MARMARA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BLOKZİNCİR TABANLI DONANIMSAL CÜZDAN VE AKILLI KARTLAR

ADİL TANRIKULU

YÜKSEK LİSANS TEZİ
Bilgi Güvenliği Mühendisliği Bölümü

DANIŞMAN
Dr. HÜSEYİN YÜCE

EŞ-DANIŞMAN
Dr. ERCAN ÖLÇER

İSTANBUL, 2021

MARMARA UNIVERSITY
INSTITUTE FOR GRADUATE STUDIES
IN PURE AND APPLIED SCIENCES

HARDWARE WALLETS AND SMART CARDS FOR BLOCKCHAIN PLATFORMS

ADİL TANRIKULU

MASTER THESIS

Department of Information Security Engineering

Thesis Supervisor
Dr. HÜSEYİN YÜCE

Thesis CO- Supervisor
Dr. ERCAN ÖLÇER

ISTANBUL, 2021

Blokzincir Tabanlı Donanımsal Cüzdan ve Akıllı Kartlar

Adil Tanrikulu

ÖZET

Bu tez çalışmasında, odak nokta olan hiyerarşik deterministik donanımsal cüzdanın işleyişi, kullandığı algoritmalar ve yöntemler derinlemesine incelenmiştir. İncelenen konular doğrultusunda örnek kod parçaları yazılarak kavramın ispatı sağlanmıştır ve örnek bitcoin transferi yoluyla işleyiş somutlaştırılmıştır.

Konuya ilk olarak paranın tarihsel serüveniyle kripto paraya dönüşümünün anlatıldığı bir girişle başlanmıştır. Çünkü son yüzyılda hızla değişen teknoloji değerli varlık olan paranın biçimini radikal olarak dijitalleşmeye doğru itmiş ve her geçen gün popülerliği daha da artan blokzincir tabanlı kripto paralar, birçok platformda ödeme aracı olarak kullanılmaya başlanmıştır.

Kripto paralar merkezi banka sistemlerinin tersine blokzincir denilen merkezi olmayan bir sistem üzerinde çalışır. Bu çalışmada genel sistemin anlaşılması adına blokzincir yapısının temel bileşenlerine detaylı şekilde yer verilmiştir.

Kripto para transferi için gönderen ve alıcıya ait ortak anahtar denilen adreslerin olması gereklidir. Göndericinin adresi, alıcının adresi ve gönderim miktarı gibi bilgilerin olduğu işlemler, birbiri ardına bağlı blokların içerisinde belli bir sisteme göre tutulur. Her adresin yani ortak anahtarın benzersiz gizli bir anahtar karşılığı bulunmaktadır. kripto para işlemleri için bu özel anahtarlar ile atılmış imzalar kullanılır. Mahremiyet söz konusu olduğu için kripto para transferinde birden fazla adres yani anahtar çifti kullanmak önemlidir ve önerilmektedir. Bu anahtarların yönetimi ve depolanması cüzdanlar sayesinde sağlanır. Çevrimdışı cüzdanların direkt olarak internet erişimi yoktur ve daha güvenlidir. Özetlemek gerekirse bu tezde çevrimdışı cüzdan çeşidi olan blokzincir tabanlı kripto paraların erişimine ve harcanabilmesine olanak sağlayan anahtarların tutulduğu donanımsal akıllı kart cüzdan konusu işlenmiştir. Anahtarın üretimi ve depolanması için hiyerarşik ve deterministik bir yöntem sunan BIP-32, BIP-39 ve BIP-44'e bu çalışmada detaylı olarak yer verilmiştir.

Yerli ve milli akıllı kart işletim sistemi olan AKİS'in yeni sürümünde kimlik, pasaport, ehliyet, e-imza, bilet gibi uygulamalar tek bir kart içinde toplanabilmektedir. Bu sürümle birlikte kartın içinde olan uygulamalardan biri de cüzdan uygulamasıdır. Bitcoin kripto para transfer işlemlerini yapabilecek özellikte tasarlanmıştır. AKİS cüzdan BIP-32, BIP-39 ve BIP-44 ve eliptik eğri kriptografik işlemlerini yapabilecek şekilde geliştirilmektedir.

Anahtar Sözcükler: Blokzincir, Hiyerarşik Deterministik Donanımsal Cüzdan, Akıllı Kart, AKİS

Hardware Wallets and Smart Cards for Blockchain Platforms

Adil Tanrikulu

ABSTRACT

In this thesis study, the mechanism of the hierarchical deterministic hardware wallet, the algorithms and methods used, are examined in depth. The proof of the concept was provided by writing sample code pieces in line with the topics examined and the operation is embodied through sample bitcoin transfer.

Initially, the subject started with an introduction to the historical adventure of money and its transformation into cryptocurrency. The rapid change in technology in the last century has pushed the form of money, which is a valuable asset, towards radically digitalization, and recently blockchain-based cryptocurrencies, which are becoming more and more popular, are being used as payment instruments in many platforms.

Cryptocurrencies operate on a decentralized system called blockchain, unlike central bank systems. In this study, basic components of blockchain structure are given in detail in order to understand the general system.

For the transfer to the cryptocurrency, the addresses of the sender and receiver must be present. Transactions with information such as the sender's address, receiver's address and amount are kept in blocks which are connected one after another according to a certain system. Each address, the public key, has a unique private key equivalent. Signatures with these private keys are used for cryptocurrency transactions. Since privacy is in question, it is important and recommended to use more than one address, ie key pairs in cryptocurrency transfer. The management and storage of these keys is provided by wallets. Offline wallets do not have direct internet access and are more secure. To summarize, in this thesis, the subject of hardware smart card wallet, which holds keys that allow access and spent of blockchain-based cryptocurrencies, is covered. BIP-32, BIP-39 and BIP-44, which provide a hierarchical and deterministic method for the production and storage of the key, are detailed in this study.

In the new version of AKiS, the national smart card operating system, applications such as identity, passport, driver's license, e-signature and ticket can be collected in a single card. With this version, one of the applications included in the card is the wallet application. It is designed to handle Bitcoin transfer. AKIS wallet is developed to be able to perform BIP-32, BIP-39 and BIP-44 and elliptic curve cryptographic transactions.

Keywords: Blockchain, Hierarchical Deterministic Hardware Wallet, Smart Card, AKIS

Teşekkür

Bilgi ve deneyimleriyle tezin tamalanmasında bana yön veren tez danışmanım Sayın Dr. Hüseyin YÜCE'ye teşekkürlerimi sunarım.

Bu uzun tez sürecinde katkılarıyla tezimi bitirmeme olanak tanıyan değerli hocam Dr. Ercan ÖLÇER'e teşekkürü bir borç bilirim.

Tezimin her aşamasında büyük fedakarlık göstererek desteğini hiç esirgemeyen biricik eşim Filiz TANRIKULU'na teşekkür ederim, iyiki varsın.

İçindekiler

Yazarlık Beyanı **ii**

Özet **iii**

Abstract **v**

Teşekkür **vi**

Şekil Listesi **ix**

Tablo Listesi **x**

Kısaltmalar **xi**

1 Giriş **1**
1.1 İlgili Çalışmalar 4
1.2 Amaç 6
1.3 Akıllı Kartlar Hakkında 6

2 Blokzincir **7**
2.1 Blokzincir Çalışma Mantığı Hakkında 9
2.1.1 Blok 9
2.1.2 Eliptik Eğri Dijital İmza Algoritması (ECDSA) 13
2.1.2.1 İmzalama 14
2.1.2.2 İmzalama Doğrulama 14
2.1.3 Fikir birliği ve Madencilik 15
2.1.4 İşlemler 16

3 Blokzincir Tabanlı Cüzdanlar **19**
3.1 Masaüstü Cüzdanlar 20
3.2 Mobil Cüzdanlar 20
3.3 Web Cüzdanlar 21
3.4 Kağıt Cüzdanlar 21
3.5 Donanımsal Cüzdanlar 21
3.5.1 Deterministik Olmayan Cüzdan 23
3.5.2 Deterministik Cüzdan 23
3.5.2.1 Sıralı Deterministik Cüzdan 24
3.5.2.2 Hiyerarşik Deterministik Cüzdan 24

4 Blokzincir Tabanlı Donanımsal HD Cüzdan Uygulaması 26
4.1 HD Cüzdanın Çalışma Mekanizması . . . 26
4.1.1 BIP-39 . . . 26
4.1.2 BIP-32 . . . 29
4.1.2.1 Temel Anahtarın Oluşturulması (Master Key Derivation) 32
4.1.2.2 Normal Alt-Geniş Özel Anahtar (Normal Child-Extended Private Key) . . . 35
4.1.2.3 Zorlaştırılmış Alt-Geniş Özel Anahtar (Hardened Child Extended Private Key) . . . 37
4.1.2.4 Normal Alt-Geniş Ortak Anahtar (Normal Child Extended Public Key) . . . 39
4.1.2.5 Zorlaştırılmış Alt-Geniş Ortak Anahtar . . . 42
4.1.2.6 Ortak Anahtarlar Hakkında Bilgilendirme . . . 42
4.1.2.7 Serileştirme . . . 45
4.1.3 BIP-44 . . . 46
4.2 Donanımsal Deterministik Cüzdan Uygulaması . . . 49
4.2.1 Uygulamada Kullanılan APDU Komutları . . . 50
4.2.2 Java Kart HD Cüzdan Uygulaması ile Para Transferi . . . 51
4.2.3 AKİS Kart Donanımsal Deterministik Cüzdan Tasarımı . . . 58

5 Sonuç 60

A BIP-32 Test 62

Kaynakça 66

Şekil Listesi

2.1 Bölgelere Göre Blokzincir Geliri, Dünya Pazarları: 2016-2025 7
2.2 Temsili Özet İşaretçisi Gösterimi 10
2.3 Merkle Ağacı Yapısı 12

3.1 Masaüstü Cüzdan Çeşitlerine Örnekler 20
3.2 Hiyerarşik Deterministik Cüzdanda Anahtarların Üretim Aşamaları 24

4.1 BIP-39'a göre tohum oluşturma adımları 28
4.2 BIP-32 uyumlu hiyerarşik deterministik cüzdanların anahtar yapıları . . . 31
4.3 Temel Anahtar Oluşturulması 34
4.4 Örnek Temel Anahtar 35
4.5 Normal Alt Geniş Özel Anahtar Hesaplama Örneği 36
4.6 Normal-Alt Geniş Özel Anahtar Hesaplama 37
4.7 Örnek Zorlaştırılmış Alt Geniş Özel Anahtar 38
4.8 Zorlaştırılmış Alt- Geniş Özel Anahtar Hesaplama 39
4.9 Normal Alt Geniş Ortak Anahtar Hesaplama Örneği 40
4.10 Normal Alt-Geniş Ortak Anahtar Hesaplama 41
4.11 Ortak Anahtar 42
4.12 Alt Ortak Anahtar 43
4.13 Alt Ortak Anahtar 43
4.14 Aynı Alt Ortak Anahtarın Üretilmesi 44
4.15 Serileştirilmiş Özel Anahtar 46
4.16 BIP-44 Derinlik Seviyeleri 47
4.17 Uygulamanın (Applet) derlenip karta ya da simülatöre yüklenmesi 51
4.18 Electrum'a Cüzdan Uygulamasını Tanıtma 52
4.19 Cüzdana PIN Atanması 53
4.20 Tohumun hesaplanması için 12 hatırlatıcı kelimenin üretilmesi 54
4.21 Türetme Yolu Belirleme 55
4.22 Cüzdana ait adresler 56
4.23 İşlem bilgieri 57

Tablo Listesi

2.1 Blok Başlığı Alanları 10
2.2 İşlem Alanları 17

3.1 Cüzdan Çeşitlerinin Karşılaştırılması 22

4.1 Entropi-Kombinasyon Tablosu 29
4.2 Serileştirilmiş Anahtardaki Alanlar 45
4.3 Java Kart HD Cüzdan Uygulaması APDU Komut Başlıkları 50

Kısaltmalar

AAA	**A**çık **A**anahtar **A**ltyapısı
AID	**A**pplication **ID**entifier
AKiS	**A**kıllı **K**art **i**şletim **S**istemi
APDU	**A**pplication **P**rotocol **D**ata **U**nit
ARM	**A**corn **R**ISC **M**achine
ATM	**A**utomatic **T**eller **M**achine
BIP	**B**itcoin **I**mprovement **P**roposals
CA	**C**ertificate **A**uthority
CKD	**C**hild **K**ey **D**erivation
ECC	**E**lliptic **C**urve **C**ryptography
ECDSA	**E**lliptic **C**urve **D**igital **S**ignature **A**lgorithm
EFT	**E**lektronik **F**on **T**ransferi
Hex	**Hex**adecimal
HD	**H**iyerşik **D**eterministik
HMAC	**H**ash-based **M**essage **A**uthentication **C**ode
Nonce	**N**umber only used **once**
PIN	**P**ersonal **I**dentification **N**umber
PBKDF	**P**assword **B**ased **K**ey **D**erivation **F**unction
SHA	**S**ecure **H**ash **A**lgorithm
TXID	**T**ransaction **ID**entification
VOUT	**V**ector of an **OUT**put

Bölüm 1

Giriş

İnsanoğlunun toplum düzenine geçmesiyle mal ve mülk edinme ihtiyacı doğmuştur. Bu da mal veya hizmet satın almak için belli bir değeri olan nesnelerin yani paranın icadını sağlamıştır. Akıllı kartlar ve blokzincir tabanlı cüzdana giriş yapmadan önce paranın tanımını ve tarihsel gelişimini bilmek gerekir. Para, genellikle mal ve hizmetlerin ödenmesi ve vergiler gibi borçların belirli bir ülkede veya sosyo-ekonomik bağlamda geri ödenmesi olarak kabul edilen herhangi bir kalem veya doğrulanabilir kayıttır [1] [2].

Aslında paranın birçok tanımı yapılabilir. Fakat bir şeye para denmesi onun fonksiyonuna bağlıdır. Değiştirilebilir olması, belli bir birime sahip olması, kendi değerini koruması vb. özellikler parayı niteler. Paranın ana işlevleri şu şekilde ayırt edilir: bir değişim aracı, bir hesap birimi, bir değer deposu ve bazen de ertelenmiş bir ödeme standardıdır [3] [4]. Bu işlevleri yerine getiren herhangi bir madde veya doğrulanabilir kayıt para olarak kabul edilebilir. Eskiden taşınılabilir olması, sert ve dayanıklı olması birimler halinde bölünebilmesi, değerinin her yerde aynı olması, güvenilir olup taklit edilemeyecek olması gibi özellikleri aranan paranın teknoloji ile birlikte belirgin özellikleri de biçimsel olarak evrilmiştir. Para yıllar içinde biçim değiştirse de insanlık tarihi boyunca değerli varlık olarak yerini korumaktadır. Lidyalılardan günümüze kadar uzanan taş paraların, gümüş ve altınların, kâğıt banknotların ve kripto paraların tümünün amacı değeri niteliğinde mal ve hizmetlerin değiş tokuşu için ödeme aracı olarak kullanılmasıdır [5]. Son yüzyılda da hızla değişen teknoloji değerli varlık olan paranın biçimini de radikal olarak dijitalleşmeye doğru itmiştir. Akçe, sikke kâğıt gibi fiziksel özelliklerin yerini dijital para sistemleri almıştır. 1946'da John Biggins tarafından kredi kartının icat edilmesiyle

paranın serüvenine yeni bir dönem eklenmiştir. 1950'lerden sonra teknolojideki büyük gelişmeler para sistemini de etkilemiştir. Dünyada ilk defa bankalarda paranın elektronik olarak transferi yani EFT kullanımı yapılmıştır [6]. Son 30 yıldır tarih sahnesine çıkan dijital paranın önemi akıllı kartlar yoluyla benzin alınması, esnafların pos cihazlarıyla kullanılmaya başlanması, otomatik para çekme makineleri olan ATM'lerin üretilmesiyle daha da artırmıştır. Alışveriş ve ticaret başta olmak üzere çoğu alanda kredi kartlarının kullanımı fiziksel ortamın sınırlılığını ortadan kaldırarak istenilen saatte daha çok mal ve hizmete ulaşma imkânı sağlamıştır. Bunun en büyük getirileri arasında zamandan tasarruf sağlaması ve mekâna olan bağımlılığın azalması sayılabilir.

Satoshi Nakamoto'nun 2008 yılının Ekim ayında ara katmanlara ihtiyaç duymayan uçtan uca elektronik ödeme sistemi hakkında yayınladığı makale ve 2009 yılında ilk Bitcoin yazılımıyla her geçen gün popülerliği daha da artan blokzincir tabanlı kripto paralar, birçok platformda ödeme aracı olarak kullanılmaya başlanmıştır [7]. Kripto paralara örnek olarak Bitcoin, Etherium, Altcoin vs. verilebilir. Kripto kelimesi gizli ve şifreli olduğu anlamına gelir. Kripto para ise finansal işlemlerin kriptolojik algoritmalarla desteklendiği alternatif değiş tokuş aracı olan dijital bir değerdir. Kripto paralar merkezi banka sistemlerinin tersine blokzincir denilen merkezi olmayan bir sistem üzerinde çalışır. Sanal olması ve belli bir merkezi yapısının olmaması sebebiyle kripto paraların kontrolü ledger denen herkese açık kayıt defteriyle sağlanır. Tüm işlemler bu dağıtık kayıt defterinde birbiri ardına bağlanmış bloklar şeklinde tutulur ve isteyen herkesin erişimine açıktır. Bitcoin için sağlanan Açık Blokzincir platformundan sonra Hyperledger ve akıllı kontratlar gibi çeşitli uygulamalar için Açık ve İzinli Blokzincir, Gizli ve İzinli Blokzincir platform türleri de türetilmiştir.

Kripto para transferi için gönderen ve gönderilene ait ortak anahtar denilen adreslerin olması gereklidir. Temelinde bütün blokzincir yapısı açık anahtar kriptografisine dayanır. Bu adreslerin birbirine transfer ettiği kripto para miktar bilgilerinin olduğu işlemler, birbiri ardına belli bir sisteme göre kayıt defterinde tutulur. Her adresin yani açık anahtarın benzersiz gizli bir anahtar karşılığı bulunmaktadır. Kripto para işlemleri için bu özel anahtarlar ile atılmış imzalar kullanılır. Mahremiyet söz konusu olduğu için kripto para transferinde birden fazla adres yani anahtar çifti kullanmak önemlidir ve önerilmektedir. Bu anahtarların yönetimi ve depolanması cüzdanlar sayesinde sağlanır. Temelde cüzdanlar çevrimiçi ve çevrimdışı olmak üzere ikiye ayrılmaktadır. Çevrimiçi cüzdanların

internet erişimi bulunur. Çevrimdışı cüzdanların ise direkt olarak internet erişimi yoktur ve daha güvenlidir. Bu çalışmada çevrimdışı cüzdan çeşidi olan blokzincir tabanlı kripto paraların erişimine ve harcanabilmesine olanak sağlayan anahtarların tutulduğu donanımsal akıllı kart cüzdan konusu işlenecektir.

Blokzincir tabanlı kripto para birimlerine erişimi sağlayacak, işlemler için imza atacak ve blokzincir ağına bağlanmayı sağlayacak uygulama için akıllı kart çeşidi olarak java kart seçilmiştir. Tezin hazırlanması genelden özele doğru ilerlemiştir. Çalışmada paranın tanımı ve tarihiyle yapılan girişten blokzincir, akıllı kart ve donanımsal cüzdana kadar olan konular sırasıyla işlenmiştir.

Bu doküman, ilk bölüm giriş; ikinci bölüm Blokzincir; üçüncü bölüm Blokzincir tabanlı cüzdanlar; dördüncü bölüm Blokzincir tabanlı donanımsal hiyerarşik deterministik (HD) cüzdan uygulaması; beşinci bölüm sonuç bölümlerini kapsamaktadır.

İlk bölüm icadından beri sürekli bir değişim içerisinde olan paranın kripto paraya doğru evrilmesinin, mevcut merkezi sistemden dağıtık sisteme geçilmesinin anlatıldığı bir giriş ile başlamıştır. Önem verilen bu değiş-tokuş aracı olan paranın güvenli bir ortamda tutulabilmesini sağlayan cüzdan kavramından bahsedilmiştir. Mevcut durumda en güvenli cüzdan çeşidinin akıllı kartlar gibi donanımsal cüzdan olduğu vurgulanmış ve akıllı kartlar hakkında genel bilgiler verilmiştir.

İkinci bölümde cüzdanın çalışacağı blokzincir ortamı hakkında, blokzincirin yapısını, elementlerini ve işleyişini içeren önemli bilgiler anlatılmıştır.

Üçüncü bölümde blokzincir tabanlı cüzdan çeşitleri ayrıntılı incelenmiştir ve karşılaştırılması yapılmıştır.

Dördüncü bölümde HD cüzdanın çalışma mekanizması ve uygulaması iki alt bölümde detaylı olarak incelenmiştir. Çalışma mekanizması kısmında BIP-32, BIP-39, BIP-44 prokollerine göre cüzdanın anahtarılarının üretilme adımları sıralanmış ve görselleştirilmiştir. Bu adımların doğruluğu ekte paylaşılan örnek Python koduyla kanıtlanmıştır. Uygulama kısmında ise kullanılan APDU komutları listelendikten sonra java akıllı kartına HD cüzdan uygulaması yüklenmiş, akıllı kartın ilklendirilmesi yapılmış ve Bitcoin testnetinde örnek bir para transferi gösterilmiştir.

1.1 İlgili Çalışmalar

Dünya genelinde Goldman Sachs, Morgan Stanley, Citibank, HSBC, Accenture, Microsoft, IBM, Cisco, Tencent, Alibaba, Samsung, LG ve diğer küresel ünlü finansal kurumlar, danışmanlık firmaları ve internet devleri blokzincir teknolojisinde laboratuvar araştırmalarını ve sermaye düzenini hızlandırmaktadır. Ayrıca IBM ve Apache vakfı tarafından desteklenen Hyperledger projesi, Ethereum, FileCoin gibi çeşitli girişimler, blokzincir araştırma ve geliştirme için açık kaynak havuzları ve platformları sağlamaktadır [8] [9]. Diğer bir teknoloji devi Sony, blokzincir teknolojisinin tüm dünyaya yayılmasını sağlamak için 2018'de çoğu kripto parayı destekleyen yeni çevrimdışı donanım cüzdanı geliştirmiş ve tanıtmıştır [10]. Dünya genelinde devletler, blokzincir teknolojisinin gelişimine yönelik olumlu tutumlarını göstermek adına blokzincir hakkında teknik raporlar yayınlamaktadır. İngiltere'de hükümet bilimsel başdanışmanı, dağıtık defter teknolojisinin geleceğini açıklayan yeni bir rapor yayınlamıştır [11]. Çin hükümeti blokzincir teknolojisi ve Çin'deki gelişimi hakkında teknik inceleme yapmıştır [12]. Amerika Birleşik Devletleri'nde (ABD) Delaware valisi, blokzincir teknolojisinin geliştirilmesi adına yasal ve düzenleyici bir ortam oluşturmak için kapsamlı bir program olan "Delaware Blokzincir Girişimi" ni başlatmıştır [13]. Akademia'de son beş yıl içinde blokzincirle alakalı çalışma raporları da dâhil binlerce makale yayınlanmıştır.

2014'te T. Bamert, C. Decker, R. Wattenhofer, ve S. Welten, BlueWallet ismindeki donanımsal cüzdanın temelini atmış ve tanıtmışlardır [14].

2014'te donanımsal cüzdan örneği olan bitsafe cüzdanı açık kaynak olarak GitHub sayfasında paylaşılmıştır [15]. Bitsafe firmasının internet sayfasında akıllı kart olarak Mastercard kullanıldığından bahsedilmektedir.

2014'ün Ocak ayında Trezor adlı donanımsal cüzdan açık kaynak olarak ilk kez piyasaya sürülmüştür [16].

2015'in Eylül ayında Ledger adlı donanımsal cüzdanın java kart versiyonu açık kaynak olarak sürülmüştür. Şu an LedgerNanoS, LedgerNanoX isimlerinde ürünleri mevcuttur. Ürünün Github sayfasında farklı dilde yazılmış versiyonları ve mobil ile bilgisayar bağlantısı için de farklı sürücüleri ve servisleri mevcuttur [17] [18]. Bu cüzdan akıllı karta yüklenen 25 farklı özel anahtarı saklayabilme kapasitesine sahiptir. Kartın ilklendirilmesi

sırasında kurtarma ya da onarma durumları için bir kısım kelimelerin saklanması gerekmektedir. İşlemlerin imzalanması sırasında mobil ve bilgisayar ara bağlantısına ihtiyaç duymaktadır. Donanımsal cüzdanlar arasında en bilinenlerindendir.

2016'da Christian Decker "On the Scalability and Security of Bitcoin" adlı doktora tezinde Bitcoin, blokzincir ve cüzdan konularını geniş çerçevede incelemiştir. Güvenli Bitcoin cüzdanı olarak BlueWallet adlı donanımsal cüzdanının işleyişi üzerinde durmuştur [19].

2016'da Miraje Gentilal "Trustzone backed Bitcoin Wallet" adlı makalesinde daha çok TrustZone olmakla birlikte donanımsal cüzdanları ve Bitcoin protokolünü incelemiş, TrustZone destekli cihazlarda cüzdanın nasıl daha güvenli hale getirilebileceğini analiz etmiştir. Ayrıca ARM tabanlı akıllı kartın güvenli elementinin faydası ve donanımsal cüzdanlarda kullanımının önemi vurgulanmıştır [20].

