ساخت رمز ارز در بستر بلاک‌چین با پایتون

چکیده

تکنولوژی بلاک‌چین فناوری است که در دهه گذشته بیشترین تأثیر را بر زندگی ما گذاشته است. اولین کلمه‌ای که هنگام صحبت درباره ی بلاک‌چین به ذهن می‌رسد بیت کوین است. بسیاری از افراد هنوز تکنولوژی بلاک‌چین را با کریپتوکرنسی‌ها و بیت‌کوین اشتباه می‌گیرند؛ با این‌حال بیت‌کوین و کریپتوکرنسی تنها یکی از کاربردها و برنامه‌های بلاک‌چین است. این سیستم علاوه بر عدم تمرکز پول و پرداخت‌ها می‌تواند برای ثبت، تأیید و ارسال انواع قراردادها به مشارکت کنندگان دیگر در شبکه نیز ارسال شود.

این تکنولوژی می‌تواند در کشور ما با توجه به فشارهای دولت‌های خارجی زمینه‌ساز تغییرات بزرگی از جمله امکان انتقال ارز به تمامی کشورهای دیگر بدون نیاز به تأیید کشورهای تحریم‌کننده باشد، همچنین از این نوع تکنولوژی می‌توان برای سرعت بخشیدن، افزایش امنیت و شفافیت به تراکنش‌های درون کشور نیز استفاده کرد.

برای معرفی بیشتر بلاک‌چین باید گفت که یک دفترکل و پایگاه داده دیجیتال و غیرمتمرکز است که برای تبادل ایمن اطلاعات، ارز دیجیتال، انجام معاملات و تراکنش‌ها استفاده می‌شود. هریک از اعضای شبکه به جدید ترین نسخه دفترکل رمزگذاری‌شده دسترسی پیدا می‌کند تا بتواند معامله جدیدی را تأیید و یا ارسال کند. اصولاً این یک پایگاه داده توزیع شده است که یک بلوک را در خود نگه می‌دارد و رشد می‌دهد. بلوک‌های تکمیل شده به ترتیب خطی زمانی اضافه می‌شوند. هر بلوک حاوی یک نشانگر زمانی و پیوند اطلاعاتی است که به بلوک قبلی اشاره دارد.

واژه‌هاي كليدي: **بلاک‌چین-کریپتوکرنسی-بیت‌کوین**

فصل 1:
مقدمه

1.1- شرح مسأله

بلاک‌چین نوعی از پایگاه داده و ذخیره‌سازی است که غیر متمرکز، قابل اعتماد و استفاده از آن برای اهداف کلاه بردارانه دشوار است. از طرف دیگر بیت کوین و بقیه ارزهای دیجیتال از یک دفترکل عمومی بلاک‌چین برای انجام معاملات در شبکه‌های همتا به همتا استفاده می‌کند.
فناوری بیت‌کوین و بلاک‌چین شروع به شکل‌گیری و تعریف جنبه‌های جدید در علم کامپوتر و فناوری اطلاعات کرده است. نیاز به ارز غیرمتمرکز و جدا از کنترل و دستکاری دولت‌ها به صورت تئوری از دیرباز برسر زبان‌ها بوده است، اما در دهه گذشته به لطف مقاله معروف ساتوشی ناکاماتو در سال ۲۰۰۸ و معرفی تکنولوژی بلاک‌چین عملی شد.
در حالیکه در مورد هویت واقعی ناکاماتو اختلاف نظر وجود دارد، بدون شک او بزرگ‌ترین و انقلابی ترین تکنولوژي بعد از اینترنت را به جهان معرفی کرد. حال کاربران باید تصمیم بگیرند که می‌خواهند با آن چه کاری انجام دهند. برخی از این فرصت استفاده کرده و برنامه و اپلیکیشن خود را برای حل‌کردن مشکلات مختلف و حفره‌های جامعه توسعه می‌دهند. عده‌ای بر روی آن ایده‌ها سرمایه‌گذاری می‌کنند یا با امواج و بالا پایین شدن قیمت رمزارز سود کسب می‌کنند.
صرفاً نسخه همتا به همسالان پول نقد الکترونیکی امکان پرداخت مستقیم پرداخت آنلاین را از یک طرف به طرف دیگر بدون گذراندن نهاد مؤسسه مالی فراهم می‌کند. برای این مشکل ما انتقالات را از روش همتا به همتا پیشنهاد میدهیم.
در این پژوهش به دنبال پیدا کردن ساختار بلاک‌چین نحوه کار آن و ساخت یک رمزارز برای رقابت در بازارهای رمز ارزهای بین‌المللی و همینطور رفع مشکلات کنونی امنیت تراکنش‌های بانکی، رفع عدم شفافیت، کمترکردن هزینه تراکنش‌ها و جلوگیری از جرایم مالی است. علاوه بر موارد گفته شده باعث درست‌شدن مشاغل از جمله ساخت مزرعه برای حفاری و نیاز به برنامه‌نویس برای ساخت و ادامه توسعه این رمز ارزها و شبکه بلاک‌چین است. البته از این موضوع نباید چشم پوشید که این تکنولوژی بسیار نو پا، در حال تکمیل خود و ایراداتی به آن وارد است ولی با کارکردن روی آن‌ها این مشکلات را می‌توان برطرف کرد. با وجود همه‌ی مشکلات کنونی نسبت به ارزهای فیات برتری دارد.

1.2- انگیزه‌هاي پژوهش

ساخت یک کریپتوکرنسی ملی با هدف عرضه عمومی در بازار ایران و کشورهای خارجی برای راحت‌تر کردن و یکسان سازی تراکنش‌ها.

1.3- اهداف پژوهش

۱- ساخت گره با رابط CLI
2- ساخت گره بر بستر وب و مروگر
۳- ساخت ماژول‌های بلاک‌چین
۴- ساخت کیف پول
۵- ساخت روش ماین کردن
۶- ساخت روش حل اختلاف
۷- رفع ایرادها و بهینه کردن آن‌ها

۸- ارائه برنامه بر روی git برای توسعه بیشتر آن

فصل۲:

مروري بر كارهاي مرتبط

1.4- مقدمه

پروژه‌های مختلف و با هدف‌های مختلفی از زمان ساخت اولین کریپتوکرنسی یا بیت‌کوین تا کنون انجام شده است. هدف از استفاده از این تکنولوژی، بالا بردن شفافیت، امنیت و سرعت است و معمولاً در سرویس‌هایی مورد استفاده قرار می‌گیرد که این سه فاکتور در آن‌ها از اهمیت خاصی برخوردار است.

به طور مثال در بیت‌کوین هرسه فاکتور استفاده شده است. برنامه‌هایی مانند اسپاتیفای و استیم از ویژگی شفافیت آن در برنامه‌های خود استفاده کرده‌اند. و برنامه‌هایی مانند استورج امنیت را از این تکنولوژی قرض گرفته‌اند.

1.5- نمونه‌هایی از برنامه‌های قراردادهای هوشمند در بلاک‌چین

در پزشکی

این تکنولوژی در زمینه‌های مختلف و رشته‌های گوناگون نیز مورد استفاده قرار گرفته است، سوابق بهداشت شخصی می‌توانند با یک کلید خصوصی روی رمز و بسته رمزگذاری و ذخیره شوند و فقط به افراد خاص دسترسی پیدا کنند. همین استراتژی می‌تواند برای اطمینان از انجام تحقیقات از طریق قوانین HIPAA (به روشی مطمئن و محرمانه) مورد استفاده قرارگیرد. دریافت جراحي را مي‌توان در زنجیره نگهداري کرد و به‌صورت اتوماتيك به عنوان اثبات زايمان براي بيمه گذاران ارسال شد. این دفترچه نیز می‌تواند برای مدیریت مراقبت‌های بهداشتی عمومی مانند نظارت بر داروها، رعایت مقررات، نتایج آزمایش و مدیریت منابع مراقبت‌های بهداشتی مورد استفاده قرارگیرد.

در موسیقی

مشکلات اصلی در صنعت موسیقی شامل حقوق مالکیت، توزیع حق امتیاز و شفافیت است. صنعت موسیقی دیجیتال بر کسب درآمد از تولید متمرکز است، در حالی که حقوق مالکیت غالباً نادیده گرفته می‌شود. فناوری بلاک‌چین و قراردادهای هوشمند می‌توانند با ایجاد یک بانک اطلاعاتی غیرمستقیم جامع و دقیق از حقوق موسیقی، این مشکل را حل کنند. در همان زمان، مدیر دفتر با ارائه شفاف حق امتیاز هنرمندان و قرارداد نوازندگان و هنرمندان را با ارز دیجیتال پرداخت می‌کنند.

در فرآیندهای دولتی

در انتخابات سال 2016 در آمریکا، دموکرات‌ها و جمهوری‌خواهان امنیت سیستم رای‌گیری را زیر سوال بردند. حزب سبز خواستار بازشماری در ویسکانسین، پنسیلوانیا و میشیگان شد. دانشمندان رایانه می‌گویند هکرها می‌توانند با دستکاری سیستم الکترونیکی، در نتیجه آرا تقلب کنند. دفترچه راهنما از زمان رمزگذاری از تقلب در آرا جلوگیری می‌کند. افراد خصوصی می‌توانند تأیید کنند که آراء آنها شمارش شده است و تأیید می‌کنند که به چه کسی رای داده‌اند. این سیستم، صرفاً برای دولت نیز پس‌انداز می‌کند.

بر اساس گزارش سال 2013 از مک کینزی و کمپانی، داده‌های باز - داده‌های منبع آزاد دولت که از

طریق اینترنت در دسترس همه شهروندان قرار دارد- بستری را فراهم می‌کند که کلاهبرداران از داده‌های آن برای کلاهبرداری استفاده کنند، کشاورزان می‌توانند از آن برای بررسی زمان دقیق برداشت محصول در مزرعه استفاده کنند و والدین می‌توانند در مورد عوارض جانبی دارو برای فرزندان بیمار خود تحقیق کنند. در حال حاضر، این داده‌ها فقط یک بار در سال منتشر می‌شود و تا حد زیادی برای ورودی شهروندان پاسخگو نیست. سیستم بلاک‌چین به عنوان یک دفترچه عمومی می‌تواند این داده‌ها را در هر زمان و هر کجا که می‌خواهد برای شهروندان باز کند.

