



پایان نامه مقطع کارشناسی مهندسی کامپیوتر - نرم افزار

پیاده سازی و ایجاد یک رمز ارز جدید مبتنی بر بلاک چین

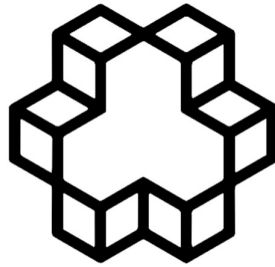
حامد محمدی

استاد راهنما:

دکتر سعید صدیقیان کاشی

پاییز ۹۷

الحمد لله الذي
خلقنا من
الحمم



دانشگاه صنعتی خواجه نصیرالدین طوسی
دانشکده مهندسی کامپیوتر

تأییدیه هیات داوران

هیات داوران پس از مطالعه پایان نامه و شرکت در جلسه دفاع از پایان نامه تهیه شده تحت عنوان:

پیاده سازی و ایجاد یک رمز ارز جدید مبتنی بر بلاک چین

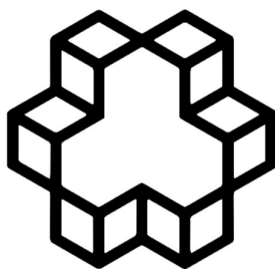
توسط آقای حامد محمدی صحت و کفایت تحقیق انجام شده را برای اخذ درجه کارشناسی
در رشته مهندسی کامپیوتر گرایش نرم افزار مورد تأیید قرار می دهند.

امضاء

استاد راهنما آقای دکتر سعید صدیقیان کاشی

امضاء

استاد ارزیاب آقای دکتر سید حسین خواسته



دانشگاه صنعتی خواجه نصیرالدین طوسی

دانشکده مهندسی کامپیوتر

اظهارنامه دانشجو

اینجانب حامد محمدی دانشجوی مقطع کارشناسی رشته مهندسی کامپیوتر گرایش نرم افزار

گواهی می‌نمایم که تحقیقات ارائه شده در پروژه با عنوان:

پیاده سازی و ایجاد یک رمز ارز جدید مبتنی بر بلاک چین

با راهنمایی استاد محترم جناب آقای دکتر سعید صدیقیان کاشی توسط شخص اینجانب انجام

شده است. صحت و اصالت مطالب نگارش شده در این پروژه مورد تأیید می‌باشد و در تدوین متن پروژه

چارچوب (فرمت) مصوب دانشگاه را به طور کامل رعایت کرده‌ام.

امضاء دانشجو:

تاریخ:

حق طبع، نشر و مالکیت نتایج

- ۱- حق چاپ و تکثیر این پروژه متعلق به نویسنده و استاد راهنمای آن می‌باشد. هرگونه تصویربرداری از کل یا بخشی از پروژه تنها با موافقت نویسنده یا استاد راهنما یا کتابخانه دانشکده مهندسی برق و کامپیوتر دانشگاه صنعتی خواجه‌نصیرالدین طوسی مجاز می‌باشد.
- ۲- کلیه حقوق معنوی این اثر متعلق به دانشگاه صنعتی خواجه‌نصیرالدین طوسی می‌باشد و بدون اجازه کتبی دانشگاه به شخص ثالث قابل واگذاری نیست.
- ۳- استفاده از اطلاعات و نتایج موجود پروژه بدون ذکر مرجع مجاز نمی‌باشد.

چکیده

در دنیای امروز باتوجه به پیشرفته‌های انجام شده و نیاز به انجام پرداخت‌های سریع و مطمئن به صورت اتوماتیک توسط عامل‌های کامپیوتری و حذف واسطه‌ها از پرداخت به نظر می‌رسد مدل‌های پرداختی فعلی دنیا قادر به پاسخگویی به نیازهای این زمینه نباشند. از این رو رمز ارزها^۱ به عنوان پاسخی بر نیاز به پرداخت‌های سریع و امن و بدون واسطه مورد توجه قرار گرفته‌اند. اولین رمز ارز دنیا با نام بیت کوین^۲ در سال ۲۰۰۹ توسط فرد یا افرادی ناشناس با نام مستعار ساتوشی ناکاماتو^۳ معرفی شده است که از این تاریخ به بعد تلاش‌های زیادی در راستای پیش‌برد تکنولوژی بلاک چین انجام شده است، یکی از این تلاش‌ها رمز ارز ParsiCoin می‌باشد که در اینجا به بررسی آن می‌پردازیم.

رمز ارز گفته شده یک رمز ارز مبتنی بر بلاک‌چین‌های نسل سوم می‌باشد که برای فعالیت‌های مرتبط با اینترنت اشیا^۴ و پرداخت‌های ریز نیز می‌تواند مورد استفاده قرار گیرد. همچنین در این رمز ارز برای نگه داری اطلاعات حساب‌های کاربری افراد از سیستم ماشین حالت^۵ استفاده می‌شود که ریشه درخت مرکله^۶ اطلاعات سیستم در هر لحظه حالت سیستم نامیده می‌شود.

^۱ Cryptocurrency

^۲ Bitcoin

^۳ Satoshi Nakamoto

^۴ IOT

^۵ State machine

^۶ Merkle tree

فهرست مطالب

۱ فصل اول : مقدمه	۱
۱-۱ تاریخچه مختصری از پول	۲
۱-۱-۱ استاندارد طلا	۲
۱-۱-۲ پول بی پشتوانه	۳
۱-۲ تراکنش های مالی و نظام بانکی	۳
۱-۲-۱ اشکالات تراکنش های مالی فعلی	۴
۱-۳ ارزهای دیجیتال	۵
۱-۳-۱ تاریخچه ارزهای دیجیتال	۵
۲ فصل دوم : برخی مفاهیم پایه بکار رفته در رمز ارزها	۷
۲-۱ رمز نگاری	۸
۲-۱-۱ رمز نگاری متقارن	۸
۲-۱-۲ رمز نگاری نامتقارن	۹
۲-۱-۲-۱ الگوریتم های تبادل کلید	۱۰
۲-۱-۲-۲ امضای دیجیتال	۱۱
۲-۱-۲-۳ رمزنگاری ECC	۱۱
۲-۱-۲-۴ تهدیدهای متوجه رمزنگاری نامتقارن	۱۲
۲-۲ توابع درهم سازی یک طرفه	۱۳
۲-۲-۱ تابع درهم سازی رمزنگارانه	۱۳
۲-۳ درخت مرکله	۱۴
۲-۴ کدبندی نویسه	۱۵

۱۶.....	۲-۵ شبکه نظیر به نظیر
۱۷.....	۲-۶ ماشین مجازی
۱۹.....	۳ فصل سوم : بلاک چین
۲۰.....	۳-۱ تعریف مفهومی بلاک چین
۲۰.....	۳-۲ تعریف ساختاری بلاک چین
۲۱.....	۳-۳ ساختار هربلاک
۲۱.....	۳-۳-۱ بخش دادهای بلاک
۲۲.....	۳-۳-۲ نماد یا امضا بلاک
۲۲.....	۳-۳-۳ نماد بلاک قبل از خود
۲۲.....	۳-۴ بررسی ساختار بلاک چین
۲۳.....	۳-۵ خواص بلاک چین
۲۳.....	۳-۶ بررسی کارکرد شبکه‌های بلاک چین
۲۴.....	۳-۷ انواع بلاک چین
۲۴.....	۳-۸ اثبات کار
۲۵.....	۳-۸-۱ حمله ۵۱ درصد
۲۶.....	۳-۸-۲ تغییرات سختی
۲۶.....	۳-۹ استخراج رمزارز
۲۷.....	۳-۱۰ دیگر اثبات‌ها
۲۷.....	۳-۱۰-۱ اثبات سهام (POS)
۲۸.....	۳-۱۰-۲ اثبات حافظه
۲۸.....	۳-۱۰-۳ گواهی سوزاندن
۲۸.....	۳-۱۱ بلاک چین‌های نسل دوم

۳-۱۱-۱	قرار دادهای هوشمند	۲۹
۳-۱۲	قاعده طولانی ترین زنجیره	۲۹
۳-۱۳	فورک در بلاک چین	۳۰
۳-۱۳-۱	فورک نرم	۳۱
۳-۱۳-۲	فورک سخت	۳۱
۳-۱۴	مشکلات بلاک چینها	۳۲
۳-۱۴-۱	هش گراف	۳۳
۳-۱۵	بلاک چینهای نسل سوم	۳۴
۳-۱۵-۱	نحوه کارکرد بلاک چینهای نسل سوم	۳۴
۳-۱۵-۲	پردازش تراکنشها در بلاک چینهای نسل سوم	۳۵
۳-۱۶	بلاک چینهای نسل چهارم	۳۶
۳-۱۷	بلاک چین در خارج از رمز ارزها	۳۷
۴	فصل چهارم : بررسی کارکرد چند رمز ارز مهم	۳۹
۴-۱	بیت کوین	۴۰
۴-۱-۱	ساختار بلاک چین بیت کوین	۴۰
۴-۱-۲	تراکنشها در بیت کوین	۴۱
۴-۱-۳	ساختار UTXO در بیت کوین	۴۲
۴-۱-۴	کیف پول بیت کوین	۴۲
۴-۲	اتریوم	۴۲
۴-۲-۱	بررسی ساختار EVM	۴۳
۴-۲-۲	زبان Solidity	۴۳
۴-۲-۳	برنامههای توزیع شده	۴۳

۴۴	۴-۲-۴ توکن‌های ERC20
۴۴	۴-۲-۵ ساختار کیف پول‌های اتریوم
۴۴	۴-۲-۶ شبکه Ropsten
۴۵	۴-۲-۷ Web3.js
۴۵	۴-۳ آیوتا
۴۵	۴-۳-۱ توکن‌های IOTA
۴۵	۴-۳-۲ نظیر یابی در IOTA
۴۶	۴-۳-۳ امکان جدا شدن و پیوستن تعدادی از گره‌ها در IOTA
۴۷	۵ فصل پنجم: پارسی کوین
۴۸	۵-۱ حالت سیستم
۴۸	۵-۲ حساب کاربری
۴۹	۵-۳ ساختار تراکنش‌های در پارسی کوین
۵۰	۵-۴ ساختار گره‌های DAG در پارسی کوین
۵۲	۵-۵ کیف پول‌ها در پارسی کوین
۵۲	۵-۶ نویسه بندی‌ها در پارسی کوین
۵۳	۵-۷ انواع توابع درهم سازی در پارسی کوین
۵۳	۵-۸ رمز نگاری در پارسی کوین
۵۳	۵-۹ اجزا سامانه
۵۴	۵-۹-۱ کامپوننت Base
۵۵	۵-۹-۱-۱ کلاس AES
۵۸	۵-۹-۱-۲ کلاس Util
۵۸	۵-۹-۲ کامپوننت اصلی ParsiCoin

۵۹.....	۵-۹-۲-۱ کلاس سازنده درخت مرکله.....
۵۹.....	۵-۹-۲-۲ کلاس مرتبط با پایگاه داده.....
۵۹.....	۵-۹-۳ کامپوننت PVM.....
۶۰.....	۵-۹-۳-۱ اجزا PVM.....
۶۲.....	۵-۹-۳-۲ دستورات PVM.....
۶۳.....	۵-۹-۴ کامپوننت CLI.....
۶۳.....	۵-۹-۵ دستورات CLI.....
۶۳.....	۵-۹-۵-۱ دستور Init.....
۶۴.....	۵-۹-۵-۲ دستور Exite.....
۶۴.....	۵-۹-۵-۳ دستور Sync.....
۶۵.....	۵-۹-۵-۴ دستور Account.....
۶۵.....	۵-۹-۵-۵ دستور PrivateKey.....
۶۶.....	۵-۹-۵-۶ دستور Send.....
۶۶.....	۵-۹-۵-۷ دستور Recive.....
۶۶.....	۵-۹-۵-۸ دستور Peer.....
۶۷.....	۵-۹-۵-۹ دستور Help.....
۶۷.....	۵-۹-۵-۱۰ دستور cls.....
۶۷.....	۵-۱۰ لایه شبکه.....
۶۸.....	۵-۱۰-۱ ارتباط امن در شبکه.....
۷۱.....	۵-۱۱ بسته‌های نرم افزاری استفاده شده.....
۷۱.....	۵-۱۲ سورس کنترل.....

فهرست تصاویر

تصویر ۲-۱: نمونه یک درخت درهم سازی	۱۵
تصویر ۲-۲: نمونه یک شبکه سرویس گیرنده-سرویس دهنده	۱۶
تصویر ۲-۳: نمونه یک شبکه نظیر به نظیر	۱۷
تصویر ۲-۴: نحوه کارکرد CLR	۱۷
تصویر ۳-۱: شمای کلی یک بلاک چین	۲۱
تصویر ۳-۲: ساختار بلاک چین	۲۳
تصویر ۳-۳: نمودار تغییرات سختی شبکه بیت کوین	۲۶
تصویر ۳-۴: قاعده طولانی ترین شاخه	۳۰
تصویر ۳-۵: نمونه فورک سخت در بلاک چین	۳۲
تصویر ۳-۶: مقایسه هش گراف با بلاکچین	۳۴
تصویر ۳-۷: نمونه یک DAG	۳۵
تصویر ۳-۸: وضعیت تراکنشها در یک DAG	۳۶
تصویر ۳-۹: رهگیری مالکیت خودرو بدون بلاک چین	۳۸
تصویر ۳-۱۰: رهگیری مالکیت خودرو با بلاک چین	۳۸
تصویر ۴-۱: ساختار بلاکهای بیت کوین	۴۱
تصویر ۴-۲: ساختار تراکنشهای بیت کوین	۴۱
تصویر ۴-۳: ساختار EVM	۴۳
تصویر ۴-۴: جدا شدن و اتصال مجدد	۴۶
تصویر ۵-۱: ارتباط بین اجزا در پارسی کوین	۵۴
تصویر ۵-۲: اجرای دستور INIT	۶۴
تصویر ۵-۳: صفحه اول برنامه	۶۴

تصویر ۴-۵: راهنمای دستور ACCOUNT.....	۶۵
تصویر ۵-۵: اجرا دستور SEND و اطلاعات تراکنش ایجاد شده و امضا شده.....	۶۶
تصویر ۶-۵: راهنمای دستور PEER.....	۶۶
تصویر ۷-۵: اجرا دستور HELP.....	۶۷

فهرست معادلات

معادله ۱-۲: منحنی‌های بیضوی در دو بعد.....	۱۲
معادله ۲-۲: گره‌های درخت درهم سازی.....	۱۴

فهرست کدها

قطعه کد ۱-۵: تابع سازنده تراکنش در حساب.....	۴۹
قطعه کد ۲-۵: محاسبه هش تراکنش.....	۵۰
قطعه کد ۳-۵: ایجاد تراکنش.....	۵۰
قطعه کد ۴-۵: تابع مسئله POW برای هر گره.....	۵۲
قطعه کد ۵-۵: کلاس AES.....	۵۷
قطعه کد ۶-۵: مقایسه سختی.....	۵۸
قطعه کد ۷-۵: سازنده درخت مرکله.....	۵۹
قطعه کد ۸-۵: تابع PUSH در پشته.....	۶۰
قطعه کد ۹-۵: تابع فراخوان دستورات در واحد پردازش.....	۶۱

۶۱	قطعه کد ۵-۱۰: تابع POP در پشته
۶۲	قطعه کد ۵-۱۱: تابع اجرا دستور DOUBLES512
۶۲	قطعه کد ۵-۱۲: تابع اجرا دستور CHEKCSIG
۶۹	قطعه کد ۵-۱۳: کلاس RSAKEYEXCHCLIENT
۶۹	قطعه کد ۵-۱۴: کلاس SECURELINECLIENT
۷۰	قطعه کد ۵-۱۵: کلاس RSAKEYECHSERVER
۷۰	قطعه کد ۵-۱۶: کلاس SECURELINESERVER
۷۱	قطعه کد ۵-۱۷: استفاده از SECURELINE

۱ فصل اول : مقدمه

در طول تاریخ بشر همواره ردپای معامله به روش‌های مختلف دیده شده است، این معاملات در ابتدای تاریخ به صورت مبادله کالا با کالا صورت می‌گرفتند که به مرور و با گذر زمان به روش‌های کارآمد تری نظیر انتخاب یک کالا به عنوان مرجع روی آورده شد، تا اینکه در نهایت با ضرب اولین سکه‌ها از طلا توسط اقوام باستانی در آسیای صغیر مفهوم پول ایجاد شد.

۱-۱ تاریخچه مختصری از پول

اولین پول‌ها توسط قوم باستانی لیدی‌ها ساکن در منطقه آسیای صغیر تقریباً در قرن هفت قبل از میلاد ضرب شدند این سکه‌ها اغلب از جنس فلزات گرانبها مانند طلا، نقره و یا مس ضرب می‌شدند و حاوی مهر حکومت ضرب کننده آن‌ها به همراه تصویر پادشاه و یا اشخاص برجسته آن قوم بودند که به این سکه‌ها رسمیت و اعتبار می‌بخشید رفته رفته دیگر اقوام نیز به ضرب سکه روی آوردند و این گونه بود که دوران سکه‌ها در تاریخ بشر آغاز شد، به طوری که امروزه نیز در اکثر تحقیقات باستان شناسی از سکه‌های هر قوم و ملیت به عنوان نماد آن قوم و ملیت یاد شده و دارای اهمیت زیادی می‌باشد.

۱-۱-۱ استاندارد طلا^۱

با گذر زمان و افزایش حجم معاملات تجاری، لزوم حمل و نقل مقدار زیادی سکه ایجاد می‌شد که حمل این سکه‌ها دردسرهای زیادی را به همراه داشت، بنابر این سازمان‌هایی به وجود آمدند که در ازای تعداد مشخصی سکه حواله‌هایی صادر می‌کردند که در اکثر مناطق دنیای آن زمان با مراجعه به دفتر آن سازمان در هر شهر قابل تبدیل شدن به آن مقدار سکه بودند بنابر این این حواله‌ها دارای ارزشی معادل به سکه طلا بودند که به مرور زمان در بین مردمان رواج پیدا کردند به طوری که امروز نیز در قوانین اساسی

¹ Gold Standard

اکثر کشورها میزان مشخصی از طلا به ازای هر واحد پول آن کشور اختصاص داده می‌شود^۱. اما درنهایت با توجه به لزوم وجود نقدینگی در کشورها و با افزایش جمعیت و برخی عوامل دیگر دولت‌ها مجبور به چاپ مقادیر پول بیشتر از ذخایر طلای خود شدند که به این ترتیب ارزهای بی‌پشتوانه^۲ ایجاد شدند.

۱-۲ پول بی‌پشتوانه

پول بی‌پشتوانه، پول حکمی، پول دستوری یا پول اعتباری پولی است که ارزشش ذاتی نبوده و تنها ناشی از دستور دولتی یا قانون باشد. نام آن از واژه لاتین فیات به معنی بگذارید انجام شود گرفته شده‌است. بنابر این در این نوع پول به دلیل عدم وجود یک پشتوانه محکم ارزش پول معادل ارزش چاپ کننده آن در نظر گرفته می‌شود و با تغییرات شرایط سیاسی یا اقتصادی یک کشور ارزش پول آن کشور نیز تغییر می‌کند. همچنین دولت‌ها در زمان‌های کمبود نقدینگی در کشورهايشان با اقدام به چاپ بی‌رویه این نوع پول باعث ایجاد تورم و کاهش ارزش پول خود می‌شوند، مجموع عوامل ذکر شده دلیل تغییراتی است که امروزه در قیمت ارزهای کشورهای مختلف نسبت به یکدیگر شاهد هستیم؛ لازم به ذکر است که تمامی ارزهای مرجع امروزی از نوع پول بی‌پشتوانه می‌باشند.