2016'nın Şubat ayında KeepKey adlı donanımsal cüzdan Trezor cüzdanın çatalı şeklinde ilk versiyonunu duyurmuştur. Güvenlik açısından Trezor cüzdan ile özellikleri aynıdır denilebilir. Aralarındaki göze çarpan ilk fark dış tasarımıdır. Bundan farklı olarak KeepKey daha stabil çalışmaktadır ve kullanıcı dostu ara yüze sahiptir. Ancak, Trezor piyasada KeepKey'den daha uzun süre bulunduğundan, çok sayıda kullanıcıya sahip olmasından dolayı daha güvenli olduğu düşünülmektedir. 2018'de Matouš Skála, BIP-32'ye göre tanımlı Trezor adlı donanımsal hiyerarşik Bitcoin cüzdanın yapısını ve imzalama şemasını incelemiş ve Andorid desteği ile iletişimi sağlayacak temel bir uygulama üzerinde çalışmıştır [21].

2018'de Wouter de Boer'in, "Power Analysis of the Ledger NanoS" adlı makalesinde BIP-32'ye göre tanımlanmış LedgerNanoS donanımsal hiyerarşik cüzdanın güvenlik durumu incelenmiş ve güvenlik zafiyetleri üzerinde durulmuştur. Cüzdana yan kanal saldırısı yapılmış ve sonuçları gözlenmiştir [22].

2019'da M. Arapinis, A. Gkaniatsou, D. Karakostas, A. Kiayias "A Formal Treatment of Hardware Wallets" adlı makalesinde donanımsal cüzdanları incelemiş ve güvenli resmi bir cüzdan modelinin tanımlanması için çalışmışlardır [23].

2019'da Yasir Emre Bulut "Secure Hardware Cryptocurrency Wallet within Common Criteria Framework" adlı tezinde kripto para cüzdanlarının güvenlik açıkları ve riskleri gibi güvenlik durumunu Ortak Kriterler çerçevesinde incelemiş ve gereksinimlere göre cüzdan

tasarımı önerisinde bulunmuştur. Ayrıca çalışmasında hâlihazırda kullanılan cüzdanların karşılaştırması yapılmıştır [24].

2019'da Aleksandr Tsõganov "Integrating User Identity with Ethereum Smart Contract Wallet" adlı tezinde hâlihazırda kullanılan blokzincir tabanlı cüzdanları tanıtmış ve karşılaştırmasını yapmıştır [25].

1.2 Amaç

Blokzincirin ve donanımsal hiyerarşik cüzdanın yapısı, işleyişi ve akıllı kartlar ile kullanımı detaylı şekilde incelenecektir. Bu tez çalışmasında donanımsal cüzdanın implementasyonunda kullanılan yöntemler irdelenecek olup çalışma sonucunda donanımsal HD cüzdan ile kripto para transferi uygulaması yapılacaktır. Tezin çalışma yöntemi kavram ispatıdır. Çalışan bir sistemin cüzdan gibi son kullanıcıya hitap eden elementlerini incelemek için yapılmış bu çalışma yerli çözümlere temel atma görevi görmektedir.

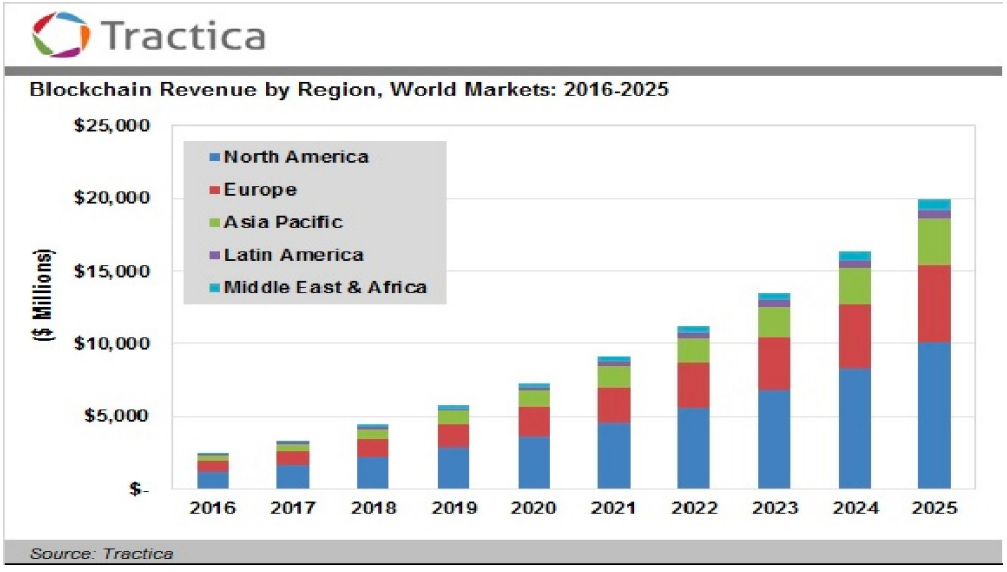
1.3 Akıllı Kartlar Hakkında

Akıllı kart, güvenlik donanımı olarak işlev gören yerleşik çipe sahip fiziksel bir elektronik karttır. Akıllı kartlar genellikle ehliyet veya kredi kartı boyutunda çip ve/veya anten entegre edilmiş plastik bir parçadır. Bir okuyucuya doğrudan fiziksel temas (çip) ile ya da yakın alan iletişimi (NFC) gibi kısa menzilli bir kablosuz bağlantı standardı ile bağlanırlar. Takıldıkları kart okuyucu ve bigisayarla APDU adı verilen komutlarla iletişim sağlamaktadır. Akıllı kartlar anahtar ve PIN gibi önemli verileri yapısındaki güvenli elementi sayesinde dışarıya vermediği ve kurcalamaya karşı koruma sağladığı için blokzincir tabanlı donanımsal cüzdanlarda en fazla kullanılan ve tercih edilen donanım aracıdır. Ledger Nano, CoolWallet, Satochip cüzdanları özel anahtarın üretilmesi, saklanması ve işlemlerin imzalanmasını sağlayacak donanım modülü olarak akıllı kart kullanmaktadır. Ledger Nano ve Satochip cüzdanı mevcut durumda akıllı kart çeşidi olarak Hollanda merkezli NXP firmasının çipi olan java kart üzerinde koşmaktadır. CoolWallet ise merkezi Almanya'da bulunan Infeneon firmasının ürünü olan SLE 97 çipini tercih etmiştir.

Bölüm 2

Blokzincir

Blokzincir teknolojisi, internete açık ağa ait bir sistemde merkezi otorite olmadan güvenli bilgi işlemenin yeni bir atılımıdır. İnternet kadar önemli bir buluş olduğu söylenmektedir [26]. Veri yönetimi perspektifinden bakıldığında blokzincir, birbiri ardına bağlanmış zincir blokları şeklinde büyüyen işlem kayıtlarının listesini tutan dağıtılmış bir veritabanıdır. Güvenlik açısından ise blokzincir, merkezi yapıya bağlı kalmaksızın eşler arası uçtan uca paylaşım ağı kullanılarak oluşturulan ve kriptografik algoritmalarla güvence altına alınan bir yapıdır. Dünya çapındaki blokzincir tabanlı kurumsal uygulamaların yıllık gelirinin, 2016 yılında yaklaşık 2,5 milyar dolardan yıllık % 26,2'lik bir büyüme oranı olan 2025'e kadar 19,9 milyar dolara ulaşacağı tahmin edilmektedir [27].



Şekil 2.1: Bölgelere Göre Blokzincir Geliri, Dünya Pazarları: 2016-2025 [27]

Blokzincirin ilk belgelenmiş tasarımı 2008'de yapıldı ve blokzincirin açık kaynak kodlu uçtan uca ilk uygulaması olan Bitcoin, 2009 yılında dağıtıma çıkarıldı. Her ikisi de önceden de bahsedildiği gibi Satoshi Nakamoto olarak bilinen anonim kişi ya da kişilerce öne sürüldü [7]. Bitcoin, uçtan uca ağ sistemindeki tüm bitcoin işlemlerini kaydeden ve doğrulayan dağıtık açık defteri olarak blokzinciri kullanır. Blokzincir tabanlı bitcoinin dikkat çekici bir yeniliği, herhangi bir merkeze güvenmek zorunda kalmaksızın yani merkezi olmayan uçtan uca ağda alımı ve gönderimi yapılan bitcoin işlemlerinde çifte harcamaları (double-spending) önleme yeteneğidir. Güvenli bir defter olarak blokzincir, büyüyen işlem kayıt listesini hiyerarşik olarak birbiri ardına bloklar zinciri halinde düzenler [28] ve her blok, işlem kayıtlarının bütünlüğünü sağlamak için Açık Anahtar Altyapısı temeli olan kriptografi teknikleri kullanır. Yeni bloklar merkezi olmayan uzlaşma prosedürü gereği bir yarışma sonucunda sadece başarılı en uzun zincirde işlenir. Rekabet sonucu başarısız olan diğer bloklar iptal edilir. Ayrıca bir blok, tüm bloğun kalıp özetini ve kendinden bir önceki bloğa ait özet değerini bulundurur. Ağ tarafından merkezi olmayan bir konsensüs (uzlaşma) prosedürü uygulanmaktadır ve bu; (i) yeni blokların blok zincirine kabulünü, (ii) blok zincirinin güvenli bir şekilde doğrulanması için okuma protokolünü ve (iii) her bir düğümde yer alan işlem kayıtlarının veri içeriğinin tutarlılığını kontrol etmektedir. Sonuç olarak, blokzincir, bir bloğa işlem kaydının eklenip ve blok başarılı şekilde oluşturulup işlendikten sonra işlem kayıtlarının geriye doğru değiştirilememesini, zincirin her bloğundaki veri içeriğinin bütünlüğünü garanti eder. Böylece blokzincir, açık ağa bağlı bir sistemin iki tarafı arasındaki tüm işlemleri etkili, kalıcı ve doğrulanabilir bir şekilde arşivleyen güvenli ve dağıtılmış bir defter görevi görür. Bitcoin kripto para sistemleri bağlamında blokzincir, Bitcoin ağında bitcoin ticareti yapan tüm işlemler için güvenli, özel ve güvenilir halka açık arşiv olarak kullanılır. Bu, tüm bitcoin işlemlerinin, doğrulanabilir ve kalıcı bir şekilde zincirlenmiş, şifreli olarak güvenli bloklarda kaydedildiğini, organize şekilde saklandığını garanti eder. Blokzincir, çifte harcama, özel işlemlerin ifşası, merkezi bir yapıya güvenme zorunluluğu gibi gizlilik ve güven sorunlarından bitcoin işlemlerini güvence altına almak için çok önemli bir koruyucudur. Blokzincir teknolojisini kullanan bitcoin sağlık, lojistik, eğitim sertifikası, güvenli depolama gibi diğer birçok uygulama alanları için de ilham kaynağı olmuştur.

2.1 Blokzincir Çalışma Mantığı Hakkında

Bir blokzincir, işlevsel olarak işlem kayıtlarının tutulduğu dağıtık ve güvenli bir veritabanıdır. Bir Bitcoin ağında, A istemcisi başka bir B istemcisine bitcoin göndermek istiyorsa, A istemcisi tarafından bir bitcoin işlemi oluşturulur. İşlem, Bitcoin ağı tarafından işlenmeden önce madenciler tarafından onaylanmalıdır. Madencilik sürecini başlatmak için işlem, ağdaki diğer tüm düğümlere dağıtılır. İşlemleri bir blok halinde toplayacak olan madencilerin kastedildiği bu düğümler, bloktaki işlemleri doğrular ve ağdan onay almak için bir uzlaşma protokolü kullanarak bloğu ve doğrulamasını yayınlar (İş İspatı). Diğer düğümler blokta bulunan tüm işlemlerin geçerli olduğunu doğruladığında, blok artık blokzincire eklenebilir. İşlemi içeren "blok" diğer düğümler tarafından onaylandığında ve blok zincirine eklendiğinde, artık A'dan B'ye bu bitcoin transferi sağlanmış ve meşru hale gelmiş olur. Bitcoin'deki blokzincir uygulaması tarafından desteklenen üç temel ve önemli özellik şunlardır: (1) özet zinciri, (2) dijital imza ve (3) kabul edilmiş en uzun zincire yeni bir blok eklemek için taahhüt konsensüsüdür. Özet zinciri, Merkle ağacı, dijital imza, uzlaşma mekanizmaları gibi bir dizi popüler güvenlik teknikleriyle Bitcoin blokzinciri hem bitcoinlerin çift harcama problemini önlemektedir hem de blok başarıyla blokzincir içine işlendikten sonra blokta herhangi bir işlem verisinin geriye dönük modifikasyonunu engellemektedir.

2.1.1 Blok

İçerisinde işlemleri barındıran ve blokzincire eklenen madencilerin ürettiği bir yapıdır. Madenciler hafıza havuzunda tutulan aday (onaylanmamış) işlemlerden blok oluşturmak için yarışa girer. İstenen hedef değerinin altında blok üretmeyi başaran madenci ürettiği bloğu diğer düğümlerdeki madencilere dağıtır. Diğer madenciler de bloğun istenen hedef değerin altında üretilip üretilmediğini kontrol ettikten sonra blok, blokzincire eklenir ve onaylanır. Daha sonraki blok için işlem havuzundan işlemler seçilir ve yarış tekrar eder. Bitcoin için blok üretimi yaklaşık 10 dakikaya ayarlanmıştır. Her blokta blok kimliği, blok başlığı ve işlemler bölümü yer alır.

Blok Başlığı

Blok başlığı Versiyon, Önceki Blok Özeti, Merkle Ağacı, Zaman damgası, Bits, Nonce gibi bloğa ait bilgi alanlarından oluşur. Önceki Blok Özeti ve Merkle Ağacı bütünlük

kontrolünde kullanılır ve özet zinciri yapısını oluşturur. Bir bloğun içerisindeki işlemlerin bütünlük kontrolü Merkle ağacıyla, bu bloktan önceki blok zincirinin bütünlük kontrolü ise Önceki Blok Özetiyle sağlanır.

TABLO 2.1: Blok Başlığı Alanları

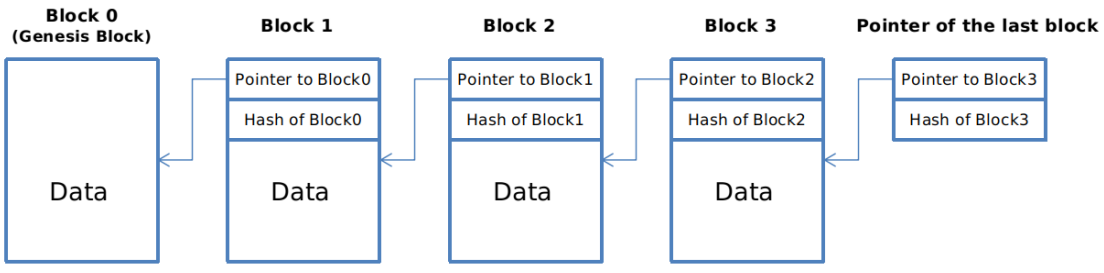
Alan	**Açıklama**	**Uzunluk**	**Veri yapısı**
Versiyon	Bloğa ait versiyon bilgisidir.	4 bayt	Küçük-endian
Önceki Blok Özeti	Bir önceki bloğa ait blok özeti veya kimliğidir.	32 bayt	Küçük-endian
Merkle Ağacı	Bloktaki tüm işlemlerin parmak izini temsil eden özet değeridir.	32 bayt	Küçük-endian
Zaman damgası	Bloğun madenci tarafından oluşturulduğu andaki Unix saatine göre bloğa verilen zaman bilgisidir.	4 bayt	Küçük-endian
Bits	Hedef değerinin kısaltılmış halidir.	4 bayt	Küçük-endian
Nonce	Madencilerin blok başlık özet değerini istenen hedef değerin altında üretmek için ayarladıkları sayıdır.	4 bayt	Küçük-endian

Blok Özeti (Kimliği)

Blok özeti Blokzincirde bir bloğa verilen referans numarası yani blok kimliğidir. Blok başlığının iki kez SHA256 özet fonksiyonundan geçirilerek küçük-endian yapısına göre oluşturulur.

Özet İşaretçisi

Özet işaretçisi, verinin depolandığı konumu işaret eden ve işaret ettiği verinin özeti olan değerdir [29]. Verilerin kurcalanıp kurcalanmadığını kontrol etmek için özet işaretçi kullanılır. Özet işaretçisi bir önceki bloğu işaret ederek bir önceki bloğun verilerinin depolandığı yeri gösterir. Veri bloklar sıralı şekilde birbirine bağlanarak zincir yapısı oluşturulur.

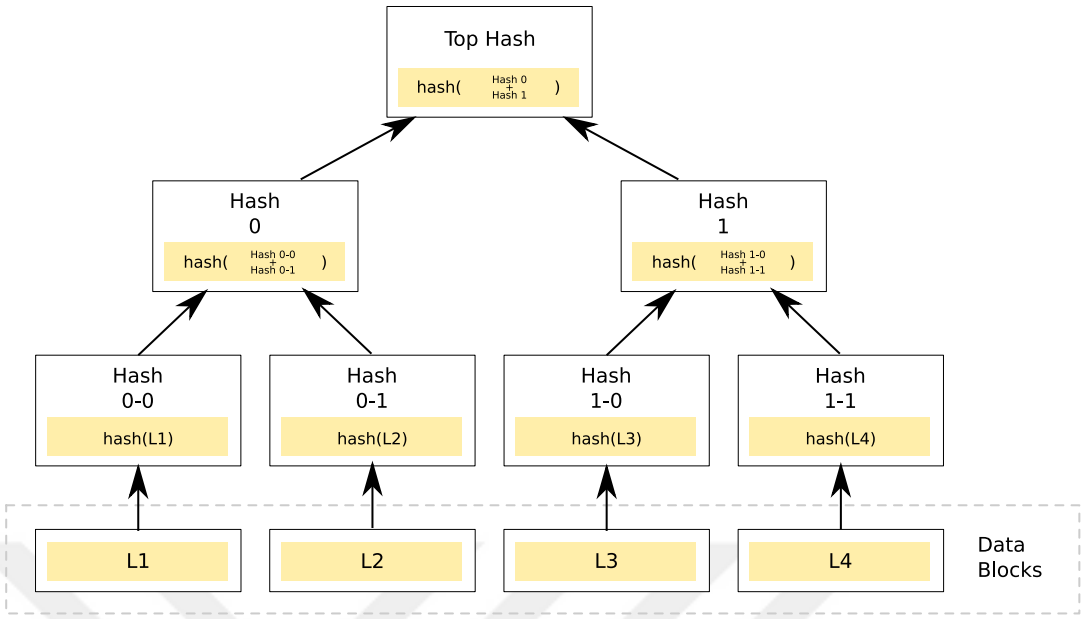


ŞEKIL 2.2: Temsili Özet İşaretçisi Gösterimi [29]

Ayrıca, saklanan verilerin özeti, verilerin kurcalanıp kurcalanmadığını kanıtlamak için kullanıcılar tarafından genel olarak doğrulanabilir. Bir saldırgan zincirin herhangi bir bloğundaki verileri değiştirmeye çalışırsa, kurcalamayı gizlemek için, önceki tüm blokların özet işaretlerini değiştirmek zorundadır. Nihayetinde saldırgan en baştaki bloğu değiştiremez çünkü bu baş blok herkesin bildiği bloktur. Bu yüzden saldırgan en sonunda bu bloğu değiştirip zinciri herkese kabul ettirmeye çalıştığında açığa çıkacaktır. Bu baş blok kurcalamayı durduran herkesin bildiği referans bloktur ve Genesis blok olarak adlandırılır. Kullanıcılar geriye doğru baş bloğa kadar giderek tüm zinciri doğrulayabilmektedir.

Merkle Ağacı

Merkle ağacı bir bloktaki tüm işlemlerin genel parmak izi olarak düşünülebilir, tüm işlemlerin özet değeridir. Bloktaki tüm işlemlere ait işlem kimlikleri sırasıyla ikili gruplar haline ayrılıp özeti alınır. Tek bir özet sonucu kalana kadar özet sonuçları da ikili gruplar halinde özeti alınır ve hesaplanan en son özet sonucu Merkle ağacını verir. Oluşturulan bu Merkle ağacı daha sonra blok başlığında ilgili alana yazılır. Böylece her blok başlığında, bloğun içindeki tüm işlemlerin parmak izi bilgisi olan Merkle ağacı bulunmaktadır. Blokzincir oluşturmak için kullanılan başka bir yararlı veri yapısıdır. Tüm işlemleri temsil edecek parmak izi verisini oluşturmak için ikili özetler almak yerine, tüm işlem kimliklerini toplayıp hepsinin basit bir özeti de alınabilirdi. Fakat daha sonra bu parmak izi kontrol edilmek istendiğinde tüm işlemleri toplamak gerekir. Yani aynı özet değerini alabilmek için bloktaki tüm işlem kimliklerini toplayıp özeti alınarak kontrol etmek gerekir. Bunun yerine Merkle ağacıyla bloğu kontrol etmek için ise sadece ağaçtaki bazı özet değerlerinin toplanması yeterli olacaktır, tüm işlemlere ait kimliklerin toplanması gerekmez. Böylece Merkle ağacıyla bloktaki işlemlerin kurcalanıp kurcalanmadığını anlamak için yapılan özet kontrolü hızlı şekilde sağlanmış olacaktır.



Şekil 2.3: Merkle Ağacı Yapısı [30]

Merkle ağacının az sayıda işlemi bulunan bloklarda kullanılması pek mantıklı gözükmese de 2000 küsur gibi ya da daha fazla işlemi bulunan büyük blokların kontrolünde kullanımı çok etkilidir. Örnek olarak 2000 adet işleme sahip bir bloğun bütünlüğü kontrol edilmek istendiğinde Merkle ağacı kullanılmazsa 2000 x 32 bayt = 64000 bayt işlem kimlik bilgisinin indirilmesi gerekir. Merkle ağacı kullanılarak ise 11 x 32 bayt = 352 bayt veri yeterlidir. 11 değeri bloktaki herhangi bir işlem kimliğinden Merkle ağacına kadar uzanan dalların sayısıdır. (İşlem kimliğinin boyutu 32 bayt)

Sonuç olarak blok başlığında bulunan Merkle ağacı, işlemlerin hepsini bilmeye gerek kalmadan işlemlerin var olduğunu ya da değiştirilmemiş olduğunu kanıtlamaya yaramaktadır.

Takma isim olarak Ortak Anahtarlar

Dijital imza kullanımının avantajı, bir mesajın göndericisinin özel anahtarının açığa çıkmadan imzalandığı ve mesajın geçerliliğini kanıtlamak için bu imzalı mesajın alıcısının gönderenin ortak anahtarıyla doğrulayabileceği şekilde AAA'yı (Açık Anahtar Altyapısı-PKI) kullanarak ortak anahtarın sahibinin göndericiye ait olduğunun kanıtlanmasıdır. Çoğu uygulama senaryosunda güvenilir bir üçüncü taraftan anahtar çifti elde edilebilir. AAA, ilgili anahtar kimlikleri (ad soyad, e-posta ve kimlik gibi) ve ortak anahtarları

arasında bağlayıcı bir sözleşme oluşturarak ortak anahtarları yönetmek için kullanır. Bu tür bir bağlanma, bir sertifika otoritesine (CA) kayıt ile ve karşılığında otoriteden sertifika alınarak yapılır. İmza doğrulama süreci, bağlayıcının güvence düzeyine bağlı olarak imzalayıcının kimlik doğrulamasına otomatik olarak çevrilir. Dolayısıyla, ortak anahtar bu gibi merkeziyetçi senaryolarda bir kimlik olarak görülebilir. Bitcoin blokzinciri, bir kullanıcıyı sisteme kaydettirmek için merkezi bir yapıya gerek duymaksızın yani merkezi olmayan bir kimlik yönetimini benimser. Anahtar çiftleri kullanıcılar tarafından oluşturulur. Kullanıcılar istedikleri kadar anahtar çifti oluşturabilirler. Bu kimliklere (ortak anahtarların özetleri) Bitcoin'de adres denmektedir. Ortak anahtarların merkezi yönetimi olmadığından, bu kimlikler aslında kullanıcılar tarafından oluşturulan takma adlardır.

2.1.2 Eliptik Eğri Dijital İmza Algoritması (ECDSA)

Dijital imza, kriptografik algoritma kullanarak bir veri parçasının geçerliliğini belirler. Ayrıca, bir veri parçasının kurcalanıp kurcalanmadığını doğrulamak için bir şema sunar. Dijital imza şemasını formüle eden üç temel bileşen vardır. İlk bileşen, iki anahtar oluşturan anahtar oluşturma algoritmasıdır, biri iletileri imzalamak için kullanılır ve özel olarak saklanır ve diğeri ortak anahtar olarak adlandırılır ve mesajda ilgili özel anahtarla imzalanmış bir imzayı doğrulamak için kullanılır. İkinci temel bileşen imzalama algoritmasıdır. Verilen özel anahtar kullanılarak onaylanan giriş mesajında bir imza oluşturur. Üçüncü temel bileşen doğrulama algoritmasıdır. Bir imzayı, bir mesajı ve bir ortak anahtarı girdi olarak alır ve ortak anahtarı kullanarak mesajın imzasını doğrular ve imzanın doğru ya da yanlış olduğunu belirten boolean değeri döndürür. Bitcoin blokzincirinde, imzalama işlemleri için dijital imza şeması olarak ECDSA kullanılır. Standart eliptik eğri "secp256k1" üzerinde ECDSA kullanılarak, Bitcoin blok zinciri için 128 bit güvenlik sağlanır [35]. ECDSA, gruba ve çarpışmalara dayanıklı özet fonksiyonuna dayalı seçilmiş bir mesaj saldırısında sahteciliğe karşı dirençli olduğunu kanıtlamaktadır.

