

**دانشکده مهندسی کامپیوتر**

**پایان نامه دوره کارشناسی مهندسی کامپیوتر-نرم افزار**

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

**نگارش:**

**حامد محمدی**

**استاد راهنما:**

**دکتر سعید صدیقیان کاشی**

**پاییز 1397**

**چکیده**

# فصل اول : مقدمه

در طول تاریخ بشر همواره ردپای معامله به روش های 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 روش های کارآمد تری نظیر انتخاب یک کالا به عنوان مرجع روی اورده شد، تا اینکه در نهایت با ضرب اولین سکه ها از طلا توسط اقوام باستانی در آسیای صغیر مفهموم پول ایجاد شد.

## تاریخچه مختصری از پول

اولین پول ها توسط قوم باستانی لیدی ها ساکن در منطقه آسیاس صغیر تقریبا در قرن 7 قبل از میلاد ضرب شدند این سکه ها 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 سکه ها رسمیت و اعتبار می­بخشید رفته رفته دیگر اقوام نیز به ضرب سکه روی آوردند و این گونه بود که دوران سکه ها در تاریخ بشر آغاز شد، به طوری که امروزه نیز در اکثر تحقیقات باستان شناسی از سکه های هر قوم و ملیت به عنوان نماد آن قوم و ملیت یاد شده و دارای اهمیت زیادی می­باشند.

### استاندارد طلا [[1]](#footnote-1)

با گذر زمان و افزایش حجم معاملات تجاری لزوم حمل و نقل مقدار زیادی سکه ایجاد می­شد که حمل این سکه ها دردسر های زیادی را به همراه داشت، بنابر این سازمان هایی به وجود آمدند که در ازای تعداد مشخصی سکه حواله هایی 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 حواله ها دارای ارزش معادلی به سکه طلا بودند که به مرور زمان در بین مردمان رواج پیدا کردند به طوری که امروز نیز در قوانین اساسی اکثر کشور ها میزان مشخصی از طلا به ازای هر واحد پول آن کشور اختصاص داده می­شود [[2]](#footnote-2) . اما در نهایت با توجه به لزوم وجود نقدینگی در کشور ها و با افزایش جمعیت و برخی عوامل دیگر دولت ها مجبور به چاپ مقادیر پول بیشتر از ذخایر طلای خود شدند که به این ترتیب ارز های بی پشتوانه [[3]](#footnote-3) ایجاد شدند.

### 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 ارز پول معادل ارز چاپ کننده آن در نظر گرفته می­شود و با تغییرات شرایط سیاسی یا اقتصادی یک کشور ارزش پول آن کشور نیز تغییر می­کند. همچنین دولت ها در زمان های کمبود نقدینگی در کشور هایشان با اقدام به چاپ بی رویه این نوع پول باعث ایجاد تورم و کاهش ارزش پول خود می­شوند، مجموع عوامل ذکر شده دلیل تغییراتی است که امروزه در قیمت ارز های کشور های مختلف نسبت به یکدیگر شاهد هستیم؛ لازم به ذکر است که تمامی ارز های مرجع امروزی از نوع پول بی پشتوانه می­باشند.

## تراکنش های مالی و نظام بانکی

با گذر زمان و نیاز به انجام تراکنش های مالی با مبالغ سنگین یا انجام پرداخت در زمان مشخصی در آینده بانک های کشور های مختلف اقدام به ایجاد سازکار های مختلفی نظیر دسته چک ها و یا چک های رمز دار بین بانکی و همچنین انتقال از حساب یک شخص به حساب شخص دیگر نمودند که در نهایت با ورود تکنولوژی های جدید مواردی از قبیل کارت های اعتباری [[4]](#footnote-4) و یا کارت های نقدی [[5]](#footnote-5) امروزه به عنوان یکی از روش های اصلی انجام تراکنش های مالی شناخته می­شوند که این کارتها قابلیت انجام انواع تراکنش های مالی مختلف را با کمک دستگاه هایی نظیر خود پرداز ها [[6]](#footnote-6) و یا پایانه های فروشگاهی [[7]](#footnote-7) فراهم می­آوند همچنین به کمک درگاه های پرداخت اینترنتی قابلیت پرداخت با کمک اینترنت نیز فراهم شده است.

### اشکالات تراکنش های مالی فعلی

اگرچه انجام تراکنش های مالی امروزه به راحتی از طریق کارت هایی 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.

1. زمان بين انجام تراكنش و تسويه مي­تواند طوﻻني باشد.
2. دوباره كاريها و نياز به شخص سوم براي تاييد اعتبار و حضور واسطه ها.
3. كلاهبرداري، حمله هاي سايبري و حتي اشتباه هاي كوچك به هزينه و پيچيدگي كسب وكار مي افزايند و اگر يك سامانهي مركزي مانند بانك به خطر بيفتد، همگي مشاركت كنندگان در شبكه با مخاطره روبه رو خواهند شد.
4. برای استفاده از خدمات این کارت های اعتباری اغلب نیاز به پرداخت هزینه های اولیه زیاد می­باشد.
5. دریافت این نوع کارت های شامل فرآیند های وقت گیر و کاغذ بازی ها و بررسی سابقه افراد است.
6. این نوع سامانه ها دارای شفافیت کافی در برخی موارد نمی­باشند، در حقیقت هیچ کس نمی­داند که آن سازمان مرکزی صادر کننده کارت ها و حسابها چگونه به انجام این کارها می­پردازد.
7. وجود چنین سامانه های مرکزی با نفوذ افراد و دولت های مختلف رخ می­دهد که در برخی موارد می­تواند باعث حذف برخی افراد از سامانه و قطع سرویس دهی به آنها شود نظیر تحریم های اعمال شده علیه کشور ایران که دسترسی این کشور و مردم آن را به بیشتر سامانه های 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 ارز ها که ارز های 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 تراکنش های مالی جهانی در نظر گرفته شوند.

## ارز های دیجیتال [[8]](#footnote-8)

به طور کلی هر ارزی که نمود فیزیکی نداشته باشد و به عنوان واحد های کامپیوتری در شبکه های 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 [[9]](#footnote-9) که در ادامه به بررسی آن خواهیم پرداخت نمی­باشد و هر نوع واحد پولی دیجیتالی حتی سکه ها و یا اعتبارات موجود در اکثر بازی های کامپیوتری را نیز می­توان نوعی ارز دیجیتال دانست.

### تاریخچه ارز های دیجیتال

ابتدایی ترین نوع ارز های دیجیتال که در حقیقت مفاهیم استفاده شده در برخی از آنها مبنای کارکرد رمز ارزهای امروزی است اغلب دارای یکاهایی معادل با یکی از ارز های 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 ارز ها رزرو آزادی [[10]](#footnote-10) بود که در دهه 1990 میلادی همزمان با حباب دات-کام ایجاد شد. اگرچه اغلب این ارز های دیجیتال از مفاهیمی نظیر امضا دیجیتال [[11]](#footnote-11) و مکانیزم هایی شبیه به گواهی اثبات کار[[12]](#footnote-12) و یا به اختصار POW نیز استفاده می­کردند و در برخی موارد تا حدی نیز به صورت نظیر به نظیر [[13]](#footnote-13) کار می­کردند اما همه آنها در نهایت برای ایجاد توافق[[14]](#footnote-14) به یک سازمان مرکزی وابسته بودند که در حقیقت این بدان معنی بود که در انجام بزرگترین هدف خود که حذف واسطه ها از انجام تراکنش ها بود نا موفق بودند بنابر این اغلب این ارز های دیجیتال به سرعت به فراموشی سپرده شدند با این حال مفاهیم استفاده شده در بسیاری از آنها به عنوان مبنای رمز ارز های امروزی قرار گرفت. در نهایت در سال 2009 میلادی با معرفی بیت کوین [[15]](#footnote-15) مشکل نیاز به واسطه برای انجام تراکنش ها حل شد و برای انجام توافق از یک الگوریتم قوی کامپیوتری به نام بلاک چین [[16]](#footnote-16) استفاده شد.

# فصل دوم : برخی مفاهیم پایه بکار رفته در رمز ارز ها

برای بررسی بیشتر رمز ارز ها و بلاک چین ابتدا لازم به نظر می­رسد تا با برخی مفاهیم پایه و الگوریتم های به کار رفته در این سامانه ها آشنا شویم.

## رمز نگاری[[17]](#footnote-17)

از ابتدای تاریخ نوع بشر همواره نیاز به رمزکردن پیام های 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 کد های انیگما [[18]](#footnote-18)نوشته شده نازی ها توسط دانشمند انگلیسی و پدر علم کامپیوتر آلن تورینگ [[19]](#footnote-19) سر نوشت جنگ به نفع متحدین تمام شد.

در ابتدا بشر برای رمزنگاری پیام های خود از روش ها مختلفی نظیر قرار دادن سیمبل ها و رمز های خاص استفاده می­کرد تا اینکه پیوند این رمز ها با ریاضیات و الگوریتم های خاص ریاضی پیدا شد و با بررسی رمزنگاری به عنوان شاخه ای از علم ریاضی بشر موفق به ایجاد رمز های پیچده شد و در نهایت با ورود کامپیوتر ها عرصه رمز نگاری نیز توسط این ماشین های قدرتمند دچار تغییراتی شد و الگوریتم های پیچیده رمزنگاری کامپیوتری ایجاد شدند به طوری که امروزه برخی از آنها نطیر الگوریتم AES [[20]](#footnote-20) به قدری امن در نظر گرفته می­شوند که شکستن رمز های آنها تقریبا غیر ممکن به نظر می­رسد.