۱-۲ تراکنش‌های مالی و نظام بانکی

با گذر زمان و نیاز به انجام تراکنش‌های مالی با مبالغ سنگین یا انجام پرداخت در زمان مشخصی در آینده بانک‌های کشورهای مختلف اقدام به ایجاد سازکارهای مختلفی نظیر دسته چک‌ها و یا چک‌های رمز دار بین بانکی و همچنین انتقال از حساب یک شخص به حساب شخص دیگر نمودند که در نهایت با ورود تکنولوژی‌های جدید مواردی از قبیل کارت‌های اعتباری^۳ و یا کارت‌های نقدی^۴ امروزه به عنوان یکی از روش‌های اصلی انجام تراکنش‌های مالی شناخته می‌شوند که این کارتها قابلیت انجام انواع تراکنش‌های مالی

^۱ به طور مثال هر یک ریال ایران برابر یکصد و هشت هزار و پنجاه و پنج دهه میلیونی (۵۵۰۸۰۱۰۰۰) گرم طلای خالص است.

^۲ Fiat Currency

^۳ Credit Card

^۴ Debit Card

مختلف را با کمک دستگاههایی نظیر خود پردازها^۱ و یا پایانه‌های فروشگاهی^۲ فراهم می‌آوند همچنین به کمک دستگاه‌های پرداخت اینترنتی قابلیت پرداخت با کمک اینترنت نیز فراهم شده است.

۱-۲-۱ اشکالات تراکنش‌های مالی فعلی

اگرچه انجام تراکنش‌های مالی امروزه به راحتی از طریق کارت‌هایی نظیر Master Card و یا VISA Card در سطح جهانی انجام می‌پذیرند اما این سبک انجام تراکنش دارای اشکالاتی است که در ادامه به معرفی آن‌ها می‌پردازیم.

۱. زمان بین انجام تراکنش و تسویه می‌تواند طولانی باشد.
۲. دوباره کاری‌ها و نیاز به شخص سوم برای تایید اعتبار و حضور واسطه‌ها.
۳. کلاهبرداری، حمله‌های سایبری و حتی اشتباه‌های کوچک به هزینه و پیچیدگی کسب و کار می‌افزایند و اگر یک سامانه‌ی مرکزی مانند بانک به خطر بیفتد، همگی مشارکت کنندگان در شبکه با مخاطره روبه رو خواهند شد.
۴. برای استفاده از خدمات این کارت‌های اعتباری اغلب نیاز به پرداخت هزینه‌های اولیه زیاد می‌باشد.
۵. دریافت این نوع کارت‌های شامل فرآیندهای وقت گیر و کاغذ بازی‌ها و بررسی سابقه افراد است.
۶. این نوع سامانه‌ها دارای شفافیت کافی در برخی موارد نمی‌باشند، در حقیقت هیچ کس نمی‌داند که آن سازمان مرکزی صادر کننده کارت‌ها و حساب‌ها چگونه به انجام این کارها می‌پردازد.
۷. وجود چنین سامانه‌های مرکزی با نفوذ افراد و دولت‌های مختلف رخ می‌دهد که در برخی

^۱ ATM

^۲ POS

موارد می‌تواند باعث حذف برخی افراد از سامانه و قطع سرویس دهی به آن‌ها شود نظیر تحریم‌های اعمال شده علیه کشور ایران که دسترسی این کشور و مردم آن را به بیشتر سامانه‌های پرداخت جهانی قطع کرده است.

بنابر این و باتوجه به موارد فوق و همچنین گسترش روز افزون اینترنت در جهان برخی دانشمندان حوزه کامپیوتر اقدام به ایجاد نوع خاصی از ارزها که ارزهای الکترونیکی هستند نمودند که بتواند مشکلات ذکر شده در فوق را برطرف نماید و به عنوان یک سبک پرداخت جهانی با شفافیت عملکرد و بدون نیاز به اعتماد به یک شخص ثالث به عنوان مرجع تراکنش‌های مالی جهانی در نظر گرفته شوند.

۳-۱ ارزهای دیجیتال^۱

به طور کلی هر ارزی که نمود فیزیکی نداشته باشد و به عنوان واحدهای کامپیوتری در شبکه‌های خاص مورد استفاده قرار گیرد می‌تواند نوعی ارز دیجیتالی به حساب آید بنابر این باید توجه داشت که مفهوم ارز دیجیتالی به طور کلی برابر با رمز ارز^۲ که در ادامه به بررسی آن خواهیم پرداخت نمی‌باشد و هر نوع واحد پولی دیجیتالی حتی سکه‌ها و یا اعتبارات موجود در اکثر بازی‌های کامپیوتری را نیز می‌توان نوعی ارز دیجیتال دانست.

۱-۳-۱ تاریخچه ارزهای دیجیتال

ابتدایی ترین نوع ارزهای دیجیتال که در حقیقت مفاهیم استفاده شده در برخی از آن‌ها مبنای کارکرد رمز ارزهای امروزی است اغلب دارای یکاهایی معادل با یکی از ارزهای رایج بودند که کاربران با پرداخت میزان معینی ارز دولتی، ارز دیجیتال معادل آن را دریافت کرده و می‌توانستند در معاملات اینترنتی خود از آن‌ها استفاده کنند نمونه این ارزها رزرو آزادی^۳ بود که در دهه ۱۹۹۰ میلادی همزمان با حباب دات-کام

^۱ Digital Currency

^۲ Crypto Currency

^۳ Liberty Reserve

ایجاد شد. اگرچه اغلب این ارزهای دیجیتال از مفاهیمی نظیر امضا دیجیتال^۱ و مکانیزم‌هایی شبیه به گواهی اثبات کار^۲ و یا به اختصار POW نیز استفاده می‌کردند و در برخی موارد تا حدی نیز به صورت نظیر به نظیر^۳ کار می‌کردند اما همه آن‌ها در نهایت برای ایجاد توافق^۴ به یک سازمان مرکزی وابسته بودند که در حقیقت این بدان معنی بود که در انجام بزرگترین هدف خود که حذف واسطه‌ها از انجام تراکنش‌ها بود نا موفق بودند بنابر این اغلب این ارزهای دیجیتال به سرعت به فراموشی سپرده شدند با این حال مفاهیم استفاده شده در بسیاری از آن‌ها به عنوان مبنای رمز ارزهای امروزی قرار گرفت. در نهایت در سال ۲۰۰۹ میلادی با معرفی بیت کوین^۵ مشکل نیاز به واسطه برای انجام تراکنش‌ها حل شد و برای انجام توافق از یک الگوریتم قوی کامپیوتری به نام بلاک‌چین^۶ استفاده شد.

^۱ Digital signature

^۲ Proof-of-Work

^۳ Peer-To-Peer

^۴ consensus

^۵ BitCoin

^۶ Block Chain

۲ فصل دوم : برخی مفاهیم پایه بکار رفته در رمز ارزها

برای بررسی بیشتر رمز ارزها و بلاک چین ابتدا لازم به نظر می‌رسد تا با برخی مفاهیم پایه و الگوریتم‌های به کار رفته در این سامانه‌ها آشنا شویم.

۱-۲ رمز نگاری^۱

از ابتدای تاریخ نوع بشر همواره نیاز به رمز کردن پیام‌های خود را احساس کرده است، تا به طور مثال در شرایط جنگی و یا موارد خاص بتواند پیام‌های خود را بین متحدین خود به گونه ای امن منتقل کند شاید اهمیت رمز نگاری را بتوان با بررسی تاریخ جنگ جهانی دوم مشخص کرد که با شکسته شدن کدهای انیگما^۲ نوشته شده نازی‌ها توسط دانشمند انگلیسی و پدر علم کامپیوتر آلن تورینگ^۳ سر نوشت جنگ به نفع متحدین تمام شد.

در ابتدا بشر برای رمزنگاری پیام‌های خود از روش‌ها مختلفی نظیر قرار دادن سیمبل‌ها و رمزهای خاص استفاده می‌کرد تا اینکه پیوند این رمزها با ریاضیات و الگوریتم‌های خاص ریاضی پیدا شد و با بررسی رمزنگاری به عنوان شاخه ای از علم ریاضی بشر موفق به ایجاد رمزهای پیچیده شد و در نهایت با ورود کامپیوترها عرصه رمز نگاری نیز توسط این ماشین‌های قدرتمند دچار تغییراتی شد و الگوریتم‌های پیچیده رمزنگاری کامپیوتری ایجاد شدند به طوری که امروزه برخی از آنها نظیر الگوریتم AES^۴ به قدری امن در نظر گرفته می‌شوند که شکستن رمزهای آنها تقریباً غیر ممکن به نظر می‌رسد.

۱-۱-۲ رمز نگاری متقارن^۵

این نوع رمزنگاری به نوعی خاصی از رمزنگاری گفته می‌شود که در آن برای رمز کردن و رمزگشایی

^۱ cryptography

^۲ Enigma Machine

^۳ Alan Mathison Turing

^۴ Advanced Encryption Standard

^۵ Symmetric-key

پیام از یک کلید استفاده می‌شود، کلیدها ممکن است مشابه باشند یا ممکن است رابطه‌ای ساده بین دو کلید وجود داشته باشد. کلید، در عمل، نشان دهنده یک راز مشترک بین دو یا چند طرف است که می‌تواند برای حفظ اطلاعات خصوصی مورد استفاده قرار گیرد. این نیاز که هر دو طرف، دسترسی به کلیدهای مخفی داشته باشند یکی از اشکالات اصلی رمزنگاری کلید متقارن است، چرا که در حقیقت با توجه به نیاز دو طرف برای دانستن یک کلید مشخص این نوع رمزنگاری بین دو طرف که به یکدیگر اعتماد ندارند قابل استفاده نیست و دو طرف باید از قبل یکدیگر را بشناسند و آن رمز مشخص را بین خود پذیرفته باشند. برای این نوع رمزنگاری اغلب الگوریتم‌های جریانی و بلوکی مورد استفاده قرار می‌گیرند که الگوریتم‌های جریانی غالباً به رمز کردن یک واحد کوچک داده یا همان بیت به همان صورت که داده در جریان است می‌پردازند و الگوریتم‌های بلوکی به رمز کردن یک تعداد مشخص از واحد داده‌ها به عنوان بلوک می‌پردازند، به طور مثال الگوریتم AES به رمزنگاری داده در بلوک‌های ۱۲۸ بیتی می‌پردازد.

۲-۱-۲ رمزنگاری نامتقارن^۱

این نوع رمزنگاری در مقابل رمزنگاری متقارن قرار دارد و در آن‌ها برای رمز کردن و رمزگشایی پیام از دو کلید متفاوت استفاده می‌شود که یکی از آنها به کلید عمومی^۲ و دیگری به کلید خصوصی^۳ شهرت دارند و همچنین یک رابطه ریاضی بین این دو کلید برقرار است به طوری که همواره از کلید خصوصی می‌توان کلید عمومی را به دست آور اما از کلید عمومی نمی‌توان کلید خصوصی را استخراج کرد.

کارکرد الگوریتم‌های رمزنگاری نامتقارن به این صورت است که هر فرد ابتدا با ایجاد یک کلید خصوصی و نگه داشتن آن در نزد خود به صورت امن و به دست آوردن کلید عمومی نظیر آن کلید خصوصی و در اختیار عموم قرار دادن آن کلید عمومی به رد و بدل کردن پیام‌های رمز شده می‌پردازد، به این صورت که هر پیام توسط کلید عمومی که در اختیار همه هست رمز می‌شود اما فقط با استفاده از کلید خصوصی که

^۱ Public-key cryptography, or asymmetric cryptography

^۲ Public Key

^۳ Private Key

در اختیار خود فرد قابل رمزگشایی خواهد بود، بنابر این همه پیام‌های مورد نظر برای یک نفر فقط و فقط توسط همان فرد قابل رمزگشایی خواهند بود در حالی که همه افرادی که کلید عمومی را در اختیار دارند قادر به رمز کردن پیام‌های خود خواهند بود از الگوریتم‌های معروف رمزنگاری نامتقارن می‌توان از الگوریتم ¹ RSA که بر پایه هم‌نهشتی و با کمک اعداد اول بزرگ کار می‌کند و ² ECC بر اساس ساختاری جبری از منحنی‌های بیضوی بر روی میدان‌های متناهی طراحی شده‌است نام برد، البته اغلب این الگوریتم‌ها با کمک الگوریتم‌های دیگر مورد استفاده قرار می‌گیرند.

۲-۱-۲ الگوریتم‌های تبادل کلید

الگوریتم‌های رمزنگاری نامتقارن اغلب دارای یک مشکل اساسی می‌باشند که محدودیت سایز پیام قابل رمزنگاری توسط آن‌ها می‌باشد به طوری که در صورتی که اندازه پیام از حدی بزرگتر باشد برای رمزنگاری آن نیاز به کلید طولانی تری خواهد بود و طولانی کردن کلید نیز تا حدی ممکن است، بنابر این برای استفاده از آنها یا باید پیام را به قطعا کوچک شکست و رمز نمود و یا از راهکار دیگری بر مبنای الگوریتم‌های تبادل کلید استفاده نمود، در حقیقت همانطور که قبلا اشاره شد الگوریتم‌های رمزنگاری متقارن دارای مشکل اساسی نیاز به دانستن کلید یکسان توسط هر دو طرف قبل از شروع انتقال پیام می‌باشند اما در بسترهای ناامن که نیاز به رمزنگاری هست در صورتی که دو طرف از قبل با یکدیگر در ارتباط نبوده باشند چطور می‌توان کلید مورد نظر را انتقال داد؟ در اینجا با کمک الگوریتم‌های انتقال کلید و با کمک رمزنگاری نامتقارن می‌توان ابتدا به تبادل کلید الگوریتم رمزنگاری متقارن پرداخت و سپس با کمک آن کلید در ادامه پیام‌ها را به صورت متقارن رمزنگاری کرد از الگوریتم‌های معروف این دسته می‌توان از پروتکل تبادل کلید دیفی-هلمن^۳ نام برد الگوریتم استفاده شده در پروتکل امن انتقال ابرمتن^۴ و یا به اختصار <https> نیز تقریبا کارکردی مشابه آنچه گفته شد دارد.

¹ Rivest-Shamir-Adleman

² elliptic curve cryptography

³ Diffie-Hellman key exchange

⁴ Hypertext Transfer Protocol Secure

۲-۱-۲-۲ امضای دیجیتال

از دیگر کاربردهای رمزنگاری نامتقارن می‌توان به امضا دیجیتال اشاره کرد که در حقیقت الگوریتم امضا دیجیتال در رمز ارزها نیز بسیار کاربرد دارد که به اثبات هویت ارسال کننده یک پیام و یا تراکنش می‌پردازد به مانند امضا عادی در سیستم های سنتی که امضا هر فرد در انتهای هر نامه و پیامی به معنی تایید فرد مورد نظر است.

کارکرد امضا دیجیتال به این صورت است که ابتدا فرد نویسنده یک پیام و یا تراکنش برای اثبات هویت خود با کمک توابع درهم سازی یک طرفه^۱ با طول ثابت به هش کردن پیام خود می‌پردازد و سپس این بار با کمک کلید خصوصی خود رشته به دست آمده را رمز می‌کند حال پیام رمز شده نهایی را به انتهای پیام اصلی می‌افزاید که به این فرایند امضا کردن پیام گفته می‌شود حال با ارسال این پیام به شخص یا اشخاص مورد نظر آنها با داشتن کلید عمومی فرد می‌توانند امضا فرستاده شده را رمزگشایی کرده و اصل پیام را نیز هش کنند و رشته خروجی را با رشته بدست آمده از رمزگشایی امضا تطبیق دهند در صورت برابر بودن این دو رشته اثبات هویت ارسال کننده پیام صورت می‌گیرد که در حقیقت فرد متناظر با آن کلید عمومی است.

۲-۱-۲-۳ رمزنگاری ECC

باتوجه به اینکه در اغلب رمز ارزها و سامانه‌های امروزی از الگوریتم ECC به عنوان الگوریتم رمزنگاری نامتقارن استفاده می‌شود به نظر می‌رسد لازم است که به اختصار نحوه کارکرد آن را توضیح دهیم.

همانطور که پیش تر ذکر شد این الگوریتم بر اساس ساختاری جبری از منحنی‌های بیضوی بر روی میدان‌های متناهی طراحی شده‌است. که این امر باعث نیاز به کلید کوچک تری نسبت به دیگر روش‌های رمزنگاری نامتقارن می‌شود، در حقیقت برای اهداف امروزی رمزنگاری، منحنی بیضوی یک منحنی مسطح است که متشکل از نقاط رضایت بخش معادله می‌باشد.

¹ Hash Function

معادله ۲-۱: منحنی های بیضوی در دو بعد $Y^2 = X^3 + aX + b$

همراه با یک نقطه برجسته در (مختصات در اینجا از یک حوزه ثابت متناهی از مشخصه که با دو یا سه برابر نیست انتخاب می شوند، یا اینکه معادله منحنی تا حدودی پیچیده تر خواهد بود) این مجموعه همراه با عملیات گروهی از نظریه گروه بیضوی از گروه Abelian، با نقطه ای در بینهایت به عنوان عنصر هویت می باشند. ساختار گروه از گروه مقسوم علیه تنوع جبری زیرین ارث بری می کند. همان طور که برای دیگر سیستم های رمزنگاری کلید عمومی محبوب، بدون اثبات ریاضی برای امنیت ECC از سال ۲۰۰۹ منتشر شد.

درنهایت باید دانست که امنیت کامل ECC بستگی به توانایی محاسبه ضرب نقطه ای و عدم توانایی برای محاسبه حاصل ضرب با توجه به نقاط اصلی و نقاط تولید شده دارد.

۲-۱-۲-۴ تهدیدهای متوجه رمزنگاری نامتقارن

امنیت شیوه های امروزی رمز نگاری نامتقارن اغلب بر پایه سخت بودن حل مسائلی نظیر تجزیه اعداد اول و یا مسئله حل لگاریتم گسسته برای کامپیوترهای امروزی مطرح می شوند که حل این مسائل به طور عادی از روابط نمایی پیروی می کند بنابر این برای حل آنها صرف زمان بسیار بسیار زیادی توسط کامپیوترهای امروزی لازم خواهد بود به طوری که می توان به دست آوردن پاسخ آنها را عملاً غیر قابل دست یابی دانست، اما نوع دیگر کامپیوترهای که کامپیوترهای کوانتومی^۱ معروف هستند. با کمک الگوریتم کوانتومی شور^۲ قادر به حل این مسائل در زمان معقولی می باشند که در حقیقت این نگرانی را در بسیاری از افراد به وجود آورده که این نوع کامپیوترها می توانند تهدیدی برای اغلب روش های رمزنگاری و سامانه های مبتنی بر رمزنگاری نامتقارن امروزی باشند، که از همین رو تلاش هایی برای ایجاد الگوریتم های رمزنگاری نامتقارن کوانتومی^۳ شده است، البته لازم به ذکر است که هنوز کامپیوترهای کوانتومی قدرتمند ای که توانایی اجرای الگوریتم

^۱ Quantum computer

^۲ Shor's algorithm

^۳ Quantum cryptography

گفته شده را داشته باشند ساخته نشده اند، اما تخمین زده می‌شود که تا سال ۲۰۳۰ میلادی نمونه اولیه چنین کامپیوترهایی ساخته شوند، نکته قابل توجه دیگر آن است که کامپیوترهای کوانتومی تهدیدی برای الگوریتم‌هایی نظیر AES نمی‌باشند.

۲-۲ توابع درهم سازی یک طرفه

توابع درهم سازی یک طرفه به توابعی گفته می‌شود که اغلب با دریافت یک رشته بیتی با درهم ریختن رشته ورودی به ایجاد یک رشته خروجی می‌پردازند به طوری که به دست آوردن رشته ورودی از رشته خروجی امکان پذیر نباشد و همچنین به ازای هر رشته منحصر به فرد در ورودی یک رشته منحصر به فرد در خروجی ایجاد شود. به طوری که با داشتن رشته ورودی همواره بتوان به یک رشته خروجی رسید و به ازای هیچ دو رشته ورودی غیر یکسانی، یک رشته خروجی یکسان حاصل نشود. به طور کلی کاربرد این نوع توابع در شماره گذاری رشته‌ها و جداول داده درهم^۱ می‌باشند.