Blokzincir teknolojisinde dijital imzanın kullanım amaçlarından biri ortak anahtarın sahibi olduğunuzun yani karşılık gelen özel anahtarı bildiğinizin kanıtlanması ve bunun özel anahtarı açığa çıkarmadan yapılmasıdır (Donanımsal cüzdanda özel anahtar çevrimdışı alanda kullanılır). Başka bir deyişle imzanın ve ortak anahtarın aynı özel anahtarla üretildiğini kanıtlamaya yaramaktadır. Böylece blokta yer alan sahibi olduğunuz cüzdanın ortak anahtarına tanımlı harcanmamış işlemdeki kripto paraya ulaşabilir ve harcama

veya gönderim gibi yeni bir işlem yapabilirsiniz. Aynı dijital imza ile aynı adresteki farklı harcanmamış işlemlere erişilemez çünkü her işleme özel benzersiz bir imza üretilmektedir. Dijital imza, hangi işleme erişim gerekiyorsa o işlemin kendisi de imzaya dâhil edilerek oluşturulur. Dijital imzayı, imza oluşturma ve imzayı doğrulama olmak üzere iki bölümde inceleyecek olursak [31] [32];

2.1.2.1 İmzalama

Bitcoin blokzincirinde imzalama algoritması olarak ECDSA (eliptik eğri dijital imza algoritması) kullanılmaktadır. İmzalama adımları bu algoritmaya göre anlatılacaktır.

1) Rastgele bir sayı oluşturulur. Bu sayı üreteç noktası (G) ile EC'ye göre çarpılır. Elde edilen nokta değerinin x koordinatı olan r alınır.

2) Özel anahtar ile r değeri çarpımına "mesaj" değeri eklenir. Bitcoinde mesaj kısmı, kilidinin açılması istenen çıktıyı içeren tüm işlem bilgisinin SHA256'ya göre özetidir.

3) 2'de bulunan değer, 1'de oluşturulan rasgele sayıya EC'ye göre bölme işlemi yapılır ve eğri derecesi = N'ye göre modu alınır, hesaplanan değer s'dir.

4) Her işleme özel oluşturulan dijital imza (r,s)'dir.

2.1.2.2 İmzalama Doğrulama

Doğru özel anahtarla oluşturulmuş imzayı kanıtlamak için eliptik eğride iki noktayı bulmak gerekir;

1) Mesaj değerini s değerine EC'ye göre bölme işlemi uygulanır ve bulunan yeni değer üreteç noktası G ile EC'ye göre çarpılır. Bulunan değer Nokta 1'dir.

2) EC'ye göre r değeri, s değerine bölünür ve ortak anahtar ile çarpılır. Bulunan değer Nokta 2'dir.

3) Nokta 1 ile Nokta 2 EC'ye göre toplanır ve bulunan noktanın x koordinatı ile r değerinin eşit olup olmadığı kontrol edilir. Eşitse imza doğru şekilde atılmış demektir.

Böylece ortak anahtarın ve karşılık gelen özel anahtarın size ait olup olmadığı kanıtlanmış olur.

2.1.3 Fikir birliği ve Madencilik

Merkezi olmayan blokzincir bağlamında, yeni bir blok üretilip ağa yayınlandığında, her düğümün bu bloğu kendi blokzincir kopyalarına ekleme veya yok sayma seçeneği vardır. Fikir birliği, ağın büyük bir kısmının blokzincirin genişlemesini sağlamak ve hileli girişimleri veya kötü niyetli saldırıları önlemek için tek bir durum güncellemesi üzerinde anlaşmaya varmak için kullanılır. Somut olarak, blokzincirin büyük, paylaşılan bir küresel defteri olduğu göz önüne alındığında, herkes onu güncelleyebilir, bir düğüm blokzincir kopyasına müdahale etmeye çalışabilir veya birkaç düğüm toplu olarak bu tür müdahalelere teşebbüs edebilir. Örneğin, A istemcisi cüzdanından B istemcisine 10 Bitcoin gönderiyorsa, ağdaki hiç kimsenin işlem içeriğini kurcalayamayacağından yani 10 Bitcoini 100 Bitcoine değiştiremeyeceğinden emin olmak ister. Böyle bir durum yalnızca blokzincir ağındaki %50'den fazla işlem gücüne sahip grubun zinciri değiştirmek için yaptığı saldırılarda meydana gelebilir. O yüzden bir grubun hiçbir zaman ağdaki işlem gücünün %50'den fazlasına sahip olamayacağı varsayılmıştır. Bu varsayım dışında fikir birliği ve zincir yapısı sayesinde diğer saldırıların hiçbiri blokzincir güvenliğini kırabilecek nitelikte değildir.

Bitcoin blokzincirinde işlemleri zincire ekleme işlemine madencilik denmektedir. Yeni yapılan tüm işlemler ağ üzerinden düğümlere gönderilir. Bütün düğümler bu işlemleri geçici depolama alanı olan hafıza havuzlarına alır. Düğümler bu hafıza havuzundaki işlemleri kayıt defteri olan blokzincire istenen kriterlerde yeni bir blok eklemek için uğraşırlar. Havuzdaki işlemlerden yeni bir blok üretip zincire eklenmesi için işlem gücü gerekmektedir. Bu güçlüğün sebebi aday bloğun başlığından oluşturulan özetin hedef değerin altında olmasının istenmesidir. Özetin hedef değerinin altında olması demek belli sayıda sıfırla başlamasına zorlamaktır. Bu istenen kriteri ilk gerçekleştiren madenci ürettiği bloğu artık zincire ekleyebilmektedir. Zincire eklenen blok diğer düğümlere yayınlanır. Diğer düğümler de bloğu kontrol edip onaylarsa yani fikir birliğine varılırsa bu sayede madenci, ödülü ve işlem ücretlerini hesabına aktarır. Madencilerin teşviki bu ödül ve işlem ücretleridir. Yeni bir blok ekleme süresinin yaklaşık 10 dakika olması için ise hedef değer her 2016 blok üretiminde bir güncellenir. Her bloğun 10 dakika sürdüğü varsayıldığında 2016 blok 20160 dakika = 336 saat = 14 gün = 2 hafta sürmektedir. Yaklaşık 2 haftada bir hedef değer güncellenir. Hedef değerin güncellenme nedeni ise madenci ağına yeni madencilerin eklenmesi veya ağdan madencilerin ayrılmasıdır. Ağdaki madenci sayısı arttığında hedef

değer düşürülür, azaldığında ise hedef değer yükseltilir. Hedef değer blok başlığında Bits alanında saklanır. Bits hedef değerinin sıkıştırılmış kompakt versiyonudur. Hedef değerin çarpmaya göre tersine ise zorluk denmektedir.

2.1.4 İşlemler

Bitcoin'de işlem göndericinin adresi, gönderilen adres ve gönderim miktarı gibi bilgilerin olduğu bir bilgi grubudur. Cüzdanla bir işlem yapıldığında bu işlem blokzincir ağına gönderilir ve bloğa eklenip onaylanması beklenir. Mevcut banka sisteminde istenilen tutar hesabınızdan alınıp gönderilir. Fakat Bitcoin sisteminde bir gönderim yapıldığında aslında gönderim miktarı hesabınızdan istenilen tutar kadar alınıp gönderme yapılmaz, bunun yerine işlemlerde girdiler ve çıktılar kullanılır. Örneğin bir gönderim işlemi için adresteki (hesaptaki) harcanmamış para girdi olarak seçilip iki çıktı oluşturulur, çıktıların biri gönderilecek miktar kadarı gönderilmek istenilen adrese, diğer çıktı ise paranın geri kalanını para üstü olarak kendi adresinize gönderilerek oluşturulur. Bunun yanında çıktıların miktarı girdilerin miktarını aşmayacak şekilde birden fazla girdi ile tek çıktı, bir girdi ile birden fazla çıktı ve birden fazla girdi ve birden fazla çıktı halinde oluşan işlemler de geçerlidir ve yaygın olarak kullanılmaktadır. Sonuç olarak işlemler para transferine ait girdiler ve çıktılardan oluşan bilgi yığınlarıdır.

Bir işlem; versiyon, girdi sayısı, girdi(ler), çıktı sayısı, çıktı(lar), kilitlenme süresi (locktime) alanlarından oluşur. Girdiler; işlem kimliği (TXID), çıktı vektörü (VOUT), imzalama scriptinin boyutu, kilit açma veya imzalama scripti (ScriptSig) ve sıra alanlarından oluşur. Çıktılar; miktar, kilitleme veya ortak anahtar scriptinin boyutu, kilitleme veya ortak anahtar scripti (ScriptPubKey) alanlarından oluşur. Bütün bu alanlar hex yapıdadır. Aşağıda alanlar tablo olarak sunulmuştur.

Tablo 2.2: İşlem Alanları

<table>
<tr><th>Alan</th><th>Boyut</th><th>Açıklama</th></tr>
<tr><td>Versiyon</td><td>4 bayt</td><td>İşleme ait veri yapısının versiyonudur.</td></tr>
<tr><td>Girdi Sayısı</td><td>Değişken</td><td>İşlemde kullanılacak girdilerin sayısını belirtir.</td></tr>
<tr><td>Girdi(ler)</td><td colspan="2"><table>
<tr><th>Alan</th><th>Boyut</th><th>Açıklama</th></tr>
<tr><td>TXID</td><td>32 bayt</td><td>Mevcut bir işleme atıfta bulunur.</td></tr>
<tr><td>VOUT</td><td>4 bayt</td><td>Atıfta bulunulan işlemden çıktı seçilir.</td></tr>
<tr><td>ScriptSig uzunluğu</td><td>Değişken</td><td>Sonraki alanda kullanılacak kilit açma scriptinin uzunluğunu belirtir.</td></tr>
<tr><td>ScriptSig</td><td>Değişken</td><td>Girdinin kilidini açmak için kullanılan scripttir.</td></tr>
<tr><td>Sıra</td><td>4 bayt</td><td></td></tr>
</table></td></tr>
<tr><td>Çıktı Sayısı</td><td>Değişken</td><td>İşleme ait çıktıların sayısını belirtir.</td></tr>
<tr><td>Çıktı(lar)</td><td colspan="2"><table>
<tr><th>Alan</th><th>Boyut</th><th>Açıklama</th></tr>
<tr><td>Miktar</td><td>8 bayt</td><td>Satoshis birimi olarak çıktıdaki miktardır.</td></tr>
<tr><td>ScriptPubKey uzunluğu</td><td>Değişken</td><td>Sonraki alanda kullanılacak kilitleme scriptinin uzunluğunu belirtir.</td></tr>
<tr><td>ScriptPubKey</td><td>Değişken</td><td>Çıktıyı kilitlemek için kullanılan scripttir.</td></tr>
</table></td></tr>
<tr><td>Locktime</td><td>4 bayt</td><td>İşlemin blokzincire dâhil edilebileceği minimum blok yüksekliği veya Unix süresi olarak ayarlanır.</td></tr>
</table>

Bir çıktı sonraki işlem için girdi olarak kullanılarak harcanır ve aynı çıktı daha sonraki işlemler için tekrar kullanılamaz. Kullanılabilir çıktılara harcanmamış çıktılar (UTXO) denilmektedir. Madenciler gelen işlemlerin daha önceden kullanılıp kullanılmadığını kontrol eder. Her çıktıda kilit mekanizması bulunmaktadır ve harcanmamış işlemleri kullanabilmek için bu kilidin açılması yani çıktının sahibinin doğrulanması gerekmektedir. Bu doğrulama yukarıdaki tabloda gösterilen kilit açma scripti veya diğer ismiyle imzalama scripti ile yapılır. Bir çıktı sadece kilidi açıldığında girdi olarak kullanılabilir. İmzalama işlemi ile çıktıdaki ortak anahtara karşılık gelen özel anahtarın sahipliği doğrulandıktan sonra kilidi açılmış çıktı artık harcanabilir durumdadır. Bu aşamadan sonra çıktı için kullanılacak miktar belirlenir ve sadece gönderimi yapılacak kullanıcının açabileceği şekilde kilitleme scripti ile kilitleme işlemi gerçekleştirilir. Kilit açma ve kilitleme scriptleri ile diğer veri alanları bir işlemde birleştirildikten sonra geçerlenmek üzere madencilere gönderilir.

Bölüm 3

Blokzincir Tabanlı Cüzdanlar

Cüzdan bir banka hesabı gibi düşünülebilir, ama ortada banka yoktur. Cüzdan olmadan para gönderme işlemi yapılamaz. Bu yönüyle bakıldığında tüm kripto para blokzincirinin temel yapıtaşı cüzdandır denilebilir. Blokzincirde cüzdanın görevi, özel anahtarları ve bunlara karşılık gelen ortak anahtarları saklamak, bu anahtarların yönetimini ve blokzincir ağında hesaplara erişimini sağlayarak para transferini gerçekleştirebilmektir. 2020 yılı Mart ayındaki verilere göre Etherium, Bitcoin, Litecoin, Altcoin gibi 5255 adet kripto para çeşidi bulunmaktadır [33]. Bu gibi kripto para hesaplarının anahtarları cüzdanlarda tutulur. Temelde internet erişimleri bakımından 2 farklı cüzdandan bahsedilebilir; çevrimiçi cüzdanlar ve çevrimdışı cüzdanlar. Çevrimiçi cüzdanların internete erişimi bulunmaktadır ve bu cüzdanlara yazılımsal cüzdanlar da denilmektedir. Çevrimiçi cüzdanlar masaüstü, mobil, web cüzdanlar olarak üçe ayrılırlar. Çevrimdışı cüzdanlar donanımsal ve kâğıt cüzdan olmak üzere ikiye ayrılmaktadır. Bu cüzdanlar da kendi içlerinde anahtarların üretilme şekline göre deterministik ve deterministik olmayan olmak üzere ikiye ayrılmaktadır. Bu bölümde cüzdan çeşitleri irdelenecek ve asıl odak nokta olan donanımsal cüzdanlar üzerinde durulacaktır.

3.1 Masaüstü Cüzdanlar

Masaüstü cüzdan oluşturulan anahtarların saklanması ve yönetimini bilgisayar üzerinden yapan çevrimiçi cüzdan çeşididir. Eğer cüzdan dosyası şifrelenip depolandıysa cüzdana erişim için doğrulama yapılması gereklidir. Bu cüzdanın kullanımı basittir fakat donanımsal akıllı kart cüzdan kadar güvenli olmasa da web cüzdanlara göre daha güvenilir kabul edilir [34].



Şekil 3.1: Masaüstü Cüzdan Çeşitlerine Örnekler

Electrum, Exodus, Guarda, Jaxx Liberty, Armory, Green Address, Atomic Wallet, MultiBit HD, Bitcoin Core, CoPay en popüler masaüstü cüzdanları arasındadır [35].

3.2 Mobil Cüzdanlar

Mobil ortamda yüklenen çevrimiçi cüzdan çeşididir. Tablet ya da telefonlara göre tasarlanmışlardır, kullanımı basit ve pratiktir. İki faktörlü doğrulama, parmak izi ile doğrulama, PIN ile doğrulama gibi ileri seviye doğrulama gerektiren güvenlikler bu cüzdan çeşidine kolaylıkla entegre edilebilmektedir fakat yine de mobil cüzdanlar kötü amaçlı yazılım, keylogger ve virüslere açık bir ortamdır. Masaüstü cüzdanlar için verilen örneklerdeki çoğu üreticinin mobil cüzdan çözümü de bulunmaktadır. Jaxx, Green Address, Mycelium, Edge, Abra, Coinomi bilinen mobil cüzdan çeşitleri arasındadır.

3.3 Web Cüzdanlar

Web adresi üzerinden üretilmiş çevrimiçi cüzdan çeşididir. Bu cüzdanlara ait hesaplar bulut ya da veritabanı sunucuları üzerinde tutulur. Anahtarlarınızın güvenliği ve kontrolü kişinin kendisinde değildir ve tamamen güvenirliliği olmayan cüzdan çeşididir. Birçok web cüzdan üreticisi anahtarların kontrolünün sizde olduğunu kanıtlamak adına multi-sig denilen çoklu imzalama gibi yöntemlere girmişlerdir. Guarda Wallet, Freewallet, Cryptonator, Blokzincir.com web cüzdan çeşidine örnek olarak verilebilir.

3.4 Kağıt Cüzdanlar

Üzerinde açık ve özel anahtarın ve olduğu kâğıda yazılmış çevrimdışı cüzdan çeşididir. Anahtarlar tamamen kapalı ağ alanına sahip makine ile üretilip kâğıda kaydedilir. Hacker ya da kötü amaçlı kişilerin saldırıları da bu cüzdanda işe yaramayacaktır fakat kullanımı oldukça zordur.

3.5 Donanımsal Cüzdanlar

kripto para cüzdanlarının görevleri arasında olan anahtarların üretilmesi, saklanması, yedeklenmesi ve yönetilmesi arasında kripto paraların benimsenmesinin önündeki en büyük on engelden biri anahtar yönetimidir [36]. Mevcut en güvenli cüzdan, özel anahtarlar ve imza işlemleri oluşturmak ve saklamak için özel bir şifreleme cihazı olan donanım cüzdanıdır [9]. Coinbase müşterilerinin % 98'inin kripto paraları çevrimdışı saklanmaktadır [37]. Donanımsal cüzdanın en temel görevi özel anahtarları güvenilir olmayan internet, bilgisayar ve mobil ortamlarından tamamen izole etmektir. Bunun da ötesinde akıllı kart donanımsal cüzdanlar özel anahtarları hiçbir şekilde dışarıya vermediğini garanti eden özel güvenli element alanına sahiptir. Tüm imzalama, şifreleme ve şifre çözme işlemleri bu çipte gerçekleşir. Bundan dolayı kripto para hesaplarına ait özel anahtarı saklayacak çevrimdışı cüzdan çeşidi olarak büyük ilgi görmektedir. Bu tez çalışmasının ana odağı da donanımsal akıllı kart cüzdan ile ilgilidir.

TABLO 3.1: Cüzdan Çeşitlerinin Karşılaştırılması

Cüzdan Türü	**Açıklaması**	**Özellikleri**	**Bilenen Örnekleri**
Masaüstü Cüzdan	-Özel anahtarlar, bitcoin üzerinde daha fazla kontrol sağlayan ana makinede saklanır.	-Yüklemesi hızlı ve kullanması oldukça pratikdir.	-Electrum -Exodus -Armory -MultiBit
Mobil Cüzdan	-Mobil cihazlarda kullanılır. -Masaüstü Cüzdanı veya Web Cüzdanı olarak çalışan bir uygulama olabilir.	-Yaygın kullanım kapasitesine sahiptir. -QR kod ve NFC kullanımıyla gelişmiş özelliklere sahip olma kapasitesine sahiptir.	-Jaxx -Mycelium -Edge
Web Cüzdan	-Özel anahtarlar, bir hesap aracılığıyla erişilebilen üçüncü taraf sunucularda saklanır.	-Güvenliği, müşterinin bitcoinleri üzerinde tam kontrole sahip olan servis sağlayıcısına emanettir.	-Guarda Wallet -Freewallet -Cryptonator
Kağıt Cüzdan	-Özel ve ortak anahtarların yazılı olduğu kâğıttır.	-Pratik değildir. -Korunduğu takdirde oldukça güvenlidir -Web üzerinden kolayca üretilebilir.	
Donanımsal Cüzdan	-Anahtarların üretilmesi, saklanması ve imzalama işlemi güvenli bir donanım cihazında yapılır.	-Cüzdanlar arasında en güvenlisidir.	-Ledger Nano -BlueWallet -KeepKey -Trezor -CoolWallet

Donanımsal cüzdanları anahtar üretimine göre deterministik ve deterministik olmayan şeklide ikiye ayrılabilir.

3.5.1 Deterministik Olmayan Cüzdan

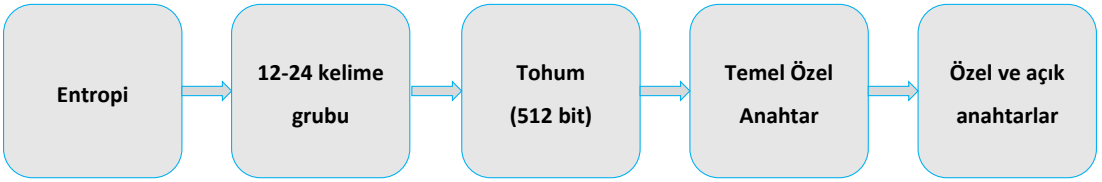
Deterministik olmayan cüzdanda özel anahtar ve bunlara karşılık gelen ortak anahtarları üretmek için özel bir disiplin uygulanmaz ve üretilen anahtar çiftleri birbirinden bağımsızdır. Diğer bir ifadeyle özel anahtarlar tek bir anahtardan türetilmez yani belli bir algoritmadan geçirilmez, bu yüzden bu cüzdanlara non-deterministic yani deterministik olmayan cüzdan denir. Bu cüzdan çeşidinde anahtarların yedeğini almak için hepsiyle tek tek ilgilenmek gereklidir.

3.5.2 Deterministik Cüzdan

Bitcoin ve neredeyse diğer tüm popüler kripto para birimleri, işlemleri imzalamak ve doğrulamak için eliptik eğri şifreleme (ECC) kullanır. Bitcoinin ilk zamanlarında kullanıcının sadece özel ve ortak anahtar olan bir çift anahtarı bulunur ve bir işlemi imzalayarak fonunu ya da parasını başka bir kullanıcının ortak anahtarına (adresine) aktarmak için özel anahtarı kullanır. Gönderen kullanıcı, bir işlemi gerçekleştirebilmesi için alıcının ortak anahtarını bilmelidir ve tüm kullanıcılar ortak anahtarlarını adres adı verilen belirli bir biçimde yayınlar. Bu nedenle, bir kullanıcı özel anahtarını gizli tutar ve adresini diğer tüm kullanıcılara yayınlar. Gizlilik veya başka bir deyişle anonimlik, Bitcoin ve Bitcoinin alternatifleri olarak bilinen altcoinler için bir mahremiyet zorluğudur, çünkü tüm işlem geçmişi blokzincirindedir, sonuçta bilgisayar korsanları bir kullanıcının tüm satın alma ve transfer faaliyetlerini ortaya çıkartabilir. Bu sorunu çözmek için, kullanıcı her işlemdeki adresini değiştirebilir ve her seferinde yeni bir özel anahtar oluşturabilir. Bu şekilde, sadece yapılan işlemlerin geçmişi izlenerek kullanıcının takip edilememesi sağlanır. Blokzincirde gizliliğin maksimum derecede sağlandığı en pratik yöntem budur [38]. Bununla birlikte, her işlem için rastgele bir özel anahtar oluşturmak, cüzdan için zorluğa neden olur, çünkü birçok anahtarı korumalıdır. Hiyerarşik Deterministik (HD) cüzdan yeni özel anahtarlar oluşturmak için deterministik bir algoritma kullanarak ve hiyerarşik şekilde bu sorunu çözer [39]. HD cüzdanında özel anahtarlar ağacı bulunur ve her bir düğümdeki özel anahtarlar bir üst (parent) öğesinden türetilir. Bu ağacın kökü, "ana ya da temel tohum" adı verilen rastgele bir sayıdan türetilen "ana özel anahtar" ya da “temel özel anahtar” adı verilen özel bir anahtardır. Başka bir deyişle, ana tohuma sahip olan herkes tüm anahtar ağacını oluşturabilir ve tüm özel anahtarları türetebilir. Sonuç olarak, HD cüzdanda kullanıcının tüm özel anahtarları korumak için yalnızca bir tohum

değerini güvende tutması gerekir. Deterministik cüzdan 12 ila 24 kelimeden türetilir. Bu kelimelere hatırlatıcı kelimeler denmektedir. Çünkü "Piece wonder vapour... " gibi hatırlanması veya kaydetmesi basittir. Cüzdan kaybolursa veya herhangi bir sebepten kullanılamaz hale gelse bile bu kelimelerle tekrar restore edilebilir. Bu kelimelerden 512 bit tohum oluşturulur. Bu 512 bitlik tohum dosyasından ise temel özel anahtar (master private key) oluşturulur.

Deterministik cüzdan Sıralı Deterministik (Sequential Deterministic: SD) ve Hiyerarşik Deterministik (Hierarchical Deterministic: HD) cüzdan olmak üzere ikiye ayrılır.



Şekil 3.2: Hiyerarşik Deterministik Cüzdanda Anahtarların Üretim Aşamaları

3.5.2.1 Sıralı Deterministik Cüzdan

SD cüzdanda özel anahtarlar SHA256(tohum + n) fonksiyonuyla özet alınarak üretilir. Buradaki n indeks numarasıdır. 0'dan başlar ve ihtiyaç duyulan adres sayısına göre artırılır.