### رمز نگاری متقارن [[21]](#footnote-21)

این نوع رمزنگاری به نوعی خاصی از رمزنگاری گفته می­شود که در آن برای رمز کردن و رمزگشایی 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 رو گروه از الگوریتم های جریانی و بلوکی مورد استفاده قرار می­گیرند که الگوریتم های جریانی اغلب به رمز کردن یک واحد کوچک داده یا همان بیت به همان صورت که داده در جریان است می­پردازند و الگوریتم های بلوکی به رمز کردن یک تعداد مشخص از واحد داده ها به عنوان بلوک می­پردازند ، به طور مثال الگوریتم AES به رمزنگاری داده در بلوک های 128 بیتی می­پردازد.

### رمز نگاری نامتقارن [[22]](#footnote-22)

این نوع رمز نگاری در مقابل رمزنگاری متقارن قرار دارد و در آن­ها برای رمز کردن و رمزگشایی پیام از 2 کلید متفاوت استفاده می­شود که یکی از آنها به کلید عمومی [[23]](#footnote-23) و دیگری به کلید خصوصی [[24]](#footnote-24) شهرت دارند و همچنین یک رابطه ریاضی بین این 2 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 الگوریتم های رمزنگاری نا متقارن به این صورت است است که هر فرد ابتدا با ایجاد یک کلید خصوصی و نگه داشتن آن نظیر خود به صورت امن و به دست آوردن کلید عمومی نظیر آن کلید خصوصی و دراختیار عموم قرار دادن آن کلید عمومی به رد و بدن کردن پیام های 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 در اختیار خود فرد است قابل رمزگشایی خواهد بود ،بنابر این همه پیام های 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 پیام های خود خواهند بود از الگوریتم های معروف رمزنگاری نامتقارن می­توان از الگوریتم RSA [[25]](#footnote-25) که بر پایه هم نهشتی و با کمک اعداد اول بزرگ کار می­کند و ECC [[26]](#footnote-26) بر اساس ساختاری جبری از منحنی‌های بیضوی بر روی میدانهای متناهی طراحی شده‌است نام برد البته اغلب این الگوریتم ها با کمک الگوریتم های دیگر مورد استفاده قرار می­گیرند.

#### الگوریتم های تبادل کلید

الگوریتم های رمزنگاری نامقارن 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 رمز نگاری 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 الگوریتم های تبادل کلید استفاده نمود ،در حقیقت همانطور که قبلا اشاره شد الگوریتم های رمزنگاری متقارن دارای مشکل اساسی نیاز به دانستن کلید یکسان توسط هر دو طرف قبل از شروع انتقال پیام می­باشند اما در بستر های 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 الگوریتم های انتقال کلید و با کمک رمزنگاری نامتقارن می­توان ابتدا به تبادل کلید الگوریتم رمزنگاری متقارن پرداخت و سپس با کمک آن کلید در ادامه پیام هارا به صورت متقارن رمزنگاری کرد از الگوریتم های معروف این دسته می­توان از پروتکل تبادل کلید دیفی-هلمن [[27]](#footnote-27) نام برد الگوریتم استفاده شده در پروتکل امن انتقال ابرمتن[[28]](#footnote-28) ویا به اختصار Https نیز تقریبا کارکردی به همین صورت دارد.

#### امضای دیجیتال

از دیگر کاربرد های رمزنگاری نامتقارن می­توان به امضا دیجیتال اشاره کرد که در حقیقت این الگوریتم امضا دیجیتال است که در ارز رمز ها نیز بسیار کاربرد دارد و در حقیقت به اثبات هویت ارسال کننده یک پیام و یا تراکنش می­پردازد به مانند امضا عادی در سیستم های سنتی که امضا هر طرف 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 [[29]](#footnote-29) 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 می افزاید که به این فرایند امضا کردن پیام گفته می­شود حال به ارسال این پیام به شخص یا اشخاص مورد نظر آنها با داشتن کلید عمومی طرف 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 2 رشته اثبات هویت ارسال کننده پیام صورت می­گیرد که در حقیقت فرد متناظر با ان کلید عمومی است.

#### رمزنگاری ECC

باتوجه به اینکه در اغلب رمز ارزها و سامانه های 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 روش های رمزنگاری نا متقارن می­شود، در حقیقت برای اهداف امروزی رمزنگاری، منحنی بیضوی یک منحنی مسطح است که متشکل از نقاط رضایت بخش معادله می‌باشد.

|  |  |
| --- | --- |
| معادله ‏2‑1: منحنی های بیضوی در 2 بعد |  |

همراه با یک نقطه برجسته در (مختصات در اینجا از یک حوزه ثابت متناهی از مشخصه که با ۲ یا ۳ برابر نیست انتخاب می‌شوند، یا اینکه معادله منحنی تا حدودی پیچیده‌تر خواهد بود) این مجموعه همراه با عملیات گروهی از نظریه گروه بیضوی از گروه Abelian، با نقطه‌ای در بینهایت به عنوان عنصر هویت می‌باشند. ساختار گروه از گروه مقسوم علیه تنوع جبری زیرین ارث بری می‌کند. همان‌طور که برای دیگر سیستم‌های رمزنگاری کلید عمومی محبوب، بدون اثبات ریاضی برای امنیت ECC از سال ۲۰۰۹ منتشر شد.

درنهایت باید دانست که امنیت کاملECC بستگی به توانایی محاسبه ضرب نقطه‌ای و عدم توانایی برای محاسبه حاصلضرب با توجه به نقاط اصلی و نقاط تولید شده دارد.

#### تهدید های متوجه رمزنگاری نامتقارن

امنیت شیوه های امروزی رمز نگاری متقارن اغلب بر پایه سخت بودن حل مسائلی نظیر تجزیه اعداد اول و یا مسئله حل لگاریتم گسسته برای کامپیوتر های امروزی مطرح می­شوند که حل این مسائل به طور عادی از روابط نمایی پیروی می­کند بنابر این برای حل آنها صرف زمان بسیار بسیار زیادی توسط کامپیوتر های امروزی لازم خواهد بود به طوری که می­توان به دست آوردن پاسخ آنهارا عملا غیر قابل دست یابی دانست ، اما نوع دیگر کامپیوتر های که کامپیوتر های کوانتمی [[30]](#footnote-30) معروف هستند. با کمک الگوریتم کوانتمی شور [[31]](#footnote-31) قادر به حل این مسائل در زمان معقولی می­باشند که در حقیقت این مورد این نگرانی را در بین بسیاری از افراد به وجود آورده که این نوع کامپیوتر ها می­توانند تهدیدی برای اغلب روش های رمزنگری و سامانه های مبتنی بر رمزنگاری نامتقارن امروزی باشند ، که از همین رو تلاش هایی برای ایجاد الگوریتم های رمزنگاری نامتقارن کوانتمی [[32]](#footnote-32) شده است ، البته لازم به ذکر است که هنوز کامپیوتر های کوانتمی قدرتمند که توانایی اجرای الگوریتم گفته شده را داشته باشند ساخته نشده اند ، اما تخمین زده می­شود که تا سال 2030 میلادی نمونه اولیه چنین کامپیوتر هایی ساخته شوند ، نکته قابل توجه دیگر آن است که کامپیوتر های کوانتمی تهدیدی برای الگوریتم هایی نظیر AES نمی­باشند.

## 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 با طول ثابت 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 به توان 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 رشته ها و جداول داده درهم [[33]](#footnote-33) کاربردی می­باشند.

### تابع درهم‌سازی رمزنگارانه [[34]](#footnote-34)

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 سیستم های بلاک چینی بسیار قابل توجه است آن است که در حقیقت امکان پیش بینی خروجی از روی وردی به هیچ عنوان امکان پذیر نیست، در حقیقت کارکرد آنها به این صورت نمی­باشد 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 بسایر کوچک در رشته ورودی، رشته خروجی تغییرات قابل توجهی خواهد کرد.

## درخت مرکله [[35]](#footnote-35)

درخت مرکله و یا درخت درهم سازی به طور معمول یک درخت دودویی [[36]](#footnote-36) و یا یک درخت پیشوندی [[37]](#footnote-37) است که برگ های آن شامل یک سری داده مشخص می­باشند ، سپس در مراحل بالاتر مقدار هر گره از مقدار هش فرزندان خود به دست می­آید به همین صورت از گره های والد برای برگ ها شروع به هش کردن مقادیر میکنیم و سطح به سطح در درخت بالا می­رویم و مقدار هر گره را معادل هش مقدار فرزندان خود می­گذاریم تا به ریشه درخت برسیم.

|  |  |
| --- | --- |
| معادله ‏2‑2: گره های درخت درهم سازی |  |

کاربرد درخت های مرکر به طور معمول در نوع خاصی از امضا دیجیتال به نام امضای لمپورت [[38]](#footnote-38) و یا تایید اعتبار یک فایل و همچنین به طور گسترده در بلاک چین می­باشد.



