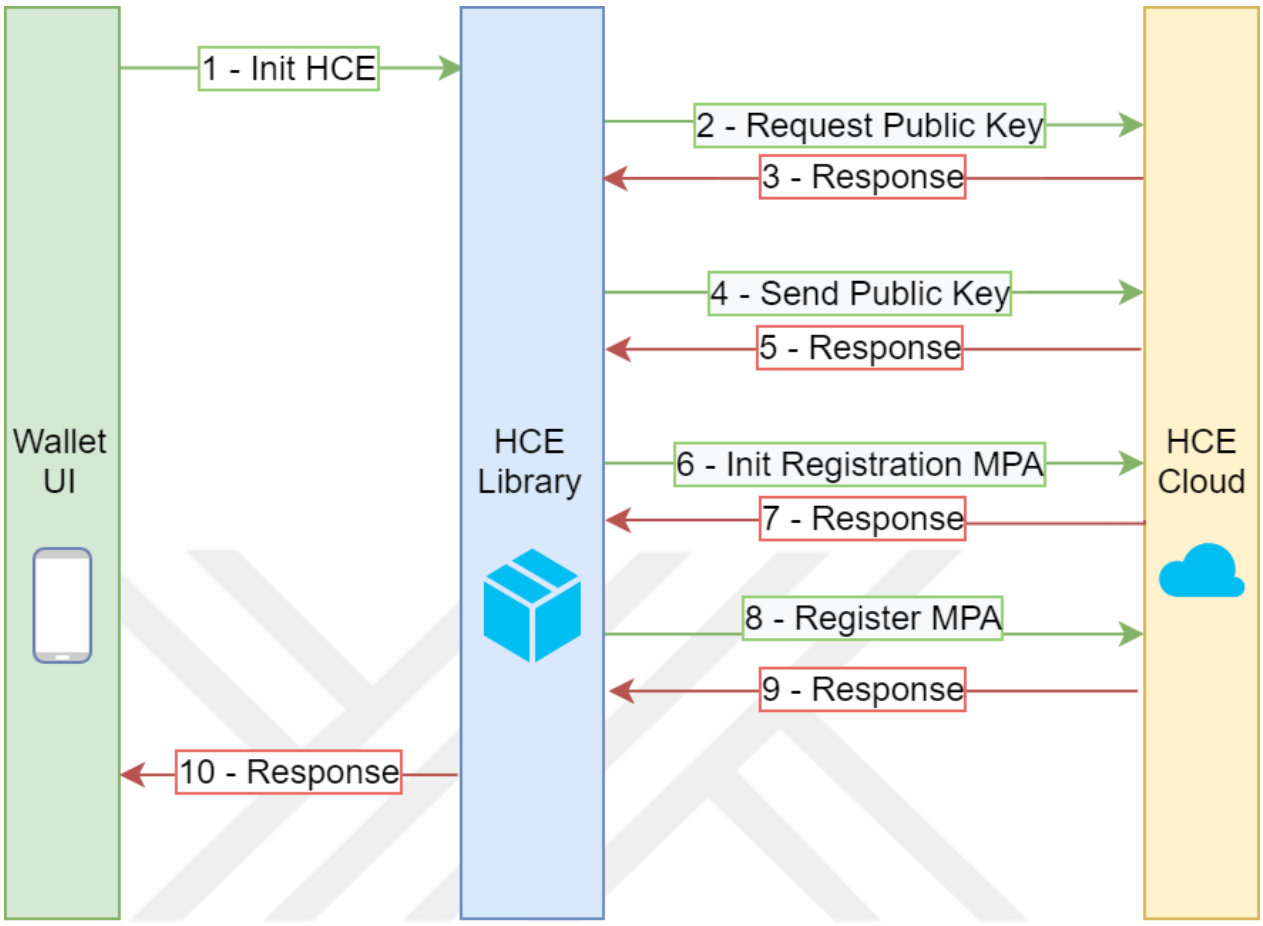
6. adımda, MPA sunucuya kayıt olmak için gerekli verileri hazırlar ve gönderir. Mobil cihazın bütün cihaz özellikleri ve sistem özellikleri alınarak, MPA cihaz bilgisi verisi oluşturulur ve MPA gizli anahtarı ile WhiteBox kütüphanesinde şifrelenir. WhiteBox kütüphanesinin hash değeri hesaplanır, kütüphanede bir değişiklik olmadığını kontrol etmek için. Cüzdanda bulunan FCM ID de MPA gizli anahtarı ile şifrelenerek gönderilir ki, sunucu tarafından FCM kanalından mesaj gönderilmek istendiğinde gideceği adresi bilmesi amacıyla. Ayrıca MPA'nın versiyon numarası ve eşsiz MPA numarası da gönderilir. Sunucuya ulaşan bu veriler, şifreli olanlar MPA gizli anahtarı ile HSM'de çözüldükten sonrasında, ilk olarak MPA versiyonunun kullanıma uygun olup olmadığı kontrol edilir. Sunucuda kayıtlı bulunan WB kütüphanesi hash değeri ile, MPA'dan gelen değer karşılaştırılır ve kütüphaneye herhangi bir müdahele olmuş mu kontrol edilir. Kontroller başarıyla sonuçlandıktan sonra, 8. adımda MPA cihaz bilgisine göre mobil cihazın parmak izini oluşturmak için cihaza özel bir cihaz parmak izi listesi oluşturulur ve MPA açık anahtarı ile şifrelenerek MPA'ya gönderilir.

8. adımda, cihaz parmak izi listesindeki veriler ile mobil cihazın sistem özellikleri alınır ve hali hazırda üretilmiş olan rastgele sayı ile uç uca eklenerek tamamen MPA'ya özel, eşsiz MPA parmak izi oluşturulur ve WhiteBox kütüphanesinde MPA gizli anahtarı ile şifrelenir. Eşsiz MPA numarası ve şifrelenmiş MPA parmak izi sunucuya gönderilir. Sunucu tarafında, MPA açık anahtarı ile şifreli MPA parmak izi çözülerek MPA parmak izi alınır. MPA parmak izi ile HSM'de mevcut tutulan doğrulama, şifreleme ve WhiteBox anahtarları ayrı ayrı HSM'de bulunan ana anahtarlar ile MPA parmak izi DES şifreleme ile şifrelenerek çıktı veriler ile

anahtar türetimi yapılır, MPA'ya özel yeni anahtarlar elde edilmiş olur. Bu anahtarlar sunucuda saklanmaz, ihtiyaç olduğunda üretilirler. 9. adımda, üretilen bu anahtarlar, MPA açık anahtarı ile şifrelenerek MPA'ya gönderilir.

10. adımda, sunucudan gelen verilerden şifreleme anahtarı ve doğrulama anahtarı MPA gizli key ile WhiteBox kütüphanesinde çözülür. WhiteBox anahtarı ise, WhiteBox kütüphanesinde MPA gizli anahtar ile çözülür ve eşsiz WhiteBox anahtarı elde edilir ve bellekte tutulur. Sonrasında, WhiteBox kütüphanesine oluşturulmuş olan rastgele sayı gönderilir, bellekteki eşsiz anahtar ile birlikte WhiteBox kütüphanesine özel büyük bir tablo oluşturulur ve WhiteBox kütüphanesi tamamen kullanıma hazır hale gelmektedir. Her kullanımda, eşsiz WhiteBox anahtarı ile şifreleme veya çözme yapılmak istendiğinde, MPA parmak izi gönderilerek ilklendirilmektedir. Şifreleme anahtarı, doğrulama anahtarı, eşsiz MPA numarası, WhiteBox kütüphanesinde, bellekteki eşsiz WhiteBox anahtarı ile şifrelenerek, şifrelenmiş veri tabanına kaydedilmektedir. Ayrıca MPA parmak izi listesi ve rastgele sayıda veritabanına kaydediliyor ve veritabanı şifrelenerek kapatılmaktadır.

Bu bölümde, MPA kurulu olan cihaz ve MPA kontrol edildi. Sunucu ile asimetrik anahtarlar ile şifreli bağlantı kuruldu ve MPA sunucuda kaydedilerek, doğrulama sağlamak için MPA'ya özel güvenli bağlantı anahtarları ve eşsiz WhiteBox anahtarı oluşturularak MPA ile paylaşıldı. MPA tarafında WhiteBox ilklendirildi, şifreli veritabanı oluşturuldu ve gerekli veriler veritabanına yazılarak MPA'da HCE ilklendirilmesi tamamlandı.