3.5.2.2 Hiyerarşik Deterministik Cüzdan

Hiyerarşik deterministik cüzdanın amaçları arasında şunlar sayılabilir;

- Sadece bir tane temel anahtar olacak ve bu anahtar ile cüzdana tekrar ulaşılabilecek.
- Yapılacak her ödeme için ayrı adresler kullanılacak ve bu sayede gizlilik sağlanacak.
- Size gelecek ödemeler için ödeme adresleri otomatik olarak üretilip değiştirilecek (Opsiyonel)

Bu amaçları sağladığı için en çok kullanılan ve tavsiye edilen cüzdan çeşididir. Bu cüzdan çeşidi BIP-32 protokolü denilen yapıya göre oluşturulur. BIP-32 genel bir hiyerarşik cüzdanın nasıl olması gerektiğini tarif eder [39]. Dünya genelinde cüzdanlar için bir standart olmamakla birlikte BIP-32, BIP-44, BIP-39 dünya genelinde endüstri standardı olarak kabul edilme yolunda ilerlemektedir. Tanınmış çoğu cüzdan üreticisi cüzdanlarını bu yapıya göre implemente eder (LedgerHQ, Trezor, CoolWallet gibi).

BIP

BIP kısaltması "Bitcoin Improvement Proposals" kelimelerinin baş harflerinden oluşturulmuştur. Bitcoin iyileştirme önerisi anlamına gelse de diğer kripto para türleri de bu iyileştirmelerden yararlanabilmektedir. Bilindiği üzere, Bitcoin tek bir varlık veya şirket tarafından kontrol edilmemektedir ve bu nedenle Bitcoin protokolünde veya kodunda iyileştirmeler önerecek resmi bir yapı da yoktur. Herhangi bir geliştirici veya dünyanın herhangi bir yerinden herhangi bir kişi BIP önerebilir. Bu önerinin uygulanıp uygulanmayacağı kararı veya oyu tamamen Bitcoin kullanıcı, madenci, geliştirici ve yatırımcı topluluğuna bağlıdır. İyileştirme önerisi olan kişi ya da kişiler fikirlerini ve belgelerini Bitcoin geliştiricileri için belirlenen posta listesine sunmaktadır. Fikir, geliştiriciler tarafından tartışılıp kabul edildikten sonra yayınlanır. Yayınlanan BIP'in nihai duruma gelmesi için ise Bitcoin kullanıcı, madenci, geliştirici, yatırımcı topluluğunda çoğunluğun BIP hakkında fikir birliğine varması gerekmektedir. 342 adet iyileştirme önerisinden 37'si nihai duruma ulaşmıştır [40] [41]. Bu çalışmada cüzdan için önerilen BIP-32, BIP-39 ve BIP-44 üzerinde durulacaktır.

HMAC (Özet Tabanlı Mesaj Doğrulama Kodu)

HMAC, tek yönlü kriptografik özet fonksiyonuna dayanan belirli bir özet doğrulama kodu (MAC) şemasıdır. HMAC'in herhangi bir kişi tarafından gizli anahtar kullanılmadan hesaplanabilen açık özet fonksiyonundan farkı, SHA-2 veya MD5 gibi özet fonksiyonları kullanılarak MAC mekanizması ile birleştirilip gizli anahtarla bir mesajı doğrulamada kullanılmasıdır. HMAC şeması, doğum günü saldırıları, uzatma saldırıları ve anahtar kurtarma saldırıları gibi bazı saldırılara karşı çarpışma direnci veya koruma sağladığı için önemli bir avantaja sahiptir. M mesajıyla aynı özeti verecek faklı bir M' mesajının bulunmasını zorlaştırıcı güçlü çarpışma direnç özelliği ve tek yönlü bir fonksiyon olması HMAC'i güçlü kılmaktadır.

Bölüm 4

Blokzincir Tabanlı Donanımsal HD Cüzdan Uygulaması

Bu bölümde HD cüzdanın çalışma mekanizması incelenecek ve bu mekanizmaya göre işleyen örnek bir uygulama üzerinde durulacaktır.

4.1 HD Cüzdanın Çalışma Mekanizması

HD cüzdan için ilk olarak 512 bitlik bir tohum gereklidir. Bu 512 bitin nasıl olması gerektiğini BIP-39 belirler. 512 bit tohum seçildikten sonra temel anahtar ve alt anahtarların üretilmesi aşamasına geçilir. Bu anahtarların nasıl üretilmesi gerektiği ise BIP-32 önerisinde sunulmuştur. BIP-32 anahtarlarının hangi amaçla kullanılacağını tarif eden BIP-44 önerisi de bu çalışmada anlatılacaktır.

4.1.1 BIP-39

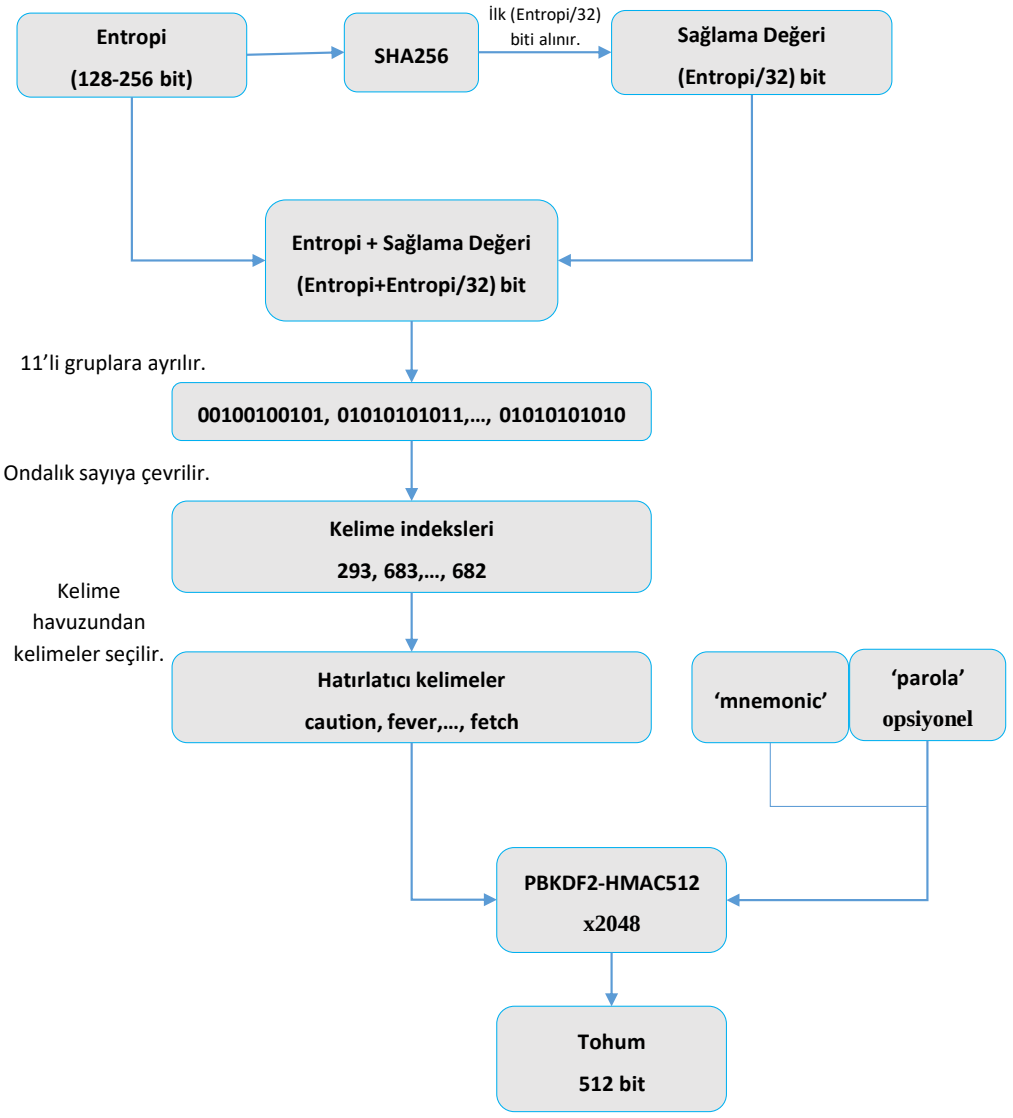
BIP-39, BIP-32'de kullanılacak 512 bitlik tohumun oluşturulması için bir takım aşamalar önermektedir. Bu önermenin amacı kullanıcının hatırlamakta fayda sağlayacağı kelime grupları oluşturarak cüzdanına ait tüm anahtarları geri getirebilmesini sağlamaktır. Uzun anahtarlar saklamak yerine kullanıcı dostu bu kelimeler tercih edilebilir. 512 bit anahtarı tek tek saymak yerine bu kelimeler kâğıda yazılabilir veya telefonla kolayca söylenerek

aktarılabilir. Sonrasında bu kelimelerden BIP-32'de veya benzeri sistemlerde kullanılacak tohum ve anahtarlar oluşturulabilir.

Bu kelimeler oluşturulurken;

1. 256-512 bit arası 32'nin katı olan rastgele bir entropi üretilir. Entropinin, daha önce hiç kimsenin üretmediği veya gelecekte üretemeyeceği çok büyük bir rasgele sayı olduğunu düşünebilirsiniz.
2. Bu entropinin SHA256 özeti alınır. Sağlama değeri olarak kullanılacak ilk (entropi /32) bit alınır. Sağlama değeri hatırlatıcı kelimeler oluşturulurken hata bulmada işe yaracak olan parmak izi değeridir.
3. Sağlama değeri entropinin sonuna eklenir. (entropi + sağlama değeri)
4. Yeni oluşan bu değer 11'li bitler şeklinde ayrılır. En büyük değer $2^{11} = 2048$ 'dir. Entropiyi 32'nin katı seçme sebebi, sonuna sağlama değeri eklendiğinde 11'li gruplara tam bölünebilmesi içindir.
5. 11 bitlik bu değerler 0 ile 2048 arası tam sayıya çevrilir. Çıkan değer kelime indeksidir.
6. 2048 adet kelime havuzundan indekse göre kelimeler seçilir. Seçilen bu kelimelere hatırlatıcı kelimeler denir.
7. Ekstra güvenlik isteyenler için opsiyonel olarak hatırlatıcı kelimelerle beraber parola eklenebilir. Parola girildiği takdirde parola "mnemonic" etiketi ile beraber tuzlama (salt) işleminden geçirilir. Parola istenmezse bu değer girilmez.
8. Hatırlatıcı kelimeler, "mnemonic" etiketi ve varsa parola ile beraber PBKDF2-HMAC512 fonksiyonuna sokulur. Bu fonksiyon için girilen tekrarlama sayısı 2048 ve elde edilecek sonuç değeri uzunluğu ise 64 bayt olarak girilir. Yani 2048 kez bu fonksiyon işletilir ve PBKDF2-HMAC512 fonksiyonu 64 bayt (512 bit) sonuç çıkartır.
9. Elde edilen sonuca tohum denmektedir ve BIP-32 önerisinde kullanılmaktadır.

PBKDF2 fonksiyonu temelinde özet fonksiyonudur ve tekrarlama sayısı girilerek kaba kuvvet saldırılarını zorlaştırıcı bir etkisi vardır. Ek olarak, PBKDF2 yukarıda belirtildiği gibi parola ile ikinci bir giriş yapmanızı sağlar.



Şekil 4.1: BIP-39'a göre tohum oluşturma adımları

Aslında BIP-39, entropi değerini verilen kelime havuzuna göre haritalandırmaktadır. Güvenlik için 24 kelimeden oluşan entropi seçilmesi tavsiye edilir. 128 bit entropi için 12 kelime, 256 bit entropi için 24 kelimedir seçilir. Buna göre 2048 bit kelimeden 128 bit entropi için; $2^{128} = 340282366920938463463374607431768211456 = 3,4 * 10^{38}$, 256 bit entropiden ise $2^{256} = 115792089237316195423570985008687907853269984665640564039457584007913129639936 = 1,16 * 10^{77}$ farklı kombinasyon elde edilir.

Bu değerlerin büyüklüğü kullanıcıların aynı entropi değerini seçme olasılığını büyük ölçüde düşürmektedir. Entropi için her zaman güvenli bir rasgele sayı üreteci kullanılmalıdır. Programlama dillerindeki varsayılan "rastgele" fonksiyonunun yeterince rastgele

olmadığı düşünülmektedir [42].

Tablo 4.1: Entropi-Kombinasyon Tablosu

Entropi	**Kelime sayısı**	**Varyasyon**	**Varyasyon**
128	12	2^{12}	$\sim 3{,}4 \times 10^{38}$
160	15	2^{15}	$\sim 1{,}46 \times 10^{48}$
192	18	2^{18}	$\sim 6{,}28 \times 10^{57}$
224	21	2^{21}	$\sim 2{,}70 \times 10^{67}$
256	24	2^{24}	$\sim 1{,}16 \times 10^{77}$

4.1.2 BIP-32

BIP-32 genel bir hiyerarşik cüzdanın nasıl olması gerektiğini tarif etmektedir. BIP-39 seedden Temel Özel Anahtar (Master Private Key) ve Zincir Kodu (Master Chain Code) üretilir.

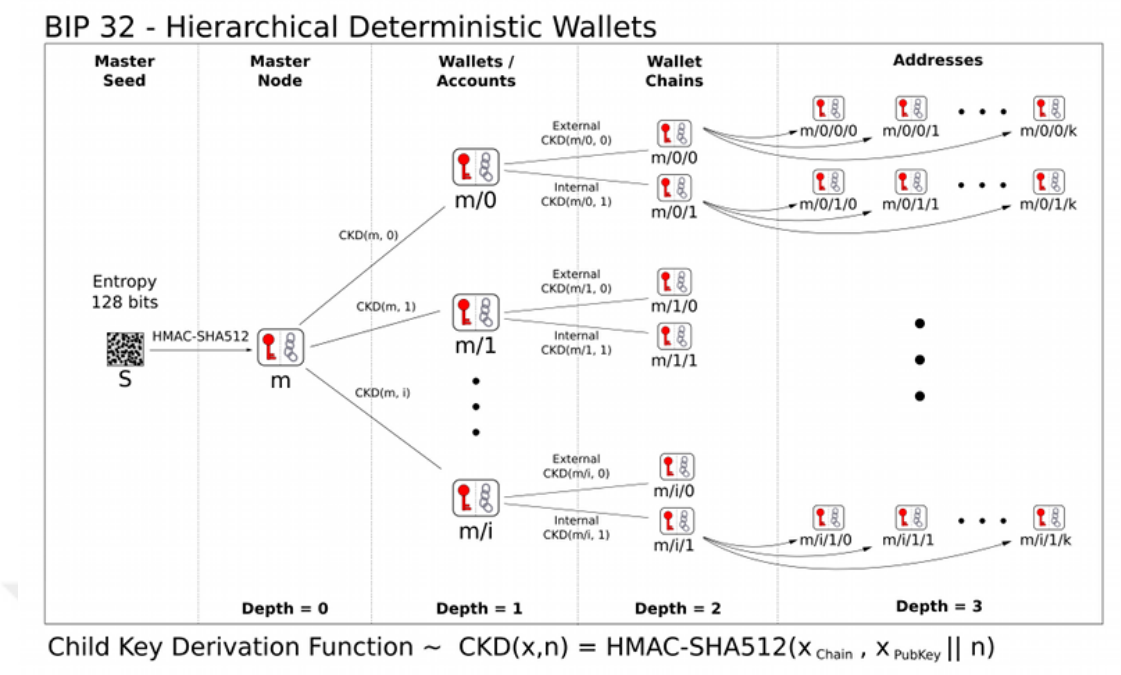
kripto para transferi için adres gereklidir. Bu adrese alınan kripto parayı gönderebilmek ya da harcayabilmek için ise ilgili özel anahtara ihtiyaç vardır. Tipik olarak, kripto para cüzdanları, kullanıcılarının birden fazla adres oluşturmasına izin vermektedir. Böylece birden fazla adres kullanan bir kullanıcının tek bir adres kullanan birine göre takip edilebilirliği daha zor olduğu için anonim kalmasına yardımcı olacaktır. Çünkü blokzincirde yaptığınız tüm işlemleri tek bir adreste toplandığınızda herhangi bir kişinin size gönderdiği para transferiyle, adresinize gelen ve adresinizden çıkan tüm transfer işlemlerini görebilmesine neden olacaktır. Her ne kadar yapılan işlemler anlaşılır düz metin formatında olmasa da bu, mahremiyet için istenen bir durum değildir. Tek bir adres ve anahtarla işlem yapmanın iyi olan tarafı ise yedekleme işlemidir. Tek adresin yedeklenmesi kolaydır.

Birden fazla adres kullanımı için ilk düşünce olarak kullanıcının her adres talebinde rastgele yeni bir adres ve özel anahtar üretilir. Bitcoin-qt çekirdeği bunu yapmaktadır. Bu yöntemin dezavantajlarından biri, cüzdanın tüm adresleri / anahtarları saklaması gerektiğidir. Bu yüzden kullanıcı, yeni oluşturulan adresleri / anahtarları kaybetmemek

için her yeni adres oluşturduğunda cüzdanını yedeklemelidir ve pratikliği azaltmaktadır. Bu zorluğa çözüm olarak BIP-32 standardı önerilmiştir.

Bir BIP-32 destekli cüzdan, kullanıcının talebi üzerine adresleri / anahtarları tekrar tekrar yedeklemeye gerek kalmadan yeni adresler oluşturmasına izin vermektedir. Yeni adres üretilirken ikinci adresin / anahtarın rastgele değil, aslında birincinin bir türevi, benzer şekilde üçüncüsü ikinciden türetilmiş olmasıdır. Bütün bunlar, temel özel anahtarı kullanmadan cüzdandaki adreslerin tahmin edilememesini sağlamak için bazı kriptografik algoritmalar ile desteklenir [39].

BIP-32 ağaç yapısına benzetilmektedir. En başta kök ya da temel anahtar diye tabir edilen anahtar ve bundan türetilen üst ve alt anahtar çeşitleri bulunmaktadır (Bu çalışmada temel anahtar denilecektir). Üst anahtarlardan alt anahtarlar üretilir, alt anahtarlardan onların da altı olan daha derine doğru alt anahtarlar üretilir ve bu şekilde devam edip dallanır. Fakat alt anahtarlardan üst anahtarlar tekrar geri getirilemez, sistem sadece üstten alta tek yönlü anahtar üretme şeklinde çalışır. Bunun sebebi üst anahtarlardan alt anahtarlar üretilirken üst anahtarların tek yönlü fonksiyon olan HMAC-SHA512 özet algoritmasından geçirilmesidir. Aşağıdaki şekil cüzdandaki anahtarların genel yapısı hakkında fikir vermektedir. HD cüzdana ait anahtarlar bu yapıya göre oluşturulmuştur. Entropiden tohum, tohumdan ise temel anahtar üretilir. En üst anahtar olan temel anahtar m ile gösterilir. Temel anahtardan bir alt "m/0, m/1, ... m/i", iki alt "m/0/0, m/0/1, ... m/i/0, m/i/1" şekline devam eden alt anahtarlar üretilir. Ne kadar alta inildiği derinlikle gösterilir. Burada i indeks numarasıdır ve bir üst anahtarın kaçıncı dalı olduğunu gösterir.



Şekil 4.2: BIP-32 uyumlu hiyerarşik deterministik cüzdanların anahtar yapıları [39]

Cüzdana ait anahtarların üretilmesini gösterebilmek için ilk temel anahtar oluşturulduktan sonra üst anahtardan alt anahtarları türetmekte kullanılan dört metot üzerinde durulacaktır. Bu metotlar:

- Normal Alt- Geniş Özel Anahtar Hesaplama
- Zorlaştırılmış Alt- Geniş Özel Anahtar Hesaplama
- Normal Alt- Geniş Açık Anahtar Hesaplama
- Zorlaştırılmış Alt-Geniş Ortak Anahtar Hesaplama

Normal ve zorlaştırılmış kavramları kullanılan indeks numarasının aralığı ile ilgilidir. İndeks numarası 0 ile 2147483647 ($2^{31}-1$) arasında olanlara normal, 2147483647 ($2^{31}-1$) ile 4294967295 ($2^{32}-1$) arasında olanlara zorlaştırılmış (hardened) anahtar denilmektedir. Anahtarlardaki "Geniş" ya da "genişletilmiş" ibaresi, anahtarın sonuna ilgili zincir kodunun eklenmiş olduğunu gösterir. Geniş anahtar hiyerarşik deterministik cüzdanda yeni anahtarlar türetmeyi sağlayan özel veya ortak anahtardır. Tek bir üst geniş özel anahtar, bu anahtarın altındaki tüm özel ve ortak anahtarların kaynağıdır. Bu yüzden tek bir üst geniş özel anahtar ile bu üst anahtara ait tüm alt özel ve ortak anahtarlar tekrar türetilebilir. Ayrıca bu üst özel geniş anahtara karşılık gelen üst geniş ortak

anahtarla aynı ortak anahtarları türetmek mümkündür. Bu özelliği farklı iş senaryolarına imkân tanır.

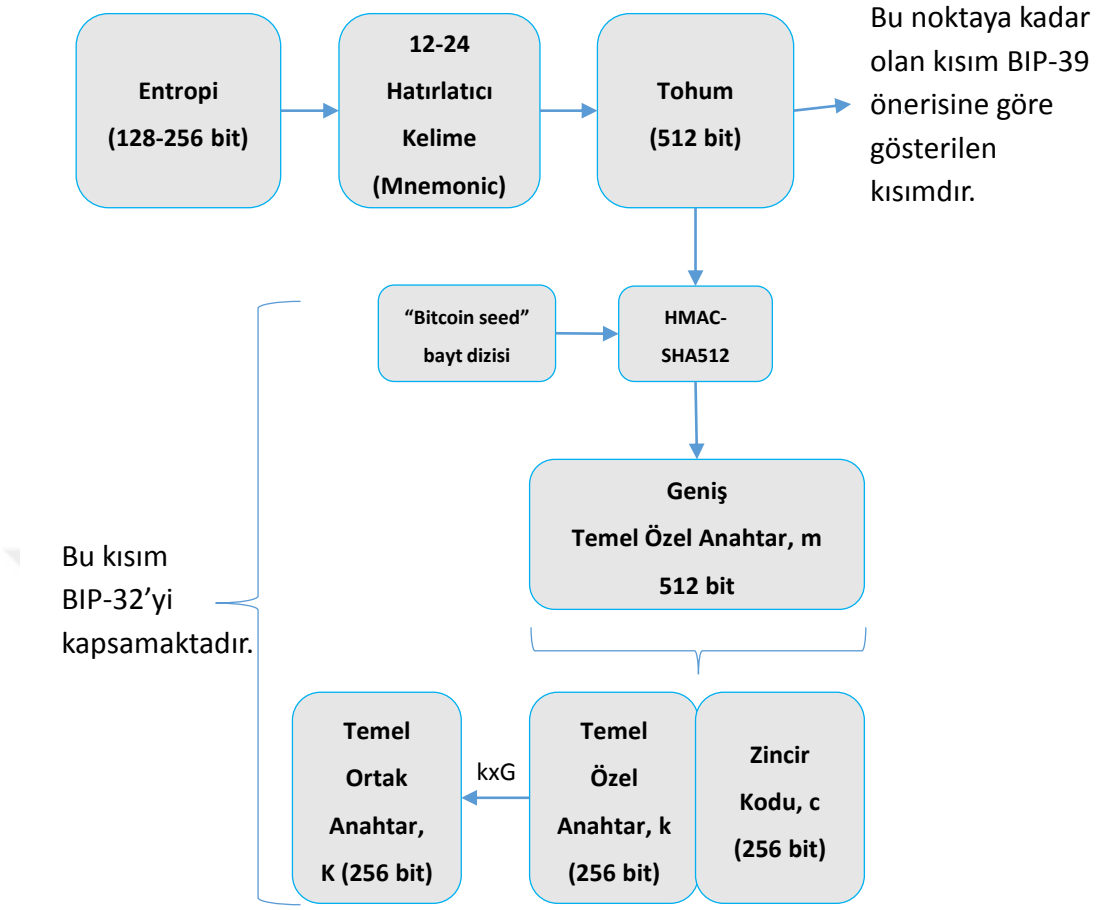
4.1.2.1 Temel Anahtarın Oluşturulması (Master Key Derivation)

Sık sık yedekleme gerekliliğinden kurtulmak için deterministik cüzdanlar, tek bir tohumdan diğer tüm anahtarları türetmeye yarayan temel anahtar oluşturur. 128 bit ile 512 bit arası uzunluğunda (512 bit tavsiye edilir) bir tohum üretilir ve çevrimdışı bir anahtar deposunda saklanır. Deterministik cüzdanlarda, türetilmiş tüm anahtarları kurtarmak için tohumun oluşturulduğu ilk zamanda tek bir yedekleme yeterli olacaktır. Fakat deterministik cüzdanlar bazı kullanım senaryolarına uygun olmayan tek bir anahtar zincirini işletir. Bu yüzden, BIP-32 standartlarında, HD cüzdanların bir ağaç yapısındaki anahtarların türetilmesi önerilmektedir, böylece HD cüzdanlar farklı iş senaryoları için birden fazla anahtar zincirini oluşturmaya imkân tanır. Örnek olarak genel ve özel ECDSA anahtarlarının oluşturulması birbirinden ayrılabilir. Yani, özel anahtarları oluşturma yeteneği olmadan, bir ECDSA anahtar çiftinin sadece ortak ECDSA anahtar yarısını oluşturmak mümkündür. Böylece, halka açık bir sistemde, ele geçirildiği takdirde saldırgana bu adreslerde alınan bitcoini harcama olanağını vermeyecek alıcı adresler oluşturulmuş olur. Çevrimdışı farklı bir makinede bu ortak anahtarlara karşılık gelen özel ECDSA anahtarları oluşturabilir ve işlemleri imzalayabilir. Bu şekilde özel ve genel anahtarların üretimi birbirinden ayrılmış olacaktır. BIP-32, secp256k1'in eliptik eğrisini kullanarak hiyerarşik anahtarların türetilmesini gösterir. Aşağıda, alt anahtar türetme fonksiyonunda (CKD) tanımlı temel anahtarın nasıl genişletileceği gösterilecektir.