۲-۲-۱ تابع درهم سازی رمزنگارانه^۲

این نوع توابع درهم سازی نوع خاصی از توابع در هم سازی می‌باشند که یک رشته با طول نا مشخص را به یک رشته با طول ثابت درهم ریزی می‌کنند به طوری که رشته خروجی نمایشی از کل محتوای متن یا رشته ورودی است و می‌توان آن را نوعی «اثر انگشت دیجیتالی» برای آن متن به حساب آورد، این نوع توابع درهم سازی در امضا دیجیتال، ذخیره اطلاعات حیاتی مانند کلمه عبور کاربران در پایگاه داده، بلاک چین و بسیاری موارد دیگر کاربرد دارند از معروف ترین توابع درهم سازی رمزنگارانه می‌توان از MD4,MD5,SHA-1,SHA-2,SHA-3 نام برد.

از دیگر خصوصیت این توابع که در سیستم‌های بلاک چینی بسیار قابل توجه است آن است که در حقیقت امکان پیش بینی خروجی از روی ورودی به هیچ عنوان امکان پذیر نیست، در حقیقت کارکرد آن‌ها

¹ Hash Table

² Cryptographic hash function

به این صورت نمی باشد که رشته خروجی به ازای تغییرات مشخص ورودی به یک رشته خاص در خروجی میل کند که بتوان آن را حدس زد و با هر تغییر بسیار کوچک در رشته ورودی، رشته خروجی تغییرات قابل توجهی خواهد کرد.

۲-۳ درخت مرکله^۱

درخت مرکله و یا درخت درهم سازی به طور معمول یک درخت دودویی^۲ و یا یک درخت پیشوندی^۳ است که برگ های آن شامل یک سری داده مشخص می باشند، سپس در مراحل بالاتر مقدار هر گره از مقدار هش فرزندان خود به دست می آید به همین صورت از گره های والد برای برگ ها شروع به هش کردن مقادیر میکنیم و سطح به سطح در درخت بالا می رویم و مقدار هر گره را معادل هش مقدار فرزندان خود می گذاریم تا به ریشه درخت برسیم.

معادله ۲-۲: گره های درخت درهم سازی
$$f(n) = Hash(f(2n + 1) + f(2n + 2))$$

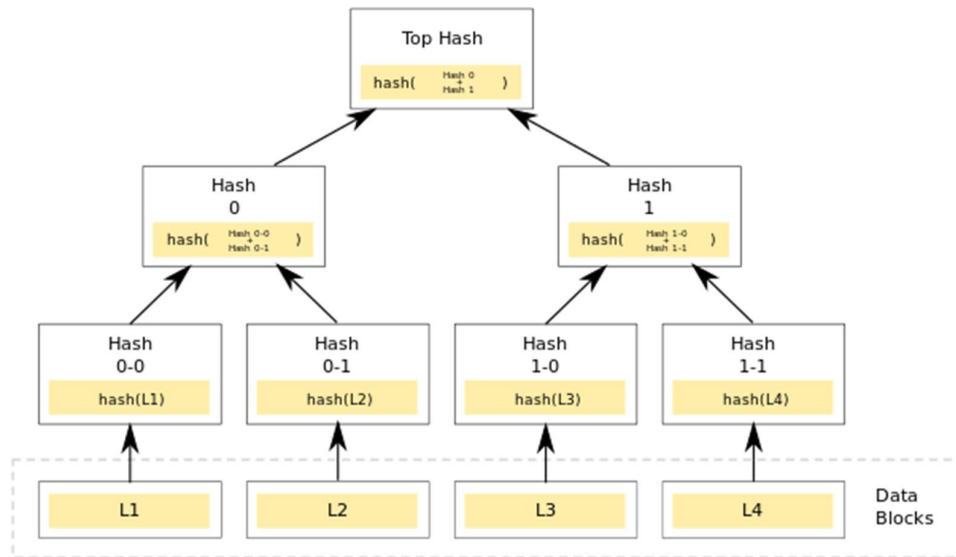
کاربرد درخت های مرکله به طور معمول در نوع خاصی از امضا دیجیتال به نام امضای لمپارت^۴ و یا تایید اعتبار یک فایل و همچنین به طور گسترده در بلاک چین می باشد.

^۱ Merkle tree

^۲ Binary Tree

^۳ Trie

^۴ Lamport Signature



تصویر ۱-۲: نمونه یک درخت درهم سازی [18]

۲-۴ کدبندی نویسه^۱

باتوجه به اینکه کوچکترین واحد قابل فهم برای کامپیوترها یک بیت است که دارای مقداری معادل ۰ یا ۱ می باشد در نتیجه همه مفاهیم از نظر کامپیوترها باید دارای یک مقدار عددی باشند، بنابر این برای نویسه^۲ های الفبایی استانداردهایی در نظر گرفته شده است که هر نویسه را به یک مقدار عددی مشخص نظیر می کند از نمونه های این استانداردها می توان به مواردی از قبیل ^۳ ASCII که فقط شامل حروف انگلیسی اعداد و برخی نویسه های خاص است و یا UTF-8 که دارای نویسه های موجود در اغلب زبان های می باشد اشاره کرد. اما این کدبندی ها اغلب برای داده هایی به کار می روند که در زبان های طبیعی دارای معنی و مفهوم می باشند و خروجی یک تابع درهم سازی یک طرفه و یا رشته خروجی حاصل از یک الگوریتم رمزنگاری اشکال سخت و عجیبی در این نوع از کدبندی ها پیدا می کنند بنابر این برای نمایش خروجی این تابع ها به صورت رشته های حرفی عددی نیز استانداردهای خاصی نظیر Hex, Base64, Base58 ایجاد شده است که Hex در حقیقت نمایش هر بایت از داده در مبنای ۱۶ است که باعث طولانی شدن رشته می شود بنابر این

^۱ Character encoding

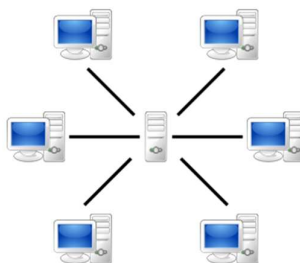
^۲ Character

^۳ American Standard Code for Information Interchange

استاندارد Base64 به طور معمول برای انتقال داده ها استفاده می شود که این استاندارد نیز به دلیل وجود نویسه هایی مانند عدد 0 و حرف o لاتین و یا حرف l کوچک لاتین و حرف I بزرگ لاتین و عدد 1 و... که در برخی فونت ها شکل یکسان دارند در صورتی که قرار باشد توسط انسان ها و نه ماشین ها مورد بررسی قرار گیرند مشکلاتی را ایجاد می کنند که برای پیشگیری از این مشکلات از استاندارد Base58 در اغلب رمز ارزها استفاده می شود که حروف و اعداد مشابه و غیر قابل تشخیص از یکدیگر در آن مشاهده نمی شوند.

۵-۲ شبکه نظیر به نظیر

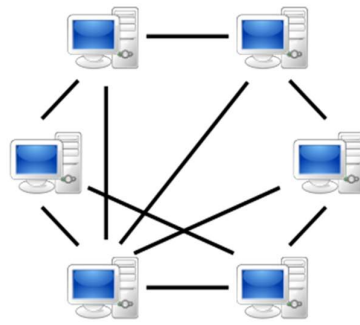
این شبکه ها در حقیقت در مقابل شبکه های سرویس دهنده-سرویس گیرنده^۱ قرار دارند و به جای آن که یک سرویس دهنده مرکزی وجود داشته باشد که همه سرویس گیرنده ها به آن متصل باشند و فقط بتوانند با سرویس دهنده انتقال پیام انجام دهند، همه نظیرها^۲ به صورت مستقیم به هم متصل هستند و در حقیقت هر نظیر حکم یک سرویس دهنده و یک سرویس گیرنده کوچک را دارد، این شبکه در رمز ارزها اهمیت بالایی دارند چراکه به کمک این شبکه ها امکان حذف شخص ثالث در تراکنش ها حذف شده است.



تصویر ۲-۲: نمونه یک شبکه سرویس گیرنده-سرویس دهنده [1]

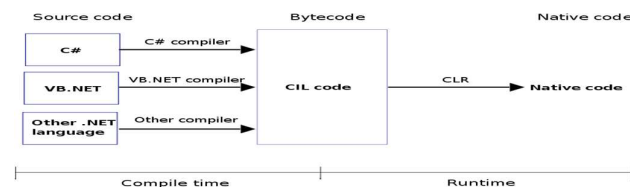
¹ Client-Server

² Peer

تصویر ۲-۳: نمونه یک شبکه نظیر به نظیر^[۱]

۲-۶ ماشین مجازی^۱

ماشین مجازی در حقیقت ماشینی است که به طور مجازی روی یک ماشین فیزیکی دیگر اجرا می‌شود و یک سری عملیات خاص انجام می‌دهد، نوع خاصی از این ماشین‌های مجازی آن‌هایی هستند که در زبان‌های برنامه نویسی سطح بالا مانند جاوا^۲ و یا زبان‌های خانواده دانت نت^۳ به ترتیب با نام‌های JVM^۴ و CLR^۵ استفاده می‌شوند. این ماشین‌های مجازی اغلب مانند ریز پردازنده‌ها^۶ دارای یک زبان اسمبلی مانند^۷ مخصوص به خود می‌باشند که کدهای زبان سطح بالاتر به این زبان‌ها ترجمه می‌شوند و سپس کدهای خروجی حاصل در ماشین مجازی مورد نظر اجرا می‌شوند در حقیقت این ماشین‌های مجازی رفتار شبیه به یک پردازنده را شبیه سازی می‌کنند.



تصویر ۲-۴: نحوه کارکرد CLR

¹ Virtual Machine

² Java

³ .Net

⁴ Java Virtual Machine

⁵ Common Language Runtime

⁶ CPU

⁷ Assembly

اغلب رمز ارزها نیز برای انجام کارهای خود دارای یک ماشین مجازی می باشند که تراکنش ها مورد نظر به صورت کدهای این ماشین مجازی در آمده و برای تایید کدهای مورد نظر اجرا می شوند به طور مثال می توان از EVM^۱ در این زمینه نامبرد.

^۱ Ethereum Virtual Machine

۳ فصل سوم : بلاک چین

همانطور که گفته شد در سال ۲۰۰۹ میلادی با معرفی بیت کوین به عنوان اولین رمز ارز توسط فرد یا افرادی با نام مستعار ساتوشی ناکاماتو^۱ مفهوم رمز ارز پایه عرصه وجود گذاشت به طوری که امروزه بیش از هزار رمز ارز مختلف در دنیا وجود دارد. بیت کوین برخلاف ارزهای دیجیتال قبل از خود توانست مشکل نیاز به فرد سوم قابل اعتماد را با کمک یک مکانیزم انقلابی با نام بلاک چین حل کند که در این فصل به بررسی این سیستم می پردازیم.

در حقیقت بلاک چین را می توان به مانند یک سیستم عامل در نظر گرفت و رمز ارزها را به عنوان برنامه های کامپیوتری ای که بر روی این سیستم عامل بزرگ و توزیع شده اجرا می شوند.

۳-۱ تعریف مفهومی بلاک چین

بلاک چین یک دفتر کل بزرگ و توزیع شده می باشد که کار ثبت و رهگیری دارایی ها را به صورت تراکنش^۲ محور انجام می دهد به طوری که امکان تغییر یا حذف اطلاعات درج شده در این دفتر کل وجود ندارد و فقط می توان اطلاعات جدیدی را به آن افزود. بلاک چین کار فرآیند ثبت تراکنش ها و ردگیری دارایی ها را در یک شبکه ی کسب و کار ساده میکند. یک دارایی می تواند ملموس مانند خانه، خودرو، پول نقد، زمین و یا ناملموس مانند مالکیت معنوی نظیر حق اختراع، حق چاپ یا نام اعتباری باشد. تقریباً هرچیز ارزشمندی می تواند در یک شبکه ی بلاک چین ردگیری و معامله شود و مخاطرات و هزینه ها را برای همه ی طرف های درگیر کاهش دهد.

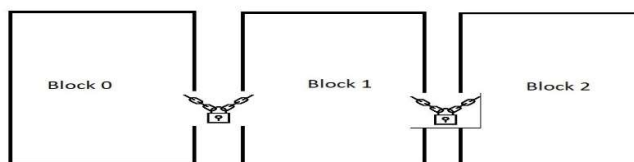
۳-۲ تعریف ساختاری بلاک چین

در ساده ترین حالت ساختار بلاک چین را همانطور که از نامش پیداست می توان به صورت زنجیره ای از بلاک های داده ای دانست که به طور متوالی پشت یکدیگر قرار می گیرند و هر بلاک به طوری به بلاک قبل

^۱ Satoshi Nakamoto

^۲ Transaction

از خود وابسته است.



تصویر ۳-۱: شمای کلی یک بلاک چین

۳-۳ ساختار هر بلاک

باتوجه به اینکه بلاک چین از زنجیره‌ای از بلاک‌ها ساخته شده است بنابر این ضروری به نظر می‌رسد که ابتدا به بررسی ساختار هر بلاک به صورت جز یک جز مستقل بپردازیم تا در ادامه بتوانیم کاربرد هر یک از این بخش‌ها را در یک بلاک چین بررسی کنیم.

هر بلاک به طور معمول حداقل دارای سه بخش زیر است:

۱. بخش داده ای
۲. نماد بلاک
۳. نماد بلاک قبل از خود

حال به بررسی هر کدام از این بخش‌ها می‌پردازیم:

۳-۳-۱ بخش داده‌ای بلاک

بخش داده ای یک بلاک در ساده ترین حالت ممکن می‌تواند یک رشته متنی حاوی یک پیام باشد و یا اطلاعات تراکنش های مالی مختلف به همراه اطلاعات دیگری نظیر یک مقدار متغییر، زمان ساخت بلاک، شماره بلاک و ... که در رمز ارزها استفاده می‌شود.

۲-۳-۳ نماد یا امضا بلاک

برای هر بلاک نماد یا امضا آن بلاک برابر است با خروجی تابع درهم ریزی یک طرف از همه بخش‌های به غیر از خود بخش نماد بلاک یا به عبارت دیگر مقدار هش بخش‌های مختلف بلاک.

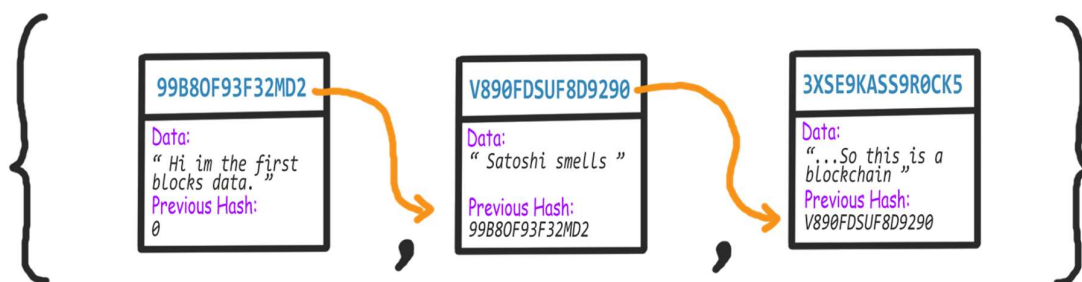
۳-۳-۳ نماد بلاک قبل از خود

هر بلاک شامل نماد بلاک قبلی خود نیز می‌شود. که در حقیقت این بخش باعث ایجاد زنجیره و مرتبط شدن بلاک‌ها به یکدیگر می‌شود.

۴-۳ بررسی ساختار بلاک چین

همانطور که گفته شد بلاک چین زنجیره‌ای از بلوک‌های داده‌ای به هم وابسته است اما این بلاک‌ها چگونه به هم وابسته می‌شوند؟

در حقیقت وظیفه ایجاد وابستگی بلاک‌های یک بلاک چین فارغ از بخش داده‌ای و بخش‌های دیگر آن بر عهده بخش نماد بلاک قبلی می‌باشد، به این صورت که با قرار دادن نماد بلاک قبلی در هر بلاک آن بلاک را به بلاک قبلی خود وابسته می‌کنیم و باتوجه به اینکه بخش نماد بلاک قبلی برای محاسبه نماد بلاک جدید استفاده می‌شود در حقیقت با ایجاد هر بلاک، این بلاک جدید به همه بلاک‌های قبل از خود وابسته می‌شود به طوری که برای ایجاد تغییر در یک بلاک نماد آن بلاک تغییر می‌کند در نتیجه نماد بلاک بعد از آن نیز تغییر می‌کند و این تغییر تا جدیدترین بلاک انتشار می‌یابد، حال اگر بلاک چین یک بلاک چین پویا باشد به طوری که پیوسته به بلاک‌های آن افزوده می‌شود عملاً ایجاد تغییر در یکی از بلاک‌های قبلی امکان پذیر نخواهد بود و یا حداقل بسیار بسیار سخت خواهد بود بنابر این همه افرادی که در یک بلاک چین مشارکت می‌کنند می‌توانند بر روی صحیح بودن اطلاعات بلاک چین توافق کنند و به این صورت توافق مورد نظر بدون نیاز به فرد سومی ایجاد می‌شود و در حقیقت همه افراد مشارکت کننده بر روی صحت داده‌های داخل بلاک چین توافق دارند.



تصویر ۳-۲: ساختار بلاک چین [19]

۳-۵ خواص بلاک چین

هر بلاک چین باید شامل خواص زیر باشد:

۱. تغییر ناپذیری: تغییر ناپذیری در بلاک چین باتوجه به ساختار آن به وجود می‌آید.
۲. اصل بودن: مشارکت کنندگان می‌دانند یک دارایی از کجا می‌آید و مالکیت آن در طول زمان چگونه تغییر کرده است.
۳. قطعیت: هر بلاک دارای قطعیت است در حقیقت داده‌های آن قطعی و صحیح و پذیرفته شده هستند.
۴. شفافیت: نحوه ساز و کار بلاک چین دارای شفافیت است و داده‌های موجود در آن برای همه افراد موجود در شبکه قابل دسترسی است.

۳-۶ بررسی کارکرد شبکه‌های بلاک چین

اغلب شبکه‌های مبتنی بر بلاک چین مانند رمز ارزها بر بستر شبکه‌های کامپیوتری نظیر به نظیر به فعالیت می‌پردازند به طوری که هر یک از نظیرهای موجود در شبکه که اینجا گره^۱ نامیده می‌شوند یک نسخه از بلاک چین را در سیستم خود نگه می‌دارند و قابلیت خواندن اطلاعات بلاک چین و افزودن بلاک جدید در شبکه را دارا می‌باشند و به ازای افزوده شدن هر بلاک جدید، گره‌ای که آن بلاک را به بلاکچین افزوده است موظف است که این موضوع را به همه نظیرهای دیگر اعلام کند تا آنها نیز اطلاعات بلاک چین خود را به روز

¹ Node

رسانی^۱ کنند تا شبکه دچار خطا نشود.

۳-۷ انواع بلاک چین

بلاک چین‌ها دارای انواع مختلفی می‌باشند اما به صورت کلی به دو دسته خصوصی و عمومی تقسیم می‌شوند که بلاک چین‌های عمومی اطلاعاتشان در اختیار همه افراد قرار دارد و هر فردی می‌تواند در هر زمان به عنوان یک گره جدید وارد این شبکه شود و یا اطلاعات آن‌ها را مورد بررسی قرار دهد نظیر بلاک چین‌های رمز ارزها، در مقابل نیز بلاک چین‌های خصوصی قرار دارند که یک سازمان و یا کسب و کار می‌تواند برای افراد موجود در شبکه خود آن‌را ایجاد کند و بر بستر شبکه‌های داخلی خود سازمان به اجرای آن بپردازد و یا با رمزگذاری همه داده‌های آن از دسترسی افراد غیر قابل اعتماد به داده‌ها جلوگیری به عمل آورد.