Şekil 3.25: HCE İlklendirilmesi Adımları

3.5.4.2 Dijital Kart Oluşumu

HCE ilklendirilmesi tamamlandıktan sonra, MPA'ya dijital kart yüklemek için gerekli güvenli ortam oluşmuştur. Sunucu ile güvenli kanal iletişimi için öncelikle oturum açılarak doğrulama yapılır ve sunucuda dijital kart oluşturularak kaydedilmek üzere MPA ile paylaşılır. **Şekil 3.26**'da dijital kart oluşum aşamaları gösterilmiştir ve alt kısımlarda açıklanmıştır.

1. adımda, arayüzden gelen istekle veritabanından ihtiyaç duyulan verileri alabilmek için, veritabanı anahtarını üreterek şifreli veritabanı açılmaktadır. Veritabanından alınan verilerden, eşsiz WhiteBox anahtarı ile şifreli olan veriler, WhiteBox kütüphanesine iletilerek, AES algoritması ile çözülüyor ve HCE kütüphanesi ile paylaşılmaktadır.

2. adımda, sunucu ile güvenli oturum başlatmak için, eşsiz MPA numarası ve FCM ID oturum başlatma isteği ile sunucuya gönderiliyor. Bunun sonucunda sunucudan, eşsiz MPA numarası aktif ise onay mesajı döndürülüyor ve oluşturulacak olan yeni oturum bilgileri FCM üzerinden iletilerek güvenlik için farklı bir kanaldan iletişim sağlanmaktadır.

4. adımda, FCM'den iletilen oturum bilgileri ile sunucuyla doğrulama gerçekleştirilecek. Doğrulama için, ilk olarak veritabanından çıkartılan şifreleme ve doğrulama anahtarları ile doğrulama kriptogramı oluşturulacak. FCM'den iletilen oturum bilgisi ve şifreleme anahtarı WhiteBox kütüphanesine iletilerek, oturum bilgisi şifreleme anahtarı altında AES-256 modunda şifrelenir ve oturum anahtarı olarak kullanılmak üzere bellekte saklanır. Oluşturulan oturum anahtarı ile sunucuya iletilecek olan veriler şifrelenecektir. Ayrıca, doğrulama anahtarı WhiteBox kütüphanesine iletilir, oturum anahtarı doğrulama anahtarı altında AES-256 modunda şifrelenerek doğrulama kriptogramını oluşturur ve HCE kütüphanesi ile paylaşır. Oluşturulan oturum anahtarı ile uygulama versiyonu, MPA açık anahtarın hash değeri ve mesaj indeksi WhiteBox kütüphanesinde AES-256 altında şifrelenmektedir. WhiteBox kütüphanesinin hash değeri hesaplanır, kütüphanede bir değişiklik olmadığını kontrol etmek amacıyla. Şifreli uygulama versiyonu MPA açık anahtarı hash değeri ve mesaj indeksi ile WhiteBox Kütüphanesi hash değeri, eşsiz MPA numarası ve doğrulama kodu sunucuya iletilir.

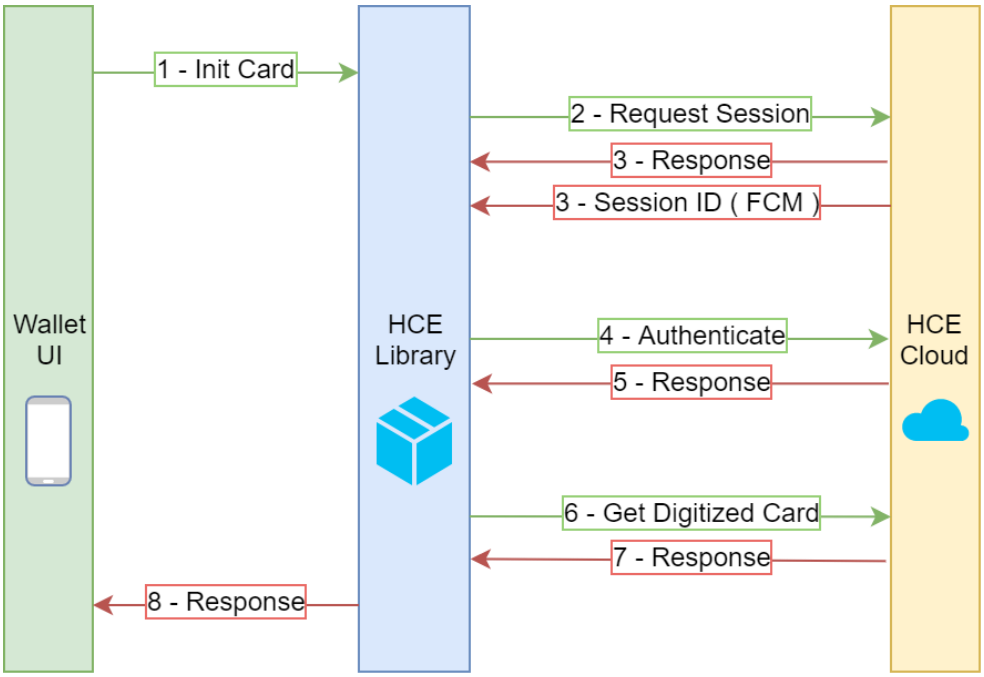
Sunucu tarafında ilk olarak, eşsiz MPA numarası için başlatılmış bir oturum var mı kontrolü yapılır, sonrasında sunucu tarafında da HSM üzerinden, MPA'ya özel olan şifreleme ve doğrulama anahtarları oluşturulur ve MPA'dan gönderilen doğrulama kodu ile HSM'de oluşturulan doğrulama kodu karşılaştırılır. Karşılaştırma sonucu olumluysa güvenli kanal

başarıyla oluşturulmuş ve doğrulama sağlanmıştır. Devamında MPA yeterlilik kontrolleri yapılır. Üretilen oturum anahtarı ile şifreli veriler HSM'de çözülür ve sonrasında mesaj indeksi, uygulamanın versiyonu, MPA açık anahtarı hash değeri ve WhiteBox kütüphanesi hash değeri karşılaştırmaları yapılır, sonuçlar olumlu ise, MPA'nın yeterliliği de doğrulanmış olur ve sonuç MPA'ya iletilir.

6. adımda, dijital kart oluşturmak için eşsiz MPA numarası ve mesaj indeksi sunucuya iletilir. İlk olarak sunucu tarafında oturum ve mesaj indeks kontrolü yapılmaktadır. Ardından dijital kart bilgileri oluşturulur ve dijital kartın işlem verileri oluşturulur. Oluşturulan dijital kartın tüm verileri oturum anahtarı ile şifrelenerek MPA'ya iletilir. MPA'da oturum anahtarı ile çözülen veriler veritabanında dijital kart tablosuna kayıt edilir.

8. adımda, dijital kart artık oluşturulmuştur ve kullanıma hazırdır. Veritabanı şifrelenerek kapatılıyor.

Bu bölümde, MPA ve sunucu arasında güvenli iletişim kuruldu, sunucu tarafında MPA doğrulandı ve dijital kart verisi oluşturuldu. MPA'da bu veriler kaydedildi ve bir kart gibi davranmaya hazır oldu.