Anahtarları genişletmek için kullanılan ve c olarak gösterilen 32 baytlık zincir kodu (chain code) vardır. Özel ve ortak anahtar olsa bile zincir kodu olmadan alt anahtarları üretmek mümkün değildir. Genişletilmiş özel anahtar (k + c) ve indeks i ile genişletilmiş (geniş) alt özel anahtarı hesaplamak mümkündür. Burada k özel anahtarı, c zincir kodunu temsil eder. Aynı şekilde genişletilmiş açık anahtar (K + c) ve indeks i ile karşılık gelen genişletilmiş alt ortak anahtar hesaplanabilir. Nokta (p), secp256k1 üreteç noktasının p tamsayısıyla eliptik eğri nokta çarpımından kaynaklanan koordinat çifti olarak tanımlanır. P = p x G, G baz noktası ya da üreteç olarak ifade edilir.

Bir temel anahtar çifti (m, M) 128 bit ile 512 bit arası uzunluğunda bir tohumdan üretilir.

1. 128 bit ile 256 bit arası uzunluğunda (256 bit tavsiye edilir) bir entropi üretilir.
2. Entropiye göre BIP-39'a uyumlu olması için 12 ile 24 adet hatırlatıcı kelime grubu oluşturulur.
3. Kelimelerden 512 bit tohum oluşturulur.
4. $I = m = HMAC-SHA512(s, tohum)$, s="Bitcoin seed" olan bir bayt dizisidir. Bu bayt dizisi BIP-32 uyumlu olması için verilen bir etiket değeri olarak düşünülebilir [39].
5. Il sol ve Ir sağ sırayı göstermek üzere I 32 baytlık 2 parçaya ayrılır.
6. Il temel özel anahtardır ve k ile gösterilir.
7. Ir temel zincir kodunu temsil eder ve c ile gösterilir.
8. Temel ortak anahtar K, k temel özel anahtarına karşılık gelen eliptik eğrideki noktasıdır. $K = kxG$
9. Temel ortak anahtarın sonuna temel zincir kodu eklenerek geniş temel ortak anahtar elde edilir. $M = (K + c)$



Şekil 4.3: Temel Anahtar Oluşturulması

HMAC fonksiyonu rastgele yeni bir bayt kümesi olan özeti oluşturmak için anahtar ve veri olmak üzere iki parametre almaktadır. Yukarıdaki şekilde temel anahtarın hesaplanma adımları gösterilmiştir. Bu adımların python programlama diliyle yazılmış örnek kod parçası Ek-A.1'da verilmiştir. Bu uygulamanın verdiği örnek çıktı Şekil 4.4'deki gibidir;

```
Temel Anahtar Hesaplama: m
Tohum (hex):
72c2876d53e48d28bb9ede3355478281301b25b9ecc243eeb66a341976e777594c148
41e0a9ceef8f6e31d454ce2edc8b88a06d61500fc7ac6627101cfc884f1
Temel geniş anahtar (hex):
6b28c78e47882c02ee7a5301eadb2a7eba4b07d92f91d255505391f1a1aa0b09c5ae0
af6d6e250c7f531ba680a88df619972cdf1f67e2c6c16bb7765c0ebddf5
Temel özel anahtar (hex):
6b28c78e47882c02ee7a5301eadb2a7eba4b07d92f91d255505391f1a1aa0b09
Temel zincir kodu  (hex):
c5ae0af6d6e250c7f531ba680a88df619972cdf1f67e2c6c16bb7765c0ebddf5
Temel ortak anahtar (hex):
02d148d7ce35a7e514598235ce7bc1affea7173597a550a3a31e1ae72f053e964f
```

Şekil 4.4: Örnek Temel Anahtar

Temel anahtardan ya da üst anahtardan alt anahtarları türetmek için dört metot bulunmaktadır, sonraki dört başlıkta bu metotlar işlenecektir.

4.1.2.2 Normal Alt-Geniş Özel Anahtar (Normal Child-Extended Private Key)

Bu anahtara normal denmesinin sebebi HMAC-SHA512 fonksiyonunda kullanılacak olan indeks numarasının 0 ile 2147483647 ($2^{31} - 1$) arasında seçilmesidir. İndeks numarası 2147483647 ile 4294967295 ($2^{32} - 1$) arasında seçilip hesaplanan alt anahtarlara, zorlaştırılmış (hardened) alt anahtar denilmektedir. Geniş anahtar ise anahtara zincir kodunun anahtarın sağına eklenmesiyle oluşur.

1. Üst ortak anahtar seçilir. Üst ortak anahtar, üst özel anahtara karşılık gelen anahtardır.
2. İndeks numarası 0 ile 2147483647 ($2^{32} - 1$) arasında seçilir. Bu aralık normal alt anahtarların üretilmesi için belirlenen aralıktır. İndeks numarası artırılarak o üst anahtara bağlı farklı alt anahtarlar üretilir.
3. HMAC-SHA512 fonksiyonuna anahtar ve veri parametreleri verilir. Bu parametreler;
    - Veri = üst ortak anahtar + indeks (yan yana eklenir.)

- Anahtar = üst zincir kodu

4. HMAC-SHA512 fonksiyonundan çıkan 512 bitlik sonucun son 256 bitlik kısmı, Ir alt zincir kodudur. Bir sonraki alt anahtarı (grand child) hesaplamak için bu zincir kodu kullanılacaktır. Başka bir deyişle bir sonraki alt anahtar hesabında bu zincir kodu üst zincir kodu olacaktır.

5. HMAC-SHA512 fonksiyonundan çıkan 512 bitlik sonucun ilk 256 bitlik kısmı, Il ile üst özel anahtar toplanır ve Secp256k1 eğrisinde belirlenen dereceye göre modu alınır. Secp256k1 eğri derecesi = n = 115792089237316195423570985008687907852837564279074904382605163141518161494337
   Normal alt özel anahtar = (hmac_sonucu[0 : 32] + üst özel anahtar) $\%n$

6. Normal alt özel anahtara karşılık gelen alt özel ortak anahtar hesaplanır. Bu genellikle alt özel anahtarın üreteç ile çarpımıyla bulunur.

7. Yeni üretilen alt özel anahtar ile alt zincir kodunun yan yana eklenmesiyle normal alt – geniş özel anahtar elde edilir.

8. Üretilen alt özel anahtara karşılık gelen ortak anahtara zincir kodu eklenerek normal alt geniş ortak anahtar hesaplanabilir.

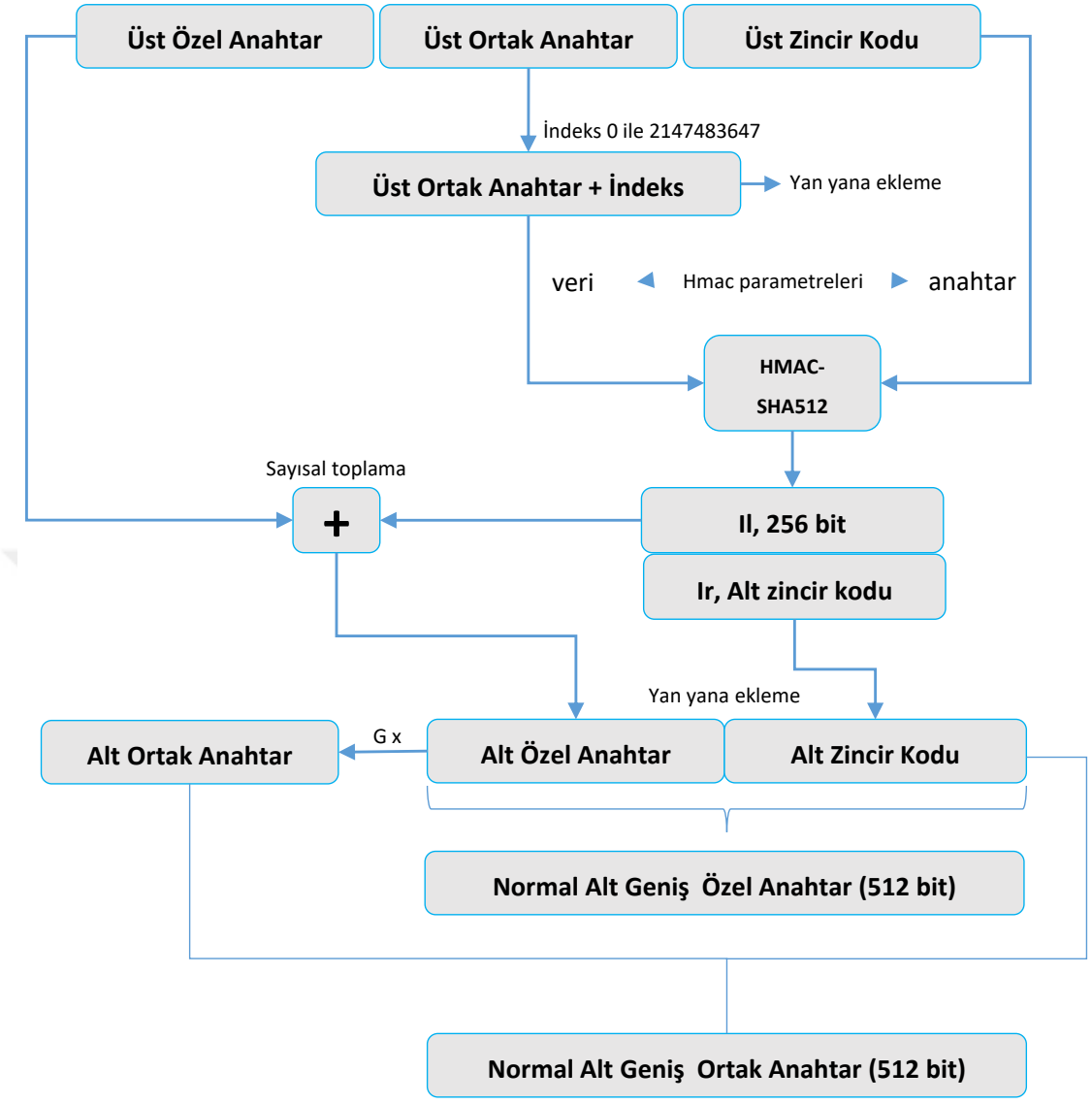
Bu adımların python programlama diliyle yazılmış örnek kod parçası Ek-A.2'da verilmiştir. Bu uygulamanın verdiği örnek çıktı Şekil 4.5'deki gibidir.

```
Normal-(Alt Geniş Özel Anahtar) Hesaplama: m/0
Alt Zincir Kodu (hex):
8d7d2ef83c1233a64b27680fb08152291ceecc89592239032ed82cd860255459
Alt özel anahtar (hex):
6de4f17275b3b178b002d1beaa8db54cee7a39bd50aa92d990aed445db561148
Alt ortak anahtar (hex):
033e0728adb800da553426e653eeaf7ec25603cbb14f7a964ee6e0ee45310670a2
```

Şekil 4.5: Normal Alt Geniş Özel Anahtar Hesaplama Örneği

Aşağıda normal-alt geniş özel anahtarın hesaplanma adımları şekil ile gösterilmiştir.

Özet olarak üst geniş özel anahtar ve indeks numarasını HMAC fonksiyonundan geçirerek (üst ortak anahtar + indeks, zincir kod) o indekse ait normal alt anahtarı üretmek mümkündür.



Şekil 4.6: Normal-Alt Geniş Özel Anahtar Hesaplama

4.1.2.3 Zorlaştırılmış Alt-Geniş Özel Anahtar (Hardened Child Extended Private Key)

Zorlaştırılmış alt özel anahtarın hesaplanması aşağıda verilmiştir.

1. Üst özel anahtar seçilir.
2. İndeks numarası 2147483647 ($2^{31} - 1$) ile 4294967295 ($2^{32} - 1$)arasında seçilir. Bu aralık zorlaştırılmış alt anahtarların üretilmesi için belirlenen aralıktır. İndeks numarası artırılarak o üst anahtara bağlı farklı alt anahtarlar üretilir.

3. HMAC-SHA512 fonksiyonuna anahtar ve veri parametreleri verilir. Bu parametreler;

   - Veri = üst özel anahtar + indeks (yan yana eklenir.)
   - Anahtar = üst zincir kodu

4. HMAC-SHA512 fonksiyonundan çıkan 512 bitlik sonucun son 256 bitlik kısmı, Ir alt zincir kodudur. Bir sonraki alt anahtarı (grand child) hesaplamak için bu zincir kodu kullanılacaktır. Başka bir deyişle bir sonraki alt anahtar hesabında bu zincir kodu üst zincir kodu olacaktır.

5. HMAC-SHA512 fonksiyonundan çıkan 512 bitlik sonucun ilk 256 bitlik kısmı, Il ile üst özel anahtar toplanır ve Secp256k1 eğrisinde belirlenen dereceye göre modu alınır. Secp256k1 eğri derecesi = n = 115792089237316195423570985008687907852837564279074904382605163141518161494337
   Zorlaştırılmış alt özel anahtar = (hmac_sonucu$[0:32]$ + üst özel anahtar) $\%n$

6. Zorlaştırılmış alt özel anahtara karşılık gelen alt özel ortak anahtar hesaplanır. Bu genellikle alt özel anahtarın üreteç ile çarpımıyla bulunur.

7. Yeni üretilen alt özel anahtar ile alt zincir kodunun yan yana eklenmesiyle zorlaştırılmış alt – geniş özel anahtar elde edilir.

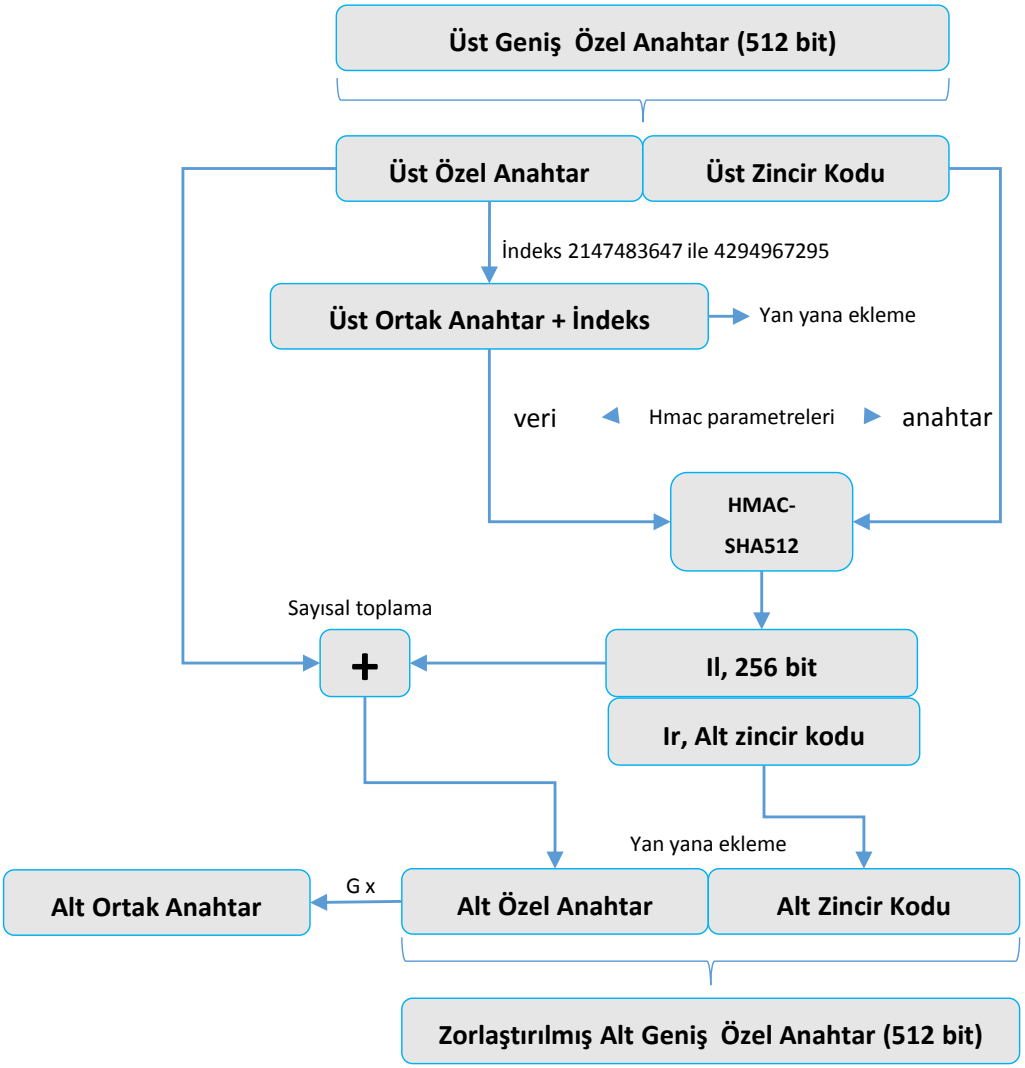
8. Üretilen zorlaştırılmış alt özel anahtara karşılık gelen ortak anahtara, alt zincir kodu eklenerek zorlaştırılmış alt geniş ortak anahtar hesaplanabilir.

Bu adımların python programlama diliyle yazılmış örnek kod parçası Ek-A.3'de verilmiştir. Bu uygulamanın verdiği örnek sonuç;

```
Zorlaştırılmış-(Alt Geniş Özel Anahtar) Hesaplama: m/2147483648
Alt Zincir Kodu (hex):
c797c4f35b0b67114b4b1c38b138354765859b63d741480cbd15c82b23799fe4
Alt özel anahtar (hex):
08726e7b5eec361540ebd0ded5d3ed6df5dc74f6a18dcaa86d808c4bd5fba020
Alt ortak anahtar (hex):
0328d2a49dbdde8b52bf4655376b8d2e7e9e662233d7c1e30db10586d7e341c81c
```

Şekil 4.7: Örnek Zorlaştırılmış Alt Geniş Özel Anahtar

Aşağıdaki zorlaştırılmış alt geniş özel anahtarın hesaplanma adımları şekil ile gösterimi mevcuttur.



Şekil 4.8: Zorlaştırılmış Alt- Geniş Özel Anahtar Hesaplama

Normal alt ortak anahtarı, özel anahtar olmadan üst ortak geniş anahtar ile hesaplamak mümkündür. Fakat zorlaştırılmış alt ortak anahtarı, üst ortak anahtar ile üretmek mümkün değildir.

4.1.2.4 Normal Alt-Geniş Ortak Anahtar (Normal Child Extended Public Key)

Normal alt ortak anahtarın hesaplanması aşağıda verilmiştir.

1. Üst ortak anahtar seçilir. Üst ortak anahtar, üst özel anahtara karşılık gelen anahtardır.

2. İndeks numarası 2147483647 ($2^{31} - 1$) ile 4294967295 ($2^{32} - 1$)arasında seçilir. Bu aralık zorlaştırılmış alt anahtarların üretilmesi için belirlenen aralıktır. İndeks numarası artırılarak o üst anahtara bağlı farklı alt anahtarlar üretilir.

3. HMAC-SHA512 fonksiyonuna anahtar ve veri parametreleri verilir. Bu parametreler;

   - Veri = üst ortak anahtar + indeks (yan yana eklenir.)
   - Anahtar = üst zincir kodu

4. HMAC-SHA512 fonksiyonundan çıkan 512 bitlik sonucun son 256 bitlik kısmı, Ir alt zincir kodudur. Bu yukarıda normal-alt geniş özel anahtar hesabında anlatılan zincir kodu ile aynıdır. Çünkü fonksiyona giren girdiler aynı girdilerdir. Bir sonraki alt anahtar hesabında bu zincir kodu kullanılır.

5. HMAC-SHA512 fonksiyonundan çıkan 512 bitlik sonucun ilk 256 bitlik kısmı, Il, üreteç ile çarpılarak eliptik eğrisi noktası haline getirilir, $GXIl$

6. Üst ortak anahtarın eliptik eğrideki noktası ile 5. Adımda hesaplanan nokta arasında EC nokta toplama işlemi yapılır.

7. Üst ortak anahtar, eliptik eğri noktasından alınır.

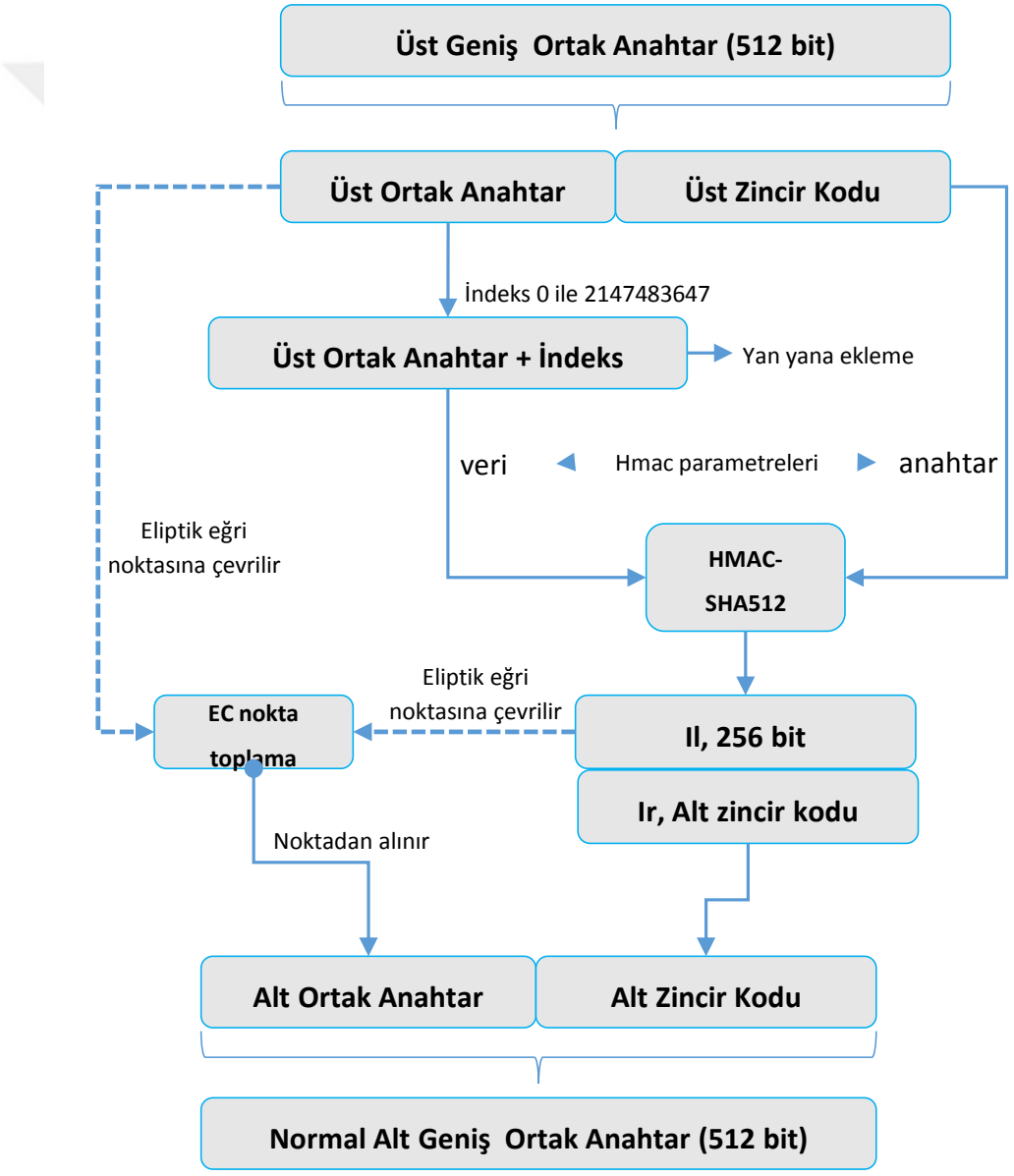
8. Yeni üretilen alt ortak anahtar ile alt zincir kodunun yan yana eklenmesiyle normal alt – geniş ortak anahtar elde edilir.