1.6- نمونه‌هایی از برنامه‌های کاربردی دولت مجهز به بلاک‌چین

ارزش عمومی / جامعه

این مجموعه می‌تواند با فراهم کردن بستر خودمدیریتی برای شرکت ها، سازمان‌های مردم نهاد، بنیادها، سازمان‌های دولتی، دانشگاهیان و شهروندان فردی، خود سازماندهی را تسهیل کند. احزاب می‌توانند در مقیاس جهانی و شفاف اطلاعات را تعامل و تبادل کنند - به iCloud و google drive فکر کنید، اما بزرگ‌تر و کم خطرتر است.

واگذاری مسئولیت

با استفاده از بلاک‌چین استفاده از مسئولیت‌ها به صورت واضح و شفاف برای افراد مشمول مسئولیت مانند دانشجویان کارمندان مدیران و مسئول‌های قسمت مختلف ارسال می‌شود.

با این روش تمامی کارهای انجام شده در سطح دانشگاهی، تیمی و جامعه مشخص شده و افراد کم کار جریمه و افرادی که مسئولیت خود را به درستی انجام می‌دهند تشویق می‌شوند.

هویت در فضای مجازی

چه آن را دوست داریم یا نه، شرکت‌های آنلاین همه چیز را در مورد ما می‌دانند. برخی از شرکت‌هایی که ما از آنها خرید اینترنتی می‌کنیم، اطلاعات هویت خود را به تبلیغ کننده‌هایی می‌فروشند که تبلیغات خود را برای شما ارسال می‌کنند. بلاک‌چین با ایجاد یک نقطه داده محافظت شده، به شما این اجازه را می‌دهد که در مقابل این سواستفاده‌ها ایمن شوید و به هر کسی که نیاز به اطلاعات شما دارد فقط آن قسمت معین شده را نشان دهید.

پاسپورت‌ها

اولین گذرنامه دیجیتال در سال 2014 در GitHub راه‌اندازی شد و می‌تواند به صاحبان کمک کند تا خود را بصورت آنلاین و حتی آفلاین خارج از کشور شناسایی کنند. چگونه کار می‌کند؟ شما از خودتان عکس می‌گیرید، آن را با یک کلید عمومی و خصوصی مهر می‌کنید، که هر دو برای اثبات آن رمزگذاری شده‌اند. گذرنامه با توجه به آدرس بیت‌کوین با یک آی پی عمومی در دفترچه ذخیره می‌شود و توسط کاربران بلاک‌چین تأیید می‌شود.

به طور کلی این کاربرد بلاک‌چین شامل ساخت اسناد مرگ، تولد، ازدواج و انواع شناسنامه می‌شود. هم چنین برای تمامی کارت‌های شناسایی نیز کاربرد خود را حفظ می‌کند.

تامین انرژی

در بعضی نقاط دنیا، موسسات اقتصادی و منازل می‌توانند از مزیت‌های شبکه‌های توزیع مبتنی بر

بلاک‌چین، برای ذخیره انرژی و ردیابی دقیق مصرف، استفاده کنند. بلاک‌چین همچنین می‌تواند برای بهبود دنبال کردن انرژی پاک استفاده شود. زمانی که برق به شبکه فرستاده می‌شود، کسی نمی‌تواند تشخیص دهد که آیا توسط سوخت‌های فسیلی تولید شده‌اند یا باد یا انرژی خورشیدی. به طور سنتی، انرژی‌های تجدید پذیر توسط مجوزهای قابل انتقال که توسط دولت تایید می‌شوند، ردیابی می‌شوند. این مجوزها دارای ایراداتی هستند که بلاک‌چین می‌تواند آن‌ها را حل کند.

حسابداری

ضبط تراکنش‌ها از طریق بلاک‌چین، به طور قابل توجهی خطای انسانی را از بین می‌برد و داده‌ها را از دستکاری احتمالی حفظ می‌کند. به یاد داشته باشید که رکوردها هربار که بلاک جدید تولید می‌شود، مورد تایید قرار می‌گیرند. البته کل فرآیند حسابداری نیز موثرتر می‌شود. به جای نگهداری رکوردهای جداگانه، کسب و کارها می‌توانند تنها یک دفتر کل واحد نگهداری کنند. یکپارچگی اطلاعات مالی یک شرکت نیز تضمین خواهد شد.

بازار بورس

مفهوم استفاده از فناوری بلاک‌چین برای امنیت و مبادله کالا مدتی است که شکل گرفته است. به دلیل خاصیت باز و امن سیستم بلاک‌چین، بازار بورس نیز به فناوری بلاک‌چین به عنوان یک نقطه پرش نگاه می‌کند. بازار بورس استرالیا مدتی است که قصد انتقال عملیات‌های خود به سیستم مبتنی بر بلاک‌چین را دارد، که توسط استارت‌آپ بلاک‌چین Digital Asset Holding طراحی شده است.

اینترنت اشیاء (IoT)

ترکیب این دو تکنولوژی با یکدیگر باعث افزایش سرعت و سهولت آن می‌شود.

1.7- نتیجه‌گیری

توجه به این نکته ضروری است که برای کار با استفاده از بلاک‌چین، در صورت رعایت این استانداردها، این تکنولوژی می‌تواند به ابزاری قدرتمند برای انجام و بهبود تجارت، عادلانه‌تر کردن اقتصاد جهانی و کمک به حمایت از جوامع بازتر و عادلانه‌تر تبدیل شود.

حال با توجه به کاربرد این تکنولوژی و ابزارهای ساخته شده به کمک آن می‌توان یک نمونه برنامه کامل برای نقاط آسیب‌پذیر جامعه نوشت. بلاک‌چین یک فناوری نوظهور است که نیاز دارد تا برنامه‌هایی سازگار با آن تولید شود. همچنین برای استفاده از این فناوری باید مجموعه‌ای مقررات تنظیم شود. این فناوری باید بتواند کارآمدی خود را اثبات کند تا به‌صورت گسترده به کار گرفته شود.

فصل۳:

روش پیکر بندی محیط توسعه

۳–مقدمه

در بخش ابتدا به محیطهایی که در آن این برنامه توسعه داده شده است می‌پردازیم، سپس مفاهیم و پیکر بندی‌های موجود در این سیستم بلاک‌چین را معرفی می‌کنیم.

۱-۳–برنامه ها و سرویس‌های استفاده شده در ساخت پروژه

۱-۱-۳- Vs Code:

ویژوال استودیو کد (Visual Studio Code یا به اختصار VSCode) یکی از ویرایشگرهای کد بسیار محبوب می‌باشد که توسط مایکروسافت ایجاد و نگهداری می‌شود. پشتیبانی VSCode از زبان‌های برنامه‌نویسی بسیار گسترده است و زبان‌هایی مانند PHP و JavaScript و HTML و CSS و ASP.NET و Java و بسیاری از زبان‌های دیگر را پشتیبانی می‌کند. دومین نکته و شاید مهم‌ترین نکته آن رایگان بودن است؛ زیرا مشکلات شکستن قفل و مشکلات قانونی را همراه خود ندارد، اما ویژگی بارزی که دارد و باعث درخشش هرچه بیشتر آن شده است، پشتیبانی از افزونه‌ها است.

قسمت‌های مختلف این برنامه عبارتند از

- **قسمت Explorer:** این قسمت ساختار پوشه‌ها و فایل‌های شما است. با کلیک روی Explorer منویی باز خواهد شد که تمام فایل ها و پوشه‌های پروژه‌ی فعلی را نمایش می‌دهد و می‌توانید با استفاده از آیکون‌های کوچک بالای آن فایل‌ها یا پوشه‌های جدیدی بسازید.
- **قسمت Search:** این قسمت به شما اجازه می‌دهد که به دنبال قسمت خاصی از کدها بگردید یا کدهای خاصی را با کدهای دیگری جایگزین کنید.
- **قسمت Source Control Management:** این قسمت مخصوص سیستم‌های کنترل نسخه مانند Git هستند. سیستم‌هایی مانند Git به شما کمک می‌کنند تا سورس کد پروژه‌تان را بهتر مدیریت کنید و از حذف ناخواسته‌ی کد جلوگیری کنید. همچنین برای کار گروهی روی یک پروژه گزینه‌ی ایده‌آلی محسوب می‌شوند.
- **قسمت Debugging:** همانطور که از نامش مشخص است، کار اشکال‌زدایی (debugging) را بر عهده دارد. توجه داشته باشید که بسته به زبان برنامه‌نویسی خود، ممکن است برای انجام debugging به افزونه‌ها نیاز پیدا کنید.
- **قسمت Extensions:** این قسمت به شما اجازه می‌دهد افزونه‌های مختلفی را جستجو کرده و آن ها را به VSCode اضافه کنید. این افزونه ها برای موارد مختلفی ساخته شده‌اند، به طور مثال: زیبایی ظاهری و تم‌ها، ایجاد پشتیبانی از فریم ورک‌هایی مانند React یا Vue، افزایش پشتیبانی از برخی از زبان‌های برنامه‌نویسی مانند Python، ارتقای کیفیت قسمت‌های خاصی از VSCode مانند debugging و غیره.