Şekil 3.26: Dijital Kart Oluşum Adımları

3.5.4.3 Token Oluşumu

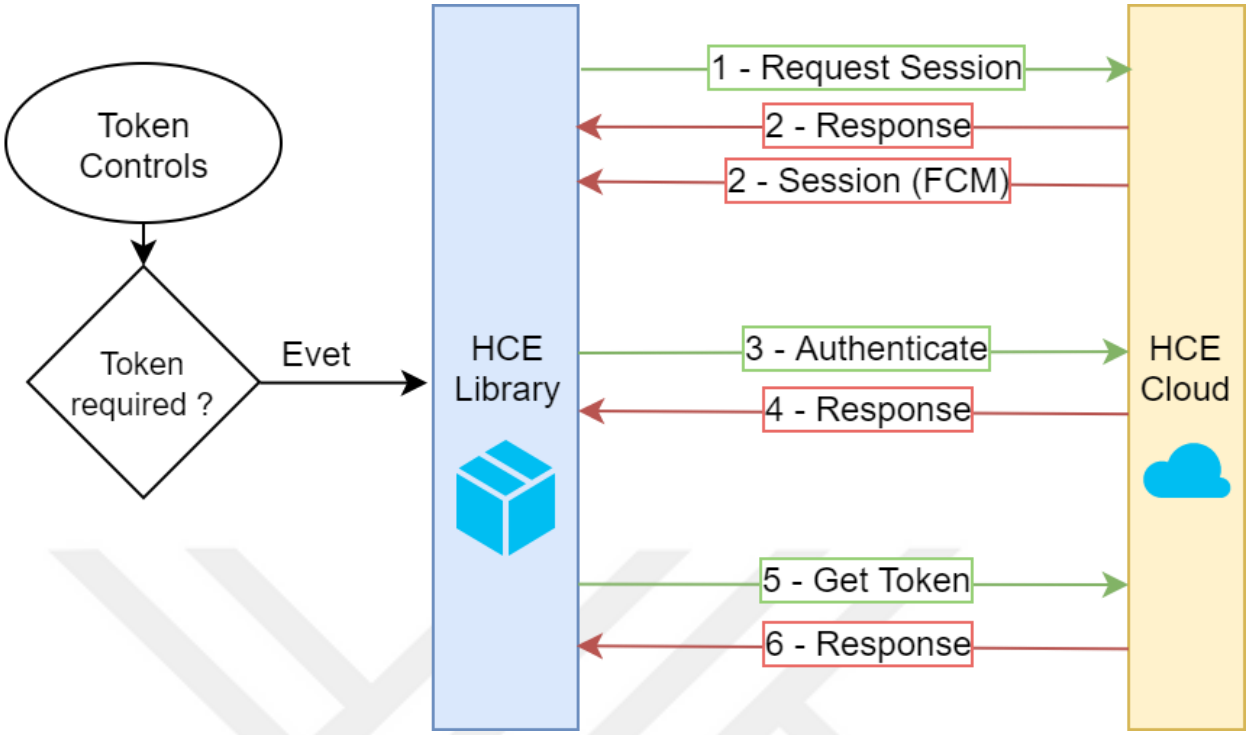
Token oluşumu ve token yönetimi, arayüzden bağımsız olarak tamamen HCE kütüphanesi tarafından yönetilir. Gerekli kontroller sonrasında token gerekliyse, sunucu ile oturum başlatılarak yeni token alınır ve MPA da kullanılmak üzere şifreli veritabanına kaydedilir. **Şekil 3.27**'de token oluşum aşamaları gösterilmiştir ve alt kısımlarda açıklanmıştır.

Token kontrollerinde, mevcutta token var mı, tokenın süresi dolmuş mu, tokenın kullanım sayacı limiti aşılmış mı veya tokenın kullanım tutarı aşılmış mı gibi kontroller sonrasında tokena ihtiyaç olup olmadığına karar verilir.

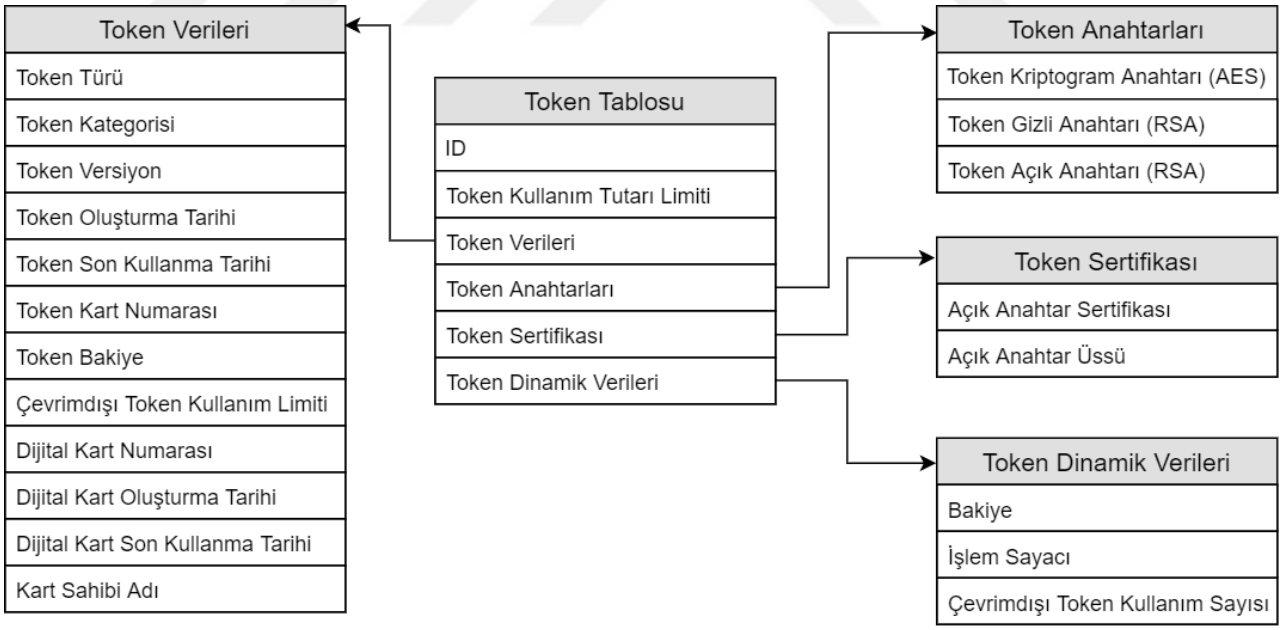
1. adımda, token ihtiyacı durumunda, ilk olarak üst başlıkta anlatıldığı gibi yeni bir oturum açma isteği gönderilir ve oturum başlatılır.

3. adımda, üst başlıkta anlatıldığı gibi MPA ile sunucu arasında kimlik doğrulaması yapılır ve başarılı sonuç sonrasında işlem devam eder.

5. adımda, sunucudan yeni bir token istemek için gerekli veriler hazırlanır. Eğer mevcutta kullam dışı olan bir token varsa harcanan ücreti doğrulamak ve kalan bakiyeyi merkez bakiyeye eklemek için, bu token'ın ID'si ve mevcut bakiyesi, genel veriler olan; oluşturulmuş dijital kartın ID'si ve mesaj indeksi oturum anahtarı ile şifrelenir ve iletilir. Sunucu tarafında şifreli veriler çözülür ve gerekli oturum kontrolleri yapılır. Yeni bir token oluşturmak için, HSM'de yeni tokena özel anahtarlar, sertifika ve veriler üretilir ve bunlar bütün olarak tokenı oluşturur. HSM'de üretilen bu tokenlar MPA'ya özel eşsiz WhiteBox anahtarı ile şifrelenir ve oturum anahtarı ile de şifrelenerek MPA'ya iletilir. MPA tarafında öncelikle veritabanındaki mevcut token silinir, sonrasında token oturum anahtarı ile çözülür ve veritabanında eşsiz WhiteBox anahtarı ile şifreli halde **Şekil 3.28**'de görüldüğü şekilde token tablosuna kaydedilir ve veritabanı şifrelenerek kapatılır.