Bu adımların python programlama diliyle yazılmış örnek kod parçası Ek-A.4'de verilmiştir. Bu uygulamanın verdiği örnek çıktı Şekil 4.9'deki gibidir;

```
Normal-Alt Geniş Açık Anahtar Hesaplama: m/0
Alt Zincir Kodu (hex):
8d7d2ef83c1233a64b27680fb08152291ceecc89592239032ed82cd860255459
Alt ortak anahtar (hex):
033e0728adb800da553426e653eeaf7ec25603cbb14f7a964ee6e0ee45310670a2
```

Şekil 4.9: Normal Alt Geniş Ortak Anahtar Hesaplama Örneği

Bulunan bu alt ortak anahtar aynı indeks numarası verildiği takdirde, b başlığında anlatılan normal alt-geniş özel anahtardan hesaplanan ortak anahtarla aynıdır. Bu önemli nokta hiyerarşik cüzdanları diğer cüzdanlardan ayıran temel özelliklerinden birisidir. Bu önemli özellik sayesinde özel ve ortak anahtarların üretilmesi birbirinden ayrılabilir. İş yerleri için çok ideal bir özelliktir. Cüzdana ait alıcı adresleri özel anahtarlar bilinmeden üretilebilir. Özel anahtarları tamamen çevrimdışı bir yerde depolayarak aynı ortamda işlemleri imzalamak için kullanabilme imkânı tanımaktadır. Böylelikle özel anahtarın çalınması gibi bir durum olasılığı en aza indirgenecektir. Aşağıda normal alt geniş ortak anahtarın hesaplanma adımları şekil ile gösterimi mevcuttur.



Şekil 4.10: Normal Alt-Geniş Ortak Anahtar Hesaplama

4.1.2.5 Zorlaştırılmış Alt-Geniş Ortak Anahtar

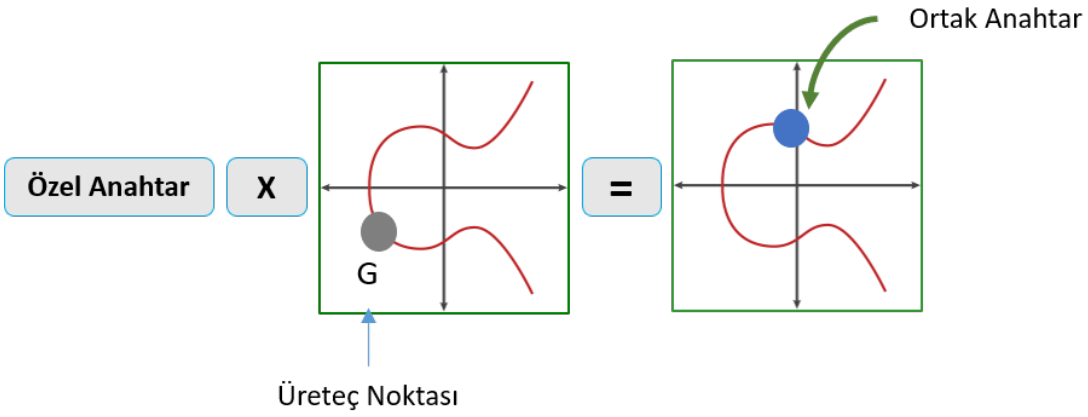
Bu anahtarı sadece üst geniş özel anahtar üretebilir yani üst geniş ortak anahtar ile üretmek mümkün değildir. Yukarıda üretimi anlatılan zorlaştırılmış alt-özel anahtarın karşılığı olan ortak anahtara alt zincir kodu eklenerek oluşturulur. Zorlaştırılmış Alt-Geniş Ortak Anahtar = Üreteç x (zorlaştırılmış alt özel anahtar)+alt zincir kodu Normal alt-geniş ortak anahtar ise aynı indeks numarası verilerek üst-geniş ortak anahtar ve üst-geniş özel anahtarın her ikisiyle de üretilebilmektedir.

Üst geniş ortak anahtar ile üretilemediği için cüzdanda gizli veya iç bir hesap açmak istenirse bu anahtar üretimi kullanılabilir. Böylelikle dışarıdan tamamen bağımsız bir ortak anahtar cüzdanda yer alabilir.

4.1.2.6 Ortak Anahtarlar Hakkında Bilgilendirme

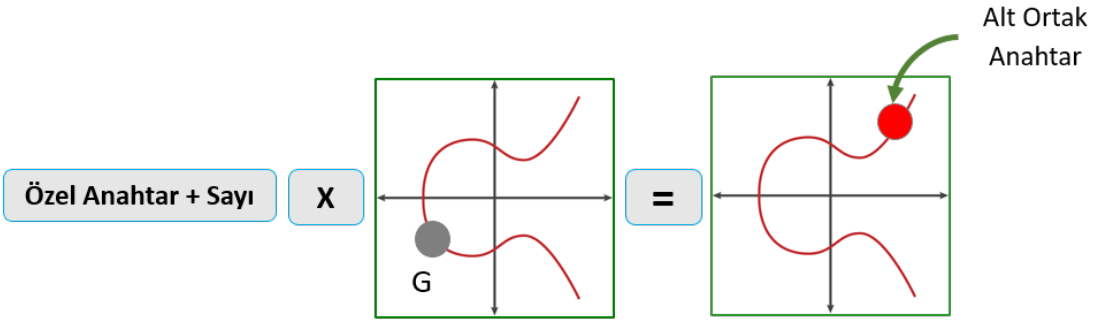
Yukarıda geniş özel anahtardan üretilen özel anahtarın geniş ortak anahtardan üretilen ortak anahtara karşılık geldiği anlatılmıştı. Başka bir deyişle geniş özel anahtardan üretilen özel anahtara karşılık gelen ortak anahtar ile geniş ortak anahtardan üretilen ortak anahtarın aynı olduğu söylenmişti. Bunun nasıl mümkün olduğu bu bölümde gösterilecektir.

Şekil 4.11'daki gibi özel anahtar üreteçle çarpıldığında ortak anahtarı verir.



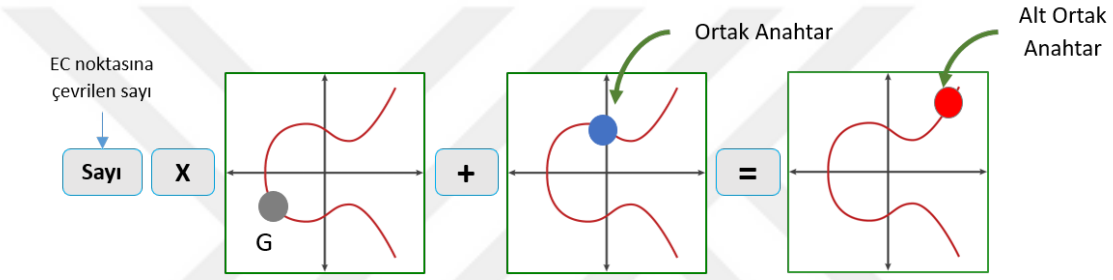
Şekil 4.11: Ortak Anahtar

Şekil 4.12'deki gibi bu özel anahtar, rastgele bir sayı ile artırılıp (256 bit HMAC sonucu gibi) üreteç noktası ile çarpıldığında yeni bir ortak anahtar elde edilir.



Şekil 4.12: Alt Ortak Anahtar

Benzer şekilde orijinal ortak anahtarı, eliptik eğri noktasına çevrilmiş aynı rastgele sayı ile toplandığı takdirde aynı alt ortak anahtar elde edilecektir (Şekil 4.13).

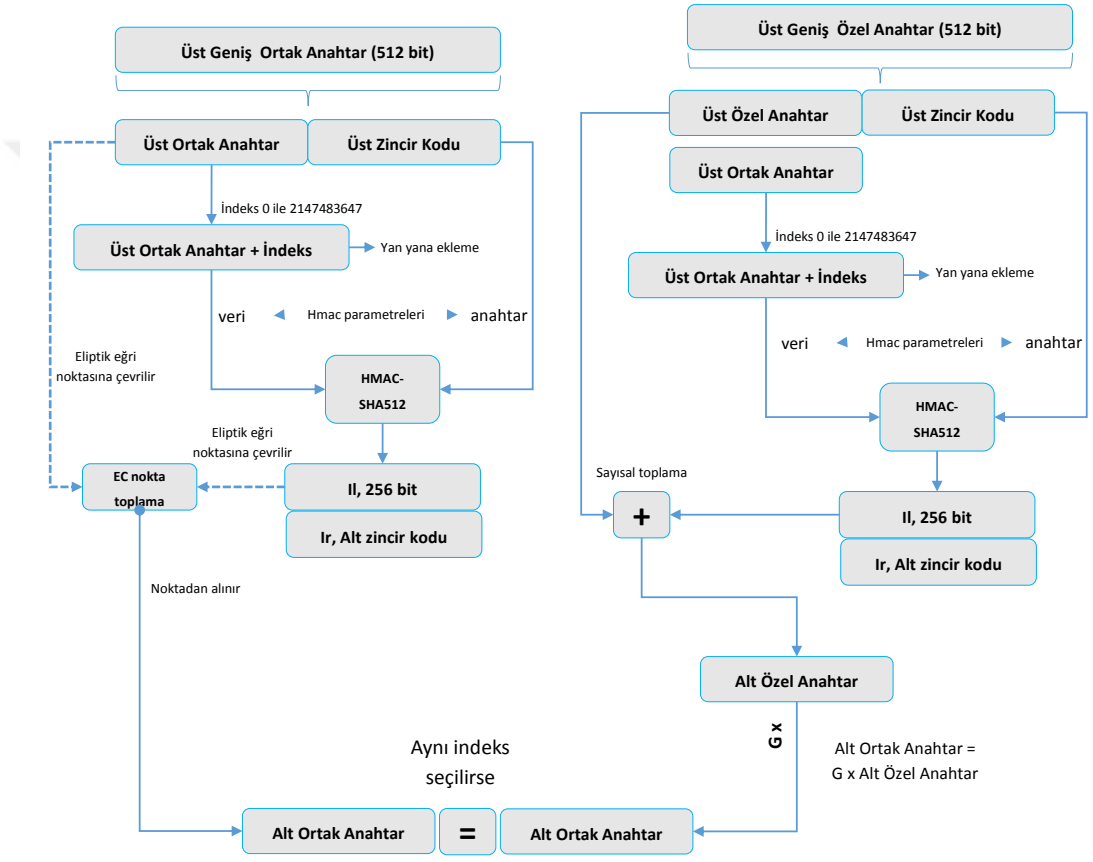


Şekil 4.13: Alt Ortak Anahtar

Alt ortak anahtar üretiminde HMAC-SHA512 fonksiyonuna aynı girdiler sokulmaktadır ve böylelikle fonksiyondan aynı çıktı elde edilir. Çıktının ilk 256 biti = L256:

- Üst ortak anahtar ile toplanarak alt ortak anahtar elde edilir.
- Üst özel anahtar ile toplanarak alt özel anahtar elde edilir. Bu anahtar üreteçle çarpılarak aynı ortak anahtar üretilmiş olur (Şekil 4.14).

Alt Ortak Anahtar = (Üst Özel Anahtar + L256) x G Alt Ortak Anahtar = (Üst Özel Anahtar x G) + (L256 x G) Alt Ortak Anahtar = Üst Ortak Anahtar + Nokta(L256)



Şekil 4.14: Aynı Alt Ortak Anahtarın Üretilmesi

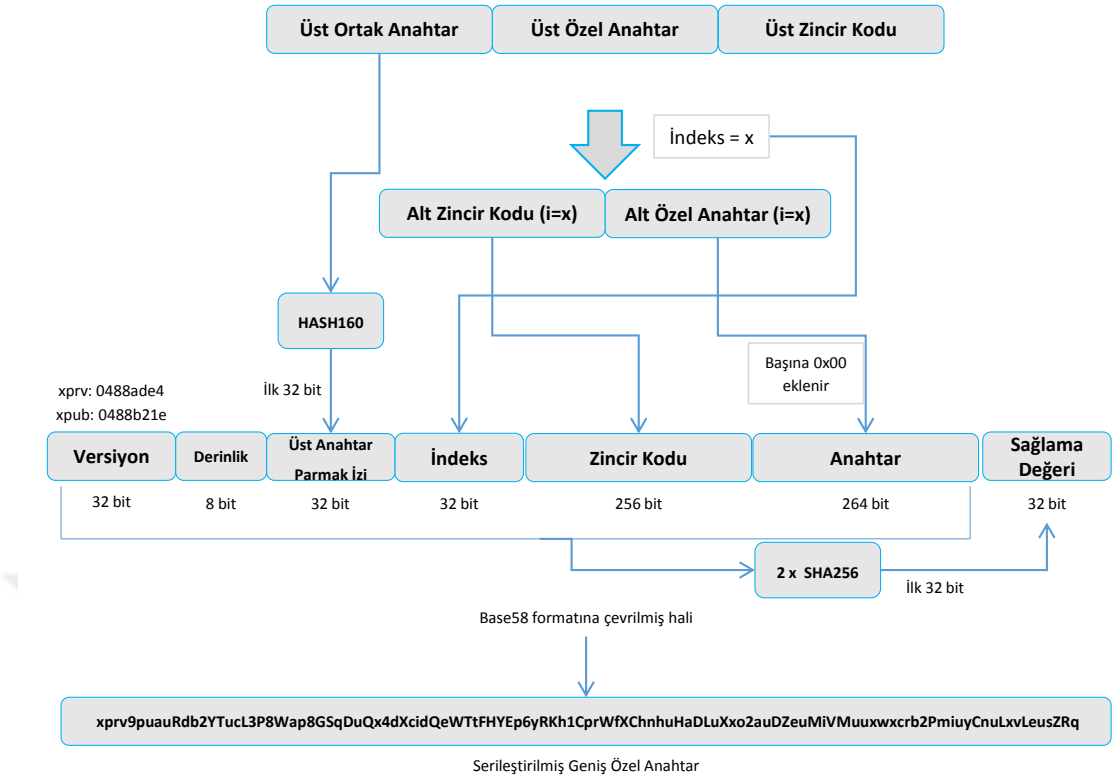
4.1.2.7 Serileştirme

Geniş anahtarlar, blokzincir tabanlı sistemlerde geçerli olmasını sağlayacak son haline kavuşması için serileştirme işleminden geçmektedir. Serileştirilmiş anahtarda Base58 formatında versiyon, derinlik, üst ortak anahtara ait parmak izi, indeks, zincir kodu, özel veya ortak anahtar, sağlama değeri bilgileri yer alır. Tablo 4.2'de bu alanlar sunulmuştur.

Tablo 4.2: Serileştirilmiş Anahtardaki Alanlar

32 bit	Versiyon	En başa konulan "xprv" 0488ade4 veya "xpub" 0488b21e hex değeri
8 bit	Derinlik	Temel anahtardan geniş anahtara kadar kaç kez türetildiği
32 bit	Üst Anahtar Parmak İzi	Üst ortak anahtarın Hash160 özetinin ilk 32 biti
32 bit	Alt Anahtar İndeksi	Üst anahtarın kaçıncı alt anahtarı ürettiğini gösteren alt anahtar indeks numarası
256 bit	Zincir Kodu	Alt anahtarlar üretebilmek için gerekli 256 bitlik ekstra anahtar
264 bit	Anahtar	264 bit ortak anahtar veya başına 0x00 eklenmiş 264 bit özel anahtar
32 bit	Sağlama Değeri	Yukarıdaki alanların iki kez SHA256'dan geçirilmiş özetin ilk 32 biti

Temel Anahtar için derinlik = 0x00, parmak izi = 0x00000000, indeks numarası = 0x00000000'dir. Ortak anahtar 33 bayt olduğu için 32 bayt olan özel anahtarın başına 0x00 eklemesiyle 33 bayt olması sağlanır. İndeks numarasının 4 bayt olmasından dolayı alt anahtarları üretmek için verilen indeks 0 ile 4.294.967.295 (0xffffffff) arasında sınırlandırılmıştır. Şekil 4.15'de örnek olması açısından özel anahtarın serileştirme işlemine ait adımların gösterimi mevcuttur.



Şekil 4.15: Serileştirilmiş Özel Anahtar

Base58, büyük sayıları daha kısa ve daha kullanıcı dostu bir biçimde göstermek için kullanılan 58 adet karakterdir. Bu karakterler: "1 2 3 4 5 6 7 8 9 A B C D E F G H J K L M N P Q R S T U V W X Y Z a b c d e f g h i j k m n o p q r s t u v w x y z" dir. Bu karakterlerde görüldüğü gibi 0, O, I, l gibi karıştırılası kolay harfler bulunmamaktadır. Versiyon, derinlik, parmak izi, indeks, zincir kodu, anahtar ve sağlama değerinden oluşan bu oldukça uzun serileştirilmiş gösterimi Base58 formatına çevrildiğinde makul bir uzunluğa düştüğünden ve karıştırılması zor olan karakterler barındırmadığından dolayı cüzdanlarda olduğu gibi blokzincir sisteminin birçok kısmında bu format tercih edilir.

4.1.3 BIP-44

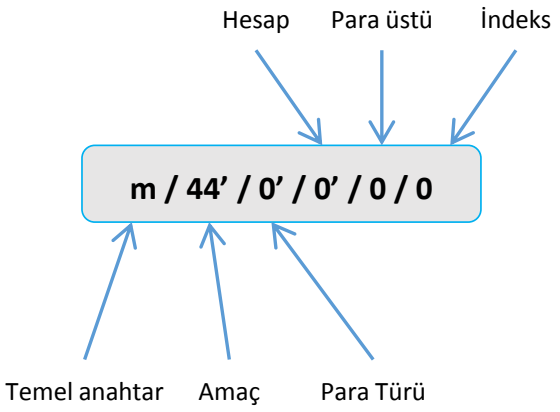
BIP-44, BIP-32'de bahsedilen algoritmaya göre deterministik cüzdan için "Amaç Alanı" sunarak mantıksal bir hiyerarşi önermektedir [43]. BIP-39'da temel anahtarın üretileceği entropi değeri hesaplandı. BIP-32'de ise ağaç yapısı şeklinde temel anahtardan her genişletilmiş anahtarın kendine ait türetme yolu ile alt ve devam eden daha alt anahtarlar

türetildi. Bu türetme yolu istenildiği gibi seçilebilir fakat hiyerarşik deterministik cüzdanın kullanımında anahtarların belli bir yapıda olması cüzdanlar arası uyumluluğu sağlamak adına önem kazanmaktadır. Böylece bir firmaya ya da kuruluşa ait cüzdan başka bir kuruluşun ürettiği cüzdana aktarmak ya da yedeklemek istenildiğinde bu uyumluluk sayesinde herhangi bir problemin çıkmaması hedeflenir. Bu tarz bir gereksinim daha önce de anlatıldığı gibi belli bir standart olmadığı için tavsiye edilen önerilerdir.

Cüzdanlar arasındaki uyumluluğa yardımcı olması için BIP-44, BIP-32'in türetme yoluna aşağıdaki yapıyı tanıtır.

m / amaç' (purpose') / para_türü' (coin_type') / hesap' (account') / para_üstü (change) / indeks

"/" işareti ağaçtaki derinlik seviyesini ayırmak için kullanılır. Zorlaştırılmış anahtarların gösterimini normal anahtarlardan ayırmak için ise tırnak işareti veya "h" soneki kullanılır. Normal anahtarlar için indeks numarası 0 ile 2^{31} arasında seçilir. Zorlaştırılmış anahtarlar için indeks numarası ise 2^{31} ile 2^{32} arasındadır. Buna göre örneğin indeks numarası 2' veya 2h değeri $(2^{31}) + 2 = 2147483650$ sayısına eşittir.



Şekil 4.16: BIP-44 Derinlik Seviyeleri

m - Tohum: Tohumdan oluşturulan Temel Anahtardır.

m/44' - Amaç: BIP-32 şemasına eklenen amaç alanıdır. Bu seviyede zorlaştırılmış türetme yöntemi kullanılır. Amaç 44' (0x8000002C = 2147483692 = $2^{31} + 44$) olarak ayarlanmış bir sabittir. Bu düğümün alt ağacının bu spesifikasyona göre kullanıldığını gösterir.

m/44'/0' - Para Türü: Anahtarların kullanılacağı Bitcoin, Etherium gibi sınırsız sayıda kripto para birimi türü için kullanılan alandır. Bu sayede farklı kripto para türleri için farklı tohumlar kullanmak yerine aynı tohumdan türetilen özel ve ortak anahtarların türetme yolları birbirinden ayrılarak kullanılabilir. kripto para türleri için sabit bir değer ayarlanmıştır. Örneğin; 0' = Bitcoin, 1' = Bitcoin (Testnet), 2' = Litecoin, 3' = Dogecoin, 60' = Etherium gibi [43]. Bu seviyede zorlaştırılmış türetme yöntemi kullanılır. Bu alan, tek bir tohum ile çeşitli farklı paraları tutmak için kullanılacak donanım cüzdanlarında çok kullanışlıdır.

m/44'/0'/0' - Hesap: Bu alan, fonlar için ayrı hesaplar oluşturmaya olanak tanımaya yarar. Varsayılan hesap 0 olarak ayarlanmıştır. Bu seviyede zorlaştırılmış türetme yöntemi kullanılır. Kullanıcılar bu hesapları banka hesaplarına benzetmek için kullanabilirler, farklı indeks numaralarıyla birikim hesabı, halka açık bağış hesabı gibi.

m/44'/0'/0'/0 - Para Üstü: Bu alan kullanılan anahtar ve adresleri "Alıcı" ve "Para üstü" olarak 0 ve 1 sabitleriyle ayırmaya yarar. Alıcı = 0: Ödeme almak için kişilere verilen adreslerdir. Para üstü = 1: kripto para gönderme yaparken gönderilen para parçalara bölünemediğinden dolayı para üstünün para üstü adresine alınması için aynı işlem içerisinde ikinci bir çıktı daha oluşturulur. Bu seviyede normal türetme yöntemi kullanılır.

m/44'/0'/0'/0/0 - İndeks: Bu alan, cüzdanda gerçek kullanımdaki yani son derinlikteki özel ve ortak anahtarları belirlemekte işe yarar. Bu seviyede normal türetme yöntemi kullanılır. Görüldüğü üzere ilk birkaç derinlik seviyesi, hiyerarşik cüzdanın kullanışlı ve pratik olması için bir yapı sunmaktadır. Adresler için kullanılan gerçek anahtarlar ise ağacın en alt seviyelerinde hesaplanır.

Hiyerarşik cüzdanda türetme yolu belirlenirken BIP-44 yapısına benzer olarak BIP-49 (m/49'/0'/0'/0/0) ve BIP-84 (m/84'/0'/0'/0/0) yapıları da yaygın olarak kullanılmaktadır. BIP-44 adresleri "1", BIP-49 adresleri "3" ve BIP-84 adresleri "bc1" ile başlamaktadır. BIP-44 P2PKH kilitleme scripti ve Base58 formatını, BIP-49 P2WPKH-P2SH kilitleme scripti ve Base58 formatını, BIP-84 ise P2WPKH kilitleme scripti ve Bech-32 formatını kullanmaktadır.

Bir diğer fark serileştirme işleminde versiyon bilgisi olarak, BIP-44 için "xprv" 0488ade4 veya "xpub" 0488b21e hex değeri, BIP-49 için "yprv" 049d7878 veya "ypub" 049d7cb2 hex değeri, BIP-84 için "zprv" 04b2430c veya "zpub" 04b24746 hex değeri kullanılır.

4.2 Donanımsal Deterministik Cüzdan Uygulaması

Cüzdan uygulaması bu kısımda anlatılacaktır. Teknik özellikleri, kullandığı APDU komutları, imzalama ve imza doğrulama adımları, Bitcoin Testnet'inde örnek para transferi bu kısmın konusudur.

Bu bölümde BIP-32 destekli donanımsal hiyerarşik cüzdan uygulamasına örnek olabilecek bir proje incelenecektir. Uygulamanın kullandığı APDU komutları listelenecek ve Bitcoin Testnet'e bağlanıp örnek bir Bitcoin transferi işlemi yapılacaktır. Genel olarak sistemin nasıl çalıştığı bu bölümde somutluk kazanacaktır.

Java kart cüzdan uygulamaları içerisinden konunun kavranması adına bu çalışma için adım adım aşağıdaki örnekler incelenmiştir;

-Java Card Kit/SDK/Sample/wallet

-Java Card Kit/SDK/Sample/WalletDemoApplet

-Java Card Kit/SDK/Sample/GPSample

-Ledger-javacard-master [17]

-Satochip [44]

Hiyerarşik deterministik cüzdan uygulaması olarak açık kaynak kodlu Ledger ve Satochip adlı projelerden yararlanılmıştır. Hiyerarşik cüzdan uygulamasının yükleneceği akıllı kart çeşidi olarak java kart seçilmiştir. Akıllı kart uygulaması kodlarının koşturulacağı ortam Java kartı programlama dili için özel olarak tasarlanmış bir entegre geliştirme ortamı olan JCIDE aracıdır. JCIDE ile kodlar derlendikten sonra oluşan "cap" dosyası Java karta ya da JCIDE ortamının sunduğu sanal Java kart simülatörüne yüklenebilmektedir. Java karta uygulama yüklendikten sonra dış dünya yani blokzincir ile iletişime geçmeyi sağlayacak arayüz istemci uygulaması ise açık kaynak olan Electrum uygulaması ile sağlanacaktır [45]. Temelinde Electrum uygulaması masaüstü cüzdanıdır fakat ledger, keepkey, trezor gibi donanımsal cüzdanlar için de arayüz desteği sunmaktadır.