تصویر ‏2‑1: نمونه یک درخت درهم سازی

## کدبندی نویسه [[39]](#footnote-39)

باتوجه به اینکه کوچکترین واحد قابل فهم برای کامپیوتر ها یک بیت است که دارای مقاداری معادل 0 یا 1 می­باشد در نتیجه همه مفاهیم از نظر کامیپوتر ها باید دارای یک مقدار عددی باشند­، بنابر این برای نویسه [[40]](#footnote-40) های الفبایی استاندارد هایی در نظر گرفته شده است که هر نویسه را به یک مقدار عددی مشخص نظیر می­کند از نمونه های این استاندارد ها می­توان به مواردی از قبیل ASCII [[41]](#footnote-41) 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 می­باشد اشاره کرد. اما این کدبندی ها اغلب برای داده هایی به کار می­روند که در زبان های طبیعی دارای معنی و مفهوم می­باشند 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 می­کنند بنابر این برای نمایش خروجی این تابع ها به صورت رشته های حرفی عددی نیز استاندارد های خاصی نظیر Hex, Base64, Base58 ایجاد شده است که Hex در حقیقت نمایش هر بایت از داده در مبنای 16 است که باعث طولانی شدن رشته می­شود بنابر این استاندارد Base64 به طور معمول برای انتقال داده ها استفاده می­شود 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 انسانها و نه ماشین ها مورد بررسی قرار گیرند مشکلاتی را ایجاد می­کنند که برای پیشگیری از این مشکلات از استاندارد Base58 در اغلب کریپتو کارنسی ها استفاده می­شود که حروف و اعداد مشابه و غیر قابل تشخیص از یکدیگر در آن مشاهده نمی­شوند.

## شبکه نظیر به نظیر

این شکبه ها در حقیقت در مقابل شبکه های سرویس دهنده-سرویس گیرنده [[42]](#footnote-42) قرار دارند و به جای آنکه 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 نظیر ها [[43]](#footnote-43) 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 ‏2‑2 : نمونه یک شبکه سرویس گیرنده-سرویس دهنده



تصویر ‏2‑3 : نمونه یک شبکه نظیر به نظیر

## ماشین مجازی [[44]](#footnote-44)

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 این ماشین های مجازی آنهایی هستند که در زبان های برنامه نویسی سطح بالا مانند جاوا [[45]](#footnote-45) و یا زبان های خانواده دانت نت [[46]](#footnote-46) به ترتیب با نام های JVM [[47]](#footnote-47)و CLR [[48]](#footnote-48) استفاده می­شوند . این ماشین های مجازی اغلب مانند ریز پردازنده ها [[49]](#footnote-49) دارای یک زبان اسمبلی مانند [[50]](#footnote-50) مخصوص به خود می­باشند که کد های زبان سطح بالاتر به این زبان ها ترجمه می­شوند و سپس کد های خروجی حاصل در ماشین مجازی مورد نظر اجرا می­شوند در حقیقت این ماشین های مجازی رفتار شبیه به یک پردازنده را شبیه سازی می­کنند.



تصویر ‏2‑4 : نحوه کارکرد CLR

اغلب رمز ارزها نیز برای انجام کارهای خود دارای یک ماشین مجازی می­باشند که تراکنش ها مورد نظر به صورت کد های این ماشین مجازی در آمده و برای تایید کد های مورد نظر اجرا می­شوند به طور مثال می­توان از EVM [[51]](#footnote-51) در این زمینه نامبرد.

# فصل سوم : بلاک چین

همانطور که گفته شد در سال 2009 میلادی با معرفی بیت کوین به عنوان اولین رمز ارز توسط فرد یا افرادی با نام مستعار ساتوشی ناکاماتو [[52]](#footnote-52) مفهوم رمز ارزها پابه عرصه وجود گذاشتند به طوری که امروزه بیش از بیش از 1000 رمز ارز مختلف در دنیا وجود دارد ، بیت کوین برخلاف ارز دیجیتال قبل از خود توانست مشکل نیاز به طرف سوم قابل اعتماد را با کمک یک مکانیزم انقلابی به نام بلاک چین حل کند که در این فصل به بررسی این سیستم می­پردازیم.

در حقیقت بلاک چین را می­توان به مانند یک سیستم عامل در نظر گرفت و رمز ارزهارا به عنوان برنامه هایی که بر روی این سیستم عامل بزرگ و توزیع شده اجرا می­شوند.

## تعریف مفهومی بلاک چین

بلاک چین یک دفتر کل بزرگ و توزیع شده می­باشد که کار ثبت و رهگیری دارایی هارا به صورت تراکنش [[53]](#footnote-53) محور انجام می­دهد به طوری که امکان تغییر یا حذف اطلاعات درج شده در این دفتر کل وجود ندارد و فقط می­توان اطلاعات جدیدی را به آن افزود ، بلاک چین كار فرآيند ثبت تراكنشها و ردگيري داراييها را در يك شبكه­ي كسب­وكار ساده ميكند. يك دارايي ميتواند ملموس مانند خانه، خودرو، پول نقد، زمين و يا ناملموس مانند مالكيت معنوي نظیر حق اختراع، حق چاپ يا نام اعتباري باشد. تقريباً هرچيز ارزشمندي ميتواند در يك شبكهي بلاک چین ردگيري و معامله شود و مخاطرات و هزينه­ها را براي همه­ي طرفهاي درگير كاهش دهد.

## تعریف ساختاری بلاک چین

در ساده ترین حالت ساختار بلاک چین را همانطور که از نامش پیداست می­توان به صورت زنجیره ای از بلاک های داده ای دانست که به طور متوالی پشت یکدیگر قرار می­گیرند و هر بلاک به طوری به بلاک قبلی خود وابسته است.



تصویر ‏3‑1: شمای کلی یک بلاک چین

## ساختار هر بلاک

باتوجه به اینکه بلاک چین از زنجیره ای از بلاک ها ساخته شده است بنابر این ضروری به نظر می­رسد که ابتدا به بررسی ساختار هر بلاک به صورت جز به جز بپردازیم تا در ادامه بتوانیم کاربرد هر یک از این بخش هارا در یک بلاک چین بررسی کنیم.

هر بلاک به طور معمول حداقل دارای 3 بخش زیر است:

1. بخش داده ای
2. نماد بلاک
3. نماد بلاک قبل از خود

حال به بررسی هر کدام از این بخش ها می­پردازیم:

### بخش داده ای 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 نذیر یک مقدار متغییر ، زمان ساخت بلاک ، شماره بلاک و ... که در رمز ارزها استفاده می­شود.

### نماد یا امضا بلاک

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

### 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 بلاک ها به یکدیگر می­شود.

## بررسی ساختار بلاک چین

همانطور که گفته شد بلاک چین زنجیره ای از بلوک های داده ای به هم وابسته است اما این بلاک ها چگونه به هم وابسته می­شوند؟

در حقیقت وظیفه ایجاد وابستگی بلاک های یک بلاک چین فارغ از بخش داده ای و بخش های دیگر آن بر عهده بخش شامل 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 بلاک های قبل از خود وابسته است 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 جدید ترین بلاک انتشار می­یابد ،حال اگر بلاک چین یک بلاک چین پویا باشد به طوری که پیوسته به بلاک های آن افزوده شود عملا ایجاد تغییر در یکی از بلاک های 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 داده های داخل بلاک چین توافق دارند.



تصویر ‏3‑2: ساختار بلاک چین

## خواص بلاک چین

هر بلاک چین باید شامل خواص زیر باشد:

1. تغییر ناپذیری: تغییر ناپذیری در بلاک چین باتوجه به ساختار آن به وجود می­آید.
2. اصل بودن : مشاركت كنندگان ميدانند دارايي از كجا ميآيد و مالكيت آن در طول زمان چگونه تغییر کرده است.
3. قطعیت : هر بلاک دارای قطعیت است در حقیقت داده های آن قطعی و صحیح و پذیرفته شده هستند.
4. شفافیت : نحوه ساز و کار بلاک چین دارای شفافیت است و داده های موجود در آن برای همه افراد موجود در شبکه قابل دسترسی است.

## بررسی کارکرد شبکه های بلاک چین

اغلب شبکه های مبتنی بر بلاک چین مانند رمز ارز ها بر بستر شبکه های کامپیوتری نظیر به نظیر به فعالیت می­پردازند به طوری که هر یک از نظیر های موجود در شبکه که اینجا ند [[54]](#footnote-54) 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 ،ندی که آن بلاک را به بلاکچین افزوده است موظف است که این موضوع را به همه نظیر های دیگر اعلام کند تا آنها نیز اطلاعات بلاک چین خود را به روز رسانی [[55]](#footnote-55) کنند تا شبکه دچار خطا نشود.

## انواع بلاک چین

بلاک چین ها دارای انواع مختلفی می­باشند اما به صورت کلی به دو دسته خصوصی و عمومی تقسیم می­شوند که بلاک چین های عمومی اطلاعاتشان در اختیار همه افراد قرار دارد و هر کسی می­تواند در هر زمان به عنوان یک ند جدید وارد این شبکه شود و یا اطلاعات آن­هارا مورد بررسی قرار دهد نظیر بلاک چین های ارز رمز ها، در مقابل نیز بلاک چین های خصوصی قرار دارند که یک سازمان و یا کسب و کار می­تواند برای افراد موجود در شبکه خود آن­را ایجاد کند و بر بستر شبکه های داخلی خود سازمان به اجرای آن بپردازد و یا با رمزگذاری همه داده های آن از دسترسی افراد غیر قابل اعتماد به داده ها جلوگیری به عمل آورد.