Şekil 3.27: Token Oluşum Adımları



Şekil 3.28: Token Yapısı

3.5.4.4 HCE Silme

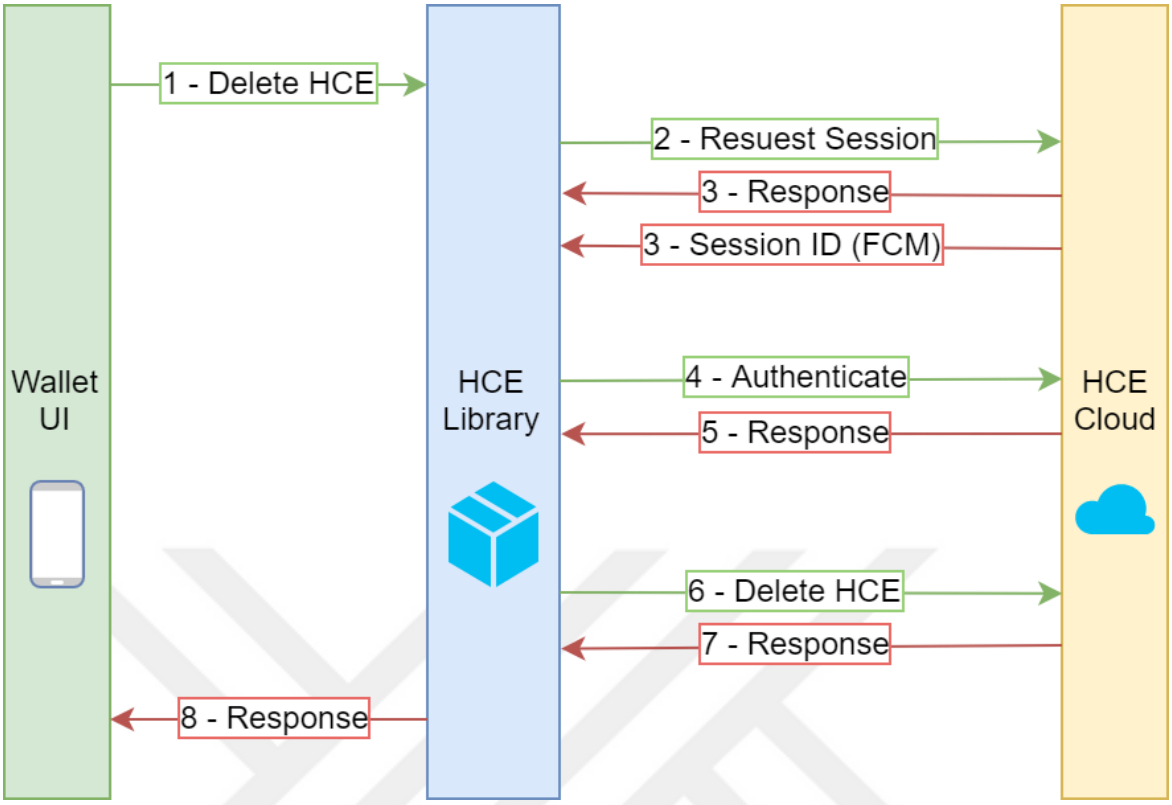
Kullanıcı tarafından veya bir saldırı tespit durumunda HCE altyapısı silinmek istendiğinde sunucu ile güvenli oturum başlatılır, kimlik doğrulaması yapılır ve sonrasında sunucudan mevcut kart ve token bilgilerinin silinmesi istenir. MPA tarafında ise, WhiteBox kütüphane tablosu ve şifreli veritabanı silinerek, hassas veriler kaldırılır ve bir daha HCE ilklendirilmesi yapılmadan MPA ile işlem yapılamaz. **Şekil 3.29**'da HCE silme aşamaları gösterilmiştir ve alt kısımlarda açıklanmıştır.

1. adımda, kullanıcı tarafından veyahut HCE kütüphanesi tarafından tespit edilen saldırı tehdidiyle HCE silme işlemi başlatılır.

2. adımda, üst başlıklarda anlatıldığı gibi oturum başlatma isteği sunucuya iletilerek, oturum başlatılır.

4. adımda, üst başlıklarda anlatıldığı şekilde, sunucu ile kimlik doğrulaması yapılır.

6. adımda, mesaj indeks, dijital kart ID'si, token ID'si ve token bakiyesi oturum anahtarı ile şifrelenerek, eşsiz MPA sayısı ile birlikte sunucuya iletilir. Sunucu tarafında oturum kontrolleri yapıldıktan sonra, token bakiyesi merkezdeki ana bakiyeye eklenerek, MPA için oluşturulmuş dijital kart ve token bilgileri sunucu tarafında deaktif edilir. MPA tarafında ise, veriler sunucuya iletildikten sonra, şifreli veritabanı ve WhiteBox için MPA'ya özel oluşturulmuş tablo cihazdan silinir.



Şekil 3.29: HCE Silme Adımları

3.5.4.5 Ödeme İşlemi

MPA da ödeme işlemleri, EMV altyapısından esinlenerek tasarlanmış olan taşıma kartı emülasyonu ile kernel arasında gerçekleşmektedir. HCE kütüphanede bulunan taşıma kartı emülasyonuna validatörlerde bulunan kernel tarafından komut gönderilir ve HCE kütüphane tarafından cevap hazırlanır ve kernele gönderilir. **Şekil 3.30**'da kernel ile HCE kütüphane arasında yapılan bilgi alışverişi gösterilmiştir ve alt kısımlarda açıklanmıştır.

1. adımda, MPA yüklü mobil cihazın validatöre yaklaşılması sonucunda, validatör tarafından yakın ödeme sistemi ortamı sorgulama komutu NFC kontrolcüsü üzerinden HCE kütüphanesine gönderilir. HCE kütüphanesi, HCE ilklendirme kontrollerini yapar ve bu komutu taşıma kartı emülasyonuna ileterek, şifreli veritabanından kart tablosuna ulaşır ve mevcutta yüklenmiş bir dijital kart var ise, bu kartın uygulama adını cevap olarak validatöre iletir.

3. adımda, ilk adımdan alınan uygulama adı ile ulaşım ödemesi gerçekleştirmek için uygulama seçme komutu kernel tarafından hazırlanır ve NFC kontrolcüsü üzerinden HCE kütüphanesine

iletilir. Kart tablosundan, sunucuda hazırlanmış olan veriler ulaşım kartı emülasyonu tarafından düzenlenir ve cevap olarak validatöre iletilir.

5. adımda, ulaşım kartı emülasyonundan kayıtları almak için, kayıt okuma komutu hazırlanır ve 7 adet kayıt ayrı ayrı komutlar gönderilerek alınır. Bu kayıtlar **Şekil 3.31**'de gösterilmiştir ve hazırlanma süreçleri şöyledir;

- Token verisi hazırlanırken, token tablosundan alınan token verisi WhiteBox kütüphanesinde eşsiz WhiteBox anahtarı ile çözülür ve validatöre iletilir. Kernel tarafında, veriler işlenerek geçiş bedeli hesaplanır ve token açık anahtar sertifikası doğrulamasında kullanılır.
- Token dinamik verisi de aynı şekilde WhiteBox kütüphanesinde çözülür ve güncel bakiye ve kullanım limiti validatöre iletilir.
- Token açık anahtar sertifikası, sunucu da token oluşumu sırasında hazırlanmış ve asimetrik RSA anahtarı olan terminal gizli anahtarı ile şifrelenmiştir. İçerisinde token açık anahtarını ve kontrol amaçlı token bilgilerini bulundurmaktadır. Kernele iletilen bu sertifika, kernelde bulunan açık anahtar ile çözülür, son kullanma tarihi, token ID'si gibi kontroller sonrası bu sertifikanın geçerli olduğunu ve bu tokena ait olduğunu doğrular ve içerisinde bulunan token açık anahtarını, token imzası geldiğinde açmak üzere saklar.
- Sertifika verileri token tablosundan alınır. Token açık anahtar üssü ve eğer token açık anahtarı token açık anahtar sertifikasına sığmamış ise, sığmayan kısmı olan anahtar kalanını içerir. Token açık anahtarının kullanımı için gerekli olan verilerdir.
- Terminal şablonu dönecektir, ulaşım kartı sağlayıcılar için ayrılmış bir bölümdür, akışı etkilemez ve HCE sistemi tarafından yönetilmez. Ulaşım sağlayıcı operatörlerin kendi verilerini yazıp okuması için ayrılmıştır. Örneğin aktarma yapmak için bilgi tutma.
- Terminal sertifikası, asimetrik RSA anahtarı olan terminal gizli anahtarını içeren, terminal açık anahtarı ile şifrelenmiş sertifikadır. Sertifikanın hazırlanması ve anahtar çiftinin üretimi ulaşım sağlayıcı operatör tarafından yapılır.
- Terminal imzası, terminal açık anahtarı ile şifrelenmiş ve içerisinde terminal şablonunun hash değerini bulunduran imzadır. Terminal tarafından, terminal şablonunun değiştirilmediğini onaylamak için bulundurulur. Terminal tarafında ulaşım sağlayıcı operatör tarafından oluşturulur.

7. adımda, validatörden MPA'ya işlem yapma isteği gönderilir. Bu istekte işlem tutarı, işlem tarihi, terminal ID, terminal tipi, terminal rastgele numarası gibi kernelde hazırlanan veri listesi gönderilir. Bu veriler işlemi kayıt altına almak, konumlandırmak ve doğrulamak için gereklidir. Şifreli kütüphanenin token tablosundan eşsiz WhiteBox anahtarı ile şifrelenmiş olan veriler WhiteBoxta çözülür ve veriler kullanıma hazırdır.