4.2.1 Uygulamada Kullanılan APDU Komutları

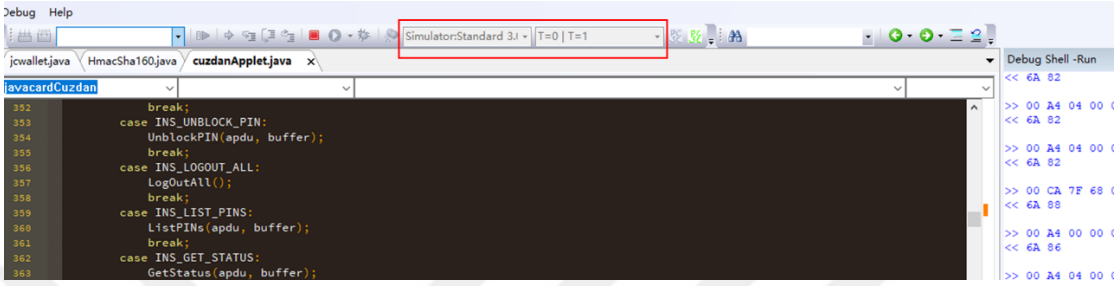
HD cüzdan uygulamasında kullanılan APDU komut başlıkları olan CLA, INS, P1 ve P2 değerleri liste olarak Tablo 4.3'de sunulmuştur.

Tablo 4.3: Java Kart HD Cüzdan Uygulaması APDU Komut Başlıkları

KOMUT	CLA	INS	P1	P2
Select	00	A4	04	00
Setup	B0	2A	00	00
Import Key	B0	32	00 - 0F	00
Get Public Key From Private	B0	35	00 - 0F	00
Create PIN	B0	40	00 - 07	Max PIN denemesi
Verify PIN	B0	42	00 - 07	00
Change PIN	B0	44	00 - 07	00
Unblock PIN	B0	46	00 - 07	00
List PINs	B0	48	00	00
Get Status	B0	3C	00	00
Import BIP32 Seed	B0	6C	Tohum boyutu (1b)	00
Get BIP32 AuthentiKey	B0	73	00	00
setBIP32AuthentikeyPubkey	B0	75	00	00
Get BIP32 Extended Key	B0	6D	00-0A	00
Set BIP32 Extended Pubkey	B0	74	00	00
Sign Message	B0	6E	00 - 0F, FF	00 - 03
Sign Short Message	B0	72	00 - 0F, FF	00
Parse Transaction	B0	71	01 - 02	00 - 01
Sign Transaction	B0	6F	00 - 0F, FF	00

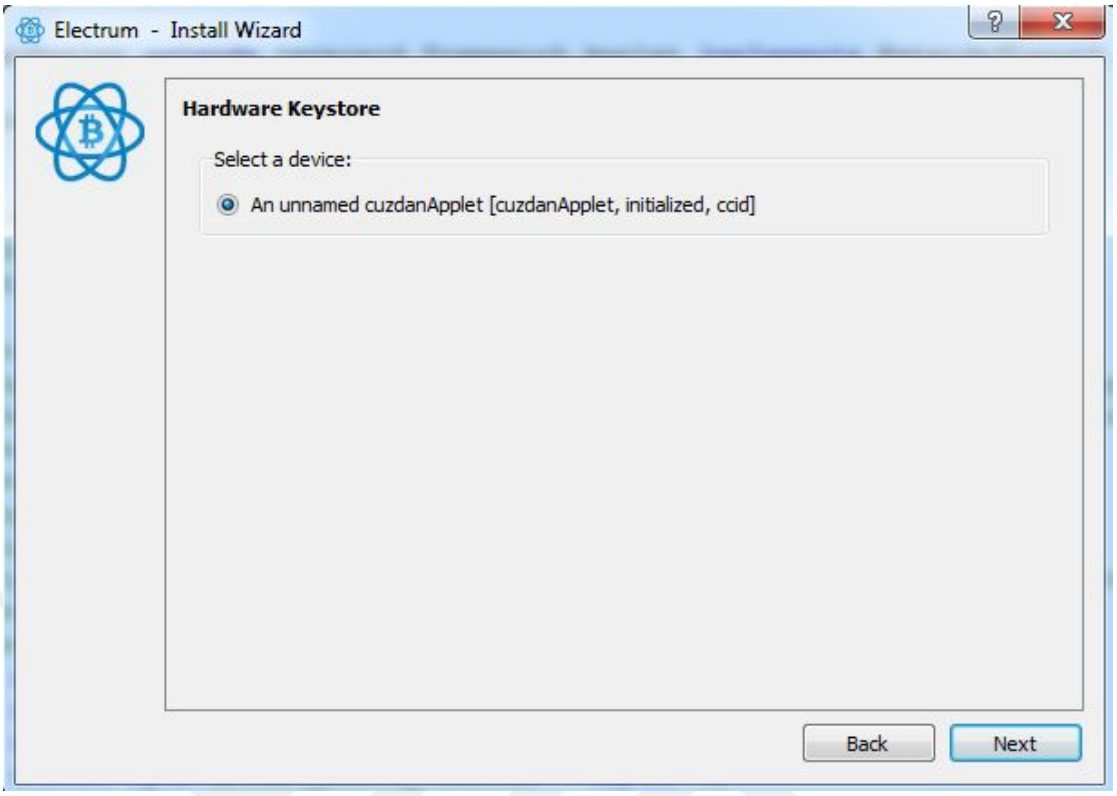
4.2.2 Java Kart HD Cüzdan Uygulaması ile Para Transferi

Bu bölümde Java akıllı kartına yüklü HD cüzdan uygulaması ile bitcoin transfreri yapılacaktır. İlk olarak uygulama kodları derlendikten sonra oluşan "cap" dosyası karta ya da yüklenecektir. Şekil 4.17'de uygulama simülatöre yüklenmiştir.



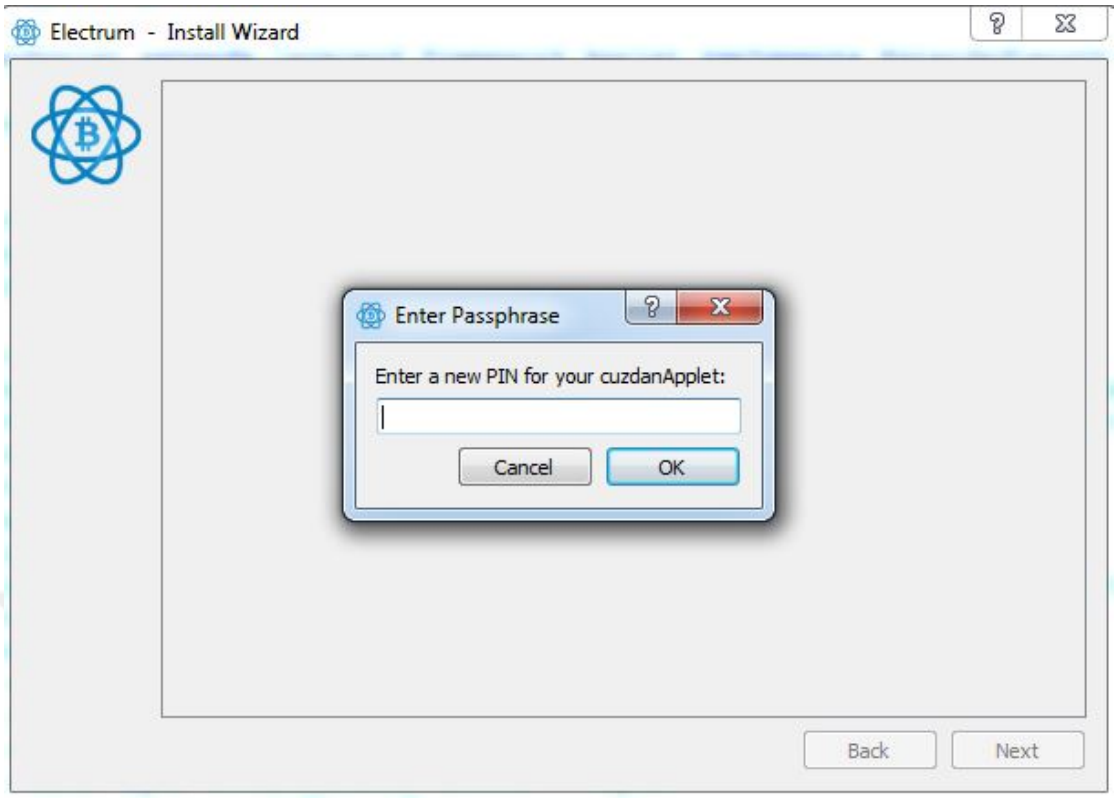
Şekil 4.17: Uygulamanın (Applet) derlenip karta ya da simülatöre yüklenmesi

Karta yüklenen uygulamanın AID'si "cuzdanApplet" kelimesinin hex halidir: 63 75 7A 64 61 6E 41 70 70 6C 65 74. Daha sonra kart ile iletişime geçecek olan Electrum uygulaması ile kart ilklendirilecek ve cüzdana ait anahtarlar üretilecektir. Electrum, plugins klasöründe tanımlı cüzdanlara ait AID'ler için kart okuyuculara Select komutu gönderir. "cuzdanApplet" uygulaması için 00 A4 04 00 0C 63 75 7A 64 61 6E 41 70 70 6C 65 74 Select komutuyla Şekil 4.18'deki gibi cüzdan uygulamasını seçilir.



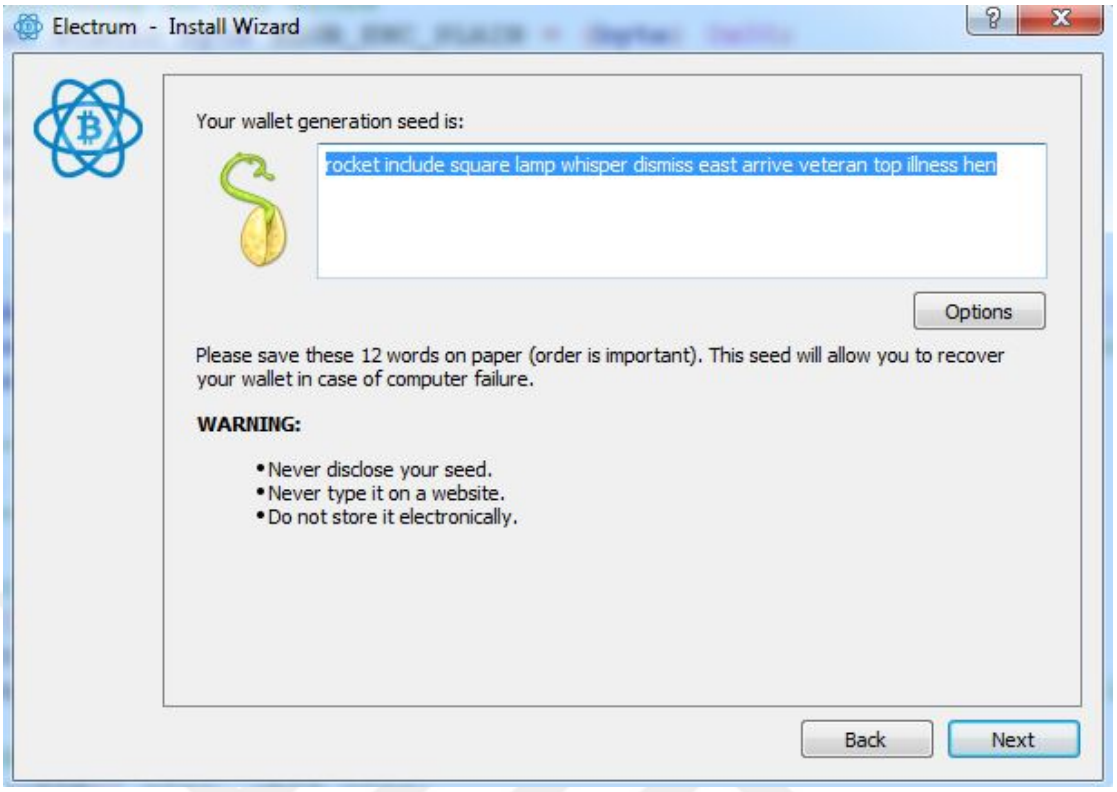
Şekil 4.18: Electrum'a Cüzdan Uygulamasını Tanıtma

B0 3C 00 00 00 Get Status komutuyla cüzdanın durumuna bakılır. Get Status komutu akıllı kartta yüklü cüzdan uygulamasına ait versiyon bilgisi (4 bayt), PIN deneme sayısı gibi genel bilgileri vermektedir. Karta Setup komutuyla PIN'ler başarılı şekilde yüklendiyse kartın Get Status komutuna döndüğü değer: versiyon bilgisi (4 bayt) + PIN0-PUK0-PIN1-PUK1 deneme sayısı (4 bayt). Karta henüz Setup komutuyla PIN yüklenmediyse 9C04 hatası döndürülür. Karttaki uygulamaya ait PIN'lerin yüklenmesi, BIP-32 objelerinin oluşturulması gibi uygulamanın hassas bilgilerinin ilklendirilmesi ve kart belleğinde yerlerinin ayrılması Setup komutuyla gerçekleştirilir. Setup komutu yapısı: INS + CLA + P1 + P2 + LC + varsayılan PIN uzunluğu (1 bayt) + varsayılan PIN + PIN0 deneme sayısı (1 bayt) + PUK0 deneme sayısı (1 bayt) + PIN0 uzunluğu (1 bayt) + PIN0 + PUK0 uzunluğu (1b) + PUK0 + PIN1 deneme sayısı (1 bayt) + PUK1 deneme sayısı (1 bayt) + PIN1 uzunluğu (1 bayt) + PIN1 + PUK1 uzunluğu (1 bayt) + PUK1 + BIP-32 objesi için kullanılacak hafıza boyutu (2 bayt) + gelecekte kullanılmak üzere rezerve edilen alan (5 bayt). Örnek bir Setup komutu; B02A00004C0830303030303030300501043132333410E788D403C4E8485150706ECE58468 4B2010110C53B3D77683019B1E8F7767AE6D1826110569E2EB60BAECE3070AD9F075 B90D65D00200000010101.



Şekil 4.19: Cüzdana PIN Atanması

Karta Setup komutuyla PIN'ler başarılı şekilde yüklendikten sonra B0 42 00 00 04 31 32 33 34 (PIN:1234) Verify komutuyla kullanıcıdan istenen PIN doğrulanır. Electrum uygulaması rastgele ürettiği entropiyi hatırlatıcı kelimelere çevirerek Şekil 4.20'deki gibi kullanıcı ekranına sunar ve kullanıcıdan bu kelimeleri kaydetmesini ister.

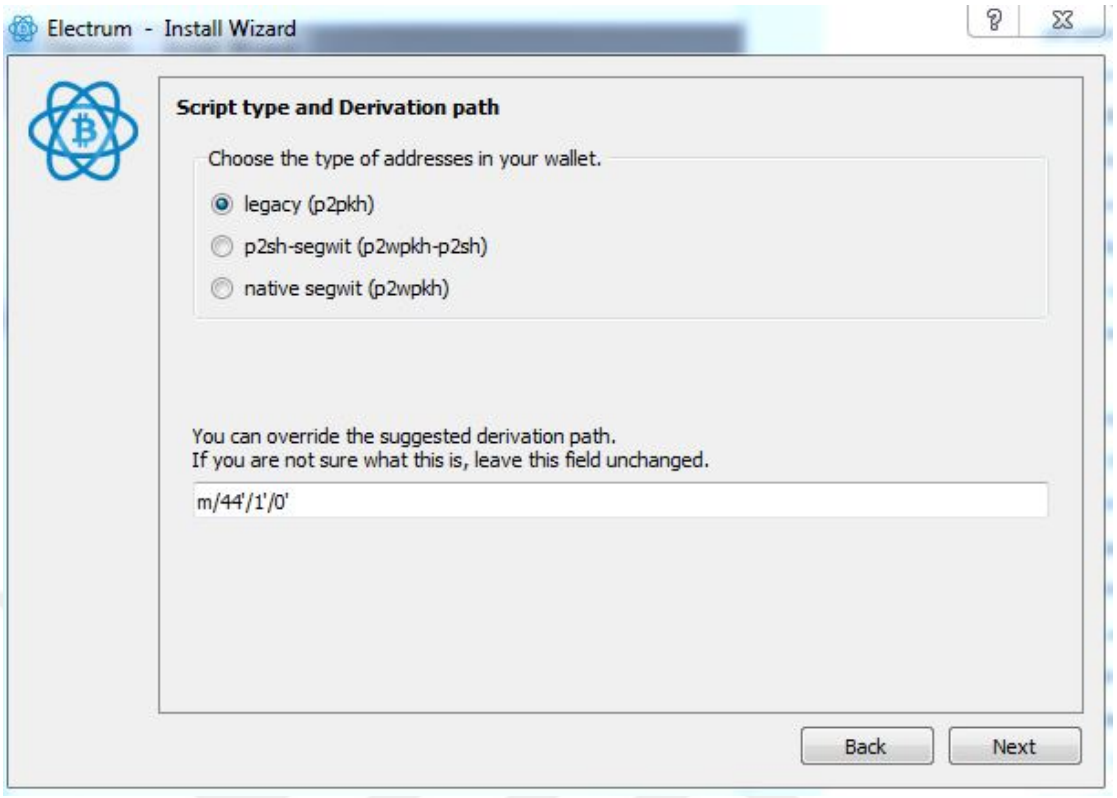


Şekil 4.20: Tohumun hesaplanması için 12 hatırlatıcı kelimenin üretilmesi

Electrum uygulaması, PBKDF2-HMAC512 fonksiyonuna girecek olan ve BIP-39 başlığında anlatılan "mnemonic" etiketi yerine "electrum" etiketini kullanmaktadır. Tohum = PBKDF2-HMAC512 ("rocket include square lamp whisper dismiss east arrive veteran top illness hen", "electrum", döngü sayısı (2048)) = "37533F462CBDB05D2001764437FB6652209F4C7F17A1F7539453C5AD0591E99DB55FBAE57C62F28B8BE89B5407C95A5733494F9CD8D786FBC40757E6E2B3DD1E".

Yukarıda üretilen tohumu karta yazmak için örnek Import BIP32 Seed komutu; "B06C40004037533F462CBDB05D2001764437FB6652209F4C7F17A1F7539453C5AD0591E99DB55FBAE57C62F28B8BE89B5407C95A5733494F9CD8D786FBC40757E6E2B3DD1E".

Karta gönderilen Import BIP32 Seed komutu ile kartta temel anahtar ve zincir kodu üretilir ve bu üretilen değerler kart içerisinde set edilir. Şekil 4.21'de BIP-44, BIP-49 veya BIP-84'e göre türetme yolu belirlenir.

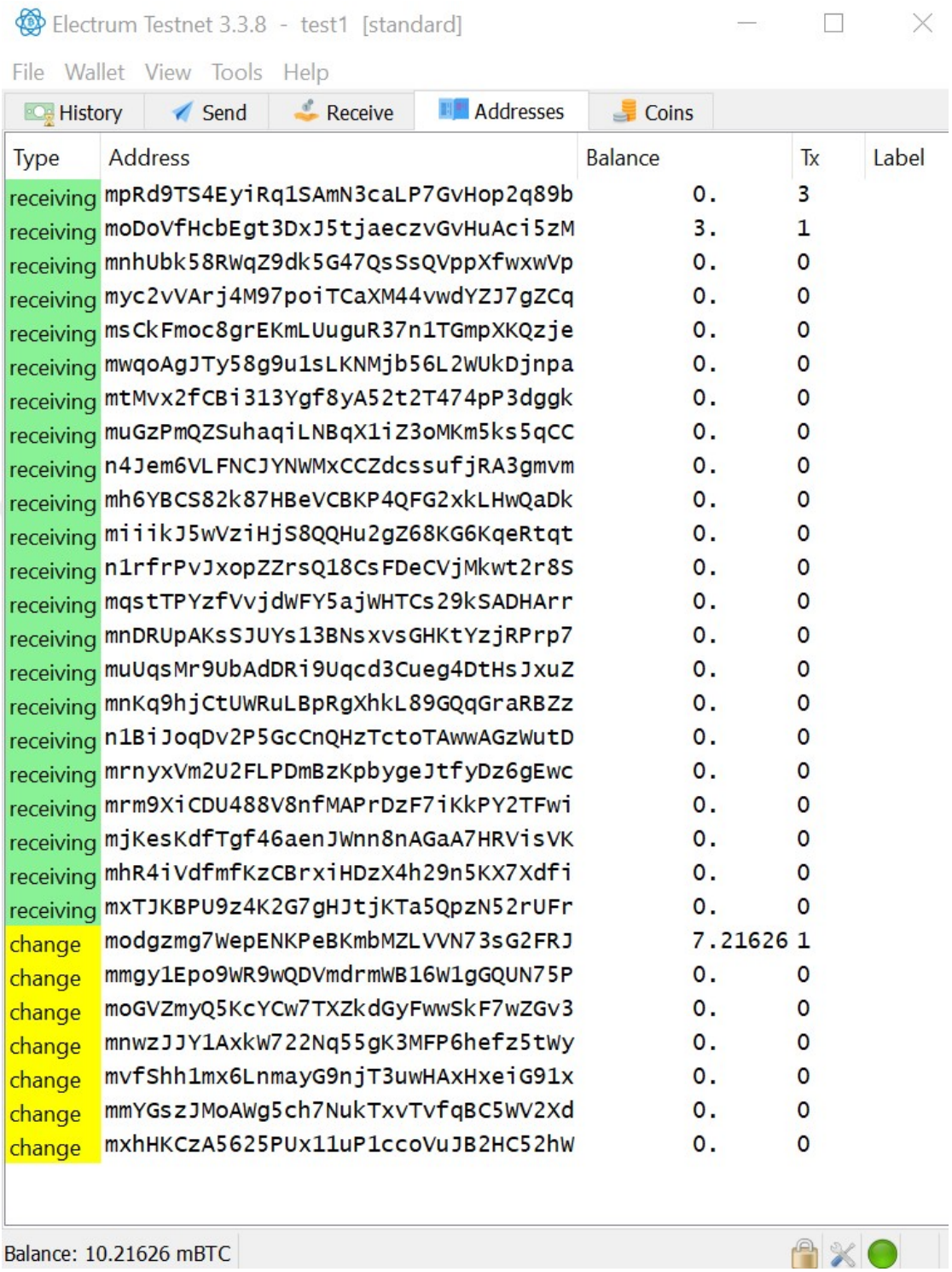


Şekil 4.21: Türetme Yolu Belirleme

Belirlenen türetme yoluna ait indeksler karta Get BIP32 Extended Key komutuyla gönderilir. m/44'/1'/0' türetme yolu için örnek Get BIP32 Extended Key komutu; B06D03400C8000002C8000000180000000.

Karta gönderilen Get BIP32 Extended Key komutu ile temel anahtardan genişletilmiş alt anahtar hesaplanır ve dönüş olarak alt zincir kodu, alt ortak anahtar ve anahtarların imzalı halleri döndürülür. Kartın bu komuta verdiği cevabın yapısı: 32 bayt zincir kodu, 4 bayt ortak anahtar uzunluğu, ortak anahtar, 4 bayt imza boyu, imza, 4 bayt imza2 boyu, imza2.

Kartın ilklendirmesi tamamlandıktan sonra cüzdana ait adresler Şekil 4.22'deki gibidir.