## اثبات کار

اثبات کار یا POW در حقیقت یک مکانیزم در شبکه های بلاک چین است که ویژگی تغییر ناپذیری بلاک چین را تقویت می­کند و همچنین از افزوده شدن بلاک های 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 [[56]](#footnote-56) در نظر گرفته می­شود و حال یک شرط جدید برای به دست آوردن بلاک هش به سازوکار شبکه افزوده می­شود به طوری که مقدار عددی هش به دست آمده باید از مقدار سختی کمتر 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 داده های ثابتی تشکیل شده اند که قابل تغییر نمی­باشد بنابر این به ازای داده های 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 داده های بلاک افزوده می­شود این مقدار متغییر فارغ از دیگر بخش های بلاک است و می­تواند هر مقداری داشته باشد حال با کمک این مقدار جدید و تغییر دادن آن و محاسبه مجدد هش بلاک می­توان هش های مختلف برای یک بلاک به دست آورد و درحقیقت مکانیزم POW مشارکت کنندگان را مجبور می­کند تا مکررا به محاسبه مقدار هش های مختلف برای یک بلاک با nonce های مختلف بپردازند تا یک مقدار هش با شرط کوچکتر بودن از سختی یافته شود؛ به محض یافته شدن این مقدار ،ندی 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 ند ها اعلام می­کند و دیگر ند ها نیز حال به سادگی با یکبار هش کردن با nonce اعلام شده می­توانند صحت ادعای ند اعلام کننده را راستی آزمایی کنند و درصورت درست بودن ادعا آن را به عنوان بلاک جدید بپذیرند.

### حمله 51 درصد

با افزوده شدن POW به بلاک چین حال تغییر دادن داده های بلاک بیش از بیش سخت می­شود به طوری که برای تغییر داده های یک بلاک این بار فرد مورد نظر باید به ازای همه بلاک های بعدی نیز الگوریتم POW را نیز اجرا کند و مقدار هشی کمتر از سختی برای همه بلاک ها بیاید، که بسیار زمانگیر خواهد بود همچنین در شبکه های پویا که همواره به بلاک های آن افزوده می­شود ند خاطی باید بتواند به قدری سریع به محاسبه POW بلاک های 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 ند مورد نظر حداقل 51 درصد قدرت محاسباتی شبکه را داشته باشد یعنی قدرت محاسباتی آن از مجموع قدرت محاسباتی همه ند های دیگر موجود در شبکه بیشتر باشد.

### تغییرات سختی

از دیگر کاربرد های POW آن است که شبکه های بلاک چینی شلوغ می­توانند زمان ایجاد شدن بلاک های خود را کنترل کنند تا به این صورت از رشد بی رویه بلاک چین جلو گیری به عمل آورند به این ترتیب که با افزوده شدن هر ند شانس یافته شدن بلاک جدید بیشتر می­شود چرا که ند های بیشتری برای بدست آوردن بلاک جدید به رقابت می­پردازند حال پس از افزوده شدن تعداد زیادی ند می­توان با افزایش دادن مقدار سختی زمان مورد نیاز برای بدست آوردن بلاک جدید را بیشتر کرد و درحقیقت انجام مکانیزم POW را سخت تر نمود و زمان میانگین برای به دست آمدن تعداد معینی ای بلاک را کنترل نمود.



تصویر ‏3‑3: نمودار تغییرات سختی شبکه بیت کوین

## استخراج [[57]](#footnote-57) رمز ارز

در شبکه های بلاک چینی رمز ارزهایی نظیر بیت کوین که تعداد زیادی ند برای بدست آوردن بلاک بعدی رقابت می­کنند مقداری از توکن های شبکه به عنوان جایزه [[58]](#footnote-58) به هر فردی که موفق به یافتن بلاک بعدی شود اختصاص داده می­شود تا به این صورت انگیزه برای پیاده کردن بلاک های 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 کوین های شبکه افزوده می­شود و در حقیقت مکانیزمی مانند استخراج کردن ایجاد می­شود.

به طور مثال برای شبکه بیت کوین در ابتدا جایزه هر بلاک 50 بیت کوین در نظر گرفته شده بود که به ازای ایجاد شدن هر 210 هزار بلاک این میزان نصف می­شود تا در نهایت به 0 میل کند به طوری که پیش بینی می­شود در نهایت تعدادی نزدیک به 21 ملیون بیت کوین وجود خواهند داشت که این مقدار تا سال 2140 میلادی کاملا استخراج می­شود ، لازم به ذکر است که با افزایش تعداد ند ها به مرور زمان سرعت به استخراج بیت کوین ها افزایش نمی­یابد چرا که همانطور که اشاره شد مقدار سختی مکررا افزوده می­شود به طوری که تقریبا هر 210 هزار بلاک زمانی معادل 4 سال برای ایجاد شدن نیاز خواهند داشت.

## دیگر اثبات ها[[59]](#footnote-59)

علی رغم تمام مزایای اثبات کار این مکانیزم دارای مشکلاتی نیز می­باشد یکی از بزرگترین ایرادات مکانیزم POW مصرف مقدار زیادی انرژی توسط کامپیوتر های ند هایی است که به اجرای آن می­پردازند به طوری که گفته می­شود به طور مثال شبکه بیت کوین در حال حاظر انرژی الکتریکی به اندازه انرژی مورد نیاز برای کل کشور ایرلند می­باشد، درحالی که این انرژی برای امن کردن شبکه استفاده می­شود اما تعداد زیادی هش تولید شده برای هر بلاک بی استفاده است و عملا 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 عملا امکان پذیر نمی­باشد بنابر این تعداد اثبات دیگر نیز برای ارز رمز ها مطرح شده است که در اینجا به معرفی برخی از آنها می­پردازیم.

### اثبات سهام [[60]](#footnote-60) (POS)

در این روش یکی از ند های موجود در شبکه به طور تصادفی برای ایجاد بلاک جدید توسط شبکه انتخاب می­شود اما با این شرط که افرادی که دارای سهم بیشتری از توکن های شبکه می­باشند شانس بیشتری برای انتخاب شدن دارند، که داشتن تعداد زیادی از توکن های شبکه مانع از تقلب فرد مورد نظر می­شود چرا که اگر این ند اقدام به تقلب کند ارزش توکن های شبکه که خود مالک تعداد زیادی از آنها است کاهش می­یابد.

### اثبات حافظه [[61]](#footnote-61)

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 آن ها شانس استفاده از مدارهای مجتمی محاسباتی درآنها کمتر است.

### گواهی سوزاندن [[62]](#footnote-62)

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

همچنین برخی روش های دیگر نیز که گاها ترکیبی از روش های فوق است نیز استفاده می­شود.

## بلاک چین های نسل دوم [[63]](#footnote-63)

همانطور که پیش­تر اشاره شد یکی از بخش های ارز رمز ها و برنامه های مبتنی بر بلاک چین یک ماشین مجازی است که به اجرای برخی دستورات شبه اسمبلی مختص آن ارز رمز و بلاک چین می­پردازد. این زبان شبه اسمبلی در نسخه های اولیه ارز رمز ها شامل دستورات بسیار محدودی می­شد و به اصطلاح دارای خاصیت کامل بودن تورینگ [[64]](#footnote-64) نبود، به این معنی که دارای دستورات پرش نبود اما با گذر زمان برخی از فعالان حوزه ارز رمز با بررسی پتانسیل های نهفته در این زبان، به استفاده از دستورات کامل تر و زبان های کامل تورینگ روی آوردند که به این ترتیب نسل دوم بلاک چین ها شروع شدند، یکی از نمونه های ارز رمز های با بلاک چین نسل دوم اتریوم [[65]](#footnote-65)می­باشد.

### قرار داد های هوشمند [[66]](#footnote-66)

قرار داد های هوشمند برخلاف آنچه که ممکن است از نام ظاهری آن­ها برداشت شود، قطعه کد هایی با زبان های کامل تورینگ ماشین های مجازی ارز رمز ها هستند که به ارسال یک تراکنش به سمت آن­ها قادر به انجام یک کار مشخص و ثبت نتیجه آن در بلاک چین می­باشند. بنابر این، این قطعه کد ها 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.

## قاعده طولانی ترین زنجیره [[67]](#footnote-67)

همانطور که گفته شد در بلاک چین های شلوغ مانند بلاک چین های رمز ارز ها تعداد زیادی ند برای به دست آوردن بلاک بعدی با یکدیگر به رقابت می­پردازند و هر ندی که بتواند زودتر از بقیه ند ها پاسخ مسئله POW را بیابد به عنوان برنده برای استخراج آن بلاک معرفی می­شود اما اگر 2 ند همزمان موفق به این کار شوند چه می­شود؟ در این حالت هر 2 ند در یک زمان پاسخ درستی برای POW بدست آورده اند و هر دو می­توانند بلاک خود را به انتهای بلاک چین بی افزایند که این اتفاق در عمل باعث 2 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 ارز ها برای چنین حالتی آن است که ابتدا میگذارند هر 2 بلاک به انتهای بلاک چین افزوده شود چرا که در مرحله اولیه هر دو بلاک برابر هستند اما با گذر و افزوده شدن بلاک های بعدی در انتهای این 2 بلاک 2 شاخه متفاوت از بلاک چین ایجاد می­شود که در نهایت همه ند ها طولانی ترین شاخه ایجاد شده را به عنوان شاخه درست می­پذیرند و کار کردن بر روی شاخه دیگر را رها می­کنند.



تصویر ‏3‑4 : قاعده طولانی ترین شاخه

همانطور که در تصویر 3-4 دیده می­شود در هر مرحله پس از 2 شاخه شدن بلاک چین بلاک چین زنجیره سیاه که طولانی تر بوده است را به عنوان زنجیره اصلی انتخاب کرده و بلاک های بنفش راه شده اند، به این بلاک های رها شده در اصطلاح یتیم شده [[68]](#footnote-68) می­گویند.

## فورک [[69]](#footnote-69) در بلاک چین

شبکه های بلاک چین نیز مانند دیگر برنامه های کامپیوتری هموراه دستخور تغییرات و به روز رسانی هایی می­شوند اما با توجه به این که این شبکه ها کار حساس تری را نسبت به دیگر برنامه ها 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 بلاک هایی با قواعد جدید شوند که با بلاک های قبل از آن تفاوت هایی دارد به چنین حالتی فورک در بلاک چین گفته می­شود. در چنین حالتی ند هایی که نسخه نرم افزار خود را به روز رسانی [[70]](#footnote-70) کرده اند بلاک هایی با خصوصیات متفاوتی از ند هایی که به روز رسانی را انجام نداده اند تولید می­کنند.