İlk olarak, işlem tutarı ve bakiye kontrol edilir, çevrimdışı token kullanım sayısı, limiti ile aşım olmuş mu kontrolü yapılır. İşlem devam etmeye uygunsa, işlemin doğru kaynaktan yapıldığını onaylatmak için uygulama kriptogramı üretilir. Ulaşım kartı uygulaması için bir adet kart rastgele sayısı üretilir, işlem sonucu ve kernelden gelen veri listesi ile kriptogram verisi oluşturulur. Eşsiz WhiteBox anahtarı altında şifreli token anahtarları ve token dinamik verileri, kriptogram verisi ile birlikte WhiteBox kütüphanesine gönderilir. WhiteBox kütüphanesi içerisinde, token anahtarları çözülür ve içerisindeki token kriptogram anahtarı ile kriptogram verisini AES-MAC işleminden geçirerek mesaj doğrulama kodunu yani uygulama kriptogramını oluşturur. Yine eşsiz WhiteBox anahtarı ile token dinamik verilerini açarak, içerisinde bulunan işlem sayacını ve çevrimdışı token kullanım sayısını 1 arttırır, ardından bakiyeden, işlem tutarını düşer ve yeni oluşturulmuş token dinamik verisini eşsiz WhiteBox anahtarı altında şifreler. Uygulama kriptogramı, şifreli ve açık token dinamik verisi, ulaşım kartı uygulamasına gönderilir.

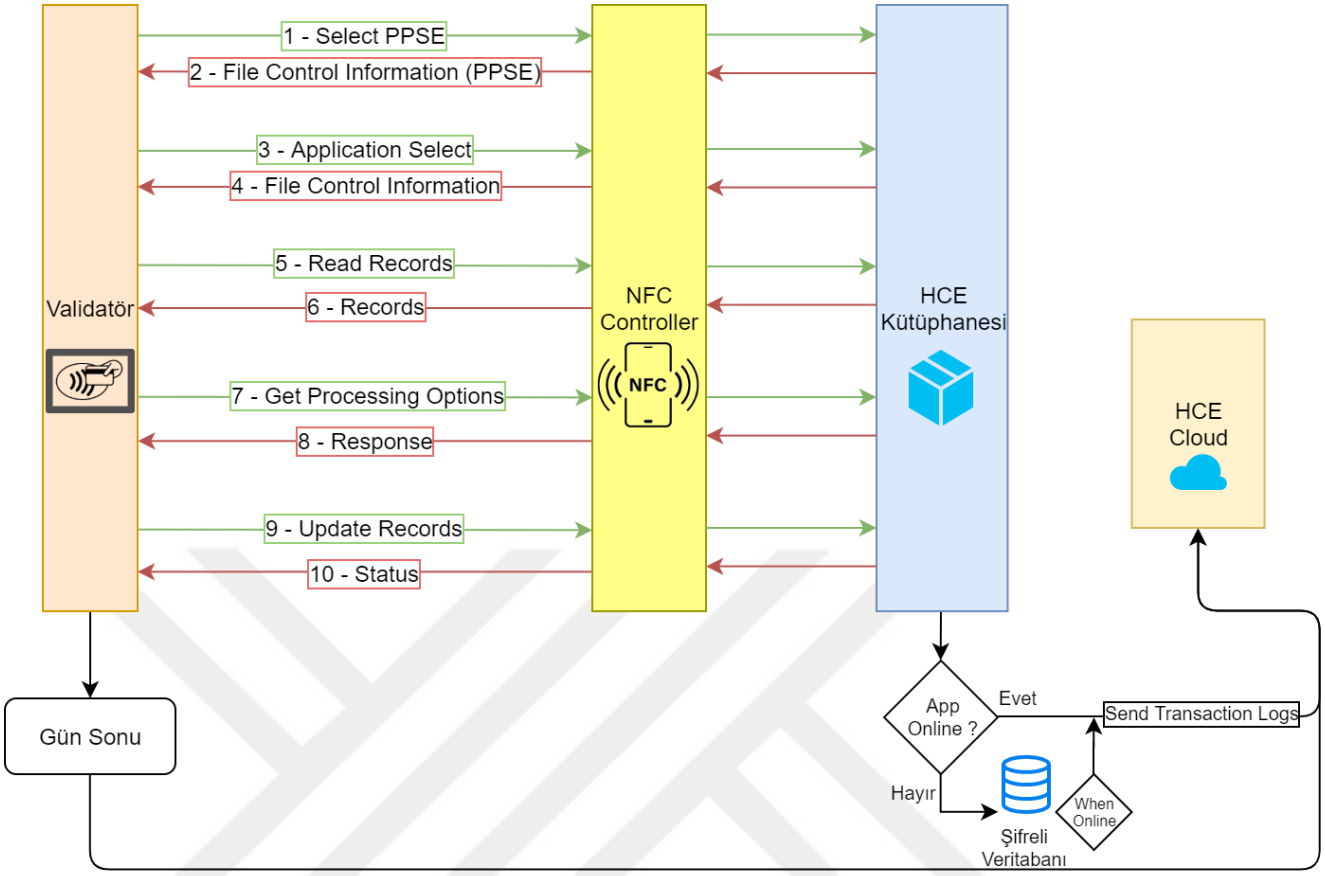
Asimetrik doğrulama yapmak için, uygulamanın dinamik imzası oluşturulur. Öncelikle veri listesinin hash değeri, kart rastgele sayısı, işlem sayacı, işlem sonucu ve uygulama kriptogramı ile uygulama dinamik verisi oluşturulur ve EMV Book 2'nin 6.5.1 Dynamic Signature Generation başlığındaki standartlara ve adımlara uyarak, dinamik imza verisi oluşturulur. Oluşturulan bu veri, şifreli token anahtarı ile WhiteBox kütüphanesine gönderilir ve eşsiz WhiteBox anahtarı ile token anahtarı çözülerek, asimetrik RSA anahtarı olan token gizli anahtarı elde edilir. Dinamik imza verisi, token gizli anahtarı ile şifrelenir ve ulaşım kartının dinamik imzası oluşturulur.

İşlem sonucu, uygulama kriptogramı ve dinamik imza oluşturulduğunda, ulaşım kartı uygulamasında işlem yapma komutu için gerekli veriler ve sonuçlar hesaplanmış olmaktadır. Güncel bakiye, güncel işlem sayacı, uygulama kriptogramı, işlem sonucu, kart rastgele sayısı ve dinamik imza validatöre cevap olarak iletilir.

Kernel tarafında, ilk olarak 5. adımdaki kayıt okuma esnasında token açık anahtar sertifikası içerisinden elde edilen token açık anahtarı ile ulaşım kartı uygulamasından dönen dinamik imza açılır ve içerisindeki formatı ve hash değerini kontrol ederek imzanın doğruluğunu kontrol eder. Ardından komut sonucu dönen uygulama kriptogramı ile dinamik imzadaki uygulama kriptogramını kontrol ederek, kriptogramın değiştirilmediğini doğrular. Diğer bilgilerle ise validatör ekranında kullanıcıya bildireceği verileri hazırlar.

9. adımda, artık kernel tarafından işlem onaylanmış ise, ulaşım kartı operatörü için ayrılan 5. 6. ve 7. kayıtların güncellenmesi beklenir. Mevcut işlemin bilgileri hazırlanır ve imzası oluşturularak kayıt güncelleme komutu ile gönderilerek ulaşım kartı uygulaması tarafından kayıtlar güncellenir.

Ulaşım işlemi bittikten sonra, MPA tarafında, eğer internet erişimi varsa, işlem kayıtları sunucuya kayıt amaçlı gönderilir. Eğer internet bağlantısı mevcut değilse, şifreli veritabanına kaydedilerek, internet erişimi sağlandığında işlem kayıtları sunucuya gönderilir. Validatör tarafında ise, gün sonu yapılacağı zaman her işlem için alınan kayıtlar sunucuya gönderilir. Sunucu tarafında, ulaşım kartından dönen uygulama kriptogramı HSM'de de oluşturularak işlemin doğru kaynaktan yapıldığı doğrulanır. Kriptogram doğrulanmaz ise, işlemi yapan MPA uygulaması kara listeye alınır ve MPA'da internet bağlantısı kurulduğu anda tüm HCE kayıtları, WhiteBox kütüphanesi ve şifreli veritabanı silinerek işlem yapması engellenir.