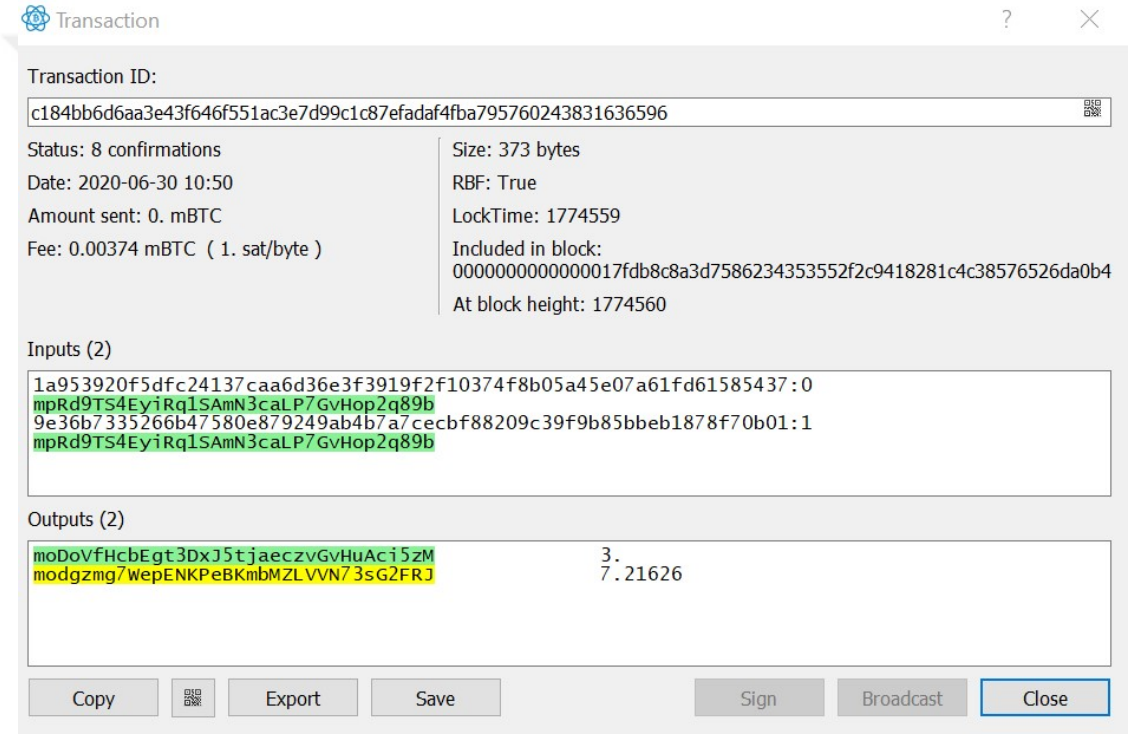


Şekil 4.22: Cüzdana ait adresler

Transfer işleminin gerçekleşmesi için kartın işlem özetini güvenli belleğinde tutulan alt özel anahtarı ile imzalaması gerekir. Karta gönderilen Sign Transaction APDU komutu bunu yapmaktadır. Örnek bir Sign Transaction komutu:

B06FFF0020E81C4E43EDC281C4B95433B961D88A0B86DEFEA89297B7EEE72EE1396

B2CCA69. Transfer işlemi gerçekleştirilmeden önce cüzdandaki "mpRd9TS4EyiRq1SAm N3caLP7GvHop2q89b" 10.22 mBTC bakiyesi bulunmaktaydı. Transfer işlemi ile birlikte cüzdandaki diğer "moDoVfHcbEgt3DxJ5tjaeczvGvHuAci5zM" adresine gönderilen 3 mBTC transferine ait işlem bilgileri Şekil 4.23'de gibidir. Girdi olarak "mpRd9TS4EyiR q1SAmN3caLP7GvHop2q89b" adresinden harcanmamış işlem olan 10 mBtc ve 0.22 mBtc bakiyelik 2 çıktı seçilmiştir. İşlemin çıktısı, gönderimi yapılan adres "moDoVfHcbEgt3DxJ 5tjaeczvGvHuAci5zM" (3 mBTC) ve para üstü adresi olan "modgzmg7WepENKPeBKmb MZLVVN73sG2FRJ" (7.21626) adresidir. $(10.22 - 3) - 7.21626 = 0.0074$ mBTC, işlemi bloğa ekleyen madenciye verilen ücret miktarıdır.



Şekil 4.23: İşlem bilgieri

Yukarıdaki işlemin tek satırlık hex yapısında gösterimi şu şekildedir:

"0200000002375458 61fd617ae0455ab0f87403f1f219393f6ed3a6ca3741c2dff52039951a0000 00006b4830450221008c9d1051aa055c185150a4f7f5040844d6e598f8211b9f015f0e0981f3ef2c 7702205790273885c78958039c2474410bdfc7a9dc499f0f9384164c0c84deb9c9049701210252a feb2f8082c7db073a7f696452d0f40f54314353a0e45edc4a4093e7bee585fdffffff010bf77818ebbb 859b9fc30982f8cbcea7b7b49a2479e88075b4665233b7369e010000006a473044022031bf1a3a3 bd8a724bfc4e2eab4401c85a142ded0d2e7931ce1ad10c2af6f2fd90220708225303790c96a750d4

4f0f9888d957b89ea5048dfe7576a370b9ba78ef9bd01210252afeb2f8082c7db073a7f696452d0f4
0f54314353a0e45edc4a4093e7bee585fdffffff02e093040000000000001976a914548296d27a6beaad
de47bdade88ebc1e1b11195188acda020b00000000001976a914590725dc2622d2118ce4ca64cda
653a5e5142a5f88acdf131b00".

Bu bölümde HD cüzdan uygulamasının akıllı karta yüklenmesi, ilklendirilmesi ve örnek bitcoin transfer işlemi incelenmiştir. Böylece önceki bölümlerde anlatılan sistemin işleyişi somut bir örnekle gösterilmiştir.

4.2.3 AKİS Kart Donanımsal Deterministik Cüzdan Tasarımı

AKİS yerli ve milli imkânlarla TÜBİTAK BİLGEM tarafından geliştirilmiş akıllı kart işletim sistemidir. Kimlik, pasaport, ehliyet, elektronik imza gibi farklı uygulamalar için altyapı sağlamaktadır. Ortak Kriter (Common Criteria) CC EAL 5+ ve üstü güvenlik değerlendirmelerinden geçmiş ve sertifikasını almış akıllı kart mikroişlemcileri üstünde koşmaktadır. Bu mikroişlemciler arasında İnfineon, NXP ve milli yonga olan UKTUM çipleri yer almaktadır.

AKiS kartlarda canlandırma, ilklendirme, kişiselleştirme, işletim ve ölüm evreleri olmak üzere 5 adet yaşam evresi bulunmaktadır. Canlandırma evresinde kart aktif edilir. AKiS kartın kullanıma hazır hale gelebilmesi için başarılı şekilde canlandırılması gerekir.Bu evrede kartın gerçekten AKiS kart olup olmadığı ve işletim sisteminin doğruluğunun tespiti yapılır. İlklendirme ve kişiselleştirme aşamalarının yapılabilmesi için gerekli ilklendirme ve kişiselleştirme anahtarları yine canlandırma evresinde belirlenip yüklenir. Bu anahtarlar AES256 veya DES3 simetrik anahtarlardır ve AKiS versiyonuna göre değişmektedir.

AKiS'in en yeni sürümünde kart canlandırıldıktan sonra ilk olarak uygulamaların bellek alanlarını kontrol eden, kısıtları belirleyen ve uygulamayla çekirdek arasında iletişimi sağlayacak olan Kart Yöneticisi uygulaması yüklenmektedir. Yeni sürüm AKiS'te kimlik, pasaport, ehliyet, e-imza, bilet gibi uygulamalar tek bir kart içinde toplanabilmektedir. Uygulamalar için karttaki bellek alanı bölünebilmektedir ve bir uygulama diğer uygulamanın bellek alanına erişmesini engelleyecek firewall mekanizması kartın donanımı tarafından sağlanmaktadır. Böylece her uygulama kendine tahsis edilen bellek alanı dışında

çıkıp diğer uygulamaların bilgilerine erişemez. Uygulama yükleme sırasında her uygulamaya özel AID değeri atanır. Uygulamaya ait bu AID değeri ile karttaki uygulamaya erişim sağlanmaktadır.

Bu sürümle birlikte kartın içinde olan uygulamalardan biri de cüzdan uygulamasıdır. AKiS cüzdan 4.1. ve 4.2. başlıklarında yer alan HD cüzdanın çalışma mekanizmasının anlatıldığı ve uygulamasının yapıldığı bitcoin kripto parasının transfer işlemlerini yapabilecek özellikte tasarlanmıştır. BIP-32, BIP-39, BIP-44 ve eliptik eğri kriptografik işlemlerini destekleyecek şekilde geliştirilmektedir. BIP-32, BIP-44, BIP-39'a göre anahtarlar türetilmekte ve bitcoin kripto para transfer işlemi için standart eliptik eğri "secp256k1" üzerinde ECDSA kullanılarak imzalama işlemleri yapılabilmektedir. AKİS cüzdan uygulamasının nihai durumuna gelebilmesi için geliştirilmesi devam etmektedir.

Bitcoin'den farklı diğer kripto paraları da destekleyecek nitelikte olabilmesi için çalışmalar başlamıştır ve devam etmektedir. Bununla birlikte cüzdan ile blokzincir ağına bağlantısını sağlayacak ara yüz çalışması yapılması planlanmaktadır. Bu bağlantı mobil ve masaüstü makineler için düşünülmektedir.

Bölüm 5

Sonuç

Bu tez çalışmasında blokzincirin işleyişinin ve mekanizmasının anlaşılması, cüzdan çeşitlerinin incelenmesi ve karşılaştırılması, donanımsal HD cüzdanın yapısı, işleyişi ve akıllı kartlar ile kullanımı detaylı şekilde incelenmiştir. Donanımsal cüzdanın implementasyonunda kullanılan yöntemler incelenerek çalışma sonucunda BIP-32 destekli donanımsal HD cüzdanda ile kripto para transferi uygulaması yapılmıştır. Tez çalışması genelden özele yöntemine göre ilerlemiştir.

Çalışmada ilk olarak, neredeyse insanlık tarihi kadar eski olan paranın tarihsel süreci ve süreklilik arz eden değişimi ile birlikte özellikle son zamanlarda kullanım alanın kripto paraya doğru genişlemesi durumundan bahsedilmiştir. Ayrıca mevcut merkezi sistemden dağıtık sisteme geçilmesinin anlatıldığı bir giriş sunulmuştur. kripto paranın güvenli bir ortamda saklanabilmesini sağlayan cüzdan kavramından bahsedilmiştir. Mevcut durumda en güvenli cüzdan çeşidinin akıllı kartlar gibi donanımsal cüzdan olduğu vurgulanmıştır. Bu yüzden bir sonraki bölümde akıllı kartlar hakkında genel bilgilere değinilmiştir.

Sonraki bölümde cüzdanın çalışacağı blokzincir ortamı hakkında, blokzincirin yapısını, elementlerini ve işleyişini içeren önemli bilgiler anlatılmıştır. Cüzdanın çalışacağı blokzincir ortamı anlatıldıktan sonra blokzincir tabanlı cüzdan çeşitleri ayrıntılı incelenmiştir ve karşılaştırılması yapılmıştır.

En son bölümde HD cüzdanın çalışma mekanizması ve uygulaması iki alt kısımda detaylı olarak incelenmiştir. Çalışma mekanizması kısmında BIP-32, BIP-39, BIP-44 prokollerine göre cüzdanın anahtarılarının üretilme adımları sıralanmış ve örnek Python kodu paylaşmıştır. Uygulama kısmında ise cüzdanda tanımlı APDU komutları listelenmiş, akıllı karta

HD cüzdan uygulaması yüklenmiş, kartın ilklendirilmesi yapılmış ve Bitcoin testnetinde para transferiyle örneklendirilmiştir.

En son kısımda ise AKİS işletim sisteminden bahsedilmiştir. Yerli ve milli işletim sistemi olan çoklu uygulamayı destekleyen AKiS'in en yeni sürümü ile cüzdan uygulamasının tasarımı yapılmıştır. Bu versiyonda anahtarlar türetilip bitcoin kripto para transfer işlemi için standart eliptik eğri "secp256k1" üzerinde ECDSA kullanılarak imzalama işlemleri yapılabilmiştir. Diğer para türlerinin de desteklenmesi için çalışma ve geliştirmeler sürmektedir.

Ek A

BIP-32 Test

```
# Ek A: BIP-32 Test
# (created by Adil TAN)
import hmac
import hashlib
import ecdsa
import struct
import binascii
from ecdsa.curves import SECP256k1
from ecdsa.ecdsa import int_to_string, string_to_int

# zorlaştırılmış-alt anahtarlar için indeks değeri
BIP0032_ZORLASTIRILMIS = 0x80000000 # 2147483648
# secp256k1 eliptik eğri tanımları
CURVE_GEN = ecdsa.ecdsa.generator_secp256k1
CURVE_ORDER = CURVE_GEN.order()
FIELD_ORDER = SECP256k1.curve.p()

##############################################################################
# 1) Temel Anahtar Hesaplama Örneği
##############################################################################
# Tohum -- m

# 512 bit tohum değeri
tohum = binascii.a2b_hex(
b'72c2876d53e48d28bb9ede3355478281301b25b9ecc243eeb66a341976e77759'
b'4c14841e0a9ceef8f6e31d454ce2edc8b88a06d61500fc7ac6627101cfc884f1')
# etiket
```

```
28 s = b'Bitcoin seed'
29
30 # hmac-sha512(anahtar = "Bitcoin seed", veri = tohum)
31 I = hmac.new(s, tohum, hashlib.sha512).digest()
32 m = I
33 Il, Ir = I[:32], I[32:]
34 # Geniş Temel Özel Anahtar değeri, k+c
35 k = Il # Temel anahtarın sol tarası = Temel Özel Anahtar, k
36 c = Ir # Temel anahtarın sağ tarası = Temel Zincir Kodu, c
37 # K Ortak Anahtarı k Özel Anahtarına karşılık gelir.
38 k = ecdsa.SigningKey.from_string(k, curve=SECP256k1)
39 K = k.get_verifying_key()
40 # veya (K_i = G * k = nokta(k)) şeklinde hesaplanır.
41 # k_int = string_to_int(k)
42 # K_noktasi = CURVE_GEN * k_int
43 # K_i = ecdsa.VerifyingKey.from_public_point(K_noktasi, curve=SECP256k1)
44
45 print("Temel Anahtar Hesaplama: m")
46 print("Tohum (hex):", tohum.hex())
47 print("Temel geniş anahtar (hex):", m.hex())
48 print("Temel özel anahtar (hex):", k.to_string().hex())
49 print("Temel zincir kodu  (hex):", c.hex())
50 print("Temel ortak anahtar (hex):", K.to_string("compressed").hex())
51 print()
52
53 ##############################################################################
54 # 2) Normal-(Alt Geniş Özel Anahtar) Hesaplama Örneği
55 ##############################################################################
56 # m -- m/0, m/1, m/2,..., m/2147483648
57
58 print("Normal-(Alt Geniş Özel Anahtar) Hesaplama: m/0")
59 indeks = 0
60 # hmac-sha512(c, (K + indeks))
61 veri = K.to_string("compressed") + struct.pack('>L', indeks)
62 hmac1 = hmac.new(c, veri, hashlib.sha512).digest()
63
64 Il1, Ir1 = hmac1[:32], hmac1[32:]
65 alt_zincir_kod = Ir1
66 print("Alt Zincir Kodu (hex):", alt_zincir_kod.hex())
67
68 # Alt özel anahtar hesaplama, (k + Il) % n, n eğri derecesi
```

```
alt_ozel_anahtar = (string_to_int(k.to_string()) + string_to_int(Il1))\
                   % (ecdsa.ecdsa.generator_secp256k1.order())

print("Alt özel anahtar (hex):", int_to_string(alt_ozel_anahtar).hex())

# Alt özel anahtara karşılık gelen alt ortak anahtar hesaplama
alt_ozel_anahtar = ecdsa.SigningKey.from_string\
    (int_to_string(alt_ozel_anahtar), curve=SECP256k1)

alt_ortak_anahtar = alt_ozel_anahtar.get_verifying_key()
print("Alt ortak anahtar (hex):",
      alt_ortak_anahtar.to_string("compressed").hex())
print()

################################################################################
# 3) Zorlaştırılmış-(Alt Geniş Özel Anahtar) Hesaplama Örneği
################################################################################
# m -- m/2147483648, m/2147483649,..., m/4294967295

print("Zorlaştırılmış-(Alt Geniş Özel Anahtar) Hesaplama: m/2147483648")
indeks2 = BIP0032_ZORLASTIRILMIS
# hmac-sha512(c, ('00' + k + i))
veri2 = struct.pack('x') + k.to_string() + struct.pack('>L', indeks2)
hmac2 = hmac.new(c, veri2, hashlib.sha512).digest()

Il2, Ir2 = hmac2[:32], hmac2[32:]
alt_zincir_kod2 = Ir2
print("Alt Zincir Kodu (hex):", alt_zincir_kod2.hex())

# Alt özel anahtar hesaplama, (k + Il) % n, n eğri derecesi
alt_ozel_anahtar2 = (string_to_int(k.to_string()) + string_to_int(Il2))\
                    % (ecdsa.ecdsa.generator_secp256k1.order())

print("Alt özel anahtar (hex):", int_to_string(alt_ozel_anahtar2).hex())

# Alt özel anahtara karşılık gelen alt ortak anahtar hesaplama
alt_ozel_anahtar2 = ecdsa.SigningKey.from_string\
    (int_to_string(alt_ozel_anahtar2), curve=SECP256k1)

alt_ortak_anahtar2 = alt_ozel_anahtar2.get_verifying_key()
print("Alt ortak anahtar (hex):",
```

```
110         alt_ortak_anahtar2.to_string("compressed").hex())
111 print()
112
113 ##############################################################################
114 # 4) Normal-Alt Geniş Açık Anahtar Hesaplama Örneği
115 ##############################################################################
116 # m -- p/0, p/1,...,p/2147483648
117
118 print("Normal-Alt Geniş Açık Anahtar Hesaplama: m/0")
119 indeks3 = 0
120 # hmac-sha512(c, (K + indeks))
121 veri3 = K.to_string("compressed") + struct.pack('>L', indeks3)
122 hmac3 = hmac.new(c, veri3, hashlib.sha512).digest()
123
124 Il3, Ir3 = hmac3[:32], hmac3[32:]
125 alt_zincir_kod3 = Ir3
126 print("Alt Zincir Kodu (hex):", alt_zincir_kod3.hex())
127
128 # Eğri noktası kurma işlemi, Il*G+K
129 Il_int = string_to_int(Il3)
130 alt_ortak_anahtar_noktasi = Il_int * CURVE_GEN + K.pubkey.point
131
132 # Ortak anahtarı eğri noktasından alma işlemi
133 K_i = ecdsa.VerifyingKey.from_public_point(alt_ortak_anahtar_noktasi,
        curve=SECP256k1)
134
135 alt_ortak_anahtar3 = K_i
136 print("Alt ortak anahtar (hex):",
137         alt_ortak_anahtar3.to_string("compressed").hex())
138 ##############################################################################
```

Kaynakça

[1] Frederic S Mishkin. *The economics of money, banking, and financial markets*. Pearson education, 2007.

[2] J.N. Smithin. *What is Money?* Routledge international studies in money and banking. Routledge, 2000. ISBN 9780415206907. URL `https://books.google.com.tr/books?id=MDU-NTEJziMC`. (accessed: 01.12.2019).

[3] *Principles of Microeconomics*, pages 22–32. SOUTH WESTERN, 2014. ISBN 9781305242784. URL `https://books.google.com.tr/books?id=cS72sgEACAAJ`.

[4] T. Greco, V. Robin, and K. Kerney. *Money: Understanding and Creating Alternatives to Legal Tender*. Chelsea Green Publishing, 2001. ISBN 9781603581882. URL `https://books.google.com.tr/books?id=SYbBxpqxx8MC`.

[5] Ahmet USTA. Paranın Serüveni-Kripto Paraların Öncesi ve Sonrası, 2018.

[6] D Karayew. *The history of credit cards*. PhD thesis, SSU Publishing House, 2012.

[7] Satoshi Nakamoto and A Bitcoin. A peer-to-peer electronic cash system. *Bitcoin.–URL: https://bitcoin. org/bitcoin. pdf*, 2008.

[8] Vitalik Buterin et al. Ethereum white paper. *GitHub repository*, 1:22–23, 2013.

[9] Christopher Ferris. IBM Blokchain based on Hyperledger Fabric from the Linux Foundation, 2020. URL `https://www.ibm.com/blockchain/hyperledger`. (accessed: 05.07.2020).

[10] Sony Computer Science Laboratories Inc. Sony CSL Develops Cryptocurrency Hardware Wallet Technology Using IC Cards, 2018. URL `https://www.sonycsl.co.jp/press/prs201810-2/?lang=en`.

[11] MGCSA Walport et al. Distributed ledger technology: Beyond blockchain. *UK Government Office for Science*, 1:1–88, 2016.

[12] Sun, Yi and Fan, Lingjun and Hong, Xuehai. Technology development and application of blockchain: Current status and challenges. *Chinese Journal of Engineering Science*, 20:27, 01 2018. doi: 10.15302/J-SSCAE-2018.02.005.

[13] Rebecca Campbell. Delaware Passes Groundbreaking Blockchain Regulation Bill, 2017. URL `https://www.ccn.com/delaware-passes-groundbreaking-blockchain-regulation-bill/`.

[14] Tobias Bamert, Christian Decker, Roger Wattenhofer, and Samuel Welten. Bluewallet: The secure bitcoin wallet. In *International Workshop on Security and Trust Management*, pages 65–80. Springer, 2014.

[15] Bitsafe. bitsafe-example-wallet, 2014. URL `https://github.com/someone42/bitsafe-example-wallet`.

[16] Trezor. Trezor Hardware wallet, 2014. URL `https://wiki.trezor.io/Trezor`. (accessed: 12.02.2020).

[17] Ledger. Java Card implementation of Ledger Bitcoin Hardware Wallet, 2015. URL `https://github.com/LedgerHQ/ledger-javacard`. (accessed: 26.11.2019).

[18] Ledger. Ledger Family Pack. URL `https://www.ledger.com/`.

[19] Christian Decker. *On the scalability and security of bitcoin*. PhD thesis, ETH Zurich, 2016.

[20] Miraje Gentilal, Paulo Martins, and Leonel Sousa. TrustZone-backed bitcoin wallet. In *Proceedings of the Fourth Workshop on Cryptography and Security in Computing Systems*, pages 25–28, 2017.

[21] Matouš Skála. Bitcoinová peněženka pro Android podporující zařízení TREZOR. B.S. thesis, České vysoké učení technické v Praze. Vypočetní a informační centrum., 2018.

[22] Wouter de Boer. Power Analysis of the Ledger NanoS, 2018.

[23] Myrto Arapinis, Andriana Gkaniatsou, Dimitris Karakostas, and Aggelos Kiayias. A formal treatment of hardware wallets. In *International Conference on Financial Cryptography and Data Security*, pages 426–445. Springer, 2019.

[24] Yasir Emre Bulut. Secure Hardware Cryptocurrency Wallet within Common Criteria Framework. Master's thesis, İstanbul Şehir Üniversitesi, 2019.

[25] Aleksandr Tsõganov. *Kasutaja identiteedi integreerimine Ethereum Nutika Lepingu Rahakotiga*. PhD thesis, 2019.

[26] Marc Andreessen. Why bitcoin matters. *https://dealbook.nytimes.com/2014/01/21/why-bitcoin-matters*, 2014.

[27] Coindesk. State of Blockchain. Technical Report 4, Coindesk, 7 2017. URL `https://www.coindesk.com/wp-content/uploads/research/state-of-blockchain/2017/q4/sob2017q4-2018.pdf`.

[28] Alan Greenspan. Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction. By Arvind Narayanan, Joseph Bonneau, Edward Felten, Andrew Miller, and Steven Goldfeder. Princeton and Oxford: Princeton University. *Journal of Economic Literature*, 55:648, 2017.

[29] Huabing Zhao. Hash Pointers and Data Structures. URL `https://medium.com/@zhaohuabing/hash-pointers-and-data-structures-f85d5fe91659`.

[30] Wikipedia contributors. Merkle Ağacı, 2004. URL `https://tr.wikipedia.org/wiki/Merkle_a%C4%9Fac%C4%B1`. (accessed: 22.06.2020).

[31] Daniel RL Brown. The exact security of ECDSA. In *Advances in Elliptic Curve Cryptography*. Citeseer, 2000.

[32] wobine. Ecdsa İmzalama ve İmza doğrulama, 2014. URL `https://github.com/wobine/blackboard101/blob/master/EllipticCurvesPart5-TheMagic-SigningAndVerifying.py`. (accessed: 11.06.2020).

[33] CoinMarketCap. All Cryptocurrencies, 2020. URL `https://coinmarketcap.com/all/views/all/`. (accessed: 24.03.2020).

[34] Binance Academy. Crypto Wallet Types, 2020. URL `https://www.binance.vision/blockchain/crypto-wallet-types-explained`. (accessed: 24.03.2020).

[35] Anisa Batabyal. Desktop Wallet, 2020. URL `https://coinswitch.co/news/desktop-wallet`. (accessed: 19.02.2020).

[36] S. Meiklejohn. Top Ten Obstacles along Distributed Ledgers Path to Adoption. *IEEE Security Privacy*, 16(4):13–19, 2018.

[37] Coinbase. Coinbase Security, 2020. URL `https://www.coinbase.com/security`.

[38] Arvind Narayanan, Joseph Bonneau, Edward Felten, Andrew Miller, and Steven Goldfeder. *Bitcoin and cryptocurrency technologies: a comprehensive introduction*. Princeton University Press, 2016.

[39] Pieter Wuille and other Bitcoin Mediawiki BIP-44 Contributors. Hierarchical Deterministic Wallets, 2012. URL `https://github.com/bitcoin/bips/blob/master/bip-0032.mediawiki`. (accessed: 19.02.2020).

[40] BIPS Contributors. Bitcoin Improvement Proposals. URL `https://github.com/bitcoin/bips/blob/master/README.mediawiki`. (accessed: 19.02.2020).

[41] Economic majority. URL `https://en.bitcoin.it/wiki/Economic_majority`. (accessed: 19.02.2020).

[42] Timur Badretdinov. Bitcoin Private Key, 2018. URL `https://www.freecodecamp.org/news/how-to-generate-your-very-own-bitcoin-private-key-7ad0f4936e6c/#cryptographically-strong-rng`. (accessed: 18.06.2020).

[43] Pavol Rusnak Marek Palatinus and other Bitcoin Mediawiki BIP-44 Contributors. Multi-Account Hierarchy for Deterministic Wallets, 2014. URL `https://github.com/bitcoin/bips/blob/master/bip-0044.mediawiki`. (accessed: 19.02.2020).

[44] Satochip. SatochipApplet, 2015. URL `https://github.com/Toporin/SatochipApplet`. (accessed: 13.01.2020).

[45] Thomas Voegtlin. Electrum Bitcoin Wallet, 2015. URL `https://github.com/spesmilo/electrum`. (accessed: 13.03.2020).