### فورک نرم [[71]](#footnote-71)

فورک نرم زمانی رخ می­دهد که تغییرات ایجاد شده اساسی نباشند و بنابر این علی رغم این که 2 نسخه مختلف از بلاک ها در حال ایجاد شد هستند اما این بلاک ها می­توانند بدون رخ دادن مشکل خاصی پشت یکدیگر قرار بگیرند و شبکه کلی دچار مشکل نمی­شود بنابر این ند هایی که به روز رسانی کرده اند و ند هایی که به روز رسانی نکرده اند می­توانند کار کردن بر روی یک بلاک چین رو ادامه دهند.

### فورک سخت [[72]](#footnote-72)

این نوع فورک زمانی رخ می­دهد که تغییرات مذکور بسیار اساسی باشند به طوری که بلاک های تولید شده با هر نسخه نمی­توانند کنار یکدیگر قرار گیرند و از نقطه ایجاد به روز رسانی عملا بلاک چین به 2 بلاک چین مجزا تبدیل می­شود در چنین حالتی ند هایی که به روز رسانی را دریافت کرده اند روی یک بلاک چین و ند هایی که به روز رسانی را دریافت نکرده اند روی یک بلاک چین دیگر کار می­کنند که به هیچ عنوان سازگار [[73]](#footnote-73) نمی­باشند و عملا در چنین حالتی ممکن است توکن های ارز رمز به 2 توکن متفاوت تجزیه شوند، به طور مثال پس از یکی از اپدیت های شبکه بیت کوین در سال 2016 میلادی با ایجاد فورک سخت توکن های شبکه به 2 نوع بیت کوین معمولی و بیت کوین کلاسیک [[74]](#footnote-74) تقسیم شدند به طوری که دارندگان کیف پول هریک نمی­توانند توکن های نوع دیگر را داشته باشند.



تصویر ‏3‑5: نمونه فورک سخت در بلاک چین

همانطور که در تصویر 3-5 مشاهده می­شود پس از یک فورک سخت بلاک چین به 2 بلاک چین متفاوت تبدیل می­شود و برخلاف حالت طولانی ترین زنجیره اینجا هر 2 زنجیره قابل قبول هستند اما نه توسط همه ند ها بلکه برخی ند ها یک زنجیره و برخی ند ها زنجیره دیگر را قبول کرده و روی زنجیره مورد نظر خود مشغول به کار می­شوند.

## مشکلات بلاک چین ها

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

یکی از ایرادات این شبکه به دلیل مکانیزم POW و مصرف انرژی بیش از حد رخ می­دهد 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 نمی­باشند.

در بسیاری از بلاک چین ها امکان پرداخت های با مبلغ بسیار کم وجود ندارد[[75]](#footnote-75) که این خود می­تواند ناشی از عوامل مختلفی باشد، به طور مثال در شبکه بیت کوین برای هر تراکنش مبلغی نیز به عنوان هزینه تراکنش [[76]](#footnote-76) دریافت می­شود که اگر این مبلغ کمتر از حد معینی باشد ماینر ها رقبتی به پردازش آن تراکنش نخواهند داشت حال اگر فردی بخواهد تراکنشی با مبلغی کمتر از هزینه آن انجام دهد عملا غیر منطقی می­باشد.

یکی از مهمترین مشکلات نیز مشکل مقیاس پذیری [[77]](#footnote-77) در شبکه های بلاک چینی 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 اطلاعاتی که بلاک می­تواند ذخیره کند وجود دارد باعث می­شود عملا در هر زمان بیش از تعداد معینی تراکنش قابل پردازش نباشند به طور مثال در شبکه بیت کوین تقریبا در هر ثانیه فقط 7 ترنزکشن 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 تراکنش ها افراد مجبور می­شوند پیوسته هزینه پردازش تراکنش بیشتری پرداخت کنند تا استخراج کنندگان [[78]](#footnote-78) رغبت بیشتری به پردازش ترکنش آن­ها داشته باشند.

### هش گراف [[79]](#footnote-79)

باتوجه به مشکل مقیاس پذیری بسیاری از فعالات حوزه بلاک چین تلاش هایی برای حل این مشکل کردند یکی از راه حل های پیشنهاد شده برای مشکل مقیاس پذیری ایجاد هش گراف ها به جای هش چین ها یا همان بلاک چین ها 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 پرداخت های با مقادیر بسیار کم نیز وجود خواهد داشت.



تصویر ‏3‑6: مقایسه هش گراف با بلاکچین

همانطور که در تصویر 3-6 مشاهده می­شود در هش گراف به جای استفاده از قاعده طولانی ترین زنجیره سعی بر آن است تا بلاک های یتیم شده به صورتی مجدادا به شبکه بازگردانده شوند در این حالت به جای یک زنجیره یک گراف ایجاد می­شود که ند های آن با هش به یکدیگر مرتبط می­باشند.

## بلاک چین های نسل سوم [[80]](#footnote-80)

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 می­آورند. همچنین کاربرد دیگر آنها در موارد مرتبط به اینترنت اشیا [[81]](#footnote-81) می­باشد یکی از نمونه های موفق پیاده سازی بلاک چین های نسل سوم IOTA می­باشد.

### 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 می­شود تا تراکنش های بعدی به صورت مستقیم به تراکنش های قبل از خود متصل شوند، به طور دقیق تر هر تراکنش به 2 تراکنش تصادفی از تراکنش های قبلی خود متصل می­شود و بنابر این یک گراف از تراکنش ها ایجاد می­شود که این گراف در حقیقت یک گراف جهت دار رو به جلو در زمان خواهد بود چرا که هر تراکنش فقط به تراکنش های قبل از خود در زمان می­تواند متصل شود بنابر این یال های متصل کننده این تراکنش ها در گراف جهت دار هستند و همچنین به دلیل جهت دار بودن امکان ایجاد دور [[82]](#footnote-82) نیز در این گراف موجود نمی­باشد از این رو به آنها گراف های جهت دار بدون دور [[83]](#footnote-83) ویا به اختصار DAG گفته می­شود.



تصویر ‏3‑7: نمونه یک DAG

### پردازش تراکنش ها در بلاک چین های نسل سوم

در این نوع بلاک چین ها 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 2 تراکنش قبل از خود را تایید کند و پس از تایید صحت آن 2 تراکنش، تراکنش خود را به انتهای آن­ها می­افزاید به این صورت ساختار DAG گفته شده از اتصال هر تراکنش به 2 تراکنش قبل از خود به وجود می­آید به این ترتیب و به دلیل عدم وجود هزینه تراکنش تراکنش های با هر مقدار ناچیز قابلیت افزوده شدن به شبکه را دارا می­باشند. همچنین به افزایش تعداد تراکنش ها مشکل ایجاد ترافیک برای تراکنش ها رخ نمی­دهد چرا که افزایش تعداد تراکنش ها به معنی افزایش تعداد ند های فعال در شبکه است که خود موظف به تایید تراکنش های قبل از خود می­باشند، بنابر این همواره با افزایش تعداد تراکنش ها تعداد تایید کنندگان تراکنش نیز افزایش می­یابد و شبکه دچار کندی نمی­شود.



تصویر ‏3‑8 : وضعیت تراکنش ها در یک DAG

همانطور که در تصویر 3-8 نشان داده شده است تراکنش ها در این نوع شبکه ها به 3 نوع مختلف تقسیم می­شوند تراکنش های کاملا تایید شده تراکنش های در دست تایید و تراکنش های جدید که به ترتیب با رنگ های سبز قرمز و خاکستری در تصویر قابل مشاهده می­باشند.

تراکنش های تایید شده تراکنش هایی 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 تراکنش ها سابقیه کاملی از تراکنش های قبلی را در خود دارند، تراکنش های تایید نشده که به رنگ قرمز هستند فقط به تعدادی از تراکنش های جدید متصل اند و تراکنش های جدید تراکنش هایی هستند که در انتظار افزوده شدن تراکنش جدید به انتهای خود و پردازش شدن توسط یک ند می­باشند.

## بلاک چین های نسل چهارم [[84]](#footnote-84)

این نوع بلاک چین ها که بلاک چین های هوشمند نیز معروفیت دارند در حقیقت تلاشی برای ایجاد اتصال میان حوزه های هوش مصنوعی [[85]](#footnote-85) و به خصوص یادگیری عمیق [[86]](#footnote-86) با بلاک چین می­باشند از پروژه های معروف بلاک چین های نسل چهارمی می­توان به SingularityNet و Deep Brain Chain اشاره کرد.

## بلاک چین در خارج از رمز ارزها

همانطور که پیشتر گفته شد بلاک چین فقط مختص به رمز ارزها نمی­شود و قابلیت های این تکنولوژی انقلابی می­تواند 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 می­پردازیم.

كارخانه­هاي سازنده­ي خودرو خريد قسطي خودرو را ساده جلوه ميدهند، اما اين كار در واقعيت ميتواند كاملا پيچيده باشد. چالش مهمي كه اين روزها شبكه­هاي خريد قسطي خودرو با آن روبهرو هستند اين است كه هرچند زنجیه تامین فیزیکی معمولا یک­پارچه است ، اما هر مشارکت کننده در شبکه و سامانه های پشتیبان از هم جدا هستند که هرکدام از این­ها 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 خودرو ها 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 مراجعه بسیار سخت صورت می­گیرد و گاها بسیار دردسر ساز است اما درصورت استفاده از بلاک چین بسیاری از مشکلات حل می­شوند. به طوری که هریک از مشارکت کنندگان می­توانند 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.