Şekil 3.30: Ödeme İşlemi Adımları

Kayıtlar
1 - Token Verileri
2 - Token Dinamik Verileri
3 - Token Açık Anahtar Sertifikası
4 - Sertifika Verileri
5 - Terminal Şablonu
6 - Terminal Sertifikası
7 - Terminal İmzası

Şekil 3.31: Ödeme İşlemi Adımları

4. BULGULAR

4.1 GÜVENLİK ANALİZİ

Ödeme sistemlerindeki en ufak bir güvenlik açığından, büyük finansal kayıplar gerçekleşebilir. Bundan ötürü, sistemin güvenliği çok büyük önem teşkil etmektedir. MPA'da güvenli işlem akışı, veri depolama ve iletişim mevcuttur. Bu kısımda, sistemin karşılaşabileceği fiziksel ve mantıksal ataklara karşı alınan önlemlerden bahsedilmiştir.

4.1.1 Donanım Saldırı Direnci

Tasarlanmış bu HCE sisteminde, özel WhiteBox anahtarına erişim MPA'da ve HCE Bulut'ta sağlanmaktadır.

MPA'da, sadece WhiteBox kütüphanesi üzerinden erişim sağlanabilen bu anahtar, bölüm 3.3 ve 3.5'te nasıl korunduğuna ve erişildiğine dair detaylıca anlatılmıştır. Ayrıca bölüm 3.1.4'te açıklanan OWASP'ın mobil platform güvenlik çalışmasıyla da MPA güvenliği güçlendirilmiştir.

HCE Bulut'ta, WhiteBox anahtarı HSM cihazında saklanarak anahtarın gizliliği sağlanmaktadır.

4.1.2 İletişim Kanalı Saldırı Direnci

Şekil 3.22'de gösterildiği üzere, MPA - HCE Bulut İletişimi, MPA – Validatör İletişimi ve Operatör – HCE Bulut İletişimi olmak üzere 3 iletişim kanalı mevcuttur.

4.1.2.1 MPA - HCE Bulut İletişimi

İletişim, SSL koruması ile ortadaki adam saldırısına karşı korunmaktadır. Her oturum için özel oluşturulan oturum anahtarı ile bütün veriler şifrelenmektedir. Kritik ve hassas veriler ayrıca MPA'ya özel WhiteBox anahtarı ile de şifrelenmektedir. Google Firebase ve güvenli kanal iletişimi sağlandığından ötürü iletişim güvenli kabul edilmiştir, farklı güvenlik katmanları ile de güçlendirilebilinir.

4.1.2.2 MPA - Validatör İletişimi

MPA, NFC kontrolcüsü aracılığıyla validatör ile iletişimi sağlamaktadır. Bu iletişim esnasında güvenlik MPA'da sağlanır ve terminal tarafından çevrimiçi ve çevrimdışı olarak doğrulanır. Çevrimdışı doğrulamada, MPA'da oluşturulan dinamik imza kontrol edilir. Çevrimiçi doğrulamada ise, MPA'da oluşturulan uygulama kriptogramı HCE sunucusuna gönderilerek doğrulanır. Her işlem için WBC'de rastgele bir sayı üretilir ve bu sayı uygulama kriptogramına ve dinamik imzaya dahil edilir. Ayrıca her işlem için, farklı bir oturum anahtarı oluşturulur.

4.1.2.3 Operatör - HCE Bulut İletişimi

Operatör ve HCE Bulut arasında güvenli kanal oluşturulur ve bu kanal üzerinden iletişim sağlanır.

4.1.3 Mantıksal Saldırılar

Mantıksal saldırılar, sisteme zarar verebilecek açıklıklar veya zafiyetler barındıran güvenlik açıklarından faydalanan saldırılardır. Bu yöntem ile saldırganlar sistemin çalışma performansını etkileyebilir, iletişim yapısına zarar verebilir. Saldırganlar tarafından sıkça tercih edilen bir yöntemdir.

4.1.3.1 Güvenilir Olmayan Mobil Ödeme Uygulaması

MPA, HCE servise ile yaptığı uçtan uca iletişimdeki paylaşılan bilgilerin gizlilik ve bütünlüğünden sorumludur. Güvenilir olmayan bir MPA'da bu bilgiler saldırgan tarafından değiştirilerek, sistemde zafiyet oluşturulabilir. Bu yüzden güvenli WBC kütüphanesinde saklanan WB Anahtarı ve oturum anahtarları ile giden gelen bütün veriler şifrelenir. MPA'daki HCE kütüphanesi, yine güvenli WBC kütüphanesinde saklanan WB anahtarı ile her yapılan her işlemde şifreli tutulan veriler çözülür. Bu veriler arasında Token Gizli Anahtarı'da bulunmaktadır. Token Gizli Anahtarı ile dinamik imza oluşturulur ve validatörde bulunan Sertifika Otoritesi Açık Anahtarıyla, HCE'den terminale gönderilen ihraççı sertifikası çözülür ve Token Açık Anahtarı elde edilerek, bu anahtar ile de dinamik imza doğrulanır. Böylece, asimetrik anahtarla sağlanan doğrulamadan geçen işlemler, HCE serviste uygulama kriptogramı da doğrulayarak işlemin doğru kaynaktan yapıldığını tespit eder. Her token, tokena özel türetilmiş farklı anahtar kullanır. Buda anahtarın sızdırılmasına karşı alınan bir önlemdir.

4.1.3.2 Güvenilir Olmayan Validatör

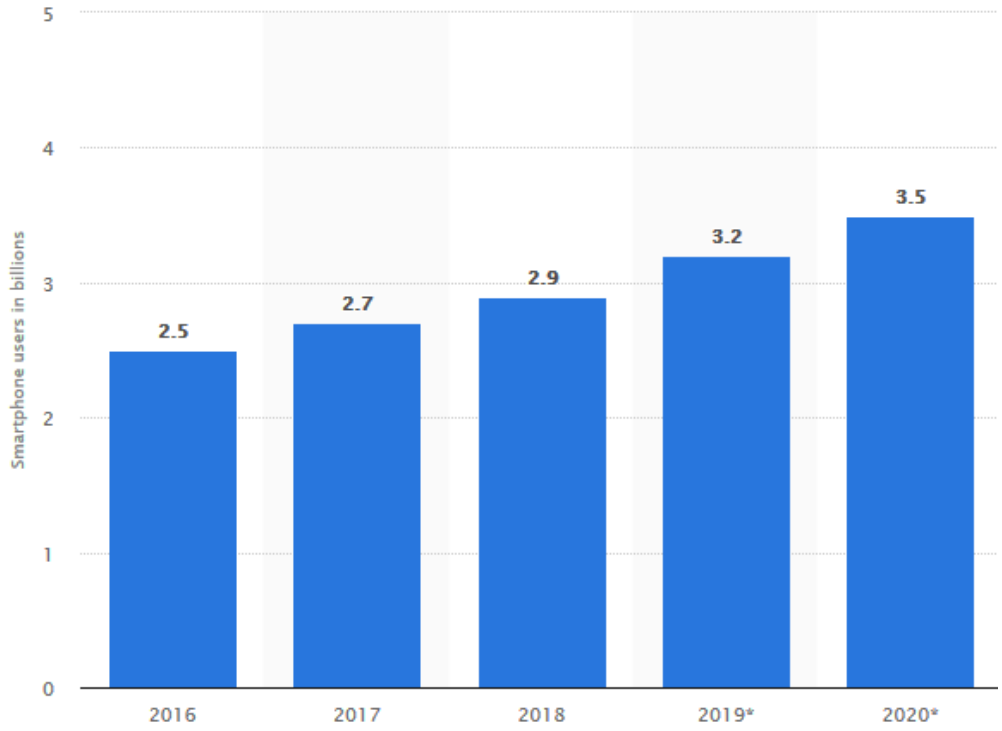
Güvenilir olmayan validatörde Sertifika Otoritesi Açık Anahtarı bulunamayacağı için yapılan işlem doğrulanamayacaktır. Ayrıca, yapılan işlem HCE buluta iletilmeyeceği için, kayıtlara geçmeyecektir ve bir geçerliliği olmayacaktır.