۳-۸ اثبات کار

اثبات کار یا POW در حقیقت یک مکانیزم در شبکه‌های بلاک‌چین است که ویژگی تغییر ناپذیری بلاک چین را تقویت می‌کند و همچنین از افزوده شدن بلاک‌های زیاد و غیر ضروری به سادگی در شبکه جلوگیری می‌کند، در حقیقت همان‌طور که از نامش پیداست POW یک مکانیزم است که باعث می‌شود هر فرد برای افزودن هر بلاک در شبکه موظف باشد میزان معینی کار انجام دهد.

سازکار این مکانیزم به این صورت است که برای هر بلاک چین مقدار مشخصی به نام سختی^۲ در نظر گرفته می‌شود و حال یک شرط جدید برای به دست آوردن نشان بلاک به سازوکار شبکه افزوده می‌شود به صورتی که مقدار عددی هش به دست آمده باید از مقدار سختی بیشتر باشد برای ساده شدن کار می‌توان این شرط را به صورت ظاهر شدن تعداد معینی 0 در ابتدای مقدار هش بلاک دانست.

همان‌طور که گفته شد هش بلاک شامل مقدار هش بخش مختلف بلاک می‌باشد اما این بخش اغلب از

^۱ Sync

^۲ Difficulty

داده‌های ثابتی تشکیل شده اند که قابل تغییر نمی‌باشد بنابر این به ازای داده‌های بلاک فقط و فقط یک مقدار هش تولید می‌شود که ممکن است شرط مورد نظر برای POW را نداشته باشد، برای حل این مشکل یک مقدار متغییر به نام nonce نیز به داده‌های بلاک افزوده می‌شود این مقدار متغییر فارغ از دیگر بخش‌های بلاک است و می‌تواند هر مقداری داشته باشد حال با کمک این مقدار جدید و تغییر دادن آن و محاسبه مجدد هش بلاک می‌توان هش‌های مختلف برای یک بلاک به دست آورد و درحقیقت مکانیزم POW مشارکت کنندگان را مجبور می‌کند تا مکرراً به محاسبه مقدار هش‌های مختلف برای یک بلاک با nonce های مختلف بپردازند تا یک مقدار هش با شرط بزرگتر بودن از سختی یافته شود؛ به محض یافته شدن این مقدار، گره‌ای که هش را پیدا کرده است این مقدار جدید را و مقدار nonce ای که به ازای آن مقدار هش را به دست آورده است به دیگر گره‌ها اعلام می‌کند و دیگر گره‌ها نیز حال به سادگی با یکبار هش کردن با nonce اعلام شده می‌توانند صحت ادعای گره اعلام کننده را راستی آزمایی کنند و درصورت درست بودن ادعا آن را به عنوان بلاک جدید بپذیرند.

۱-۸-۳ حمله ۵۱ درصد

با افزوده شدن POW به بلاک چین حال تغییر دادن داده‌های بلاک بیش از پیش سخت می‌شود به طوری که برای تغییر داده‌های یک بلاک این بار فرد مورد نظر باید به ازای همه بلاک‌های بعدی نیز الگوریتم POW را اجرا کند و مقدار هشی بیشتر از سختی برای همه بلاک‌ها بیابد، که بسیار وقت گیر خواهد بود همچنین در شبکه‌های پویا که همواره به بلاک‌های آن افزوده می‌شود گره خاطی باید بتواند به قدری سریع به محاسبه POW بلاک‌های قبلی بپردازد که بتواند به آخرین بلاک برسد و بلاک آخر را نیز خود بدست بیاورد و به شبکه اعلام کند تا همه آن‌را بپذیرند که این اتفاق فقط در حالتی ممکن است که گره مورد نظر حداقل ۵۱ درصد قدرت محاسباتی شبکه را داشته باشد یعنی قدرت محاسباتی آن از مجموع قدرت محاسباتی همه گره‌های دیگر موجود در شبکه بیشتر باشد.

۲-۸-۳ تغییرات سختی

از دیگر کاربردهای POW آن است که شبکه‌های بلاک چینی شلوغ می‌توانند زمان ایجاد شدن بلاک‌های خود را کنترل کنند تا به این صورت از رشد بی‌رویه بلاک چین جلوگیری به عمل آورند به این ترتیب که با افزوده شدن هر گره شانس یافته شدن بلاک جدید بیشتر می‌شود چرا که گره‌های بیشتری برای بدست آوردن بلاک جدید به رقابت می‌پردازند حال پس از افزوده شدن تعداد زیادی گره می‌توان با افزایش دادن مقدار سختی زمان مورد نیاز برای بدست آوردن بلاک جدید را بیشتر کرد و درحقیقت انجام مکانیزم POW را سخت‌تر نمود و زمان میانگین برای به دست آمدن تعداد معینی ای بلاک را کنترل نمود.



تصویر ۳-۳: نمودار تغییرات سختی شبکه بیت کوین [6]

۲-۹ استخراج^۱ رمز ارز

در شبکه‌های بلاک چینی رمز ارزهایی نظیر بیت کوین که تعداد زیادی گره برای بدست آوردن بلاک بعدی رقابت می‌کنند مقداری از توکن‌های شبکه به عنوان جایزه^۲ به هر فردی که موفق به یافتن بلاک بعدی شود اختصاص داده می‌شود تا به این صورت انگیزه برای پیاده کردن بلاک‌های جدید در بین افراد ایجاد شود چرا که هر فرد با پیدا کردن بلاک جدید و انجام عملیات POW در حقیقت به میزان امنیت شبکه می‌افزاید، به این ترتیب به تعداد کوین‌های شبکه افزوده می‌شود و در حقیقت مکانیزمی مانند استخراج کردن ایجاد

¹ Mining

² Block Reward

به طور مثال برای شبکه بیت کوین در ابتدا جایزه هر بلاک ۵۰ بیت کوین در نظر گرفته شده بود که به ازای ایجاد شدن هر ۲۱۰ هزار بلاک این میزان نصف می‌شود تا در نهایت به ۰ میل کند به طوری که پیش بینی می‌شود در نهایت تعدادی نزدیک به ۲۱ میلیون بیت کوین وجود خواهند داشت که این مقدار تا سال 2140 میلادی کاملاً استخراج می‌شود، لازم به ذکر است که با افزایش تعداد گره‌ها به مرور زمان سرعت استخراج بیت کوین‌ها افزایش نمی‌یابد چرا که همانطور که اشاره شد مقدار سختی مکرراً افزوده می‌شود به طوری که تقریباً هر ۲۱۰ هزار بلاک زمانی معادل چهار سال برای ایجاد شدن نیاز خواهند داشت.

علی رغم تمام مزایای اثبات کار این مکانیزم دارای مشکلاتی نیز می‌باشد یکی از بزرگترین ایرادات مکانیزم POW مصرف مقدار زیادی انرژی توسط کامپیوترهای گره‌هایی است که به اجرای آن می‌پردازند به طوری که گفته می‌شود به طور مثال شبکه بیت کوین در حال حاضر انرژی الکتریکی به اندازه انرژی مورد نیاز برای کل کشور ایرلند را مصرف می‌کند، درحالی که این انرژی برای امن کردن شبکه استفاده می‌شود اما تعداد زیادی هش تولید شده برای هر بلاک بی استفاده است و عملاً این حجم عظیم انرژی به هدر می‌رود، همچنین از دیگر اشکالات این مکانیزم آن است که برخی افراد با ایجاد مدارات مجتمع خاص و یا ASICs قادر به استخراج بیت کوین با سرعت زیاد می‌باشند به طوری که امروزه استخراج با CPU های معمولی عملاً امکان پذیر نمی‌باشد بنابر این تعدادی اثبات دیگر نیز برای رمز ارزها مطرح شده است که در اینجا به معرفی برخی از آنها می‌پردازیم.

در این روش یکی از گره‌های موجود در شبکه به طور تصادفی برای ایجاد بلاک جدید توسط شبکه

2 Proof Of Stake

انتخاب می‌شود اما با این شرط که افرادی که دارای سهم بیشتری از توکن‌های شبکه می‌باشند شانس بیشتری برای انتخاب شدن دارند، که داشتن تعداد زیادی از توکن‌های شبکه مانع از تقلب فرد مورد نظر می‌شود چرا که اگر این گره اقدام به تقلب کند ارزش توکن‌های شبکه که خود مالک تعداد زیادی از آنها است کاهش می‌یابد.

۳-۱۰-۲ اثبات حافظه^۱

این اثبات به مانند اثبات کار می‌باشد با این تفاوت که در آن از توابع در هم سازی حافظه سخت می‌باشد که فرد مورد نظر برای استخراج بلاک جدید به جای داشتن قدرت محاسباتی بیشتر باید سیستمی با حافظه بیشتر داشته باشد که باتوجه به سخت بودن ساخت حافظه و گرانتر بودن آنها شانس استفاده از مدارهای مجتمی محاسباتی در آنها کمتر است.

۳-۱۰-۳ گواهی سوزاندن^۲

این گواه نوعی گواهی است که در آن هر گره برای ایجاد کردن بلاک جدید باید تعدادی از توکن‌های خود را از بین ببرد بنابر این لزوم از بین بردن مانع از تقلب می‌شود.

همچنین برخی روش‌های دیگر نیز که گاه ترکیبی از روش‌های فوق است نیز استفاده می‌شود.

۳-۱۱ بلاک چین‌های نسل دوم^۳

همانطور که پیش‌تر اشاره شد یکی از بخش‌های رمز ارزها و برنامه‌های مبتنی بر بلاک چین یک ماشین مجازی است که به اجرای برخی دستورات شبه اسمبلی مختص آن رمز ارز و بلاک چین می‌پردازد. این زبان شبه اسمبلی در نسخه‌های اولیه رمز ارزها شامل دستورات بسیار محدودی می‌شد و به اصطلاح

^۱ Proof Of Capacity

^۲ Proof Of Burn

^۳ Block chain 2.0

دارای خاصیت کامل بودن تورینگ^۱ نبود، به این معنی که دارای دستورات پرش نبود اما با گذر زمان برخی از فعالان حوزه رمز ارز با بررسی پتانسیل‌های نهفته در این زبان، به استفاده از دستورات کامل تر و زبان‌های کامل تورینگ روی آوردند که به این ترتیب نسل دوم بلاک چین‌ها شروع شدند، یکی از نمونه‌های رمز ارزهای با بلاک چین نسل دوم اتریوم^۲ می‌باشد.

۱-۱۱-۳ قرار دادهای هوشمند^۳

قرار دادهای هوشمند برخلاف آنچه که ممکن است از نام ظاهری آن‌ها برداشت شود، قطعه کدهایی با زبان‌های کامل تورینگ ماشین‌های مجازی رمز ارزها هستند که با ارسال یک تراکنش به سمت آن‌ها قادر به انجام یک کار مشخص و ثبت نتیجه آن در بلاک چین می‌باشند. بنابر این، این قطعه کدها در محیط یک شبکه بلاک چین اجرا می‌شوند و همه مشارکت کنندگان شبکه می‌توانند با آن‌ها ارتباط برقرار کرده و انجام سرویس خاصی را مدنظر داشته باشند.

۱۲-۳ قاعده طولانی ترین زنجیره^۴

همانطور که گفته شد در بلاک چین‌های شلوغ مانند بلاک چین‌های رمز ارزها تعداد زیادی گره برای به دست آوردن بلاک بعدی با یکدیگر به رقابت می‌پردازند و هر گره‌ای که بتواند زودتر از بقیه گره‌ها پاسخ مسئله POW را بیابد به عنوان برنده برای استخراج آن بلاک معرفی می‌شود اما اگر دو گره همزمان موفق به این کار شوند چه می‌شود؟ در این حالت هر دو گره در یک زمان پاسخ درستی برای POW بدست آورده اند و هر دو می‌توانند بلاک خود را به انتهای بلاک چین بیافزایند که این اتفاق در عمل باعث دو شاخه شدن بلاک چین می‌شود، اما در نهایت باید یکی از این شاخه به عنوان شاخه صحیح پذیرفته شود و به حیات خود ادامه دهد. راه حل بیت کوین و دیگر رمز ارزها برای چنین حالتی آن است که ابتدا اجازه می‌دهند هر دو بلاک به انتهای بلاک چین افزوده شوند چرا که در مرحله اولیه هر دو بلاک برابر هستند اما با گذر زمان و

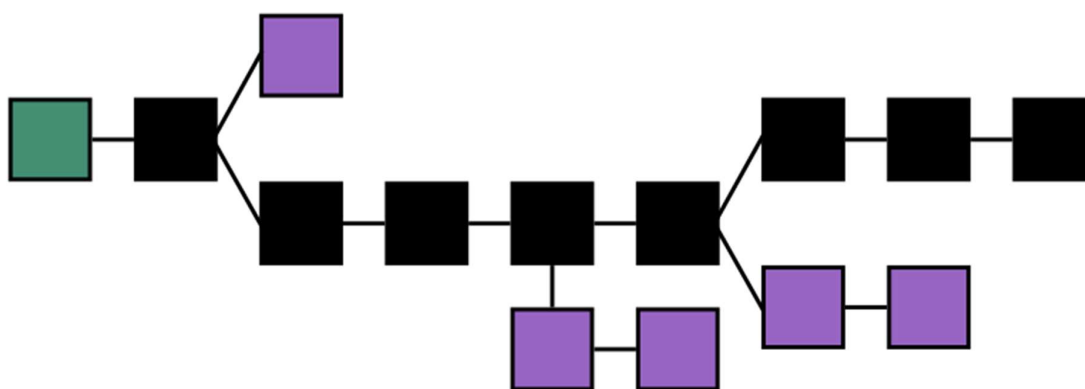
¹ Turing completeness

² Ethereum

³ Smart Contract

⁴ Longest Chain Is Valid

افزوده شدن بلاک‌های بعدی در انتهای این دو بلاک دو شاخه متفاوت از بلاک چین ایجاد می‌شود که در نهایت همه گره‌ها طولانی ترین شاخه ایجاد شده را به عنوان شاخه درست می‌پذیرند و کار کردن بر روی شاخه دیگر را رها می‌کنند.



تصویر ۳-۴: قاعده طولانی ترین شاخه [5]

همانطور که در تصویر ۳-۴ دیده می‌شود در هر مرحله پس از دو شاخه شدن بلاک چین، شبکه زنجیره سیاه که طولانی تر بوده است را به عنوان زنجیره اصلی انتخاب کرده و بلاک‌های بنفش رها شده اند، به این بلاک‌های رها شده در اصطلاح یتیم شده^۱ می‌گویند.

۳-۱۳ فورک^۲ در بلاک چین

شبکه‌های بلاک چین نیز مانند دیگر برنامه‌های کامپیوتری همواره دستخوش تغییرات و به روز رسانی-هایی می‌شوند اما با توجه به این که این شبکه ها کار حساس تری را نسبت به دیگر برنامه‌ها انجام می‌دهند و همچنین پایگاه داده آن‌ها که بلاک چین می‌باشد همواره در حال افزودن بلاک جدید است در برخی موارد ممکن است این تغییرات باعث ایجاد بلاک‌هایی با قواعد جدید شوند که با بلاک‌های قبل از آن تفاوت‌هایی دارد به چنین حالتی فورک در بلاک چین گفته می‌شود. در چنین حالتی گره‌هایی که نسخه نرم افزار خود

¹ Orphaned Blocks

² Fork

را به روز رسانی^۱ کرده اند بلاک‌هایی با خصوصیات متفاوتی از گره‌هایی که به روز رسانی را انجام نداده اند تولید می‌کنند.

۳-۱۳-۱ فورک نرم^۲

فورک نرم زمانی رخ می‌دهد که تغییرات ایجاد شده اساسی نباشند و بنابر این علی‌رغم این که دو نسخه مختلف از بلاک‌ها در حال ایجاد شد هستند اما این بلاک‌ها می‌توانند بدون رخ دادن مشکل خاصی پشت یکدیگر قرار گیرند و شبکه کلی دچار مشکل نمی‌شود بنابر این گره‌هایی که به روز رسانی کرده اند و گره‌هایی که به روز رسانی نکرده اند می‌توانند کار کردن بر روی یک بلاک چین را ادامه دهند.

۳-۱۳-۲ فورک سخت^۳

این نوع فورک زمانی رخ می‌دهد که تغییرات مذکور بسیار اساسی باشند به طوری که بلاک‌های تولید شده با هر نسخه نمی‌توانند کنار یکدیگر قرار گیرند و از نقطه ایجاد به روز رسانی عملاً بلاک چین به دو بلاک چین مجزا تبدیل می‌شود در چنین حالتی گره‌هایی که به روز رسانی را دریافت کرده اند روی یک بلاک چین و گره‌هایی که به روز رسانی را دریافت نکرده اند روی یک بلاک چین دیگر کار می‌کنند که به هیچ عنوان سازگار^۴ نمی‌باشند و عملاً در چنین حالتی ممکن است توکن‌های رمز ارز به دو توکن متفاوت تجزیه شوند، به طور مثال پس از یکی از به روز رسانی‌های شبکه بیت کوین در سال ۲۰۱۶ میلادی با ایجاد فورک سخت توکن‌های شبکه به دو نوع بیت کوین معمولی و بیت کوین کلاسیک^۵ تقسیم شدند به طوری که دارندگان کیف پول هریک نمی‌توانند توکن‌های نوع دیگر را داشته باشند.

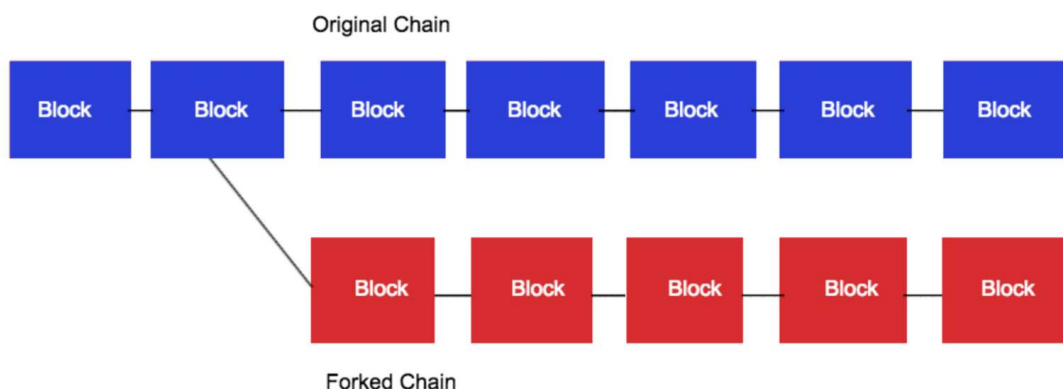
¹ Update

² Soft Fork

³ Hard Fork

⁴ Compatible

⁵ Bitcoin Classic



تصویر ۵-۳: نمونه فورک سخت در بلاک چین

همانطور که در تصویر ۳-۵ مشاهده می شود پس از یک فورک سخت بلاک چین به دو بلاک چین متفاوت تبدیل می شود و برخلاف حالت طولانی ترین زنجیره اینجا هر دو زنجیره قابل قبول هستند اما نه توسط همه گره ها بلکه برخی گره ها یک زنجیره و برخی گره ها زنجیره دیگر را قبول کرده و روی زنجیره مورد نظر خود مشغول به کار می شوند.

۱۴-۳ مشکلات بلاک چین ها

شبکه های بلاک چینی نیز علی رغم همه مزیت هایی که به ارمغان می آورند خالی از اشکال نیستند که در اینجا به برخی از اشکالات این شبکه ها اشاره مختصری می کنیم.