تصویر ‏3‑9: رهگیری مالکیت خودرو بدون بلاک چین



تصویر ‏3‑10: رهگیری مالکیت خودرو با بلاک چین

# فصل چهارم : بررسی کارکرد چند رمز ارز مهم

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

## بیت کوین

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

بیت کوین برای اولین بار در سال 2009 میلادی برای اولین بار توسط شخص یا اشخاص ناشناسی با نام ساتوشی ناکاماتو معرفی شد، معرفی این ارز رمز به صورت چند برگ مقاله با نام وایت پیپر [[87]](#footnote-87) معرفی شد در این مقاله به تشریح سازوکار این ارز رمز پرداخته شده است.

### ساختار بلاک چین بیت کوین

بلاک چین بیت کوین ساختاری مانند بلاک چین های فصل قبل دارد در اصل در اینجا باید به بررسی دقیق تر بخش داده ای هر بلاک بپردازیم. بخش داده ای هر بلاک بیت کوین حاوی اطلاعات تعدادی از تراکشن های انتقال این رمز ارز می­باشد به طوری که حجم هر بلاک به 10 کیلو بایت [[88]](#footnote-88) محدود می­باشد بنابر این تعداد تراکشن های موجود در هر بلاک محدود می­باشد. همچنین برای سرعت دادن به محاسبات POW به جای استفاده از کل اطلاعات ترنزکشن در محاسبه هش بلاک فقط از ریشه درخت مرکله تراکنش های بلاک استفاده می­شود.



تصویر ‏4‑1 : ساختار بلاک های بیت کوین

### تراکنش ها در بیت کوین

یکی از مهمترین اجزای تشکیل دهنده شبکه بیت کوین تراکنش ها هستند، در حقیقت خورد ترین بخش سازنده شبکه بیت کوین این تراکنش های 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 ‏4‑2 : ساختار تراکنش های بیت کوین

### ساختار UTXO [[89]](#footnote-89) در بیت کوین

در بیت کوین برای رهگیری مالکیت توکن های 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 درنظر گرفته نمی­شود، بلکه در این رمز ارز به محض دریافت یک تراکنش توسط یک نفر خروجی های 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.

### کیف پول بیت کوین

شبکه بیت کوین دارای تعداد مشخصی کیف پول می­باشد که در حقیقت این کیف پول ها وظیفه حفظ و نگه داری دارایی های هر فرد را دارند، در حقیقت در عمق هرکدام 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 ارز رمز می­باشد، این کیف پول در حقیقت نمونه کامل از پیاده سازی پروتوکل [[90]](#footnote-90) بیت کوین می­باشد این کیف پول برای کار کردن ابتدا لازم است تا نسخه کاملی از بلاک چین بیت کوین را که در حال حاظر تقریبا حجمی معادل 185GB دارد را دانلود نماید، یکی از قابلیت های این کیف پول امکان پیوستن به شبکه استخراج کنندگان توسط آن می­باشد.

## اتریوم

این ارز رمز نمونه موفق از پیاده سازی بلاک چین های 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 کد های این ماشین شبکه اتریوم امکان اجرای کد هایی به پیچیدگی کد های همه دیگر زبان های برنامه نویسی به صورت توزیع شده روی همه کامپیوتر های اجرا کننده شبکه اتریوم می­دهد.

### بررسی ساختار EVM

ماشین مجازی اتریوم و یا EVM در حقیقت یک ماشین مجازش مبتنی بر پشته [[91]](#footnote-91) می­باشد به این صورت که همه دستورات آن فقط مقداری را از پشته خوانده و مقداری را به پشته می­افزایند، سایز این پشته به ازای هر داده برابر با 256 بیت[[92]](#footnote-92) 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 کد ها وجود دارد.



تصویر ‏4‑3 : ساختار EVM

### زبان Solidity

همانند دیگر زبان های اسمبلی برای زبان اسمبلی EVM نیز چندین زبان سطح بالاتر ایجاد شده است که کد های این زبان ها به کد های اسمبلی EVM ترجمه [[93]](#footnote-93) می­شود. یکی از زبان های سطح بالاتر شبکه اتریوم زبان Solidity نام دارد که با کمک این زبان به راحتی می­توان به نوشتن قرار داد های هوشمند پرداخت.

### برنامه های توزیع شده [[94]](#footnote-94)

به برنامه های نوشته شده به زبان سالیدیتی در اصطلاح برنامه های توزیع شده گفته می­شود چرا که این برنامه ها درحقیقت نه بر روی یک کامپیوتر بلکه برروی همه کامپیوتر هایی که در شبکه اتریوم فعالیت می­کنند اجرا می­شوند و هر ند در شبکه می­تواند توابع موجود در این برنامه هارا فراخوانی و استفاده نماید. البته باید توجه داشت که مفهوم برنامه های توزیع شده مختص شبکه اتریوم نمی­باشند و این مفهوم از مدت ها قبل نیز وجود داشته است اما یکی از بزرگترین مشکلات این برنامه ها بحث توافق می­باشد که با کمک شبکه بلاک چین این مشکل حل می­شود.

### توکن های ERC20

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

### ساختار کیف پول های اتریوم

در ساختار شبکه اتریوم به جای مکانیزم UTXO استفاده شده در بیت کوین از مکانیزم ماشین حالت[[95]](#footnote-95) استفاده می­شود به این ترتیب که هر کیف پول در این شبکه معادل یک حساب کاربری [[96]](#footnote-96) 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 حساب های کاربری حالت سیسم گفته می­شود که با پردازش هر تراکنش این سیستم از یک حالت به حالت دیگری می­رود.

### شبکه Ropsten

باتوجه به امکان برنامه نویسی برای شبکه اتریوم و نیاز این برنامه ها به کار کردن به اتریوم علاوه بر شبکه اصلی اتریوم که توکن های آن دارای ارزش می­باشند، شبکه های دیگیری نیز برای آن ایجاد شده است که در حقیقت این شبکه ها برای تست و خطایابی قرار داد های هوشمند می­­باشند، ساختار این شبکه ها مانند شبکه اصلی اتریوم می­باشد با این تفاوت که توکن های آن­ها ارزش ندارند یکی از معروف ترین این شبکه ها Ropsten نام دارد.

### Web3.js

زبان Solidity علی رغم داشتن توانایی برای نوشتن هر نوع قرار داد هوشمند و برنامه ای قابلیت ایجاد رابط کاربری گرافیکی [[97]](#footnote-97) را دارا نمی­باشد در نتیجه کتابخانه های زیادی برای زبان های مختلف برنامه نویسی برای کار با آن ایجاد شده است، یکی از معروف ترین این کتابخانه ها که به زبان جاوا اسکریپت نوشته شده است Web3.js نام دارد که با کمک زبان جاوا اسکریپت و زبان های طراحی Html و Css امکان ایجاد رابط های کاربری تحت وب برای قرار داد های هوشمند را فراهم می­آورد.

## آیوتا

ایوتا یک نمونه از پیاده سازی بلاک چین های نسل سومی است که درسال 2016 میلادی پا به عرصه وجود نهاد این ارز رمز در حقیقت به منظور مناسب بودن برای اینترنت اشیا ایجاد شده است و از پرداخت های ریز به خوبی پشتیبانی می­کند در این شبکه به DAG ایجاد شده از تراکنش های Tangle گفته می­شود.

### توکن های IOTA

باتوجه به اینکه در IOTA ساختار استخراج وجود ندارد بنابر این همه توکن های شبکه IOTA در ابتدای ایجاد شدن آن به صورت آماده وجود داشتند و توسط یه عرضه عمومی در میان افراد مختلف پخش شده اند.

### نظیر یابی در IOTA

در شبکه IOTA عملیات نظیر یابی برخلاف بیت کوین و اتریوم نه به صورت خودکار بلکه به صورت دستی انجام می­شود در حقیقت در این شبکه افراد برای متصل شدن به نظیر ها و اتصال به شبکه باید به سایت IOTA مراجعه کرده و از پایگاه داده موجود در سایت آدرس نظیر های 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 تراکنش ها به تراکنش های دیگر متصل می­شوند هر فرد باید به همسایه های مورد اعتماد متصل شود، همچنین باتوجه به ساختار بلاک چین های نسل سومی برخلاف دیگر بلاک چین 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 نظیر ها از بین ­می­رود.

### امکان جدا شدن و پیوستن تعدادی از ند ها در IOTA

یکی از قابلیت های منحصر به فرد IOTA امکان جدا شدن تعدادی از ندها از شبکه اصلی و مجدادا پیوستن آن­ها به شبکه می­باشد در این حالت در مقطعی از زمان تعدادی از ند ها 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 تراکنش های شبکه اصلی به تراکنش های خود مجدادا به شبکه بپیوندند.



تصویر ‏4‑4: جدا شدن و اتصال مجدد

# فصل پنجم : پارسی کوین

پارسی کوین [[98]](#footnote-98) در حقیقت نمونه یک پیاده سازی از بلاک چین های نسل سومی با زبان C# و با تکنولوژی .Net Standard می­باشد 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 سامانه می­پردازیم.

## حالت سیستم

در این ارز رمز نیز به مانند ارز رمز اتریوم از ریشه درخت مرکله تمام حساب های کاربری سامانه حالت سیستم به دست می­آید که با افزوده شدن هر تراکنش جدید سیستم از حالت n-1 به حالت n می­رود. که این امر مستلزم به روز رسانی درخت مرکله می­باشد 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 کوچی می­باشد به طور مثال در صورت وجود یک ملیوم حساب کاربری در سامانه به تعداد تقریبی 20 عملیات برای به روز رسانی حالت سیستم نیاز خواهد بود.