۳-۱-۲- Post Man:

به نظر می‌رسد اگر بخواهیم در یک جمله مهم‌ترین خصوصیت و ویژگی کاربردی نرم افزار و افزونه Postman را نام ببریم، باید چنین بگوییم:

Postman امکان تست، اجرا و بررسی کدها و متدهای نوشته شده WebAPI را برای ما بسیار آسان می‌سازد.
در محیط برنامه‌نویسی وقتی یک Web API را پیاده سازی می‌کنیم، برای تست آن در همان محیط، به ناچار باید کدهایی را سمت کلاینت بنویسیم تا بتوانیم خروجی آن را بررسی نماییم، یعنی باید کد اصلی که قرار هست بنویسیم را شبیه سازی کنیم، اما همیشه انجام این کار برای ما از نظر زمانی مقرون به صرفه نیست و ما دوست داریم بسیار سریع خروجی برنامه خود را مشاهده کنیم و در زمان‌هایی که در کدهای خود تغییر می‌دهیم به آسانی بتوانیم خروجی و نتیجه تغییرات خود را نیز ملاحظه نماییم.
همچنین در برخی از مواقع تست خروجی توسط برنامه اصلی کار دشوار و پر چالشی است. بنابرین وجود یک نرم افزار واسط برای این کار بسیار کارآمد بوده و استفاده از آن به ما کمک شایانی خواهد کرد.
از دلایل استفاده از پستمن می‌توان به رایگان بودن آن، به اشتراک گذاری آسان و گرفتن خروجی کد برای هر یک از API ها که به معنی این است که هر چیزی که بتواند زمان برنامه‌نویسی و توسعه را کم کند بسیار اهمیت پیدا می‌کند. یکی از این قابلیت‌ها همین Code Snippet است که می‌توانید پس از مقداردهی و تست هر یک از API ها می‌تواند یک خروجی به زبان برنامه‌نویسی مورد علاقه خود داشته باشد. زبانهایی مانند JAVA، C#، PHP، Python باشد. هم چنین مولتی پلتفرم بودن آن باعث می‌شود بر روی همه دستگاه‌ها قابل اجرا و ویرایش باشد.
برای نصب آن کافیست از وبگاه آن به صورت رایگان آن را دانلود کنید.

۳-۱-۳- زبان پایتون:

نصب پایتون در لینوکس: نسخه 3.8

هنگام شروع توسعه‌دادن این رمز ارز پایتون نسخه ۳.۵ آخرین ورژن ارائه شده بود ولی به دلیل پشتیبانی تیم پایتون هنگام نوشتن این مقاله پایتون ۳.۸ نیز برای استفاده مهیا شده است.
به دلیل مشکلات فراوانی که بر روی سیستم عامل ویندوز وجود دارد، برای کار با سرور و برای برنامه‌نویسی پایتون، بعد از طراحی بخش‌های پایه این برنامه ادامه توسعه آن را با لینوکس آرچ ادامه داده ولی به دلیل پیچیدگی بیش از حد نصب آن در این مقاله نصب پایتون بر روی سیستم عامل لینوکس اوبونتو را توضیح داده می‌شود.
نصب پایتون با استفاده از ابزار **apt**، فرایند بسیار ساده‌ای است. مراحل لازم برای نصب پایتون در لینوکس (توزیع Ubuntu) توسط ابزار **apt** در ادامه آمده است.

در ابتدا لازم است تا لیست بسته‌های نرم‌افزاری به روز رسانی و پیش‌نیازهای لازم برای نصب پایتون در لینوکس (توزیع Ubuntu) آماده شوند:

```
sudo apt update
sudo apt install software-properties-common
```

در مرحله بعد، deadsnakes PPA به لیست مخزن‌های نرم‌افزاری در Ubuntu اضافه می‌شود:

```
sudo add-apt-repository ppa:deadsnakes/ppa
```

پس از فعال شدن مخزن نرم‌افزاری deadsnakes PPA [illegible] زیر نصب کنید:

```
sudo apt install python3.8
```

در پایان مرحله قبلی، نسخه 3.8 پایتون روی توزیع Ubuntu از سیستم عامل لینوکس (یا توزیع‌های مرتبط) نصب شده و آماده استفاده است. برای اطمینان از صحت نصب پایتون در لینوکس می‌توان از دستور زیر استفاده کرد:

```
Python3.8 --version
```

۴-۱-۳- محیط مجازی یا virtual environment:

از مهم‌ترین شیوه‌های آزموده شده پایتون هستند. هر زمان که می‌خواهید یک پروژه پایتون جدید را آغاز کنید، باید تصمیم بگیرید که قصد دارید از کدام نسخه پایتون استفاده کنید. همچنین باید از میان برخی کتابخانه‌ها یا بسته‌ها دست به انتخاب بزنید. البته یک روش دیگر هم این است که این بسته‌ها را در سطح سیستمی نصب نکنید. چون این امکان هست که هم‌زمان بر روی پروژه‌های مختلفی کار کنید که به نسخه‌های متفاوتی از پایتون نیاز دارند. درعین حال احتمال دارد به برخی بسته‌های خاص نیاز داشته باشید که صرفاً با یک نسخه از پایتون کار می‌کنند و روی نسخه‌های دیگر از کار می‌افتند. در چنین مواردی باید محیط‌های مختلفی برای پایتون راه‌اندازی کنیم. این محیط‌ها به نام «محیط مجازی پایتون» (Python Virtual Environments) نامیده می‌شوند. محیط مجازی استفاده شده در این برنامه بسته به این که پایتون را روی چه سیستمی و با چه روشی نصب کرده باشید، مکان آن روی سیستم متفاوت خواهد بود. با این حال اگر پایتون 3 را با استفاده از Homebrew نصب کرده باشید، موقعیت آن روی سیستم در مسیر زیر خواهد بود:

```
usr/local/Cellar/python/3.7.2_1/bin/
```

در این حالت، برای نصب virtualenv با استفاده از pip3 می‌توانید از دستور زیر استفاده کنید:

```
pip3 install virtualenv
```

اکنون همه بسته‌ها نصب شده‌اند و می‌توانیم شروع به راه‌اندازی محیط مجازی بکنیم. در ابتدا باید محلی که می‌خواهیم محیط خود را بسازیم معین کنیم و نامی نیز برای آن تعیین کنیم. ما محیط مجازی خود را در همان دایرکتوری که نصب شده، ایجاد می‌کنیم و نام آن را نیز virtEnv1 می‌گذاریم.

```
virtualenv -p python3 ~/virtEnv1
```

برای **ورود و فعال سازی** محیط مجازی‌ای با نام myenv به صورت زیر عمل می‌کنیم:

```
Windows #
\cd Documents\SampleENV <
Scripts\activate.bat <

<(SampleENV)

Linux #
/cd Documents/SampleENV $
source bin/activate $
```

مزیت‌های Virtualenv: ارتقای آن با استفاده از pip به آسانی میسر است و می‌تواند به راحتی با نسخه‌های مختلف پایتون کار کند. همچنین از Python 2.7 به بعد را نیز پشتیبانی می‌کند.

معایب Virtualenv: در این محیط مجازی، فایل باینری مفسر در عمل به یک مکان جدید کپی شده است و باید از آنجا خوانده شود. ضمناً اگر بخواهید از آن استفاده کنید، باید آن را به صورت مجزا نصب کنید و به همراه پایتون ارائه نمی‌شود.

توجه کنید که با ورود موفق به محیط مجازی، prompt خط فرمان چگونه تغییر می‌کند.

اکنون می‌توانیم در پروژه خود به کتابخانه‌ها، pip، دایرکتوری site-packages و مفسری اختصاصی دسترسی داشته باشیم. همچنین با فعال کردن یک محیط مجازی، فایل‌های اجرایی مربوط به این محیط درون متغیر PATH قرار می‌گیرند تا همانند تا دستورات مورد استفاده به سادگی در دسترس باشند.

در لینوکس می‌توانید با اجرای دستورات which python3 و which pip3 بررسی کنید که مسیر فایل اجرایی مفسر پایتون و pip از مسیر معمول آن (/usr/bin/python3) متفاوت است.

> پس، برای هر پروژه‌ای کافی است داخل پروژه یکبار با فراخوانی virtualenv محیط مجازی را بسازید و پس از آن هرباری که داخل دایرکتوری پروژه مورد نظر می‌شوید آن محیط را فعال کنید.

برای **خروج و غیر فعال کردن** محیط از دستور زیر استفاده می‌کنیم:

```
Windows #
(SampleENV)> deactivate.bat

Linux #
(SampleENV)$ deactivate
```

در صورت استفاده از لینوکس میتوان با alias ها عملیات ورود و خروج را ساده‌تر کرد.

```
'alias ae='deactivate &>/dev/null; source ./venv/bin/activate
'alias de='deactivate &>/dev/null
```

۳-۱-۵- نصب vue:

برای نصب ویو ابتدا باید node.js را نصب کرد. برای اینکار ابتدا از سایت https://nodejs.org/en/ باید پکیج مورد نظر را نصب کرد. برای نصب آخرین نسخه از Vue CLI می‌توانید از دستور زیر استفاده کنید:

```
npm install -g @vue/cli
```

ممکن است در سیستم‌عامل‌های مبتنی بر لینوکس و یا مک‌اواس به دستور sudo در ابتدای کدهای بالا نیاز داشته باشید. بنابراین اگر با مشکلی روبرو شدید آن را وارد کنید.