یکی از ایرادات این شبکه به دلیل مکانیزم POW و مصرف انرژی بیش از حد رخ می دهد که تلاش هایی برای حل این مشکل با سعی بر ایجاد کردن اثبات های متفاوت انجام شده است اما هیچکدام به کارآمدی اثبات کار یا POW نمی باشند.

در بسیاری از بلاک چین ها امکان پرداخت های با مبلغ بسیار کم وجود ندارد^۱ که این خود می تواند ناشی از عوامل مختلفی باشد، به طور مثال در شبکه بیت کوین برای هر تراکنش مبلغی نیز به عنوان هزینه

¹ Micro Payment

تراکنش^۱ دریافت می‌شود که اگر این مبلغ کمتر از حد معینی باشد استخراج کنندگان تمایلی به پردازش آن تراکنش نخواهند داشت حال اگر فردی بخواهد تراکنشی با مبلغی کمتر از هزینه آن انجام دهد عملاً غیر منطقی می‌باشد.

یکی از مهمترین مشکلات نیز مشکل مقیاس پذیری^۲ در شبکه‌های بلاک چین می‌باشد. که در این حالت باتوجه به نیاز انجام کار مشخصی و سپری شدن زمانی برای افزوده شدن هر بلاک و همچنین محدودیتی که در حجم اطلاعاتی‌ای که بلاک می‌تواند ذخیره کند وجود دارد باعث می‌شود عملاً در هر زمان بیش از تعداد معینی تراکنش قابل پردازش نباشند به طور مثال در شبکه بیت کوین تقریباً در هر ثانیه فقط هفت تراکنش قابلیت پردازش شدن را دارند که این باعث ایجاد محدودیت بسیاری در ساختار این شبکه و قابلیت های آن می‌شود. همچنین به دلیل محدودیت تعداد تراکنش‌ها افراد مجبور می‌شوند پیوسته هزینه پردازش تراکنش بیشتری پرداخت کنند تا استخراج کنندگان^۳ تمایل بیشتری به پردازش تراکنش آن‌ها داشته باشند.

۱-۱۴-۳ هش گراف^۴

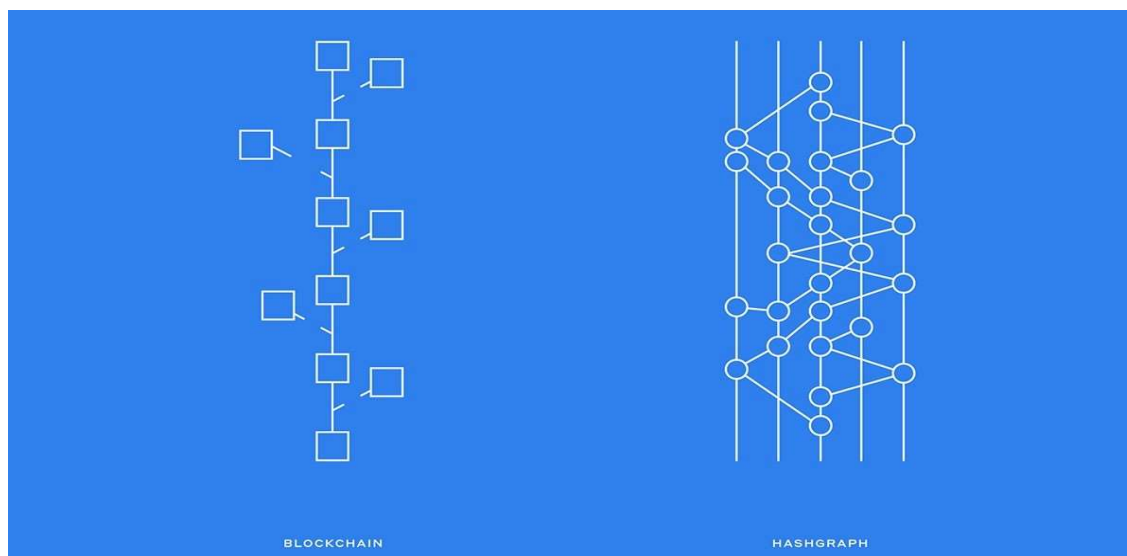
باتوجه به مشکل مقیاس پذیری بسیاری از فعالان حوزه بلاک چین تلاش‌هایی برای حل این مشکل کرده‌اند یکی از راه حل‌های پیشنهاد شده برای مشکل مقیاس پذیری ایجاد هش گراف‌ها به جای هش چین‌ها یا همان بلاک چین‌ها می‌باشد که در این حالت مشکل مقیاس پذیری تا حد زیادی حل می‌شود و تعداد هزاران تراکنش در ثانیه قابلیت پردازش شدن خواهند داشت همچنین امکان انجام پرداخت‌های با مقادیر بسیار کم نیز وجود خواهد داشت.

¹ Transaction Fee

² Scalability

³ Miners

⁴ HashGraph



تصویر ۳-۶ : مقایسه هش گراف با بلاکچین [15]

همانطور که در تصویر ۳-۶ مشاهده می شود در هش گراف به جای استفاده از قاعده طولانی ترین زنجیره سعی بر آن است تا بلاک های یتیم شده به صورتی مجدداً به شبکه بازگردانده شوند در این حالت به جای یک زنجیره یک گراف ایجاد می شود که گره های آن با هش به یکدیگر مرتبط می باشند.

۳-۱۵ بلاک چین های نسل سوم^۱

بلاک چین های نسل سوم نوع دیگری از بلاک چین های هستند که در پاسخ به مشکل مقیاس پذیری بلاک چین ایجاد شده اند به طوری که با حل این مشکل امکان پردازش تعداد نامحدود تراکنش با هر مبلغ را فراهم می آورند. همچنین کاربرد دیگر آنها در موارد مرتبط به اینترنت اشیا^۲ می باشد یکی از نمونه های موفق پیاده سازی بلاک چین های نسل سوم IOTA می باشد.

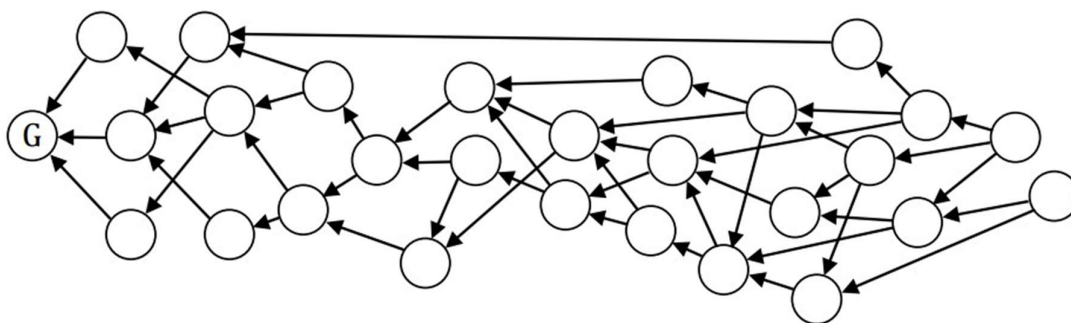
۳-۱۵-۱ نحوه کارکرد بلاک چین های نسل سوم

در این بلاک چین ها به جای ذخیره کردن تعدادی تراکنش در یک بلاک و سپس پشت هم قرار دادن بلاک ها پشت یکدیگر تلاش می شود تا تراکنش های بعدی به صورت مستقیم به تراکنش های قبل از خود

^۱ Block Chain 3.0

^۲ IOT

متصل شوند، به طور دقیق تر هر تراکنش به دو تراکنش تصادفی از تراکنش‌های قبل از خود متصل می‌شود و بنابر این یک گراف از تراکنش‌ها ایجاد می‌شود که این گراف در حقیقت یک گراف جهت دار رو به جلو در زمان خواهد بود چرا که هر تراکنش فقط به تراکنش‌های قبل از خود در زمان می‌تواند متصل شود بنابر این یال‌های متصل کننده این تراکنش‌ها در گراف جهت دار هستند و همچنین به دلیل جهت دار بودن امکان ایجاد دور^۱ نیز در این گراف موجود نمی‌باشد از این رو به آنها گراف‌های جهت دار بدون دور^۲ و یا به اختصار DAG گفته می‌شود.



تصویر ۳-۷: نمونه یک DAG

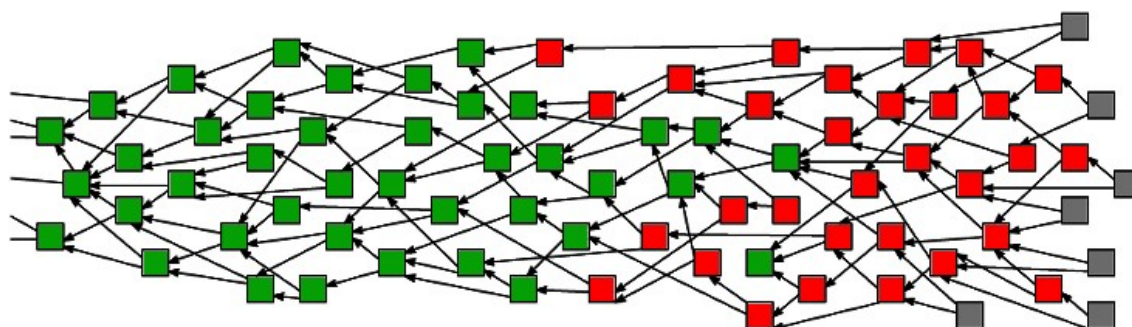
۳-۱۵-۲ پردازش تراکنش‌ها در بلاک چین‌های نسل سوم

در این نوع بلاک چین‌ها در حقیقت هر فرد برای ارسال تراکنش خود به جای پرداخت هزینه تراکنش به استخراج کنندگان تا آن‌ها تراکنش فرد را تایید و پردازش و به شبکه اعلام کنند خود اقدام به اعلام کردن تراکنش خود می‌کند با این تفاوت که برای افزودن تراکنش خود به شبکه موظف است دو تراکنش قبل از خود را تایید کند و پس از تایید صحت آن دو تراکنش، تراکنش خود را به انتهای آن‌ها می‌افزاید به این صورت ساختار DAG گفته شده از اتصال هر تراکنش به دو تراکنش قبل از خود به وجود می‌آید به این ترتیب و به دلیل عدم وجود هزینه تراکنش تراکنش‌های با هر مقدار ناچیز قابلیت افزوده شدن به شبکه را دارا می‌باشند. همچنین با افزایش تعداد تراکنش‌ها مشکل ایجاد ترافیک برای تراکنش‌ها رخ نمی‌دهد چرا

¹ Cycle

² Directed Acyclic Graph

که افزایش تعداد تراکنش‌ها به معنی افزایش تعداد گره‌های فعال در شبکه است که خود موظف به تایید تراکنش‌های قبل از خود می‌باشند، بنابر این همواره با افزایش تعداد تراکنش‌ها تعداد تایید کنندگان تراکنش نیز افزایش می‌یابد و شبکه دچار کندی نمی‌شود.



تصویر ۸-۳ : وضعیت تراکنش‌ها در یک $DAG^{[14]}$

همانطور که در تصویر ۸-۳ نشان داده شده است تراکنش‌ها در این نوع شبکه‌ها به سه نوع مختلف تقسیم می‌شوند تراکنش‌های کاملاً تایید شده تراکنش‌های در دست تایید و تراکنش‌های جدید که به ترتیب با رنگ‌های سبز قرمز و خاکستری در تصویر قابل مشاهده می‌باشند.

تراکنش‌های تایید شده تراکنش‌هایی اند که از هر تراکنش جدید شبکه پس از طی چند مرحله در گراف می‌توان به آن‌ها رسید، در حقیقت این بدان معنی است که این تراکنش‌ها سابقه کاملی از تراکنش‌های قبلی را در خود دارند، تراکنش‌های تایید نشده که به رنگ قرمز هستند فقط به تعدادی از تراکنش‌های جدید متصل اند و تراکنش‌های جدید تراکنش‌هایی هستند که در انتظار افزوده شدن تراکنش جدید به انتهای خود و پردازش شدن توسط یک گره می‌باشند.

۱۶-۳ بلاک چین‌های نسل چهارم^۱

این نوع بلاک چین‌ها که بلاک چین‌های هوشمند نیز معروف هستند در حقیقت تلاشی برای ایجاد

¹ Block Chain 4.0

اتصال میان حوزه های هوش مصنوعی^۱ و به خصوص یادگیری عمیق^۲ با بلاک چین می باشد از پروژه های معروف بلاک چین های نسل چهارمی می توان به SingularityNet و Deep Brain Chain اشاره کرد.

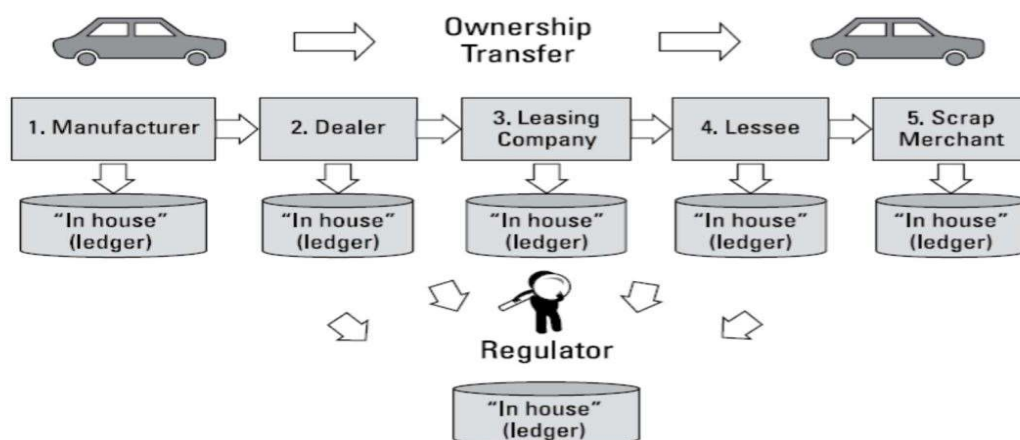
۱۷-۳ بلاک چین در خارج از رمز ارزها

همانطور که پیشتر گفته شد بلاک چین فقط مختص به رمز ارزها نمی شود و قابلیت های این تکنولوژی انقلابی می تواند در بسیاری از صنایع دیگر نیز مورد استفاده قرار گیرد به طور مثال در اینجا به بررسی کاربرد بلاک چین در یک شبکه فروش اقساطی خودرو می پردازیم.

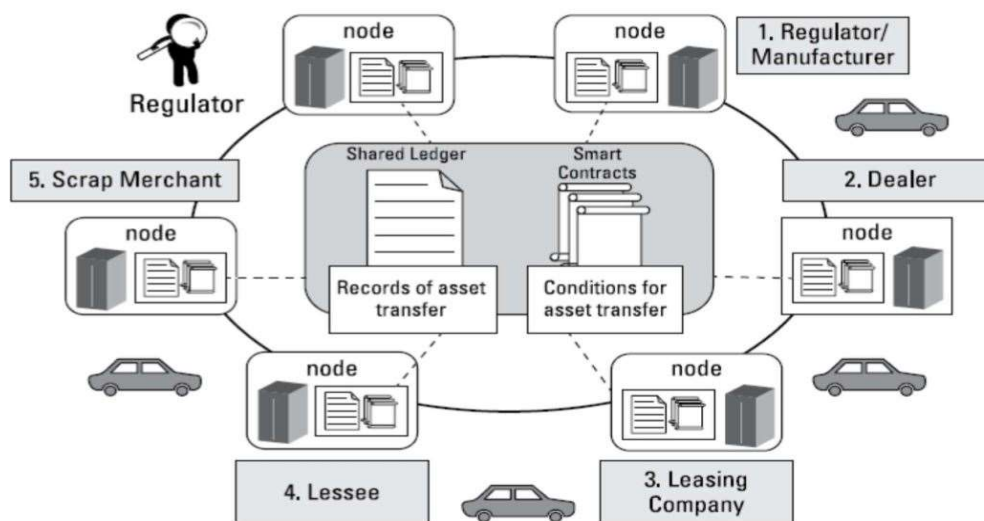
کارخانه های سازنده ی خودرو خرید قسطی خودرو را ساده جلوه میدهند، اما این کار در واقعیت میتواند کاملاً پیچیده باشد. چالش مهمی که این روزها شبکه های خرید قسطی خودرو با آن روبرو هستند این است که هرچند زنجیه تامین فیزیکی معمولاً یک پارچه است، اما هر مشارکت کننده در شبکه و سامانه های پشتیبان از هم جدا هستند که هر کدام از این ها باید دفتر کل مخصوص به خود را نگه داری کنند همچنین از طرفی نیز مراجع حقوقی که وظیفه ثبت مالکیت و سند را بر عهده دارند دفتر کل خود را دارند و پلیس نیز باید دفتر کل خود را برای شماره گذاری خودروها داشته باشد و از طرفی بانک ها نیز دفتر کل خود را برای پرداخت اقساط دارند با وجود این همه سازمان مختلف و دفتر کل های متفاوت هماهنگی میان این مراجع بسیار سخت صورت می گیرد و گاهی بسیار دردسر ساز است اما در صورت استفاده از بلاک چین بسیاری از مشکلات حل می شوند. به طوری که هریک از مشارکت کنندگان می توانند به راحتی بلاک چین را رسد کرده و اطلاعات مورد نظر خود را از آن برداشته و تراکنش های مورد نظر خود را بی افزایند.

^۱ artificial intelligence

^۲ Deep Learning



تصویر ۳-۹: رهگیری مالکیت خودرو بدون بلاک چین^[3]



تصویر ۳-۱۰: رهگیری مالکیت خودرو با بلاک چین^[3]

۴ فصل چهارم: بررسی کارکرد چند رمز ارز مهم

باتوجه به این که شبکه‌های بلاک چین اغلب برای رمز ارزها استفاده می‌شوند برای درک کامل مدل کارکرد آن‌ها لازم به نظر می‌رسد تا ابتدا به بررسی کارکرد چند رمز ارز مهم دنیای فناوری بپردازیم.

۴-۱ بیت کوین

بررسی تخصصی تر بیت کوین به عنوان اولین رمز ارز دنیا و همچنین به دلیل مشترک بودن الگوریتم-های استفاده شده در آن در بسیاری از دیگر رمز ارزها به عنوان مبنا هر رمز ارزی به نظر ضروری می‌رسد.

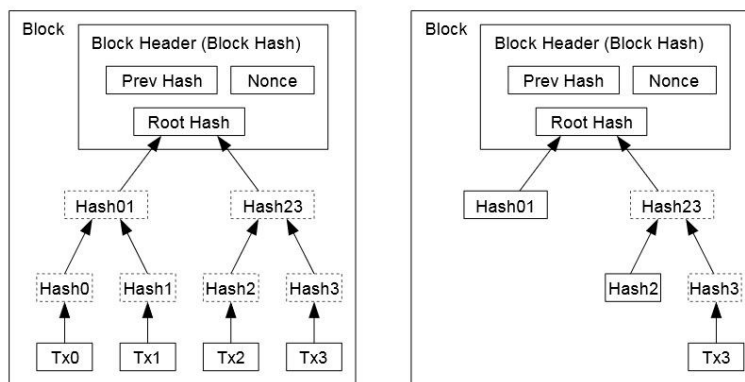
بیت کوین برای اولین بار در سال ۲۰۰۹ میلادی توسط شخص یا اشخاص ناشناسی با نام ساتوشی ناکاماتو معرفی شد، معرفی این رمز ارز به صورت چند برگ مقاله با نام وایت پیپر^۱ بود که در این مقاله به تشریح سازوکار این رمز ارز پرداخته شده است.