## حساب کاربری

هر حساب کاربری در پارسی کوین شامل موارد زیر می­باشد.

1. کلید خصوصی دارنده حساب
2. کلید عمومی دارنده حساب
3. دارایی حساب
4. مقدار هش حساب

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 تراکنش ها و امضای آن­هارا نیز برعهده دارند.

قطعه کد ‏5‑1: تابع سازنده تراکنش در حساب

public Transaction TransactionBuilder(string reciepient, double value, string message = "")

{

if (value > Balance) throw new Exception("Not enough funds.");

var t = new Transaction(reciepient, value, signtureProvider, message);

if (t.ISSigntureVerified()) return t;

throw new Exception("Something went wrong, cannot sign the transaction");

}

## ساختار تراکنش های در پارسی کوین

ساختار تراکنش های در پارسی کوین شامل موراد زیر می­باشد:

1. کلید عمومی فرسنتده/ایجاد کننده تراکنش
2. کلید عمومی دریافت کننده تراکنش
3. میزان تراکنش
4. زمان ایجاد تراکنش
5. زمان ثبت تراکنش در شبکه
6. مقدار هش تراکنش
7. پیام تراکنش به صورت اختیاری
8. آدرس تراکنش در DAG
9. امضا دیجیتال ایجاد کننده تراکنش
10. قطعه کد های لازم برای تایید امضا دیجیتال تراکنش

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

قطعه کد ‏5‑2: محاسبه هش تراکنش

public string ComputeObjectHash()

=> $"{TransactionIssuer}-{Reciepient}-{Amount}-{IsuueTime}".ComputeHashString();

در این سیستم تراکنش ها پس از ایجاد و امضا شدن و قرار گرفتن در انتهای 2 تراکنش قبلی به ند های همسایه ارسال می­شوند تا توسط یک الگوریتم به مانند تئوری سخن چینی [[99]](#footnote-99) در میان تمام ند های شبکه پخش شوند. همچنین هر ند پس از دریافت تراکنش ابتدا موظب به تایید هویت امضا آن است که توسط ماشین مجازی این سیستم انجام خواهد شد.

قطعه کد ‏5‑3 : ایجاد تراکنش

public Transaction(string reciepient, double amount, ECDSA ec, string message = "")

{

TransactionIssuer = ec.ExportPubKey;

Reciepient = reciepient;

Amount = amount;

IsuueTime = DateTime.UtcNow;

TxHash = ComputeObjectHash();

Signture = ec.Sign(TxHash);

ScriptPubKey = $"{Signture};{TransactionIssuer}";

ScriptSig = $"{ScriptPubKey};CheckSig;IsOne";

}

## ساختار ند های DAG در پارسی کوین

همانطور که گفته شد DAG در حقیقت یک گراف می­باشد بنابر این، این گراف مانند دیگر گراف ها دارای ند هایی می­باشد، این ندها به منظور نگه داری تراکنش ها ایجاد شده اند اما می­توانند حاوی اطلاعات دیگیری نیز ­باشند. در ساختار پارسی کوین این امکان دیده شده است که ندها بتوانند حاوی هیچ تراکنشی نبوده و فقط دارای یک پیام خاص باشند. ساختار ند ها شامل موارد زیر است:

1. نشانه ند
2. زمان ایجاد شدن ند
3. زمان تایید شدن ند
4. ند قبلی سمت راست
5. ند قبلی سمت چپ
6. هش ند قبلی سمت راست
7. هش ند قبلی سمت چپ
8. پیام
9. تراکنش
10. هش تراکنش
11. کلید عمومی ایجاد کننده ند
12. هش ند
13. حالت سیسم قبل از ایجاد این ند
14. حالت سیستم بعد از ایجاد این ند
15. مقدارد none برای مسئله POW ند
16. تعداد تایید های ند

که مقدار هش ند در این سیستم برابر با هش شده نشانه ند به همراه زمان ایجاد آن و اطلاعات ند های قبلی و همچنین سازنده و تراکنش موجود در ند می­باشد. همچنین حل یک مسئله POW نیز برای تایید و ایجاد هر ند درنظر گرفته شده است که این مسئله POW مقدار سختی بسیار کمتر از بلاک چین های نسل اول و دومی دارد به طوری که به طور معمول با یک کامپیوتر معمولی می­توان در زمان تقریبی حدود 30 ثانیه پاسخ آن­را به دست آورد.

قطعه کد ‏5‑4 : تابع مسئله POW برای هر ند

public string Mine()

{

byte[] s = null;

do

{

s = ComputeObjectHash().ToByteArray(StringEncoding.Base85Check);

} while (NodeHash.ToByteArray().CompareDiff());

return s.ToBase58Check();

}

## کیف پول ها در پارسی کوین

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

## نوسیه بندی ها در پارسی کوین

در این سامانه از 4 نوع نویسه بندی برای انتقال متن ها استفاده شده است:

1. نویسه بندی ASCII برای متن های انگلیسی
2. نویسه بندی UTF-8 برای دیگر متن ها
3. نویسه بندی Base64 برای انتقال متن های بی معنی در دیگر نویسه بندی ها در اجزا داخلی سیستم
4. نویسه بندی Base58Check برای انتقال متن های بی معنی در دیگر نویسه بندی ها برای مواردی که لازم است توسط افراد خوانده شود مانند کلید های عمومی

تفاوت نویسه بندی Base58Check با نویسه بندی Base58 در قرار داشتن یک CheckSum در آن است.

## انواع توابع درهم سازی در پارسی کوین

در این سامانه از 3 تابع درهم سازی MD5 برای موارد معمولی SHA256 و SHA512 برای محاسبه POW استفاده شده است البته حالت اصلی 2 بار محاسبه خروجی این توابع درهم سازی برای امنیت بیشتر می­باشد.

## رمز نگاری در پارسی کوین

در این ارزرمز از 3 سیستم رمزنگاری AES وRSA و ECCاستفاده شده است که از ترکیب AES و RSAبرای نقل و انتقال امن پیام ها استفاده می­شود و از ECC نیز برای ساخت امضا دیجیتال برای ایجاد تراکنش ها. همچنین اطلاعات خصوصی نظیر کلید عمومی هر فرد که برروی سیستم فرد ذخیره می­شود بر روی یک فایل که با AES رمزنگاری شده است نوشته می­شوند تا درصورت به خطر افتادن رایانه فرد این اطلاعات قابل بازگشایی نباشند.

## اجزا [[100]](#footnote-100) سامانه

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

1. کامپوننت Base که کامپوننت پایه برنامه می­باشد و شامل یک سری توابع اساسی که در همه بخش های دیگر مورد استفاده قرار می­گیرد می­باشد.
2. کامپوننت اصلی ParsiCoin که شامل موارد کاری سامانه می­باشد.
3. کامپوننت PVM که شامل ماشین مجازی سیستم می­باشد.
4. کامپوننت CLI که شامل یک کامند لاین [[101]](#footnote-101) برای کار با سامانه است
5. کامپونتت GUI که شامل یک رابط کاربری گرافی [[102]](#footnote-102)وب بیس برای سامانه می­باشد.

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



تصویر ‏5‑1 : ارتبابط بین اجزا در پارسی کوین

### 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 می­باشد.

این کامپوننت از کلاس هایی زیر تشکیل شده است:

1. کلاس [[103]](#footnote-103) EncoderBase58 که وظیفه کد بندی نویسه ها از حالت آرایه ای از بایت ها به Base58 را دارد.
2. کلاس AES که شامل توابع مورد نیاز برای رمزنگاری AES در سیستم می­باشد.
3. کلاس ECDSA که شامل توابع مورد نیاز برای امضا دیجیتال با کمک رمزنگاری ECC می­باشد.
4. کلاس های SecureLine که شامل توابع مورد نیاز برای رمزکردن پیام ها و انتقال آن­ها در بستر اینترنت می­باشد.
5. کلاس استاتیک [[104]](#footnote-104) Util که شامل توابعی برای کد بندی و یا هش کردن می­باشد.
6. کلاس Configuration که شامل پیکر بندی های سیستم می­باشد
7. اینترفیس [[105]](#footnote-105) IPICObject که تمام کلاس های کامپوننت اصلی موظف به پیاده سازی آن­می­باشند.

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

#### کلاس AES

این کلاس که وظیفه انجام رمز نگاری AES را دارا می­باشد دارای 4 سازنده می­باشد که کارکرد آن­ها به شرح زیر است.

1. سازنده اول که یک رشته حرفی [[106]](#footnote-106)از کاربر به عنوان رمز استفاده می­کند.
2. سازنده دوم که از GUID به عنوان رمز استفاده می­کند.
3. سازنده سوم که به مانند سازنده دوم بوده اما GUID گفته شده را خود تولید می­کند.
4. سازنده چهارم که وظیفه انجام دادن کارهای یکسان را داشته و در هر اجرای دیگر سازنده ها فراخوانی می­شود.

همچنین شامل 2 تابع برای رمزگذاری و 2 تابع برای رمزگشایی می­باشد که به ترتیب رشته های حرفی و یا رشته ای از بایت هارا 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 بردار و کلید گفته شده نیز توسط شی System.Security.Cryptography.Rfc2898DeriveBytes از رمز دریافت شده از کاربر توسط سازنده ها ایجاد می­شود.