بعد از آنکه به صورت موفقیت آمیز این پکیج را نصب کردید حال باید به صورت اجرایی به vue در ترمینال دسترسی داشته باشید. برای اطمینان از این قضیه می‌توانید دستور vue را در خط فرمان وارد کنید:

بعد از نصب Vue CLI بیایید نگاهی به این موضوع بیاندازیم که چگونه می‌توانیم یک پروژه را از ابتدا ایجاد کنیم. برای انجام چنین کاری ما از دستور create استفاده خواهیم کرد. برای انجام این کار دستور زیر را در ترمینال وارد کنید:

```
vue create example-vue-project
```

۳-۱-۶- نصب axious:

axios باید از طریق NPM یا Yarn نصب شود:

```
npm install axios --save
```

۳-۱-۷- شبکه نظیر به نظیر

نظیر به نظیر یا P2P به گونه خاصی از شبکه‌های کامپیوتری اشاره دارد که از یک معماری توزیع شده استفاده می‌کنند. به این معنا که همه کامپیوترها یا دستگاه‌های عضو این شبکه حجم کاری خود را در شبکه به اشتراک قرار می‌دهند. کامپیوترها یا دستگاه‌هایی که بخشی از یک شبکه نظیر به نظیر هستند peers نامیده می‌شوند. کامپیوترهای درون یک شبکه نظیر به نظیر هیچ‌گونه ارجحیتی نسبت به یکدیگر نداشته و همگی و با یکدیگر برابر هستند. کامپیوترهای درون یک شبکه نظیر به نظیر بدون آن‌که به یک سیستم مدیریت متمرکز نیازی داشته باشند، منابع را میان یکدیگر تقسیم می‌کنند.

شبکه‌های نظیر به نظیر ضمن آن‌که برای به‌اشتراک‌گذاری منابع مورد استفاده قرار می‌گیرند، همچنین به کامپیوترها و دستگاه‌ها کمک می‌کنند در قالب یک گروه سرویس خاصی را ارائه کرده یا یک کار خاص را انجام دهند. با این وجود شبکه‌های فوق عمدتا به منظور به اشتراک‌گذاری فایل‌ها در اینترنت مورد استفاده قرار می‌گیرند. شبکه‌های P2P به دلیل آن‌که به کامپیوترها اجازه می‌دهند به شبکه متصل شده و به‌طور همزمان فرآیند دریافت و ارسال فایل را انجام دهند ایده‌آل هستند. فرض کنید، مرورگر خود را باز کرده و برای دانلود یک فایل سایتی را باز می‌کنید. در این حالت، سایت به عنوان یک سرور کار کرده و کامپیوتر شما در نقش یک کلاینت فایل را دریافت می‌کند. این وضعیت مشابه جاده‌ای یک طرفه است. فایلی که شما دانلود می‌کنید ماشینی است که از نقطه A (سایت) به نقطه B (کامپیوتر شما) حرکت می‌کند.

اگر همان فایل را از طریق یک شبکه نظیر به نظیر دانلود کنید، به‌طور مثال از طریق یک سایت تورنت به عنوان نقطه شروع، دانلود به صورت‌های مختلف انجام می‌شود. فایل در قالب بیت‌ها و بخش‌هایی که روی کامپیوترهای مختلف درون یک شبکه نظیر به نظیر قرار دارد دانلود می‌شود. در همان زمان ممکن است فایلی از کامپیوتر شما به سمت کامپیوترهایی که فایل را درخواست کرده‌اند ارسال شود. این وضعیت شبیه به یک جاده دو طرف است.

مزایا و معایب P2P

حال نقاط قوت و ضعف این معماری را بررسی می‌کنیم.

مزایا

ساختار همتا به همتای بلاک‌چین، مزایای بسیاری را فراهم می‌کند. در میان مهم‌ترین مزایای P2P می‌توان این نکته را بیان کرد که شبکه‌های همتا به همتا، امنیت بسیار بیشتری در مقایسه با تنظیمات مشتری-سرور سنتی ارائه می‌دهند. توزیع بلاک‌چین‌ها در تعداد زیادی از نودها باعث می‌شود که آن‌ها در مقابلDos که تعداد بی شماری از سیستم ها را گرفتار می‌نماید، مصون باشند.

به همین صورت، از آن جایی که تعداد زیادی از نودها باید پیش از اضافه‌شدن داده‌ها به بلاک‌چین به اجماع برسند، تقریبا برای حمله کننده غیر ممکن می‌شود که بتواند در داده‌ها تغییر ایجاد کند. این امر مخصوصا برای شبکه‌های بزرگی همچون شبکه بیت کوین صادق است. بلاک‌چین‌های کوچک‌تر بیشتر مستعد حملات هستند، چرا که یک فرد یا یک گروه، در نهایت ممکن است بتوانند کنترل اکثریت نودها را به دست بگیرند (این امر به حمله 51 درصد مشهور است).

در نتیجه، شبکه توزیع شده همتا به همتا که با نیاز به اجماع اکثریت همراه می‌شود، به بلاک‌چین‌ها میزان زیادی از مقاومت را در برابر فعالیت‌های مخرب را اعطا می‌کند. مدل P2P یکی از دلایل اصلی است که بیت کوین (و سایر بلاک‌چین‌ها) قادر به دست یابی به **تحمل خطای بیزانس** هستند.

علاوه بر امنیت، ساختار همتا به همتا در بلاک‌چین‌های ارزهای رمز پایه، می‌تواند آن‌ها را در برابر سانسور توسط مقامات مرکزی مقاوم سازد. برخلاف حساب‌های بانکی استاندارد، کیف پول‌های ارزهای رمز پایه نمی توانند توسط دولت ها مسدود یا تخلیه شوند. همچنین این مقاومت در برابر تلاش‌های پلتفرم‌های محتوا و پردازش پرداخت‌های خصوصی برای سانسور نیز ایجاد می‌گردد.

برخی از تولید کنندگان محتوا و تاجران آنلاین، پرداخت با ارزهای رمز پایه را به عنوان روشی برای جلوگیری از مسدود شدن پرداخت‌های آنها توسط اشخاص ثالث انتخاب کرده‌اند. پس از بیان مزایا P2P به سراغ معایب آن می‌رویم.

معایب

با وجود مزایا و خوبی‌های زیادی که شبکه P2P دارد، همچون هر مفهوم دیگری دارای معایب و محدودیت‌هایی نیز هست.

از آن جایی که دفاتر کل توضیع شده، به جای یک سرور مرکزی، باید در تمامی نودها به روزرسانی شود، افزودن یک تراکنش به بلاک‌چین، نیازمند قدرت پردازشی بسیار عظیمی است. با وجود این که این امر امنیت بسیاری را فراهم می‌نماید، اما کارایی را تا حد زیادی کاهش داده، و یکی از موانع اصلی در برابر مقیاس‌پذیری و مقبولیت گسترده است.

با این وجود، رمز نگاران و توسعه دهندگان بلاک‌چین در حال جستجو برای جایگزین‌هایی هستند که بتوانند به عنوان راه حل مقیاس پذیری از آن ها استفاده کنند. مثال‌های برجسته آن عبارت‌اند ازشبکه لایتنینگ، پلاسما اتریوم، و پروتکل میمبل ویمبل.

محدودیت دیگر شبکه همتا به همتا می‌تواند مرتبط با حملاتی باشد که در رویدادهای هاردفورک اتفاق می‌افتند. از آن جایی که اکثر بلاک‌چین‌ها غیر متمرکز و متن باز هستند، گروه‌هایی از نودها می‌توانند به صورت آزادانه به کپی کردن و اصلاح کدها بپردازند و از زنجیره اصلی جدا شده و یک شبکه موازی جدید بسازند. هارد فورک‌ها به خودی خود کاملا نرمال هستند، و هیچ تهدیدی ندارند. اما اگر چند مورد امنیتی خاص به خوبی رعایت نشود، هر دو زنجیره ممکن است در معرض خطر حمله قرار بگیرند.

علاوه بر این، ذات توزیع شده شبکه‌های P2P، کنترل و قانون گذاری بر آن ها را نسبتا مشکل می‌سازد. چندین برنامه و شرکت همتا به همتا درگیر فعالیت‌های غیر قانونی و نقض حق کپی شده‌اند.

برنامه‌نویسیCLI:

در نسل قبلی رایانه ها وقتی سیستم عامل‌های مبتنی بر GUI توسعه نیافته بودند، رایانه ها از برخی سیستم عامل‌های مبتنی بر فرمان مانند MS-DOS، Apple-DOS، Unix و غیره استفاده می‌کردند.

هرگونه تعامل انسانی با این سیستم عامل از طریق برخی دستورات انجام می‌گرفت. بعداً توسط مترجمی که به عنوان واسطه بین دستورات انسانی و زبان دستگاه OS عمل می‌کرد، تفسیر شد.

برخی از رایج ترین کارهای روزانه که CLI را شامل می‌شود، رفتن به یک دایرکتوری، ایجاد یک فایل جدید، نوشتن متن به یک فایل، حذف یک پرونده، نمایش پرونده ها در یک پوشه و غیره بود. با این حال، حتی در دنیای مدرن GUI سیستم عامل‌ها هنگامی که ما با انجام کلیک با استفاده از ماوس، تعامل و آسان‌ترین راه‌های تعامل با سیستم عامل را داشته باشیم، به وب سایت‌ها مراجعه کنید و غیره.

CLI هنوز در بسیاری از مکان‌ها برای انجام برخی کارهای اساسی استفاده می‌شود. رایج‌ترین استفاده از CLI توسط برنامه‌نویسان انجام می‌شود که از این ابزار برای اضافه کردن و نصب قطعات به برنامه‌های خود استفاده می‌کنند. آنها همچنین از CLI برای انجام برخی تنظیمات مربوط به سیستم بر روی سرور به طور خودکار بر اساس سناریو استفاده می‌کنند به طوری که نیازی به کار واقعی روی سرور نیست. آنها معمولاً یک فایل فرمان یا اسکریپت PowerShell را ذخیره می‌کنند که می‌تواند دستورات CLI را روی سرور اجرا کند و عملیات/تنظیمات مورد نیاز را انجام دهد.

دلیل استفاده از CLI :

در پیکربندی سیستم رایانه، CLI هنوز کاربرد عمده ای پیدا می‌کند. CLI این روزها بیشتر توسط برنامه‌نویسان نرم افزار یا مدیر سیستم انجام می‌شود تا برخی از وظایف مهم خود را انجام دهد که در غیر این صورت وقت و تلاش زیادی را در صورت انجام GUI مصرف می‌کند.