۴-۱-۱ ساختار بلاک چین بیت کوین

بلاک چین بیت کوین ساختاری مانند بلاک چین‌های فصل قبل دارد در اصل در اینجا باید به بررسی دقیق تر بخش داده‌ای هر بلاک بپردازیم. بخش داده‌ای هر بلاک بیت کوین حاوی اطلاعات تعدادی از تراکنش‌های انتقال این رمز ارز می‌باشد به طوری که حجم هر بلاک به ۱ مگابایت^۲ محدود می‌باشد بنابراین این تعداد تراکنش‌های موجود در هر بلاک محدود می‌باشد. همچنین برای سرعت دادن به محاسبات POW به جای استفاده از کل اطلاعات تراکنش‌ها در محاسبه هش بلاک فقط از ریشه درخت مرکله تراکنش‌های بلاک استفاده می‌شود.

^۱White Paper

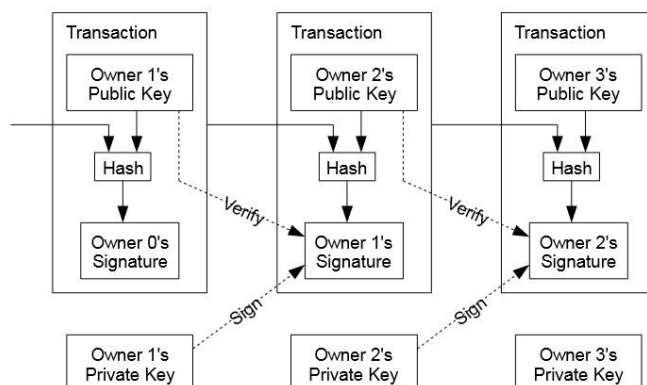
^۲ Mega byte

تصویر ۴-۱: ساختار بلاک‌های بیت کوین^[۴]

۴-۱-۲ تراکنش‌ها در بیت کوین

یکی از مهمترین اجزای تشکیل دهنده شبکه بیت کوین تراکنش‌ها هستند، در حقیقت خورد ترین بخش سازنده شبکه بیت کوین این تراکنش‌های می‌باشند که به کمک آن‌ها نقل و انتقالات بیت کوین در شبکه انجام می‌شود.

ساختار هر تراکنش در شبکه بیت کوین شامل اطلاعات کلید عمومی دریافت کننده تراکنش، مقدار تراکنش و امضای دیجیتال صادر کننده می‌باشد که این امضا دیجیتال در حقیقت به صورت یک قطعه کد برای ماشین مجازی بیت کوین به اسم Script Sig می‌باشد.

تصویر ۴-۲: ساختار تراکنش‌های بیت کوین^[۴]

۳-۱-۴ ساختار UTXO^۱ در بیت کوین

در بیت کوین برای رهگیری مالکیت توکن‌های شبکه در طول زمان دارایی هر فرد به صورت یک عدد که با هر بار ارسال توکن مقداری از آن کم می‌شود و با هر بار دریافت به آن افزوده می‌شود در نظر گرفته نمی‌شود، بلکه در این رمز ارز به محض دریافت یک تراکنش توسط یک نفر خروجی‌های آن تراکنش به عنوان UTXO در کیف پول فرد ذخیره می‌شوند و هنگام ارسال بیت کوین فرد مورد نظر باید تعداد مشخصی از این UTXO ها را به حساب فرد مورد نظر ارسال کند.

۴-۱-۴ کیف پول بیت کوین

شبکه بیت کوین دارای تعداد مشخصی کیف پول می‌باشد که در حقیقت این کیف پول‌ها وظیفه حفظ و نگه داری دارایی‌های هر فرد را دارند، در حقیقت در عمق هر کدام از این کیف پول‌ها کلید خصوصی هر فرد ذخیره می‌شود که با کمک آن و امضا می‌تواند UTXO هایی که به حساب کلید عمومی آن کلید خصوصی ارسال شده است را خرج کند، مهمترین کیف پول بیت کوین BitCoin Core می‌باشد که کیف پول رسمی این رمز ارز می‌باشد، این کیف پول در حقیقت نمونه کامل از پیاده سازی پروتوکول^۲ بیت کوین می‌باشد این کیف پول برای کار کردن ابتدا لازم است تا نسخه کاملی از بلاک چین بیت کوین را که در حال حاضر تقریباً حجمی معادل 185GB دارد را دریافت نماید، یکی از قابلیت‌های این کیف پول امکان پیوستن به شبکه استخراج کنندگان توسط آن می‌باشد.

۲-۴ اتریوم

این رمز ارز نمونه موفق از پیاده سازی بلاک چین‌های نسل دوم می‌باشد که با کمک ماشین مجازی خود که EVM نام دارد، قادر است یک زبان کامل تورینگ را اجرا کند که به کمک کدهای این ماشین شبکه اتریوم امکان اجرای کدهایی به پیچیدگی کدهای همه دیگر زبان‌های برنامه نویسی را به صورت توزیع شده

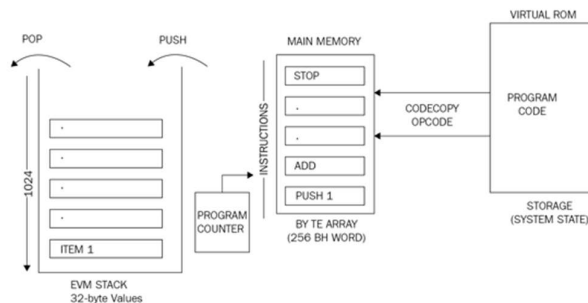
^۱ Unspent Transaction Output

^۲ Protocol

روی همه کامپیوترهای اجرا کننده شبکه اتریوم می‌دهد.

۴-۲-۱ بررسی ساختار EVM

ماشین مجازی اتریوم و یا EVM در حقیقت یک ماشین مجازش مبتنی بر پشته^۱ می‌باشد به این صورت که همه دستورات آن فقط مقداری را از پشته خوانده و مقداری را به پشته می‌افزایند، اندازه این پشته به ازای هر داده برابر با 256 بیت^۲ می‌باشد که در حقیقت برابر با خروجی تابع درهم ساز SHA3 (که به نام Keccak 256) نیز شناخته می‌شود می‌باشد همچنین در این ماشین مجازی یک حافظه دیگر نیز برای ذخیره سازی کدها وجود دارد.



تصویر ۴-۳: ساختار EVM^[۱]

۴-۲-۲ زبان Solidity

همانند دیگر زبان‌های اسمبلی برای زبان اسمبلی EVM نیز چندین زبان سطح بالاتر ایجاد شده است که کدهای این زبان‌ها به کدهای اسمبلی EVM ترجمه^۳ می‌شود. یکی از زبان‌های سطح بالاتر شبکه اتریوم زبان Solidity نام دارد که با کمک این زبان به راحتی می‌توان به نوشتن قرار دادهای هوشمند پرداخت.

۴-۲-۳ برنامه‌های توزیع شده^۴

به برنامه‌های نوشته شده به زبان سالیدیتی در اصطلاح برنامه‌های توزیع شده گفته می‌شود چرا که

¹ Stack

² Bit

³ Compile

⁴ Dapps

این برنامه‌ها درحقیقت نه بر روی یک کامپیوتر بلکه بر روی همه کامپیوترهایی که در شبکه اتریوم فعالیت می‌کنند اجرا می‌شوند و هر گره در شبکه می‌تواند توابع موجود در این برنامه‌ها را فراخوانی و استفاده نماید. البته باید توجه داشت که مفهوم برنامه‌های توزیع شده مختص شبکه اتریوم نمی‌باشند و این مفهوم از مدت‌ها قبل نیز وجود داشته است اما یکی از بزرگترین مشکلات این برنامه‌ها بحث توافق می‌باشد که با کمک شبکه بلاک چین این مشکل حل می‌شود.

۴-۲-۴ توکن‌های ERC20

زبان برنامه نویسی Solidity به قدری قوی می‌باشد که توسط آن افراد مختلف می‌توانند توکن‌های فرعی دیگر مورد نظر خود را ایجاد و برنامه نویسی کنند و به کمک بلاک چین شبکه اتریوم و کیف پول‌هایش از آنها استفاده نمایند به چنین توکن‌هایی ERC20 گفته می‌شود.

۴-۲-۵ ساختار کیف پول‌های اتریوم

در ساختار شبکه اتریوم به جای مکانیزم UTXO استفاده شده در بیت کوین از مکانیزم ماشین حالت^۱ استفاده می‌شود به این ترتیب که هر کیف پول در این شبکه معادل یک حساب کاربری^۲ در نظر گرفته می‌شود که به یک کلید خصوصی متصل می‌باشد و مقدار موجودی نیز به صورت یک عدد به آن‌ها اختصاص داده می‌شود حال به ریشه درخت مرکله ساخته شده از اطلاعات همه این حساب‌های کاربری حالت سیستم گفته می‌شود که با پردازش هر تراکنش این سیستم از یک حالت به حالت دیگری می‌رود.

۴-۲-۶ شبکه Ropsten

باتوجه به امکان برنامه نویسی برای شبکه اتریوم و نیاز این برنامه‌ها به اطمینان از صحت کارکرد آن‌ها علاوه بر شبکه اصلی اتریوم که توکن‌های آن دارای ارزش می‌باشند، شبکه‌های دیگری نیز برای آن ایجاد شده

^۱ State Machine

^۲ Account

است که در حقیقت این شبکه‌ها برای تست و خطایابی قرار داده‌ای هوشمند می‌باشند، ساختار این شبکه‌ها مانند شبکه اصلی اتریوم می‌باشد با این تفاوت که توکن‌های آن‌ها ارزش ندارند یکی از معروف ترین این شبکه‌ها Ropsten نام دارد.

۴-۲-۷ Web3.js

زبان Solidity علی‌رغم داشتن توانایی برای نوشتن هر نوع قرار داد هوشمند و برنامه‌ای قابلیت ایجاد رابط کاربری گرافیکی^۱ را دارا نمی‌باشد در نتیجه کتابخانه‌های زیادی برای زبان‌های مختلف برنامه نویسی برای کار با آن ایجاد شده است، یکی از معروف ترین این کتابخانه‌ها که به زبان جاوا اسکریپت نوشته شده است Web3.js نام دارد که با کمک زبان جاوا اسکریپت و زبان‌های طراحی Html و Css امکان ایجاد رابط‌های کاربری تحت وب برای قرار دادهای هوشمند را فراهم می‌آورد.

۴-۳ آیوتا

ایوتا یک نمونه از پیاده سازی بلاک چین‌های نسل سومی است که در سال ۲۰۱۶ میلادی پا به عرصه وجود نهاد این رمز ارز در حقیقت به منظور مناسب بودن برای اینترنت اشیا ایجاد شده است و از پرداخت‌های ریز به خوبی پشتیبانی می‌کند در این شبکه به DAG ایجاد شده از تراکنش‌های شبکه Tangle گفته می‌شود.

۴-۳-۱ توکن‌های IOTA

باتوجه به اینکه در IOTA ساختار استخراج وجود ندارد بنابر این همه توکن‌های شبکه IOTA در ابتدای ایجاد شدن آن به صورت آماده وجود داشتند و توسط یک عرضه عمومی در میان افراد مختلف پخش شده‌اند.

۴-۳-۲ نظیر یابی در IOTA

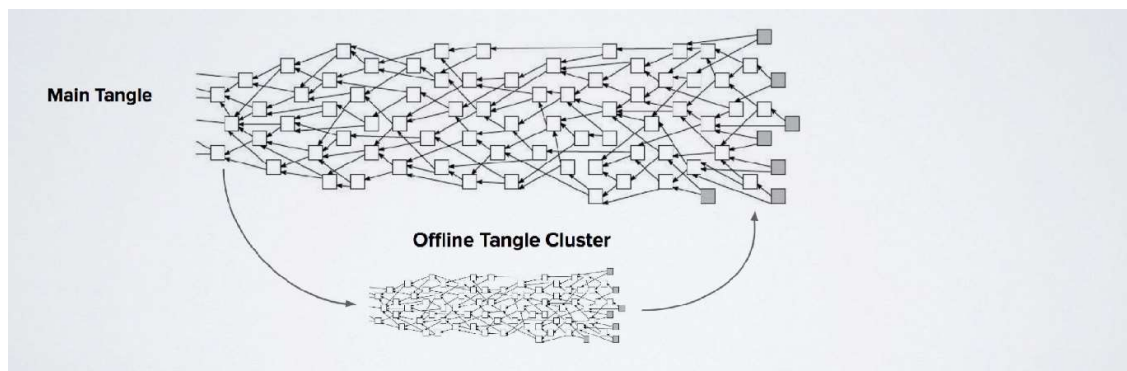
در شبکه IOTA عملیات نظیر یابی برخلاف بیت کوین و اتریوم نه به صورت خودکار بلکه به صورت

¹ GUI

دستی انجام می‌شود در حقیقت در این شبکه افراد برای متصل شدن به نظیرها و اتصال به شبکه باید به سایت IOTA مراجعه کرده و از پایگاه داده موجود در سایت آدرس نظیرهای مورد نظر خود را برداشته و برای اتصال به آن‌ها وارد نمایند، دلیل این ساختار آن است که باتوجه به اینکه در بلاک چین های نسل سوم تراکنش‌ها به تراکنش‌های دیگر متصل می‌شوند هر فرد باید به همسایه‌های مورد اعتماد متصل شود، همچنین باتوجه به ساختار بلاک چین‌های نسل سومی برخلاف دیگر بلاک چین‌ها که قدرت محاسباتی بیشتر به معنی کنترل بیشتر بر شبکه بود تعداد همسایه بیشتر به معنای کنترل بیشتر بر این شبکه است بنابر این با وجود این ساختار نظیر یابی امکان متصل شدن به تعداد زیادی از نظیرها از بین می‌رود.

۳-۳-۴ امکان جدا شدن و پیوستن تعدادی از گره‌ها در IOTA

یکی از قابلیت‌های منحصر به فرد IOTA امکان جدا شدن تعدادی از گره‌ها از شبکه اصلی و مجدداً پیوستن آن‌ها به شبکه می‌باشد در این حالت در مقطعی از زمان تعدادی از گره‌ها از شبکه جدا می‌شوند و به کار کردن بر روی یک Tangle کوچک تر در شعاع کاری خود می‌پردازند در نهایت به راحتی می‌توانند با متصل کردن تراکنش‌های شبکه اصلی به تراکنش‌های خود مجدداً به شبکه بپیوندند.



تصویر ۴-۴ : جدا شدن و اتصال مجدد

۵ فصل پنجم : پاریسی کوین

پارسی کوین^۱ در حقیقت نمونه یک پیاده سازی از بلاک چین های نسل سوم با زبان C# و با تکنولوژی Net Standard. می باشد با علامت اختصاری PIC برای توکن ها. که برای درهم سازی از تابع درهم ساز SHA-512 در آن استفاده شده است. که در ادامه به توضیح بخش های اساسی این سامانه می پردازیم.

۵-۱ حالت سیستم

در این رمز ارز نیز به مانند رمز ارز اتریوم از ریشه درخت مرکله تمام حساب های کاربری سامانه حالت سیستم به دست می آید که با افزوده شدن هر تراکنش جدید سیستم از حالت $n-1$ به حالت n می رود. که این امر مستلزم به روز رسانی درخت مرکله می باشد تا ریشه جدید آن بدست آید، نکته قابل توجه آن است که به دلیل خاصیت درخت های دو دویی هر بار به روز رسانی این ریشه به تعداد لگاریتم حساب های کاربری نیاز به انجام عملیات دارد که به مراتب عدد کوچکی می باشد به طور مثال در صورت وجود یک میلیون حساب کاربری در سامانه به تعداد تقریبی بیست عملیات برای به روز رسانی حالت سیستم نیاز خواهد بود.

۵-۲ حساب کاربری

هر حساب کاربری در پارسی کوین شامل موارد زیر می باشد.

۱. کلید خصوصی دارنده حساب

۲. کلید عمومی دارنده حساب

۳. دارایی حساب

۴. مقدار هش حساب

¹ ParsiCoin

به طوری که مقدار هش حساب از هش کردن موارد دیگر به دست می‌آید و این مقدار هش به عنوان برگ‌های درخت مرکله ای که ریشه آن حالت سیستم است استفاده می‌شود. حساب‌ها در حقیقت وظیفه ایجاد تراکنش‌ها و امضای آن‌ها را نیز برعهده دارند.

```
public Transaction TransactionBuilder(string recipient, double value, string message = "")
{
    if (value > Balance) throw new Exception("Not enough funds.");
    var t = new Transaction(recipient, value, signatureProvider, message);
    if (t.ISSignatureVerified()) return t;
    throw new Exception("Something went wrong, cannot sign the transaction");
}
```

قطعه کد ۵-۱: تابع سازنده تراکنش در حساب

۵-۳ ساختار تراکنش‌های در پارسی کوین

ساختار تراکنش‌های در پارسی کوین شامل موارد زیر می‌باشد:

۱. کلید عمومی فرستنده/ایجاد کننده تراکنش
۲. کلید عمومی دریافت کننده تراکنش
۳. میزان تراکنش
۴. زمان ایجاد تراکنش
۵. زمان ثبت تراکنش در شبکه
۶. مقدار هش تراکنش
۷. پیام تراکنش به صورت اختیاری
۸. آدرس تراکنش در DAG
۹. امضا دیجیتال ایجاد کننده تراکنش
۱۰. قطعه کدهای لازم برای تایید امضا دیجیتال تراکنش

برای ایجاد هش تراکنش کلیدهای عمومی فرستنده و دریافت کننده، میزان تراکنش و زمان ایجاد تراکنش استفاده می‌شوند.

```
public string ComputeObjectHash()
=> $"{TransactionIssuer}-{Recieipient}-{Amount}-{IssueTime}".ComputeHashString();
```

قطعه کد ۵-۲: محاسبه هش تراکنش

در این سیستم تراکنش‌ها پس از ایجاد و امضا شدن و قرار گرفتن در انتهای دو تراکنش قبلی به گره‌های همسایه ارسال می‌شوند تا توسط یک الگوریتم به مانند تئوری سخن چینی^۱ در میان تمام گره‌های شبکه پخش شوند. همچنین هر گره پس از دریافت تراکنش ابتدا موظف به تایید هویت امضا آن است که توسط ماشین مجازی این سیستم انجام خواهد شد.

```
public Transaction(string recieipient, double amount, ECDSA ec, string message = "")
{
    TransactionIssuer = ec.ExportPubKey;
    Recieipient = recieipient;
    Amount = amount;
    IssueTime = DateTime.UtcNow;
    TxHash = ComputeObjectHash();
    Signiture = ec.Sign(TxHash);
    ScriptPubKey = $"{Signiture};{TransactionIssuer}";
    ScriptSig = $"{ScriptPubKey};CheckSig;IsOne";
}
```

قطعه کد ۵-۳: ایجاد تراکنش

۵-۴ ساختار گره‌های DAG در پارسی کوین

همانطور که گفته شد DAG در حقیقت یک گراف می‌باشد بنابراین، این گراف نیز مانند دیگر گراف‌ها دارای گره‌هایی می‌باشد، این گره‌ها به منظور نگه داری تراکنش‌ها ایجاد شده اند اما می‌توانند حاوی اطلاعات

¹ Gossip

دیگری نیز باشند. در ساختار پارسی کوین این امکان دیده شده است که گره‌ها بتوانند حاوی هیچ تراکنشی نبوده و فقط دارای یک پیام خاص باشند. ساختار گره‌ها شامل موارد زیر است:

۱. نشانه گره
۲. زمان ایجاد شدن گره
۳. زمان تایید شدن گره
۴. گره قبلی سمت راست
۵. گره قبلی سمت چپ
۶. هش گره قبلی سمت راست
۷. هش گره قبلی سمت چپ
۸. پیام
۹. تراکنش
۱۰. هش تراکنش
۱۱. کلید عمومی ایجاد کننده گره
۱۲. هش گره
۱۳. حالت سیستم قبل از ایجاد این گره
۱۴. حالت سیستم بعد از ایجاد این گره
۱۵. مقدار none برای مسئله POW گره
۱۶. تعداد تاییدهای گره

که مقدار هش گره در این سیستم برابر با هش شده نشانه گره به همراه زمان ایجاد آن و اطلاعات گره‌های قبلی و همچنین سازنده و تراکنش موجود در گره می‌باشد. همچنین حل یک مسئله POW نیز برای تایید و ایجاد هر گره در نظر گرفته شده است که این مسئله POW مقدار سختی بسیار کمتر از بلاک چین‌های

نسل اول و دومی دارد به طوری که به طور معمول با یک کامپیوتر معمولی می توان در زمان تقریبی حدود

```
public string Mine()
{
    byte[] s = null;
    do
    {
        s = ComputeObjectHash().ToArray(StringEncoding.Base85Check);
    } while (NodeHash.ToArray().CompareDiff());
    return s.ToBase58Check();
}
```

قطعه کد ۴-۵: تابع مسئله POW برای هر ند

۳۰ ثانیه پاسخ آنرا به دست آورد.