4.2 PERFORMANS ANALİZİ

İşlem performansı, özellikle de kullanımın yoğun olduğu bölgelerde uzun işlem süresi ve bundan ötürü de uzun kuyruklara neden olabileceğinden ötürü kritiktir. Bu yüzden, yapılan işlemin çok hızlı olması lazımdır. Mevcuttaki çalışan RFID kartlara göre tek fark, validatör uygulamasında ödeme aracını tanımlayan kısma bir sorgu ekleyerek RFID kart mı yoksa HCE uygulaması mı olduğuna karar vermektir. Verilen karar neticesinde, HCE uygulama ile işlem yapılmak isteniyorsa, validatörde bulunan HCE kütüphanesi devreye girecek ve işlem gerçekleşecektir. Ödeme aracından ilk olarak uygulamanın AID'si istenir ve gelen AID, HCE uygulamaya ait ise HCE kütüphanesine, RFID karta ise mevcut kütüphaneye yönlendirilir. Akıllı telefon pazarında bulunan düşük – orta segment telefonlarla yapılan işlemlerde işlem süresi 350-400 ms sürmektedir. Yüksek segment telefonlarda ise bu süre 250 ms'e kadar düşmektedir.

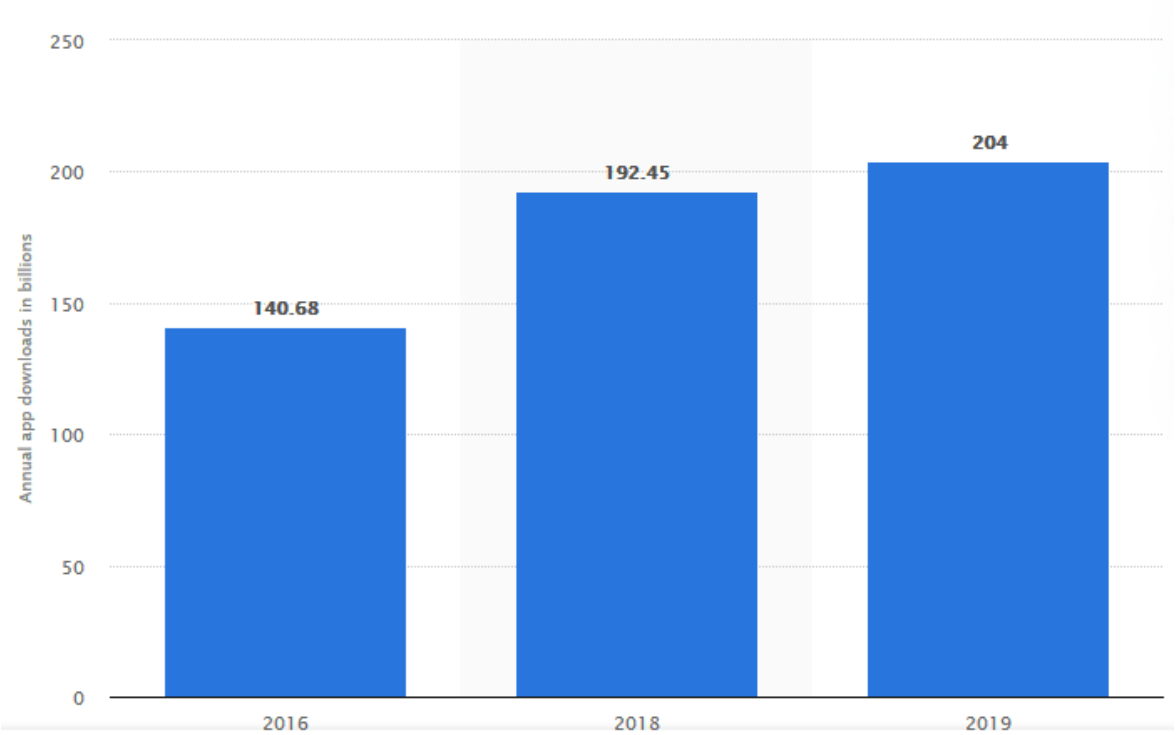
5. TARTIŞMA VE SONUÇ

Akıllı telefonlardaki donanımların gelişmesi ve standart özelliklere sahip telefonların fiyatlarının karşılanabilir olmasından ötürü akıllı telefon kullanımı dünya çapında arttı. **Şekil 5.1**'de dünya genelinde akıllı telefon kullanıcı sayısı görülmektedir. Yıllara göre artan kullanıcı sayısı ile artık kolay ulaşılabilir bir cihaz olduğu görülmektedir.



Şekil 5.1: Dünya Geneli Akıllı Telefon Kullanıcı Sayısı [33]

Akıllı telefon kullanıcıları tarafından dünya geneli mobil uygulama indirme grafiği **Şekil 5.2**'de görülmektedir. Bu grafikten anlaşılacağı üzere, kişi başına indirilen ortalama uygulama sayısı 68'tir, yani akıllı telefon kullanıcıları mobil cihazlarında güncel uygulamaları kullanmak istemektedir.



Şekil 5.2: 2016–2019 Yıllarında Dünya Genelinde Mobil Uygulama İndirme Sayısı [34]

Son yıllarda mobil uygulamalardan çeşitli işlemler yapabilmek büyük popülerlik kazanmaktadır. Mobil cihazlarda bulunan NFC teknolojisi ile de finansal işlemler yapılabilmeye başlandı. Dünya çapında milyonlarca kullanıcı tarafından NFC donanımlı akıllı telefonlarla mobil ödeme yapılmaktadır. Mobil ödeme, sadece POS cihazlarıyla bir şey satın alırken değil, ulaşım alanında da turnikelerde bulunan terminaller aracılığıyla da yapılabilir. Bu hedefe, önceki bölümlerde detaylı bir şekilde açıklandığı gibi çevrimdışı ödeme şeması için ulaşılmıştır.

Akıllı telefon kullanıcıları tarafından, mobil cihazlar aktif olarak kullanılmakta ve kullanılmak istenmektedir. Bilgisayarlar sayesinde masada bulunması gereken aletler, sanal masaüstünde bulunmaya başladı. Bu sayede dünyada gereksiz hammadde tüketimi engellendi ve kolay erişim sağlandı. RFID kartların da cüzdanlarda yer kaplaması kullanıcı açısından bir yük olmakta ayrıca kaybolma, kırılma, çalınma ve kopyalanma risklerini taşımaktadır.

Ulaşım sağlayıcı operatörler açısından ise RFID kartlar için satın alma, kişiselleştirme, baskılama, kullanıcıya ulaştırma gibi maliyetleri mevcuttur. Ayrıca bu kartlar

değiştirilemeyeceği için, kişisel veri güncellenmek istendiğinde yeni bir RFID kart gerekmektedir.

HCE teknolojisinin kullanımı, ulaşım sağlayıcı firmalar için kart maliyetinin çıkarılıp, kolayca güncellenebilir ve takip edilebilir dijital kart imkanı sunmaktadır. En büyük sakınca olan HCE teknolojisinde veri ve iletişim güvenliği, bu tez kapsamında incelenmiştir ve bir çözüm sunulmuştur.

Bu sayede, ulaşım sağlayıcı operatörler kart masraflarından kurtulup, daha kolay yönetip takip edebileceği dijital kartlar oluştururken, kullanıcılar ise akıllı telefonlarını satın alırken ücretini de ödediği NFC donanımını kullanabilecek, cüzdanlarında kart taşıma zorunluluğu kalkacak ve hesaplarını daha yakından takip edebilecekler.

Bu çalışmada, HCE teknolojisi ile çevrimdışı güvenli bir mobil ödeme işlemiyle, ulaşım alanında hizmetlerden faydalanabilmek için kart taşıma zorunluluğu sorununu çözmektedir.

Bu tez kapsamında açıklanan, çevrimdışı güvenli HCE ödeme sistemi tasarımı ile mobil cihazlardan çevrimdışı yapılacak olan ulaşım ödemelerinde güvenli bir altyapı kurmayı amaçlamıştır. Bu kapsamda;

- WhiteBox kütüphanesi ile mobil cihazlarda kritik veri korunumu,
- OWASP mobil platform güvenlik çalışmalarına uyumluluk,
- PCI DSS kurallarının sağlanması,
- Uzak sunucu ile güvenli iletişim kanalı kurulumu,
- HSM ile anahtar gizliliğinin sağlanması,

Çalışmaları tamamlanarak, günümüz teknolojisi ve standartlarınca kabul görmüş güvenlik altyapıları bu HCE sistem tasarımda kullanılarak güvenli bir HCE ekosistemi oluşturulmuştur.