برای این پروژه هم یک گره با رابط کاربری گرافیکی (GUI) و یک رابط کاربری دستوری ایجاد می‌شود.

فصل۴:

تحلیل و طراحی

4-1 مقدمه

همانطور که در مقاله‌های پیشین بررسی شد تکنولوژی نوظهور بلاک‌چین بر اساس یک سری قوانین و چهارچوب‌هایی کار می‌کند که در این بخش سعی شده است اتفاقات داخل بلوک‌ها و روند ساخت یک بلوک مورد بررسی قرار گیرد.

اگر فردی در شبکه درخواستی را ارسال کند صحت آن درخواست توسط سرویس دهنده‌های شبکه و یا اصطلاحا نود ها تایید می‌شود سپس آن درخواست به شکل یک تراکنش داخل یک بلوک قرار می‌گیرد چگونگی این اقدامات سعی شده است طی فرآیندهایی شرح داده شود.

قبل از شروع این فصل با جدولی از کلمات پر کاربرد آشنا می‌شویم.

تایید کننده‌ی تراکنش	استخراج کننده
یک روش v-پایه برای تأیید صحت تراکنش است	توافق همگانی
مشکلی که هنگام استفاده از گره برای نسخه‌های جدید بلاک‌چین بوجود می‌آورد.	فورک
کاربرد هش یک روش برای بررسی صحت معامله است	هش
یک دفتر کل در سیستم بلاک‌چین	گره
متغیر یا متغیرهایی برای ثبت زمان صرف شده در بلاک‌چین	زمان سنج

جدول شماره یک. مروری بر مفاهیم پرکاربرد(۱)

بلاک‌چین: زنجیره‌ای از بلاک‌ها است که هر بلاک حاوی داده‌های با ارزش بدون نظارت از سرور مرکزی است. این اطلاعات با رمزنگاری محفوظ و غیر قابل تغییر هستند.

غیرمتمرکز بودن بلاک‌چین: بلاک‌چین به علت نداشتن سرور مرکزی ساختاری غیر متمرکز دارد.

توافق همگانی: قبل از اضافه شدن اطلاعات جدید به بلاک‌چین، بیش از نیمی از نودها باید تایید کنند که این اطلاعات جدید برای ورورد به بلاک‌چین معتبر هستند.

تغییر ناپذیری: هنگامی که اطلاعات به بلاک‌چین اضافه شود، پس از آن قابل تغییر یا حذف نیست. اطلاعات توسط بلاک‌چین محافظت می‌شوند، بنابراین اطلاعات رمزگذاری شده هستند و تقریبا هک اطلاعات را غیرممکن می‌سازد.

استخراج کننده: کاربرهایی که قدرت کامپیوتری خود را برای استخراج بلاک ها استفاده می‌کنند.

فرآیند ۱: ساخت آدرس

شما برای ارتباط با شبکه بلاک‌چین می‌بایست یک آدرس برای خود داشته باشید، ساخت آدرس کار سخت و زمان بری نیست اما برای برقراری ارتباط با شبکه لازم است. آدرس شما از دو قسمت تشکیل

شده، آدرس عمومی و آدرس خصوصی، که در این بین آدرس خصوصی آدرسی است که حتما می‌بایست در اختیار خودتان باشد و به نوعی امضای شماست، اگر هر کسی به آدرس خصوصی شما دسترسی داشته باشد به راحتی می‌تواند هرکاری با حساب شما بکند، از جمله به سرقت بردن دارایی‌های دیجیتال در آن گره.

فرآیند ۲: رمزنگاری

درخواست شما با امضای شما (که از طریق کلید خصوصی انجام می‌شود) در شبکه ارسال می‌شود و صحت این درخواست از طریق کلید عمومیتان تایید می‌شود، حال می‌خواهد این درخواست انتقال ارز دیجیتال باشد و یا تنها فرستادن یک متن.

هر کاری بخواهید بکنید رمزنگاری میشود! اگر شما بخواهید مثلا "سلام سپهر" را به فردی ارسال کنید این کلمه از طریق تابه هش رمزنگاری می‌شود و به صورت یک سری حروف و عدد بی معنی در می‌آید مثلا "سلام محسن" به **bce9c72c24edbd3baa1d3cac4d457d45** تبدیل می‌شود.

متد ماینینگ	تابع رمزگذاری	سال	رمز ارز
با محاسبه اثبات کار، تمام مقادیر غیرمجاز ممکن را پیدا کنید و سایر کاربران موافق و اثبات آن را تأیید کنند.	SHA-256	۲۰۰۸	بیت‌کوین
همانند بیت کوین (اثبات کار)	SCRYPT	۲۰۱۱	لایت کوین
اثبات کار همراه با اثبات شرط	SHA-256d	۲۰۱۲	پیر کوین
توافق عمومی	EC-DIGITAL	۲۰۱۴	ریپل
اثبات کار	ETHASH	۲۰۱۴	اتریوم
اثبات کار	SCRYPT	۲۰۱۴	آروراکوین
اثبات کار	X11	۲۰۱۵	دارک‌کوین

جدول شماره دو. اطلاعات کلی درباره رمز ارزها(۲)

فرآیند ۴: جلوگیری از ایجاد هش تکراری

ممکن است این سوال پیش بیاید که اگر داده‌های یکسان موجود باشد چه می‌شود؟ در این‌صورت که هش‌های یکسان ایجاد می‌شود!

مثلا در نظر بگیرید یک تراکنش با داده‌های یکسان از یک آدرس به آدرس دیگری ارسال شود، در این‌صورت هش‌های تکراری تولید می‌شود، چون همه چیز یکسان است. شبکه برای این مشکل هم چاره‌اندیشیده است.

نانس(Nonce)

نانس برای همین منظور ساخته شده است، نانس یک مقدار تصادفی است که به داده ها اضافه می‌شود و پس از اضافه شدن یک هش جدید ساخته می‌شود، در این حالت داده‌های یکسان هش‌های یکسان نخواهند داشت.

فرآیند ۵: تشکیل زنجیره بلوک
در بلاک‌چین هر بلوک به بلوک قبلی خود وابسته است، به این صورت که وقتی یک سری تراکنش‌ها در یک بلوک قرار می‌گیرند و به طور کلی آن بلاک هش می‌شود، این هش (که شامل اطلاعاتی از همه تراکنش‌های قبلی است) در بلاک بعدی قرار داده می‌شود، الی آخر...

- اولا به این دلیل است که می‌گویند زنجیره بلاک‌ها، زیرا در هر بلوک هش بلوک قبلی موجود است و زنجیره وار بهم متصل هستند.
- ثانیا به همین دلیل است که تغییر در اطلاعات هر بلوک باعث به هم خوردن پیوستگی بین بلوک ها می‌شود، زیرا اگر حتی هش یک داده تغییر کند، هش تمامی بلوک‌ها تغییر می‌کند.

که این از چشم اعضای شبکه پنهان نمی‌ماند، بنابراین اگر کسی بخواهد تغییری ایجاد کند این تغییر را سرویس دهنده‌های شبکه متوجه می‌شوند و آن را تایید نمی‌کنند، مگر اینکه ۵۱ درصد سرویس دهنده‌ها این تغییر را بپذیرند و آن را قبول داشته باشند.

با توجه به مفاهیم اشاره شده می‌توان میزان امنیت و شفافیت را در ساختار بلاک‌چین مشاهده کرد و طریقه کارکرد این شبکه می‌تواند برای علاقمندان در نوع خود جالب باشد.

ساختار داده ها

دانش عمیق و کاربردی از ساختار داده‌ها وقتی که هدف تبدیل‌شدن به توسعه دهنده بلاک‌چین است ضروری است. توسعه‌دهندگان بلاک‌چین دائما در حال نمایش و بهینه‌سازی ساختار داده‌های موجود مانند درخت مرکل، درخت پترسیا و غیره برای برآورده ساختن نیازهای شبکه شخصی خود هستند. بلاک‌چین، برای ساخت یک سیستم امن و غیر قابل تغییر با استفاده از رمزنگاری پیشرفته از بسیاری از ساختارهای داده استفاده می‌کند. دانش درباره بلاک‌چین، بدون داشتن اطلاعات کافی از ساختار داده‌ها، ناقص است.

کریپتو گرافی یا الگوریتم رمزنگاری

همان‌طور که قبلا اشارهشد، بلاک‌چین ترکیبی از ساختار داده‌ها و رمزنگاری پیشرفته است. بدین ترتیب واضح است که داشتن درک خوب از رمزنگاری لازمه‌ای بر توسعه دهنده بلاک‌چین است. بسیاری از روش‌های رمزنگاری مانند توابع هش، از جمله "SHA256"و"KECCAK256" برای تولید امضاهای دیجیتالی در بلاک‌چین به جای رمزنگاری نامتقارن استفاده می‌شوند. بدون درکی از نحوه کارکرد آن‌ها، توسعه‌دهنده بلاک‌چین شدن عملا غیرممکن است.

توسعه وب

توسعه وب یک جنبه اصلی ازتوسعه دهنده بلاک‌چین است. هنگامی که یک شخص کار خود را به عنوان یک توسعه دهنده بلاک‌چین در صنعت آغاز می‌کند، اکثر آن ها برای طراحی اولیه اپلیکیشن و برنامه‌های غیرمتمرکز به کار گرفته می‌شوند. این بدان معنی است که یک توسعه دهنده بلاک‌چین باید با مفاهیم اولیه تنظیمات سمت کاربر "front-end" و تنظیمات سمت سرور "back-end" که مواردی مانند ایجاد رابط کاربری گرافیکی تعاملی برای Dappsها، بررسی API، پردازش و رسیدگی درخواست ها و غیره را شامل می‌شود، آشنا باشد.