۵-۵ کیف پول ها در پارسی کوین

هر کیف پول در پارسی کوین شامل تعداد نا محدودی حساب کاربری می باشد که یکی از آن ها به عنوان حساب کاربری اصلی در هر زمان وظیفه ایجاد و امضا تراکنش ها را برعهده خواهد داشت.

۵-۶ نویسه بندی ها در پارسی کوین

در این سامانه از چهار نوع نویسه بندی برای انتقال متن ها استفاده شده است:

۱. نویسه بندی ASCII برای متن های انگلیسی

۲. نویسه بندی UTF-8 برای دیگر متن ها

۳. نویسه بندی Base64 برای انتقال متن های بی معنی در دیگر نویسه بندی ها در اجزا داخلی سامانه

۴. نویسه بندی Base58Check برای انتقال متن های بی معنی در دیگر نویسه بندی ها برای مواردی که لازم است توسط افراد خوانده شود مانند کلیدهای عمومی

تفاوت نویسه بندی Base58Check با نویسه بندی Base58 در قرار داشتن یک CheckSum در

آن است.

۵-۷ انواع توابع درهم سازی در پارسی کوین

در این سامانه از سه تابع درهم سازی MD5 برای موارد معمولی SHA256 و SHA512 برای محاسبه POW استفاده شده است البته حالت اصلی دو بار محاسبه خروجی این توابع درهم سازی برای امنیت بیشتر می باشد.

۵-۸ رمز نگاری در پارسی کوین

در این رمز ارز از سه سیستم رمزنگاری AES و RSA و ECC استفاده شده است که از ترکیب AES و RSA برای نقل و انتقال امن پیام ها استفاده می شود و از ECC نیز برای ساخت امضا دیجیتال برای ایجاد تراکنش ها. همچنین اطلاعات خصوصی نظیر کلید عمومی هر فرد که بر روی سیستم فرد ذخیره می شود بر روی یک فایل که با AES رمزنگاری شده است نوشته می شوند تا در صورت به خطر افتادن رایانه فرد این اطلاعات قابل بازگشایی نباشند.

۵-۹ اجزای سامانه

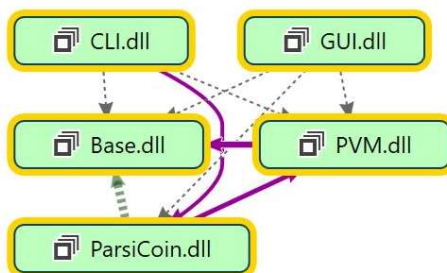
در این سامانه برای کنترل بهتر کد برنامه به قسمت ها و کامپوننت هایی به شرح زیر تقسیم شده است.

۱. کامپوننت Base که کامپوننت پایه برنامه می باشد و شامل یک سری توابع اساسی که در همه بخش های دیگر مورد استفاده قرار می گیرد می باشد.
۲. کامپوننت اصلی ParsiCoin که شامل موارد کاری سامانه می باشد.
۳. کامپوننت PVM که شامل ماشین مجازی سیستم می باشد.

۴. کامپوننت CLI که شامل یک کامند لاین^۱ برای کار با سامانه است

۵. کامپوننت GUI که شامل یک رابط کاربری گرافیکی^۲ وب بیس برای سامانه می باشد.

ارتباط اجزا برنامه بایکدیگر به صورت زیر می باشد. که در ادامه به شرح کارکرد هریک خواهیم پرداخت.



تصویر ۵-۱: ارتباط بین اجزا در پارسی کوین

۵-۹-۱ کامپوننت Base

این کامپوننت در اصل دربرگیرنده کلاس ها و توابعی است که در همه بخش های دیگر استفاده خواهند

شد این توابع شامل تغییرات لازم برای کدبندی ها و محاسبات خروجی توابع درهم سازی می باشد.

این کامپوننت از کلاس هایی زیر تشکیل شده است:

۱. کلاس^۳ EncoderBase58 که وظیفه کد بندی نویسه ها از حالت آرایه ای از بایت ها به Base58

را دارد.

۲. کلاس AES که شامل توابع مورد نیاز برای رمزنگاری AES در سیستم می باشد.

۳. کلاس ECDSA که شامل توابع مورد نیاز برای امضا دیجیتال با کمک رمزنگاری ECC می -

باشد.

۴. کلاس های SecureLine که شامل توابع مورد نیاز برای رمزکردن پیام ها و انتقال آن ها در

^۱ Command Line Interface

^۲ Graphic User Interface

^۳ Class

بستر اینترنت می‌باشد.

۵. کلاس استاتیک^۱ Util که شامل توابعی برای کد بندی و یا هش کردن می‌باشد.

۶. کلاس Configuration که شامل پیکر بندی‌های سیستم می‌باشد.

۷. اینترفیس^۲ IPICObject که تمام کلاس‌های کامپوننت اصلی موظف به پیاده سازی آن می‌باشند.

که در ادامه به شرح کارکرد برخی از این کلاس‌ها و توابع موجود در آن‌ها خواهیم پرداخت.

۵-۹-۱-۱ کلاس AES

این کلاس که وظیفه انجام رمز نگاری AES را دارا می‌باشد دارای چهار سازنده می‌باشد که کارکرد آن‌ها به شرح زیر است.

۱. سازنده اول که یک رشته حرفی^۳ از کاربر به عنوان رمز استفاده می‌کند.

۲. سازنده دوم که از GUID به عنوان رمز استفاده می‌کند.

۳. سازنده سوم که به مانند سازنده دوم بوده اما GUID گفته شده را خود تولید می‌کند.

۴. سازنده چهارم که وظیفه انجام دادن کارهای یکسان را داشته و در هر اجرای دیگر سازنده‌ها فراخوانی می‌شود.

همچنین شامل دو تابع برای رمزگذاری و دو تابع برای رمزگشایی می‌باشد که به ترتیب رشته‌های حرفی و یا رشته ای از بایت‌ها را دریافت می‌کنند.

در پیاده سازی اصلی این کلاس از شی System.Security.Cryptography.Aes خود چهارچوب Net. استفاده شده است که این از یک بردار IV و یک کلید برای رمزنگاری استفاده می‌کند که

¹ Static

² Interface

³ String

بردار و کلید گفته شده نیز توسط شی
System.Security.Cryptography.Rfc2898DeriveBytes از رمز دریافت شده از کاربر توسط
سازنده‌ها ایجاد می‌شود.

```

public class AES : IDisposable
{
    private readonly byte[] _password;
    private readonly Aes _aes;
    private readonly byte[] _salt;
    private readonly int _iterationCount;
    public byte[] PassWord { get => _password; }
    public AES(string PassWord) : this(new object())
    {
        _password = Utilities.Util.ToByteArray(PassWord);

        Rfc2898DeriveBytes pdb = new Rfc2898DeriveBytes(_password, _salt, _iterationCount);
        _aes.Key = pdb.GetBytes(32);
        _aes.IV = pdb.GetBytes(16);
        pdb.Dispose();
    }
    public AES(Guid PassWord) : this(new object())
    {
        _password = PassWord.ToByteArray();

        Rfc2898DeriveBytes pdb = new Rfc2898DeriveBytes(_password, _salt, _iterationCount);
        _aes.Key = pdb.GetBytes(32);
        _aes.IV = pdb.GetBytes(16);
        pdb.Dispose();
    }
    public AES() : this(Guid.NewGuid())
    {
    }
    private AES(object obj)
    {
        _aes = Aes.Create();
        _salt = new byte[] { 0x49, 0x76, 0x61, 0x6e, 0x20, 0x4d, 0x65, 0x64, 0x76, 0x65, 0x64, 0x65, 0x76 };
        _iterationCount = 20_000;
    }
    public byte[] Encrypt(byte[] clearBytes)
    {
        byte[] res = null;
        using (MemoryStream ms = new MemoryStream())
        {
            using (CryptoStream cs = new CryptoStream(ms, _aes.CreateEncryptor(), CryptoStreamMode.Write))
            {
                cs.Write(clearBytes, 0, clearBytes.Length);
                cs.Close();
            }
            res = ms.ToArray();
        }
        return res;
    }
    public byte[] Decrypt(byte[] cipherBytes)
    {
        byte[] res = null;
        using (MemoryStream ms = new MemoryStream())
        {
            using (CryptoStream cs = new CryptoStream(ms, _aes.CreateDecryptor(), CryptoStreamMode.Write))
            {
                cs.Write(cipherBytes, 0, cipherBytes.Length);
                cs.Close();
            }
            res = ms.ToArray();
        }
        return res;
    }
    public byte[] Encrypt(string clearText, StringEncoding encoding = StringEncoding.UTF8)
    => Encrypt(clearText.ToByteArray(encoding));
    public byte[] Decrypt(string cipherText)
    => Decrypt(cipherText.ToByteArray(StringEncoding.Base64));
}

```

۵-۹-۱-۲ کلاس Util

این کلاس شامل توابعی می باشد که مهمترین های آنها وظایف مرتبط با محاسبه سختی را دارا می باشد، در ساختار پارسی کوین باتوجه به اینکه خروجی توابع هش ۶۴ بایت می باشد برای سختی دو مقدار در نظر گرفته شده است مقدار اول تعداد بایت های برابر با ۰ می باشد و مقدار دوم عدد یک بایت بعدی و بقیه بایت ها برابر با ۱ در نظر گرفته می شوند.

۵-۹-۲ کامپوننت اصلی ParsiCoin

```
public static bool CompareDiff(this byte[] targer)
{
    for (int i = 0; i < targer.Length; i++)
    {
        if (targer[i] > Difficulty[i]) return false;
    }
    return true;
}
```

قطعه کد ۵-۶: مقایسه سختی

این کامپوننت درحقیقت شامل اجزایی از سامانه می باشند که وظیفه اجرای الگوریتم ها و پروتکل های اصلی سامانه را دارا می باشند. و شامل موارد زیر است:

۱. کلاس Account شامل حساب های کاربری
۲. کلاس DAG شامل ساختار کلی DAG
۳. کلاس MerkleTree شامل درخت مرکله اکانت ها
۴. کلاس Node شامل اطلاعات گره ها
۵. کلاس Services که شامل سرویس های مورد نیاز در سیستم می باشد.
۶. کلاس Transaction شامل اطلاعات تراکنش ها
۷. کلاس Wallet شامل اطلاعات کیف پول ها

۵-۹-۲-۱ کلاس سازنده درخت مرکله

این کلاس وظیفه ساخت درخت مرکله از اکانت‌ها را دار می‌باشد، ساختار داده موردنظر برای ساخت درخت یک آرایه از رشته‌های حرفی عددی می‌باشد همچنین در این کلاس یک آرایه دیگر نیز به تعداد اکانت‌ها از کلاس اکانت برای نگه داری اطلاعات کامل اکانت‌ها در نظر گرفته شده است.

```
public MerkleTree()
{
    Leafs = new MerkleNode[65536];
    Nodes = new string[131071];
    for (int i = 0; i < Leafs.Length; i++)
    {
        Leafs[i] = new MerkleNode();
        Leafs[i].HashString = Leafs[i].ComputeObjectHash();
        Nodes[i + Nodes.Length / 2] = Leafs[i].HashString;
    }
    var Start = Nodes.Length / 2;
    var End = Nodes.Length;
    while (true)
    {
        if (Start == 0) break;
        for (int i = Start; i < End; i += 2)
        {
            Nodes[(i) / 2] = $"{Nodes[i]} {Nodes[i + 1]}".ComputeHashString();
        }
        End = Start;
        Start /= 2;
    }
}
```

قطعه کد ۵-۷: سازنده درخت مرکله

۵-۹-۲-۲ کلاس مرتبط با پایگاه داده

یکی از کلاس‌های این بخش کلاسی است که برنامه را به یک پایگاه داده ساده و بدون نیاز به سرور و NoSQL به نام LiteDB متصل می‌کند این پایگاه داده وظیفه ذخیره اطلاعات همه تراکنش‌ها و اکانت‌ها را در کامپیوتر هر دارنده سیستم دارا می‌باشد.

۵-۹-۳ کامپوننت PVM

این کامپوننت در حقیقت ماشین مجازی سیستم را دربر می‌گیرد که یک ماشین مجازی مبتنی بر پشته می‌باشد، به طوری که همه دستورات آن مقداری را از پشته خوانده و یا مقداری را در پشته می‌نویسند.

این ماشین مجازی به دلیل محدودیت دستوراتش قادر به شبیه سازی کامل یک ماشین تورینگ نمی باشد، بنابر این زبان آن از نوع زبان های کامل تورینگ نمی باشد. که در ادامه به شرح دستورات و کارکرد آن ها و همچنین اجزا ماشین خواهیم پرداخت.

۱-۳-۹-۵ اجزا PVM

ماشین مجازی PVM شامل اجزای زیر می باشد:

۱. لیست دستورات

۲. حافظه Memory که وظیفه نگه داری رکوردهای داده ای با حجم بیش از ۶۴ بایت را دارا می باشد. این حافظه به هر داده یک نشانگر^۱ اختصاص می دهد، مقدار این نشانگر که معادل آدرس آن در حافظه است در پشته ذخیره می شود.

۳. پشته که به برای نگه داری مقادیر استفاده می شود، اندازه هر رکورد داده ای آن ۶۴ بیت و اندازه کل پشته ۱۰۲۴ رکورد می باشد بنابر این قادر به نگهداری حجمی از داده معادل ۶۴*۱۰۲۴ یا 64KB می باشد.

۴. واحد پردازش که در آن وظیفه هر دستور توسط یک تابع برنامه نویسی به آن تخصیص داده شده است، در این واحد دستورات یک به یک خوانده شده و سپس تابع مربوط به هر کدام با ورودی های مناسب اجرا می شود.

```
public bool Push(byte[] r)
{
    if (SP < -1) throw new ArgumentException();
    SP++;
    _data[SP] = r;
    return true;
}
public bool Push(string s)
{
    var r = s.ToByteArray();
    if (r.Length > 64)
    {
        return Push(_mem.Add(s));
    }
    else
    {
        return Push(r);
    }
}
```

قطعه کد ۵-۸: تابع Push در پشته

¹ Pointer

```

public bool Pop(out byte[] r)
{
    if (SP < 0)
    {
        r = null;
        return false;
    }
    r = _data[SP];
    SP--;
    return true;
}
public bool Pop(out string s)
{
    var b = Pop(out byte[] r);
    if (!b)
    {
        s = string.Empty;
        return b;
    }
    if (_mem.ContainsKey(r))
    {
        s = _mem[r];
        _mem.Remove(r);
    }
    else
    {
        s = r.FromByteArray();
    }
    return true;
}

```

قطعه کد ۵-۱۰: تابع *Pop* در پشته

```

public bool? Process()
{
    bool? res = null;
    foreach (var item in Codes)
    {
        try
        {
            res = _actions[item].Invoke();
        }
        catch
        {
            return false;
        }
    }
    return res;
}

```

قطعه کد ۵-۹: تابع فراخوان دستورات در واحد پردازش

۲-۳-۹-۵ دستورات PVM

در این بخش به بررسی لیست دستورات ماشین مجازی PVM می‌پردازیم.

۱. دستور Zeor که یک رشته با مقادیر تمام ۰ را درپشته ذخیره می‌کند.
۲. دستور One که یک رشته با مقادیر تمام ۱ را درپشته ذخیره می‌کند.
۳. دستور MD5 که به محاسبه هش از نوع MD5 بالاترین عنصر پشته می‌پردازد.
۴. دستور SHA256 که به محاسبه هش از نوع SHA256 بالاترین عنصر پشته می‌پردازد.
۵. دستور SHA512 که به محاسبه هش از نوع SHA512 بالاترین عنصر پشته می‌پردازد.
۶. دستور DoubleSHA512 که به محاسبه دو بار هش از نوع SHA512 بالاترین عنصر پشته می‌پردازد.

```
_actions.Add(Commands.SHA512, () =>
{
    _stack.Pop(out string data);
    var data2 = data.ComputeHash(HashAlgorithms.SHA512);
    _stack.Push(data2);
    return null;
});
```

قطعه کد ۵-۱۱: تابع اجرا دستور DoubleSHA512

۷. دستور DoubleSH256 که به محاسبه دو بار هش از نوع SHA256 بالاترین عنصر پشته می‌پردازد.

۸. دستور Dup که مقدار سر پشته را مجدداً در پشته ذخیره می‌کند.

۹. دستور CheckSig که به بررسی تایید هویت امضا دیجیتال می‌پردازد. و در صورت معتبر

```
_actions.Add(Commands.CheckSig, () =>
{
    _stack.Pop(out string PubKey);
    _stack.Pop(out string Sig);
    _stack.Pop(out string Message);
    var ecdsa = new ECDSA(PubKey);
    var res = ecdsa.Verify(Sig, Message);
    if (res) _stack.Push(_one);
    else _stack.Push(_zero);
    return null;
});
```

قطعه کد ۵-۱۲: تابع اجرا دستور CheckSig

بودن امضا مقدار ۱ را در پشته ذخیره می‌کند، و در غیر این صورت مقدار ۰ را ذخیره می‌کند.

۱۰. دستور IsOne که به بررسی برابر ۱ بودن آخرین مقدار داخل پشته می‌پردازد.

۱۱. دستور IsZero که به بررسی برابر ۰ بودن آخرین مقدار داخل پشته می‌پردازد.

۱۲. دستور Eq که برابری دو مقدار آخر پشته را بررسی می‌کند. و در صورت برابر بودن مقدار ۱

را در پشته ذخیره می‌کند و در غیر این صورت مقدار ۰ را در پشته ذخیره می‌کند.

۵-۹-۴ کامپوننت CLI

این کامپوننت در حقیقت به تعامل با کاربر به صورت دستورات نوشتاری می‌پردازد که با دریافت هر دستور عملیات مورد نظر آن را اجرا کرده و خروجی را در صفحه کنسول^۱ نشان می‌دهد. با دریافت دستور Exite سیستم متوقف شده و فعالیت‌های جاری ذخیره می‌شوند.

۵-۹-۵ دستورات CLI

کامپوننت CLI به منظور راحتی کاربر دستورات متنوعی را پشتیبانی می‌کند این دستورات بنابر کارکرد آن‌ها می‌توانند یک یا چند ورودی و یا سوییچ^۲ داشته باشند که اجرا دستور مورد نظر را مدیریت کنند، همچنین همه دستورات برای راهنما کامل از کارکرد آن‌ها و چگونگی استفاده از دستور می‌باشند.