KAYNAKLAR

[1] Motlagh, H.N., 2012, *Near Field Communication - a technical overview,* Yüksek Lisans Tezi, University of VAASA

[2] ECMA, 2013, *ECMA 340: Near Field Communication Interface and Protocol 1 (NFCIP-1),* ECMA International, http://www.ecma-international.org/publications/standards/Ecma-340.htm (Ziyaret 16 Nisan 2020)

[3] Coskun, V., Ok, K., Ozdenizci, B**.**, 2013, *Professional NFC Application Development for Android*,1st ed.; John Wiley Sons, Wrox: London, UK

[4] Coskun, V., 2015, *The Survey of the Near Field Communication*, Sensors, 15.

[5] Urien, P. 2013, *LLCPS: a new security framework based on TLS for NFC P2P applications in the Internet of things.* In Proceedings of the IEEE consumer communications and networking conference (CCNC 2013). Las Vegas: IEEE, 845–846.

[6] Roland M., Langer OVE J., 2013, *Comparison of the usability and security of NFC's different operating modes in mobile devices*. In Elektrotechnik & Informationstechnik 2013. page 202

[7] Mobey Forum (2014). *The Host Card Emulation in Payments: Options for Financial Institutions.* White Paper, 25 p.page 5 , https://www.mobeyforum.org/the-host-card-emulation-in-payments-options-for-financial-institutions-3/ , (Ziyaret 16 Nisan 2020)

[8] Coskun, V., Ok, K.; Ozdenizci, B., 2016, *A Tokenization-Based Communication Architecture for HCE-Enabled NFC Services,* Hindawi Publishing CorporationMobile Information Systems, Article ID:5046284, 1-20.

[9] Coskun, V., Ok, K.; Ozdenizci, B., 2016, *A cloud based framework for HCE and NFC services,*64th The IIER International Conference, Barcelona, SPAIN, 11-14, http://www.worldresearchlibrary.org/up_proc/pdf/212-145767718111-14.pdf (Ziyaret:16 Nisan 2020)

[10] Mobey Forum, 2014, *The Host Card Emulation in Payments: Options for Financial Institutions.* White Paper, 25 p. page 13, https://www.mobeyforum.org/the-host-card-emulation-in-payments-options-for-financial-institutions-3/ (Ziyaret 16 Nisan 2020)

[11] BMO Bank of Montreal, CIBC, National Bank of Canada, RBC Royal Bank, Scotiabank, TD Bank Group, 2015, *Payments Security White Paper*, 14-17, https://cba.ca/Assets/CBA/Documents/Files/Article%20Category/PDF/misc-2015-paymentssecurity-whitepaper-en.pdf (Ziyaret 16 Nisan 2020)

[12] OWASP, 2015, *Mobile Top 10 Risks* , https://owasp.org/www-project-mobile-top-10/ (Ziyaret 16 Nisan 2020)

[13] Schaefer, F., E., 1996, *A SIMPLIFIED DATA ENCRYPTION STANDARD ALGORITHM, Cryptologia*, 20:1,77-84.

[14] Sharma, A., Mitali, Kumar, V., 2014, *A Survey on Various Cryptography Techniques,* International Journal of Emerging Trends & Technology in Computer Science (IJETTCS), Vol:3, Issue:4, 307-312.

[15] Şeker Ş. E., 2008, *DES (Veri Şifreleme Standardı, Data Encryption Standard)*, http://bilgisayarkavramlari.sadievrenseker.com/2008/03/13/des-veri-sifreleme-standardi-data-encryption-standard/ (Ziyaret: 10 Mart 2020)

[16] Daemen, J., Rijmen, V., 1999**,** *AES Proposal: Rijndael,* https://csrc.nist.gov/csrc/media/projects/cryptographic-standards-and-guidelines/documents/aes-development/rijndael-ammended.pdf (Ziyaret:16 Nisan 2020)

[17] Yerlikaya, T., Gençoğlu, H., Emir, M. K., Çankaya, M. ve Buluş, E., 2013, *RSA Şifreleme Algoritması ve Aritmetik Modül Uygulaması*. İstanbul Aydın Üniversitesi Dergisi, 3 (9), 95-104.

[18] Hatun, E., 2018, *Raspberry Pi Üzerinde Gerçeklenmiş RSA Algoritmasına Yan Kanal Analizi,* Yüksek Lisans Tez, İstanbul Teknik Üniversitesi

[19] Rivest, R. L., Shamir, A. & Adleman, L., 1978, *A method for obtaining digital signatures and public-key cryptosystems*, Communications of the ACM, 21 (2), 120-126.

[20] Do, A., Ko, S. T. & Htet, A. T., 2013, *Electromagnetic side-channel analysis on Intel Atom processor.* (A major qualifying project report). Worcester Polytechnic Institute, USA.

[21] Aksuoğlu, A., 2010, *RSA Algoritmasının İyileştirilmesi için Yeni bir Yaklaşım,* Yüksek Lisans Tezi, Anadolu Üniversitesi, Fen Bilimleri Enstitüsü, Eskişehir.

[22] EMVCo, 2011, EMV Integrated Circuit Card Specifications for Payment Systems Book 2 Security and Key Management, https://www.emvco.com/wp-content/uploads/2017/05/EMV_v4.3_Book_2_Security_and_Key_Management_20120607061923900.pdf (Ziyaret:16 Nisan 2020) 126

[23] Chow, S., Eisen, P., Johnson, H. and Oorschot, P., 2002, *White-Box Cryptography and an AES Implementation*, 250–270

[24] Chow, S., Eisen, P., Johnson, H. and Oorschot, P., 2002. *A White-Box DES Implementation for DRM Applications,* 1–15

[25] Payment Card Industry, 2014, *PIN Security Requirements*, https://www.pcisecuritystandards.org/documents/PCI_PIN_Security_Requirements_v2.pdf (Ziyaret:16 Nisan 2020)

[26] Kaur, R., Li, Y., Iqbal, J., Gonzalez, H., & Stakhanova, N., 2018, *A Security Assessment of HCE-NFC Enabled E-Wallet Banking Android Apps*

[27] Lee, H., Hong, W.-C., Kao, C.-H., & Cheng, C.-M., 2014, *A User-Friendly Authentication Solution Using NFC Card Emulation on Android*

[28] Pasquet, M., & Gerbaix, S., 2016, *Fraud on host card emulation architecture: Is it possible to fraud a payment transaction realized by a mobile phone using an "Host Card Emulation" system of security ? Invited paper.*

[29] Alattar, M., & Achemlal, M., 2014, *Host-Based Card Emulation: Development, Security, and Ecosystem Impact Analysis.*

[30] Umar, A., Mayes, K., & Markantonakis, K., 2015, *Performance variation in host-based card emulation compared to a hardware security element.*

[31] Umar, A., & Mayes, K., 2017, *Trusted Execution Environment and Host Card Emulation.*

[32] Mavrovouniotis, S., & Ganley, M., 2013, *Hardware Security Modules. Secure Smart Embedded Devices, Platforms and Applications.*

[33] O'Dea S., 2020, *Smartphone users worldwide 2016-2021* , https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/ (Ziyaret: 15 Nisan 2020)

[34] Clement J., 2020, *Annual number of global mobile app downloads 2016-2019,* https://www.statista.com/statistics/271644/worldwide-free-and-paid-mobile-app-store-downloads/ (Ziyaret: 15 Nisan 2020)

ÖZGEÇMİŞ



Kişisel Bilgiler	
Adı Soyadı	Ahmet Zahit ÖZDEMİR
Doğum Yeri	Sakarya
Doğum Tarihi	22.02.1994
Uyruğu	☒ T.C. ❒ Diğer:
Telefon	0543 521 45 87
E-Posta Adresi	ahmet-ozde@hotmail.com
Web Adresi	

Eğitim Bilgileri	
Lisans	
Üniversite	İstanbul Ticaret Üniversitesi
Fakülte	Mühendislik ve Tasarım Fakültesi
Bölümü	Bilgisayar Mühendisliği
Mezuniyet Yılı	2016

Yüksek Lisans	
Üniversite	İstanbul Üniversitesi-Cerrahpaşa
Enstitü Adı	Lisansüstü Eğitim Enstitüsü
Anabilim Dalı	Bilgisayar Mühendisliği Anabilim Dalı
Programı	Bilgisayar Mühendisliği Tezli Yüksek Lisans Programı

Makale ve Bildiriler