بررسی ریاضیات به کار رفته در بیت کوین

یکی از دلایلی که بیت کوین برای مبتدیان می‌تواند سردرگم کننده باشد این است که فناوری بیت کوین از مفهوم مالکیت، تعریف جدید/ ارائه می‌دهد. به طور سنتی مالک چیزی نظیر خانه یا مقداری پول بودن یعنی تصدی آن را در اختیار دارد یا تصدی آن را به نهاد مورد اعتماد نظیر بانک واگذار کرده است.

در مورد بیت کوین شرایط متفاوت است. بیت کوین به صورت مرکزی یا محلی ذخیره نمی‌شود و بدین ترتیب هیچ نهادی تصدی آن را به عهده ندارد. بیت کوین به عنوان سوابق در دفتر کل توزیع شده ای به اسم بلاک‌چین وجود دارند. نسخه‌های یکسانی از این لجر توسط شبکه ای از رایانه‌های به هم متصل به اشتراک گذاشته می‌شود. داشتن بیت کوین به معنای داشتن قابلیت انتقال دادن کنترل آن به فرد دیگر است. این امر با ایجاد سابقه انتقال در بلاک‌چین امکان پذیر است. اما چه چیزی باعث به وجود آمدن این قابلیت می‌شود؟ جواب این سوال دسترسی آن جفت کلید عمومی و خصوصی ECDSA می‌باشد. این پاسخ به چه معناست و چگونه بیت کوین را ایمن می‌کند؟

با نگاهی دقیق تر متوجه میشویم که **الگوریتم ECDSA** مخفف عبارت Elliptic Curve Digital Signature Algorithm و به معنای الگوریتم امضای دیجیتال مبتنی بر منحنی بیضوی است. ECDSA فرآیندی است که از منحنی بیضوی و میدان متناهی برای امضای اطلاعات استفاده می‌کند. امضای اطلاعات در این الگوریتم به شیوه ای انجام می‌شود که اشخاص ثالث می‌توانند اعتبار امضا را تایید کنند در حالی که فقط امضا کننده قابلیت ایجاد امضا را در اختیار دارد. در خصوص بیت کوین، اطلاعات امضا شده همان تراکنشی است که مالکیت را انتقال می‌دهد.

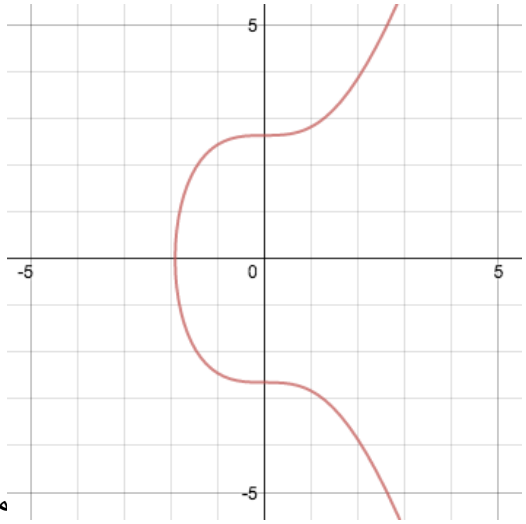
الگوریتم ECDSA، رویه‌های امضا و تایید تفکیک شده دارد. هر رویه یک الگوریتم شامل چند عملیات محاسباتی است. الگوریتم امضا و فرآیند تایید به ترتیب از کلید خصوصی و کلید عمومی استفاده می‌کنند. در ادامه این مقاله، مثالی در این خصوص بیان خواهیم کرد.

منحنی‌های بیضوی

منحنی بیضوی به صورت جبری طبق فرمول زیر بیان می‌شود:

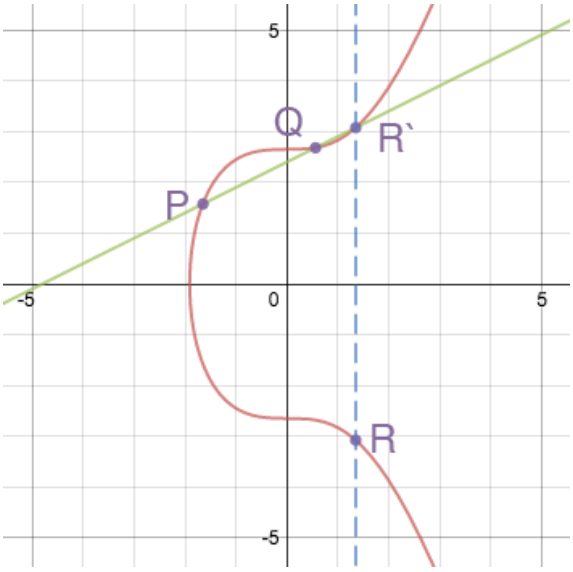
$$y^2 = x^3 + ax + b$$

اگر a برابر با صفر و b برابر با ۷ باشد (نسخه مورد استفاده توسط بیت کوین) سهمی به شکل زیر خواهد بود:

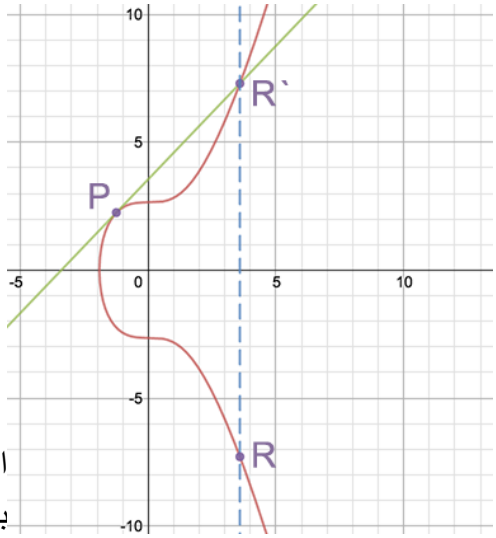


منحنی‌های بیضوی ویژگی‌های مفیدی دارند. برای مثال، خط غیرعمودی که دو نقطه غیر مماس در منحنی را قطع می‌کند همواره از روی نقطه سوم بر روی منحنی عبور می‌کند. ویژگی دیگر این است که خط غیر عمودی مماس با منحنی در یک نقطه، دقیقا یک نقطه دیگر بر روی منحنی را قطع می‌کند. با استفاده از این ویژگی ها می‌توانیم دو عملیات دیگر را نیز تعریف کنیم: افزودن نقطه و دو برابر کردن مختصات نقطه.

افزودن نقطه با فرمول P + Q = R به صورت دستیابی به مختصات محور x نقطه تلاقی سوم R بر روی خطی است که شامل P و Q است. درک این تعریف با شکل زیر آسان‌تر می‌شود:



دو برابر کردن مختصات نقطه هم با فرمول P + P = R و با یافتن خط مماس با نقطه‌ای که قرار است مختصاتش دو برابر شود و به دست آوردن مختصات محور x نقطه قطع کننده R می‌باشد. در ادامه به مثالی در این خصوص می‌پردازیم:



این دو عملیات با یکدیگر برای ضرب اسکالر با فرمول R = a P و جمع نقطه P با خودش به تعداد a بار می‌باشد. برای مثال داریم:

$$R = 7P$$

$$R = P + \Big(P + \big(P + \big(P + (P + P)\big)\big)\Big)$$

فرآیند **ضرب اسکالر** معمولا با استفاده از ترکیب عملیات افزودن نقطه و دو برابر کردن مختصات نقطه، ساده تر می‌شود. برای مثال داریم:

$$R = 7P$$
$$R = P + 6P$$
$$R = P + 2(P + 2P)$$

میدان‌های متناهی

میدان متناهی در بافت ECDSA را می‌توان به صورت طیف از پیش تعیین شده ای از اعداد مثبت در نظر گرفت که در این طیف هر محاسبه‌ای قابل انجام است. هر عددی خارج از این محدوده گرد می‌شود و داخل طیف قرار می‌گیرد.

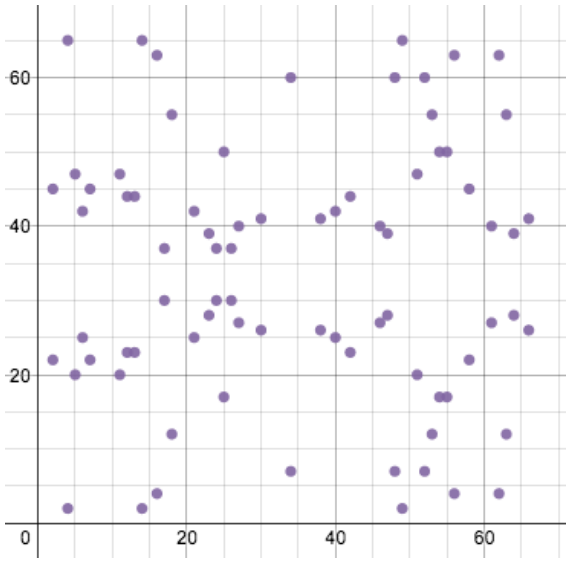
ساده ترین روش برای بیان این مورد، محاسبه باقی مانده هاست که توسط اپراتور ماژول ها (mod) بیان می‌شود. برای مثال داریم:

$$9 mod 7 = 2$$

در اینجا میدان متناهی، ماژول ۷ است و حاصل تمام عملیات mod در این میدان در طیف صفر الی ۶ قرار می‌گیرد.