۵-۹-۵-۱ دستور Init

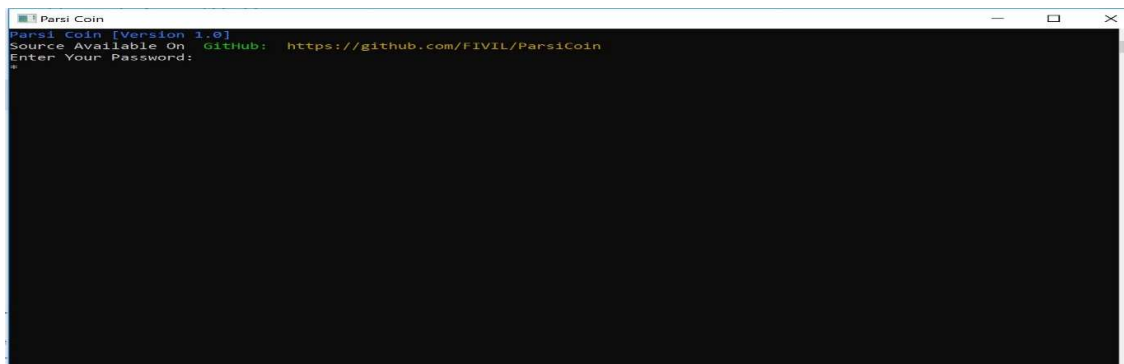
این دستور در حقیقت به فعال سازی اولیه برنامه می‌پردازد و بررسی می‌کند که آیا برنامه در کامپیوتر میزبان برای اولین بار اجرا می‌شود و یا قبلاً اجرا شده است سپس بر اساس نتایج این بررسی به ایجاد و به روز رسانی اطلاعات حساب‌ها و همگامسازی^۳ اطلاعات DAG می‌پردازد.

این دستور هیچ ورودی یا سوییچی نمی‌پذیرد.

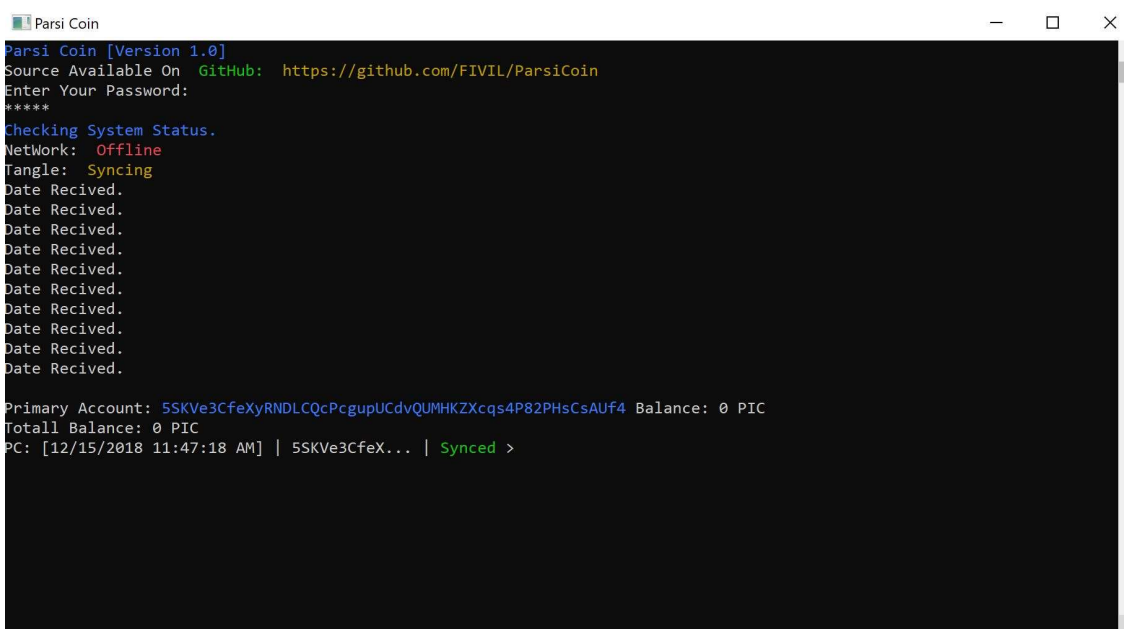
^۱ Console

^۲ Switch

^۳ Sync



تصویر ۵-۲: اجرای دستور Init



تصویر ۵-۳: صفحه اول برنامه

۵-۹-۵-۲ دستور Exite

این دستور ابتدا همه فعالیت ایجاد شده در نشست^۱ جاری را ذخیره کرده و سپس از برنامه خارج می‌شود.

این دستور هیچ ورودی یا سوئیچی نمی‌پذیرد.

۵-۹-۵-۳ دستور Sync

این دستور به خاموش و یا روشن کردن سیستم همگام‌ساز می‌پردازد که باروشن بودن آن همگام‌سازی

¹ Session

به صورت خودکار انجام می‌شود و با خاموش کردن آن سیستم به حالت برون خط^۱ می‌رود.

این دستور هیچ ورودی ندارد اما از دو سویچ -N و -F پشتیبانی می‌کند که به ترتیب به خاموش یا روشن کردن همگام‌ساز می‌پردازند.

۴-۵-۹-۵ دستور Account

این دستور برای مدیریت حساب‌های کاربری فرد می‌باشد.

این دستور باتوجه به حالت درخواستی استفاده از آن ورودی‌های متنوعی را دریافت می‌کند همچنین از چهار سویچ -A، -G، -P و -N پشتیبانی می‌کند که به ترتیب به نمایش اطلاعات همه حساب‌های کاربری، نمایش اطلاعات کامل یک حساب کاربری، انتخاب یک حساب کاربری به عنوان پیشفرض و ایجاد حساب کاربری جدید می‌پردازد.

```
PC: [12/15/2018 11:48:15 AM] | 5SKVe3CFeX... | Synced > Help Account
-A : Show All Accounts. Usage: Account -A
-G <Specified account index> : Get Specific Account Full Details. Usage: Account -G 1
-P <Specified account index> : Set Specific Account As Primary Account. Usage: Account -P 1
-N : Create New Account. Usage: Account -N
PC: [12/15/2018 11:48:21 AM] | 5SKVe3CFeX... | Synced >
```

تصویر ۴-۵: راهنمای دستور Account

۵-۵-۹-۵-۵ PrivateKey دستور

این دستور برای کنترل کلیدهای خصوصی کاربر طراحی شده است که به کمک آن می‌توان به ذخیره اطلاعات کلیدهای خصوصی بر روی فایل یا خواندن آن‌ها از روی فایل متنی پرداخت.

این دستور ورودی‌های متنوعی بر اساس حالت استفاده از آن دارد که به تعیین مسیر فایل مورد نظر برای خواندن یا نوشتن اطلاعات می‌پردازند، سویچ‌های آن شامل -E برای نوشتن اطلاعات و -I برای خواندن اطلاعات می‌باشند.

¹ Offline

۵-۹-۵-۶ دستور Send

این دستور به ایجاد و ارسال یک تراکنش و انتقال تعدادی توکن می‌پردازد.

این دستور داری سوییچ خاصی نمی‌باشد اما ورودی‌های آن شامل مواردی از قبیل مقدار توکن برای ارسال و آدرس دریافت کننده و اکانت ارسال کننده می‌باشند.

```
PC: [12/15/2018 11:49:02 AM] | 5SKVe3CfeX... | Synced > Send 0 6veU98pyowQnJJZDgbUFLU4BM3DLjzmE7U43aUBJL1gVqJrnb
{
  "TransactionIssuer": "5SKVe3CfeXyRNDLCQcPcgupUCdvQUMHKZXCqs4P82PHsCsAUf4",
  "Recieipient": "6veU98pyowQnJJZDgbUFLU4BM3DLjzmE7U43aUBJL1gVqJrnb",
  "Amount": 0.0,
  "IssueTime": "2018-12-15T08:19:14.1003629Z",
  "ApprovalTime": "0001-01-01T00:00:00",
  "TxHash": "WmXVfKRib13boGHcmYhKXudW6ZkU1nUQXWdNLBEetn8HMTNcKeM7xHMSVGyVbBSvPHiy24yHdTC7iqEZ59dNR6QH4D5B",
  "TxMessage": null,
  "NodeID": "00000000-0000-0000-0000-000000000000",
  "Signature": "HyX60jVKHL+B1HczKA64eVsOI39GWhqBH3dyCCcOVCJNHrLHNop5PRS+/fpVPuL0cN2J2Ddw18iCe6t8noemRnc=",
  "ScriptPubKey": "HyX60jVKHL+B1HczKA64eVsOI39GWhqBH3dyCCcOVCJNHrLHNop5PRS+/fpVPuL0cN2J2Ddw18iCe6t8noemRnc;5SKVe3CfeXyRNDLCQcPcgupUCdvQUMHKZXCqs4P82PHsCsAUf4",
  "ScriptSig": "HyX60jVKHL+B1HczKA64eVsOI39GWhqBH3dyCCcOVCJNHrLHNop5PRS+/fpVPuL0cN2J2Ddw18iCe6t8noemRnc;5SKVe3CfeXyRNDLCQcPcgupUCdvQUMHKZXCqs4P82PHsCsAUf4;CheckSig;IsOne"
}
```

تصویر ۵-۵: اجرا دستور Send و اطلاعات تراکنش ایجاد شده و امضا شده

۵-۹-۵-۷ دستور Receive

این دستور برای نمایش کلید عمومی حساب‌های کاربری خودی برای دریافت می‌پردازد.

۵-۹-۵-۸ دستور Peer

این دستور به منظور نمایش اطلاعات نظیرها و یا افزودن نظیر جدید و حذف نظیر موجود به کار می‌رود.

این دستور بر اساس حالت استفاده از آن ورودی‌هایی را دریافت می‌کند، سوییچ‌های آن شامل A- برای نمایش اطلاعات نظیرها، N- برای افزودن نظیر جدید و D- برای حذف نظیر فعلی می‌باشد.

```
PC: [12/15/2018 11:49:32 AM] | 5SKVe3CfeX... | Synced > Help Peer
-A : Show all connected peers. Usage: Peer -A
-N <peerIP> : Set new Peer. Usage: Peer -N [0.0.0.0]
-D <peerID> : Delete specified peer from connected peers. Usage: Peer -D 0
PC: [12/15/2018 11:49:40 AM] | 5SKVe3CfeX... | Synced >
```

تصویر ۵-۶: راهنمای دستور Peer

۵-۹-۵-۹ دستور Help

این دستور به نمایش راهنمای دستورات می‌پردازد، در صورتی که هیچ دستوری برای نمایش راهنما انتخاب نشود به نمایش لیست دستورات می‌پردازد.



```
PC: [12/15/2018 11:47:18 AM] | 5SKVe3CFeX... | Synced > Help
Init
Exite
Sync
Account
PrivateKey
UpdatePassword
Send
Recive
Peer
Help
cls
PC: [12/15/2018 11:47:45 AM] | 5SKVe3CFeX... | Synced >
```

تصویر ۵-۷: اجرا دستور Help

ورودی این دستور می‌تواند اسم دستور مورد نظر برای دریافت راهنمای آن یا خالی باشد.

۵-۹-۵-۱۰ دستور cls

این دستور به پاک کردن اطلاعات نوشته شده در صفحه می‌پردازد.

۵-۱۰ لایه شبکه

این لایه در اصل وظیفه نقل و انتقال بسته های اطلاعاتی بر روی یک شبکه نظیر به نظیر را در گره های اجرا کننده سرویس داره می باشد در این لایه از پروتوکل ¹TCP برای نقل و انتقال بسته ها استفاده شده است، یکی از چالش های پیش رو در این لایه عدم امکان وجود سرویس دهنده ای با آدرس IP مشخص و قابل مشاهده برای همه سرویس گیرنده ها می باشد. بنابر این ضروری به نظر می رسد که نظیر های موجود در شبکه که هر کدام پشت یک برگردان نشانی شبکه ² و یا به اختصار NAT قرار دارند بتوانند به صورت مستقیم با یکدیگر ارتباط برقرار کنند از این رو عملیاتی با نام NAT Traversal لازم است انجام شود که راه انجام این عملیات نیز یک راهکار شبکه با نام Port Forwarding می باشد که با انجام این عملیات در حقیقت یکی از

¹ Transmission Control Protocol

² Network Address Translation

Port های مسیر یاب^۱ شبکه فرد اجرا کننده سرویس گیرنده باز شده و تمامی ترافیک دریافتی خود را به یک Port مشخص شده در سیستم ارسال می کند به این ترتیب سیستم های موجود می توانند اطلاعات ارسالی خود را به آدرس IP مشخص آن NAT ارسال می کنند و سپس مسیر یاب مربوط بسته ها را به سمت کامپیوتر اجرا کننده سرویس ارسال می کنند بنابر این دو نظیر می توانند توسط شبکه اینترنت بایکدیگر ارتباط برقرار کنند.

۵-۱۰-۱ ارتباط امن در شبکه

باتوجه به اینکه بسته های ارسالی و دریافتی بسته های حساسی هستند به نظر می رسد که لازم است این بسته ها که گاهی از مسیر های نا امن در شبکه منتقل می شوند بتوانند ابتدا رمز گذاری شوند در این سامانه برای این رمز گذاری از AES برای نقل و انتقال بسته ها استفاده شده است که رمز مورد نیاز برای هر ارتباط توسط الگوریتم های رشته تصادفی ساز به صورت تصادفی ایجاد می شوند و سپس با پروتکل انتقال رمزی به مانند پروتکل انتقال رمز Diffie Hellman و با کمک رمزنگاری نامتقارن RSA بین دو نظیر موجود در شبکه جابه جا می شوند و سپس دو نظیر می توانند بر روی یک کانال^۲ امن شده ارتباطی با یکدیگر به نقل و انتقال داده بپردازند.

این ارتباط امن توسط دو کلاس در برنامه با نام های SecureLineClient برای نظیری که سرویس گیرنده است و SecureLineServer برای نظیر سرویس دهنده پیاده سازی شده است. که خود این دو کلاس نیز از دو کلاس برای انجام پروتکل انتقال رمز با رمزگذاری RSA با نام ها RsaKeyExchClient و RsaKeyExchServer استفاده می کنند.

^۱ Router

^۲ Channel

```

public class SecureLineClient
{
    private AES _aes;
    private readonly RsaKeyExchClient _rsaKeyXchC;

    public SecureLineClient()
    {
        _rsaKeyXchC = new RsaKeyExchClient();
    }

    public string PubKey { get => _rsaKeyXchC.PubKey; }

    public void InitiateServer(string send)
    {
        _aes = _rsaKeyXchC.ImportPassword(send);
    }

    public string Encrypt(string message)
    {
        if (_aes is null) throw new Exception("Not initiated");
        return _aes.Encrypt(message).ToBase64();
    }

    public string Decrypt(string message)
    {
        if (_aes is null) throw new Exception("Not initiated");
        return _aes.Decrypt(message).FromByteArray();
    }
}

```

قطعه کد ۵-۱۴: کلاس *SecureLineClient*

```

class RsaKeyExchClient : IDisposable
{
    private readonly RSACryptoServiceProvider _rsa;

    private readonly string _pubKey;

    public string PubKey { get => _pubKey; }
    #region ctor
    public RsaKeyExchClient()
    {
        _rsa = new RSACryptoServiceProvider();
        _pubKey = Convert.ToBase64String(_rsa.ExportCspBlob(false));
    }
    #endregion

    public AES ImportPassword(string Pass)
    {
        var key = Convert.FromBase64String(Pass);
        var K = new Guid(_rsa.Decrypt(key, false));
        return new AES(K);
    }

    public void Dispose()
    {
        _rsa.Dispose();
    }
}

```

قطعه کد ۵-۱۳: کلاس *RsaKeyExchClient*

```

public class SecureLineServer:IDisposable
{
    private AES _aes;
    private readonly RsaKeyExchServer _rsaKeyXchS;

    public SecureLineServer(string pubKey)
    {
        _rsaKeyXchS = new RsaKeyExchServer(pubKey);
        _aes = new AES();
    }

    public string InitaitClient() => _rsaKeyXchS.ExportPassWord(_aes);

    public string Encrypt(string message)
    {
        if (_aes is null) throw new Exception("Not initaited");
        return _aes.Encrypt(message).ToBase64();
    }

    public string Decrypt(string message)
    {
        if (_aes is null) throw new Exception("Not initaited");
        return _aes.Decrypt(message).FromByteArray();
    }

    public void Dispose()
    {
        ((IDisposable)_aes).Dispose();
    }
}

```

قطعه کد ۵-۱۶: کلاس *SecureLineServer*

```

class RsaKeyExchServer : IDisposable
{
    private readonly RSACryptoServiceProvider _rsa;

    #region ctor
    public RsaKeyExchServer(string pubKey)
    {
        _rsa = new RSACryptoServiceProvider();
        _rsa.ImportCspBlob(Convert.FromBase64String(pubKey));
    }
    #endregion

    public string ExportPassWord(AES Aes)
    {
        var EncryptedKey = _rsa.Encrypt(Aes.PassWord, false);
        return Convert.ToBase64String(EncryptedKey);
    }

    public void Dispose()
    {
        _rsa.Dispose();
    }
}

```

قطعه کد ۵-۱۵: کلاس *RSACryptoServiceProvider*


```

public Usage()
{
    var c = new SecureLineClient();
    var s = new SecureLineServer(c.PubKey);
    var password = s.InitaiteClient();
    c.InitaiteServer(password);
    while (true)
    {
        var data = Console.ReadLine();
        var h = c.Encrypt(data);
        Console.WriteLine(s.Decrypt(h));
        data = Console.ReadLine();
        var h2 = s.Encrypt(data);
        Console.WriteLine(c.Decrypt(h2));
    }
}

```

قطعه کد ۵-۱۷؛ استفاده از SecureLine

۵-۱۱ بسته‌های نرم‌افزاری استفاده شده

در این سامانه از بسته‌های نرم‌افزاری زیر استفاده شده است:

۱. بسته نرم‌افزاری Json.net که برای سرالایز^۱ کردن اطلاعات از فرمت Json و به فرمت Json است.

۲. بسته LiteDB برای کار کردن با api های LiteDB

۳. بسته NbitCoin برای استفاده از توابع مرتبط با امضا دیجیتال ECC موجود در این بسته.

۴. بسته NetStandardLibrary به عنوان بسته اصلی دربرگیرنده چهارچوب دات نت

۵-۱۲ سورس کنترل^۲

نسخه کاملی از کد برنامه به صورت بسته‌های سورس کنترل گیت^۳ در سایت گیت‌هاب و در مسیر

^۱ Serialize

^۲ Source Control

^۳ Git

<https://github.com/FIVIL/ParsiCoin> موجود می باشد.

- [1] Imran Bashir, Mastering Blockchain, Packt Publishing, 2018
- [2] Andreas M. Antonopoulos, Mastering Bitcoin, O'Reilly, 2017
- [3] Wiley Brand, Blockchain for dummies, IBM Limited Edition, 2017
- [4] Satoshi Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System", 2009
- [5] BlockChain, <https://en.wikipedia.org/wiki/Blockchain>, 2018
- [6] bitcoin difficulty chart at <https://bitinfocharts.com/comparison/bitcoin-difficulty.html>, 2018
- [7] Andrew S. Tanenbaum, Distributed Systems, Maarten van Steen, 2017
- [8] Chris Dannen, Introducing Ethereum and Solidity, Apress, 2017
- [9] Nicolas Dorier, Programming the blockchain in C#, GitBook, 2018
- [10] Narayan Prusty, Building Blockchain Projects, Packt Publishing, 2017
- [11] Ethereum foundation, "Ethereum white paper", 2014
- [12] Ethereum foundation (Dr.Gavin Wood), "Ethereum yellow paper", 2015
- [13] IOTA (Serguei Popov) "The Tangle", 2018
- [14] Leemon Baird, Mance Harmon, and Paul Madsen "Hedera: A Governing Council & Public Hashgraph Network", 2018
- [15] IOTA documentation, <https://docs.iota.org>, 2018
- [16] Directed acyclic graph, [WIKIPEDIA]Directed_acyclic_graph, 2018
- [17] Merkle tree, [WIKIPEDIA] Merkle_tree, 2018
- [18] Kass, "Creating Your First Blockchain with Java" available on medium, 2017