قطعه کد ‏5‑5: کلاس AES

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));

}

#### کلاس Util

این کلاس شامل توابعی می­باشد که مهمترین های آن­ها وظایف مرتبط با محاسبه سختی را دارا می­باشند، در ساختار پارسی کوین باتوجه به اینکه خروجی توابع هش 64 بایت می­باشد برای سختی 2 مقدار در نظر گرفته شده است مقدار اول تعداد بایت های برابر با 0 می­باشد و مقدار دوم عدد یک بایت بعدی و بقیه بایت ها برابر با 1 درنظر گرفته می­شوند.

قطعه کد ‏5‑6 : مقایسه سختی

public static bool CompareDiff(this byte[] targer)

{

for (int i = 0; i < targer.Length; i++)

{

if (targer[i] > Difficulty[i]) return false;

}

return true;

}

### کامپوننت اصلی ParsiCoin

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

1. کلاس Account شمامل حساب های کاربری
2. کلاس DAG شامل ساختار کلی DAG
3. کلاس MerkleTree شامل درخت مرکله اکانت ها
4. کلاس Node شامل اطلاعات ندها
5. کلاس Services که شامل سرویس های مورد نیاز در سیستم می­باشد.
6. کلاس Transaction شامل اطلاعات تراکنش ها
7. کلاس Wallet شامل اطلاعات کیف پول ها

#### کلاس سازنده درخت مرکله

این کلاس وظیفه ساخت درخت مرکله از اکانت هارا داری می­باشد، ساختار داده موردنظر برای ساخت درخت یک ارایه از رشته های حرفی عددی می­باشد همچنین در این کلاس یک آرایه دیگر نیز به تعداد اکانت ها از کلاس اکانت برای نگه داری اطلاعات کامل اکانت ها درنظر گرفته شده است.

قطعه کد ‏5‑7 : سازنده درخت مرکله

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;

}

}

#### کلاس مرتبط با پایگاه داده

یکی از کلاس های 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 تراکنش ها و اکانت ها را در کامپیوتر هر دارنده سیستم دارا می­باشد.

### کامپوننت PVM

این کامپوننت در حقیقت ماشین مجازی سیستم را دربر می­گیرد که در ادامه در فصلی جداگانه به بررسی آن خواهیم پرداخت.

### کامپوننت CLI

### کامپوننت GUI

## بسته های نرم افزاری استفاده شده

در این سامانه از بسته های نرم افزاری زیر استفاده شده است:

1. بسته نرم افزاری Json.net که برای سرالایز [[107]](#footnote-107) کردن اطلاعات از فرمت Json و به فرمت Json است.
2. بسته LiteDB برای کار کردن با api های LiteDB
3. بسته NbitCoin برای استفاده از توابع مرتبط با امضا دیجیتال ECC موجود در این بسته.
4. بسته NetStandardLibrary به عنوان بسته اصلی دربرگیرنده چهارچوب دات نت

## سورس کنترل [[108]](#footnote-108)

نسخه کاملی از کد برنامه در به صورت بسته های سورس کنترل گیت [[109]](#footnote-109) در سایت گیت هاب و در مسیر <https://github.com/FIVIL/ParsiCoin> موجود می­باشد.

1. Gold Standard [↑](#footnote-ref-1)
2. به طور مثال هر یک ریال ایران برابر یکصدوهشت‌هزاروپنجاه‌وپنج‌ده‌میلیونیم (۰٫۰۱۰۸۰۵۵) گرم [طلای](https://fa.wikipedia.org/wiki/%D8%B7%D9%84%D8%A7) خالص است. [↑](#footnote-ref-2)
3. Fiat Currency [↑](#footnote-ref-3)
4. Credit Card [↑](#footnote-ref-4)
5. Debit Card [↑](#footnote-ref-5)
6. ATM [↑](#footnote-ref-6)
7. POS [↑](#footnote-ref-7)
8. Digital Currency [↑](#footnote-ref-8)
9. Crypto Currency [↑](#footnote-ref-9)
10. Liberty Reserve [↑](#footnote-ref-10)
11. Digital signature [↑](#footnote-ref-11)
12. Proof-of-Work [↑](#footnote-ref-12)
13. Peer-To­-Peer [↑](#footnote-ref-13)
14. consensus [↑](#footnote-ref-14)
15. BitCoin [↑](#footnote-ref-15)
16. Block Chain [↑](#footnote-ref-16)
17. cryptography [↑](#footnote-ref-17)
18. Enigma Machine [↑](#footnote-ref-18)
19. Alan Mathison Turing [↑](#footnote-ref-19)
20. Advanced Encryption Standard [↑](#footnote-ref-20)
21. [Symmetric-key](https://en.wikipedia.org/wiki/Symmetric-key_algorithm) [↑](#footnote-ref-21)
22. Public-key cryptography, or asymmetric cryptography [↑](#footnote-ref-22)
23. Public Key [↑](#footnote-ref-23)
24. Private Key [↑](#footnote-ref-24)
25. Rivest–Shamir–Adleman [↑](#footnote-ref-25)
26. elliptic curve cryptography [↑](#footnote-ref-26)
27. Diffie–Hellman key exchange [↑](#footnote-ref-27)
28. Hypertext Transfer Protocol Secure [↑](#footnote-ref-28)
29. Hash Function [↑](#footnote-ref-29)
30. Quantum computer [↑](#footnote-ref-30)
31. Shor's algorithm [↑](#footnote-ref-31)
32. Quantum cryptography [↑](#footnote-ref-32)
33. Hash Table [↑](#footnote-ref-33)
34. Cryptographic hash function [↑](#footnote-ref-34)
35. Merkle tree [↑](#footnote-ref-35)
36. Binary Tree [↑](#footnote-ref-36)
37. Trie [↑](#footnote-ref-37)
38. Lamport Signature [↑](#footnote-ref-38)
39. Character encoding [↑](#footnote-ref-39)
40. Character [↑](#footnote-ref-40)
41. American Standard Code for Information Interchange [↑](#footnote-ref-41)
42. Client-Server [↑](#footnote-ref-42)
43. Peer [↑](#footnote-ref-43)
44. Virtual Machine [↑](#footnote-ref-44)
45. Java [↑](#footnote-ref-45)
46. .Net [↑](#footnote-ref-46)
47. Jave Virtual Machine [↑](#footnote-ref-47)
48. Common Language Runtime [↑](#footnote-ref-48)
49. CPU [↑](#footnote-ref-49)
50. Assembly [↑](#footnote-ref-50)
51. Ethereum Virtual Machine [↑](#footnote-ref-51)
52. Satoshi Nakamoto [↑](#footnote-ref-52)
53. Transaction [↑](#footnote-ref-53)
54. Node [↑](#footnote-ref-54)
55. Sync [↑](#footnote-ref-55)
56. Difficulty [↑](#footnote-ref-56)
57. Mining [↑](#footnote-ref-57)
58. Block Reward [↑](#footnote-ref-58)
59. Proofs [↑](#footnote-ref-59)
60. Proof Of Stake [↑](#footnote-ref-60)
61. Proof Of Capacity [↑](#footnote-ref-61)
62. Proof Of Burn [↑](#footnote-ref-62)
63. Block chain 2.0 [↑](#footnote-ref-63)
64. Turing completeness [↑](#footnote-ref-64)
65. Ethereum [↑](#footnote-ref-65)
66. Smart Contract [↑](#footnote-ref-66)
67. Longest Chain Is Valid [↑](#footnote-ref-67)
68. Orphaned Blocks [↑](#footnote-ref-68)
69. Fork [↑](#footnote-ref-69)
70. Update [↑](#footnote-ref-70)
71. Soft Fork [↑](#footnote-ref-71)
72. Hard Fork [↑](#footnote-ref-72)
73. Compatible [↑](#footnote-ref-73)
74. Bitcoin Classic [↑](#footnote-ref-74)
75. Micro Payment [↑](#footnote-ref-75)
76. Transaction Fee [↑](#footnote-ref-76)
77. Scalability [↑](#footnote-ref-77)
78. Miners [↑](#footnote-ref-78)
79. HashGraph [↑](#footnote-ref-79)
80. Block Chain 3.0 [↑](#footnote-ref-80)
81. IOT [↑](#footnote-ref-81)
82. Cycle [↑](#footnote-ref-82)
83. Directed Acyclic Graph [↑](#footnote-ref-83)
84. Block Chain 4.0 [↑](#footnote-ref-84)
85. artificial intelligence [↑](#footnote-ref-85)
86. Deep Learning [↑](#footnote-ref-86)
87. White Paper [↑](#footnote-ref-87)
88. Kilo byte [↑](#footnote-ref-88)
89. Unspent Transaction Output [↑](#footnote-ref-89)
90. Protocol [↑](#footnote-ref-90)
91. Stack [↑](#footnote-ref-91)
92. Bit [↑](#footnote-ref-92)
93. Compile [↑](#footnote-ref-93)
94. Dapps [↑](#footnote-ref-94)
95. State Machine [↑](#footnote-ref-95)
96. Account [↑](#footnote-ref-96)
97. GUI [↑](#footnote-ref-97)
98. ParsiCoin [↑](#footnote-ref-98)
99. Gossip [↑](#footnote-ref-99)
100. Component [↑](#footnote-ref-100)
101. Command Line Interface [↑](#footnote-ref-101)
102. Graphic User Interface [↑](#footnote-ref-102)
103. Class [↑](#footnote-ref-103)
104. Static [↑](#footnote-ref-104)
105. Interface [↑](#footnote-ref-105)
106. String [↑](#footnote-ref-106)
107. Serialize [↑](#footnote-ref-107)
108. Source Control [↑](#footnote-ref-108)
109. Git [↑](#footnote-ref-109)