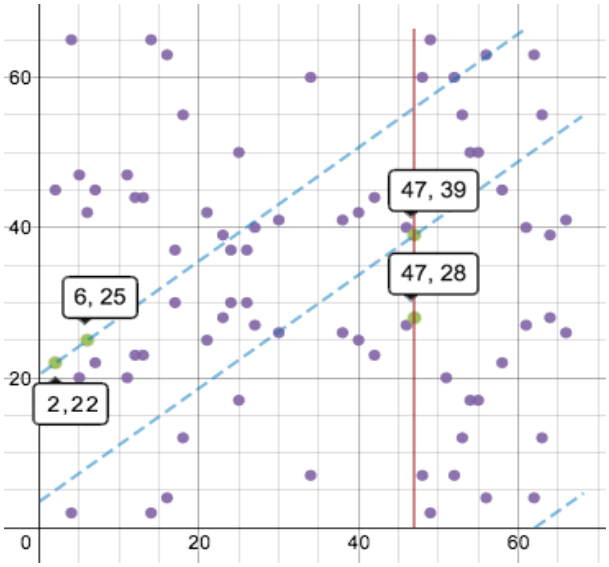
ساماندهی کردن

الگوریتم ECDSA از منحنی‌های بیضوی در بافت میدان متناهی استفاده می‌کند که ظاهر آن را به شدت تغییر می‌دهد اما در فرمول‌ها و ویژگی‌های خاص آن تغییری ایجاد نمی‌کند. فرمول مشابه شکل فوق در میدان متناهی ماژول ۶۷ به صورت زیر می‌باشد:



اکنون مجموعه ای از نقاط داریم که تمام مقادیر x و y اعداد صحیح بین صفر الی ۶۶ می‌باشند. به خاطر داشته باشید که منحنی هم چنان تقارن افقی خود را حفظ می‌کند.

افزودن نقطه و دو برابر کردن مختصات نقطه از نظر ظاهری مقداری با یکدیگر فرق دارند. خطوط این گراف در جهت‌های عمودی و افقی قرار می‌گیرند. بنابراین افزودن نقاط (2, 22) و (6, 25) به صورت شکل زیر خواهد بود:



نقطه تلاق سوم (47, 39) و نقطه بازتاب (47, 28) می‌باشد.

الگوریتم ECDSA و بیت کوین

پروتکلی نظیر بیت‌کوین، مجموعه ای از پارامترها را برای منحنی بیضوی و میدان متناهی خود انتخاب می‌کند که برای تمام کاربران پروتکل ثابت می‌باشد. این پارامترها شامل فرمول استفاده شده، ماژول اصلی میدان و نقطه پایه‌ای است که بر روی منحنی قرار می‌گیرد. ترتیب نقطه پایه‌ای که به صورت مستقل انتخاب نمی‌شود و تابعی از پارامترهای دیگر است را از نظر تصویری می‌توان به عنوان تعداد دفعاتی در نظر گرفت که نقطه می‌تواند به مختصات خودش اضافه شود تا شیب به بی‌نهایت برسد.

بیت کوین از اعداد بسیار بزرگی برای نقطه پایه‌ای، ماژول اصلی و ترتیب خود استفاده می‌کند. در واقع تمام کاربردهای عملی از ECDSA از مقادیر بسیار بزرگ استفاده می‌کنند. امنیت الگوریتم به بزرگ‌بودن این مقادیر بستگی دارند، در نتیجه حملات بروت فورس و مهندسی معکوس غیرممکن می‌شود.

در مورد بیت کوین شرایط زیر برقرار است:

معادله منحنی بیضوی:

$$y2 = x3 + 7$$

ماژول اصلی:

ترتیب:

FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE BAAEDCE6 AF48A03B BFD25E8C D0364141

چه کسی و چرا این اعداد را انتخاب کرده است؟ تحقیقات و تمهیدات بسیار زیادی در خصوص انتخاب پارامترهای مناسب صورت گرفته است. علیرغم این موارد، به نظر می‌رسد که عدد تصادفی بسیار بزرگی ممکن است روش بکدور ساخت مجدد کلید خصوصی را پنهان کرده باشد. به طور خلاصه می‌توان گفت این تعریف خاص به اسم secp256k1 بوده و بخشی از خانواده راهکارهای منحنی بیضوی در میدان‌های متناهی مورد استفاده در رمزنگاری است.

کلیدهای خصوصی و کلیدهای عمومی

با کنار گذاشتن این تشریفات، اکنون در موقعیتی قرار داریم که می‌توانیم کلیدهای خصوصی و عمومی و نحوه ارتباط آن ها را متوجه شویم. به صورت خلاصه می‌توان گفت: در ECDSA، کلید خصوصی عدد انتخابی غیرقابل پیش بینی بین ۱ الی سفارش می‌باشد. کلید عمومی از کلید خصوصی و ضرب اسکالر نقطه پایه ای به تعداد مقدار کلید خصوصی حاصل می‌شود. این تعریف را به صورت فرمول می‌توان این‌گونه بیان کرد:

کلید عمومی= کلید خصوصی * نقطه پایه ای

این فرمول نشان می‌دهد که حداکثر تعداد کلیدهای خصوصی برابر با سفارش است.

در میدان ممتد می‌توانیم خط مماس را رسم کنیم و کلید عمومی را بر روی گراف قرار دهیم، اما فرمول‌هایی وجود دارد که در بافت میدان‌های متناهی به همین نتیجه دست می‌یابند. افزودن نقطه p+q برای یافتن r به صورت زیر تعریف می‌شود:

$$c = (qy - py)/(qx - px)$$

$$rx = c2 - px - qx$$

$$ry = c(px - rx) - py$$

و دو برابر کردن نقطه p برای یافتن r نیز به صورت زیر است:

$$c = (3px2 + a)/2\,py$$

$$rx = c2 - 2px$$

$$ry = c(px - rx) - py$$

در عمل، محاسبه کلید عمومی به عملیات دو برابر کردن نقطه و افزودن نقطه تقسیم می‌شود.

مثال زیر را با اعداد کوچک بیان می‌کنیم تا مفهوم مناسبی از نحوه ساخت کلیدها و استفاده از آنها در امضا و تایید ارائه شود. پارامترهای مورد استفاده عبارتند از:

فرمول: $y2 = x3 + 7$

ماژول اصلی: ۶۷

نقطه پایه ای: (2, 22)

ترتیب: ۷۹

کلید خصوصی: ۲

ابتدا به یافتن کلید عمومی می‌پردازیم. از آنجایی که ساده ترین کلید خصوصی ممکن با مقدار ۲ را انتخاب کرده ایم، فقط به یک‌بار دو برابر کردن مختصات نقطه از نقطه پایه‌ای نیاز داریم. محاسبات این عملیات به شرح زیر می‌باشد:

$$c = (3 * 2^2 + 0)/(2 * 22)\, mod 67$$

$$c = (3 * 4)/(44)\, mod 67$$

$$c = 12/44\, mod 67$$

در اینجا سوال دیگری بیان می‌کنیم: در بافت میدان متناهی که نتیجه باید همواره عدد صحیح باشد، چگونه دسته‌بندی انجام می‌شود؟ ما باید عملیات ضرب معکوس را انجام بدهیم که از حوصله این مقاله خارج است. اما به محاسبات زیر می‌توانید اعتماد کنید:

$$44^{-1} = 32$$

سپس در ادامه داریم:

$$c = 12 * 32 \bmod 67$$
$$c = 384 \bmod 67$$
$$c = 49$$

$$r_x = (49^2 - 2 * 2) \bmod 67$$
$$r_x = (2401 - 4) \bmod 67$$
$$r_x = 2397 \bmod 67$$
$$r_x = 52$$

$$r_y = (49 * (2 - 52) - 22) \bmod 67$$
$$r_y = (49 * (-50) - 22) \bmod 67$$
$$r_y = (-2450 - 22) \bmod 67$$

$$r_y = -2472 \bmod 67$$
$$r_y = 7$$

بنابراین کلید عمومی ما متناظر با نقطه(7,52) است. تمام این اقدامات برای کلید خصوصی ۲ می‌باشد.

این عملیات یعنی دریافت کلید عمومی از کلید خصوصی، در مقایسه با معکوس این اقدام یعنی دریافت کلید خصوصی از کلید عمومی از نظر محاسباتی آسان‌تر است؛ زیرا این اقدام اگرچه از نظر تئوری امکان پذیر است اما به دلیل پارامترهای بزرگ به کاررفته در رمزنگاری بیضوی، از نظر محاسباتی اجرا شدنی نیست. بنابراین مسیر کلید خصوصی به کلید عمومی به صورت یک‌طرفه طراحی شده است.

همانند کلید خصوصی، کلید عمومی نیز معمولا با رشته هگزادسیمال بیان می‌شود. اما چگونه از روی نقطه ای بر روی صفحه که با دو عدد بیان می‌شود، چگونه می‌توان به یک عدد دست یافت؟ در کلید عمومی فشرده نشده، دو عدد ۲۵۶ بیتی که بیانگر مختصات x و y می‌باشند در یک رشته بلند به یکدیگر چسبیده‌اند. هم‌چنین می‌توانیم از تقارن منحنی بیضوی برای تولید کلید عمومی فشرده بهره ببریم. این امر با حفظ مقدار x و توجه به اینکه نقطه بر روی کدام نیمه منحنی قرار دارد انجام می‌شود. یا توجه به این اطلاعات جزیی، می‌توانیم هر دو مختصات را بازیابی کنیم.

امضای اطلاعات با کلید خصوصی

اکنون که کلید خصوصی و کلید عمومی را در اختیار داریم، اطلاعات را امضا می‌کنیم.

اطلاعات می‌تواند به هراندازه ای وجود داشته باشد. قدم اول معمولا، هش کردن اطلاعات برای تولید عددی است که میزان بیت یکسان (۲۵۶ بیت) داشته باشد. در اینجا به منظور ساده‌سازی، از مراحل هش‌کردن می‌گذریم و فقط اطلاعات خام z را امضا می‌کنیم. در ادامه به نقطه پایه G، ترتیب n و کلید خصوصی d می‌گوییم. دستور العمل امضا به شرح زیر است:

عدد صحیحی بین ۱ و n-۱ انتخاب کنید.

نقطه (x, y) را به صورت زیر با استفاده از ضرب اسکالر محاسبه کنید:

$$(x, y) = k * G$$

مقدار r را با فرمول زیر به دست آورید. اگر r=۰ باشد آنگاه به مرحله اول بر می‌گردیم:

$$r = x mod n$$

مقدار s را با فرمول زیر به دست آورید. اگر s=۰ باشد آنگاه به مرحله اول بر می‌گردیم:

$$s = (z + r * d)/k \, mod n$$

امضا به صورت (r, s) به دست می‌آید.

یادآور می‌شویم که در مرحله چهارم، اگر جواب‌ها به صورت اعشاری باشند، صورت باید در معکوس مخرج ضرب شود. در مرحله اول، این نکته مهم است که k در امضاهای مختلف تکرار نشده باشد و این امر توسط شخص ثالث قابل حدس نیست. بدین ترتیب، k باید تصادفی باشد یا توسط ابزارهای

تعیین کننده‌ای که از اشخاص ثالث پنهان شده است تولید شود. در غیر این صورت، استخراج کلید خصوصی از مرحله ۴ امکان‌پذیر می‌شود زیرا k ،z ،s و n مشخص می‌باشند.

فرض می‌کنیم که داده ما دارای عدد ۱۷ باشد. متغیرهای ما به شکل زیر خواهند بود:

z= ۱۷ (داده)

n = ۷۹ (ترتیب)

G = (2, 22) (نقطه پایه)

d = ۲ (کلید خصوصی)

انتخاب عدد تصادفی نیز به شرح زیر است:

عدد k = عدد صحیحی بین ۱ الی n-۱

k = ۳ عدد (برای ساده‌تر شدن مثال، این عدد را انتخاب کردیم)

نقطه را محاسبه کنید. این امر به همان روش تعیین کلید عمومی انجام می‌شود با این تفاوت که به منظور خلاصه کردن این عملیات، محاسبات را برای افزودن نقطه و دوبرابرکردن مختصات آن حذف کرده‌ایم:

$$3G = (x, y)$$

$$(x, y) = G + 2G$$

$$(x, y) = (2, 22) + (52, 7)$$

$$(x, y) = (62, 63)$$

$$x = 62$$

$$y = 63$$

یافتن r

$$r = x \bmod n$$

$$r = 62 \bmod 79$$

$$r = 62$$

یافتن s

$$s = (z + r * d) / k \bmod n$$

$$s = (17 + 62 * 2) / 3 \bmod 79$$

$$s = (17 + 124) / 3 \bmod 79$$

$$s = 141 / 3 \bmod 79$$

$$s = 47 \bmod 79$$

$$s = 47$$

توجه داشته باشید که در مثال فوق، می‌توانیم اعداد را بر ۳ تقسیم کنیم زیرا جواب ها اعداد صحیح بودند. در موارد واقعی، از معکوس k استفاده می‌کنیم که خواهیم داشت:

$$s = (z + r * d) / k \bmod n$$
$$s = (17 + 62 * 2) / 3 \bmod 79$$
$$s = (17 + 124) / 3 \bmod 79$$
$$s = 141 / 3 \bmod 79$$
$$s = 141 * 3^{-1} \bmod 79$$
$$s = 141 * 53 \bmod 79$$
$$s = 7473 \bmod 79$$
$$s = 47$$

امضای ما در حالت $(r, s) = (62, 47)$

همانند کلید عمومی و کلید خصوصی، این امضا نیز معمولا توسط رشته هگزادسیمال بیان می‌شود.

تایید امضا با استفاده از کلید عمومی

اکنون چندین داده و امضا برای داده مورد نظر داریم. شخص ثالثی که کلید عمومی داشته باشد می‌تواند اطلاعات و امضا را دریافت کند و تایید کند که ما ارسال کننده هستیم. در ادامه به نحوه کار این مورد می‌پردازیم.

در این مثال Q را کلید عمومی در نظر می‌گیریم و سایر متغیرات در مراحل تایید امضا به شرح زیر می‌باشند:

- تایید مقدار r و s بین ۱ الی n-۱
- $w = s^{-1} \bmod n$
- $u = z * w \bmod n$
- $v = r * w \bmod n$
- نقطه $(x, y) = uG + vQ$

تایید $r = x \bmod n$. در غیر این صورت امضا نامعتبر است.

چرا این مراحل درست می‌باشند؟ ما از مرحله اثبات می‌گذریم. در این خصوص، از دستورالعمل زیر پیروی کنید. بار دیگر متغیرات ما به شرح زیر است:

$$z = 17$$

$$(62, 47) = (r, s)$$

$$n = 79$$

$$G = (2, 22)$$

$$Q = (52, 7)$$

تایید مقدار r و s بین ۱ الی ۱-n

$$r: 1 <= 62 < 79$$

$$s: 1 <= 47 < 79$$

محاسبه w:

$$w = s^{-1} \bmod n$$
$$w = 47^{-1} \bmod 79$$
$$w = 37$$

محاسبه u:

$$u = zw \bmod n$$

$$u = 17 * 37 \bmod 79$$

$$u = 629 \bmod 79$$

$$u = 76$$

محاسبه v:

$$v = rw \bmod n$$

$$v = 62 * 37 \bmod 79$$

$$v = 2294 \bmod 79$$

$$v = 3$$

محاسبه نقطه (x, y):

$$(x, y) = uG + vQ$$

دو برابر کردن مختصات نقطه و افزودن نقطه را در uG و vQ به طور جداگانه محاسبه می‌کنیم.

$$uG = 76G$$

$$uG = 2(38G)$$

$$uG = 2(2(19G))$$

$$uG = 2(2(G + 18G))$$

$$uG = 2(2(G + 2(9G)))$$

$$uG = 2(2(G + 2(G + 8G)))$$

$$uG = 2(\ 2(G + 2(G + 2(4G)\ )\ )\ )$$

$$uG = 2(\ 2(G + 2(G + 2(\ 2(2G)\ )\ )\ )\ )$$

برای راحتی کار، ۷۵ عملیات افزودن متوالی را به ۶ عملیات افزودن نقطه و دو برابر کردن مختصات نقطه کاهش می‌دهیم. هنگامی که اعداد بسیار بزرگ باشند این امر بسیار کارآمد است.

در نتیجه، عملیات را از آخر به اول بیان می‌کنیم:

$$uG = 2(\ 2(G + 2(G + 2(\ 2(\ 2(2, 22)\ )\ )\ )\ )\ )$$

$$uG = 2(\ 2(G + 2(G + 2(\ 2(52, 7)\ )\ )\ )\ )$$

$$uG = 2(\ 2(G + 2(G + 2(25, 17)\ )\ )\ )$$

$$uG = 2(\ 2(G + 2(\ (2, 22) + (21, 42)\ )\ )\ )$$

$$uG = 2(\ 2(G + 2(13, 44)\ )\ )$$

$$uG = 2(\ 2(\ (2, 22) + (66, 26)\ )\ )$$

$$uG = 2(\ 2(38, 26)\ )$$

$$uG = 2(27, 40)$$

$$uG = (62, 4)$$

اکنون برای vQ داریم:

$$vQ = 3Q$$

$$vQ = Q + 2Q$$

$$vQ = Q + 2(52, 7)$$

$$vQ = (52, 7) + (25, 17)$$

$$vQ = (11, 20)$$

با کنار هم قرار دادن این موارد خواهیم داشت:

$$(x, y) = uG + vQ$$

$$(x, y) = (62, 4) + (11, 20)$$

$$(x, y) = (62, 63)$$

مشخصا مرحله پنجم، قسمت عمده ای از عملیات را به خود اختصاص می‌دهد. در مرحله آخر داریم:

تایید $r = x \bmod n$

$$r = x \bmod n$$

$$\bmod 79\ 62 = 62$$

$$62 = 62$$

اکنون امضای ما معتبر است.

نتیجه گیری

ما در این مقاله به توضیح رابطه پیچیده ریاضیاتی بین کلید عمومی و کلید خصوصی پرداختیم. هم‌چنین مشاهده کردیم که چطور حتی در ساده ترین مثال‌ها نیز، اصول ریاضیاتی امضاها و تایید به چه سرعتی پیچیده می‌شوند. این پیچیدگی بسیار زیاد را هنگامی که پارامترهای حاضر اعداد ۲۵۶ بیتی می‌باشند، می‌پذیریم. ما در این مقاله مشاهده‌کردیم که کاربرد هوشمندانه ساده‌ترین رویه‌های ریاضیاتی می‌تواند مسیر یک‌طرفه ضروری برای حفظ عدم تقارن اطلاعات ایجاد کند که مالکیت بیت کوین را تعریف می‌کند. هم چنین به اعتماد مجددی به استحکام این سیستم دست یافتیم که شامل محافظت کاملا ایمن از کلیدهای خصوصی ما می‌باشد.

به عبارت دیگر به این دلیل است که می‌گویند بیت کوین با علم ریاضیات پشتیبانی می‌شود. اگر به مسائل پیچیده کدنویسی علاقه دارید امیدواریم که این مقاله به شما کمک کند تا قدم بعدی را در بحث مورد نظر بردارید.

طراحی

حال با الگوریتم‌های گفته شده و فرآیندهای لازم در بلاک‌چین زبان‌های موجود برای این کار را بررسی می‌کنیم.

انتخاب زبان

اگر هم بخواهیم به یک توسعه دهنده بلاک‌چین تبدیل شویم، اولین قدم انتخاب یک زبان برنامه‌نویسی است که بر اساس نوع پروژه شما تعیین می‌گردد. در حقیقت باید بگوییم که یک زبان برنامه‌نویسی واحد برای برنامه‌نویسی بلاک‌چین وجود ندارد، چراکه با تغییر نوع کارکرد پروژه شما زبان برنامه‌نویسی آن هم تغییر می‌کند. به عنوان مثال ممکن است که شخصی زبان پایتون را برای انجام پروژه بلاک‌چینی خود انتخاب کند، درحالی‌که توسعه دهنده‌ی دیگری از جاوا اسکریپت استفاده کند.

بنابراین باید در ابتدا مشخص کنید که چه ارز دیجیتالی می‌تواند پلتفرم پایه پروژه شما باشد، همچنین باید مشخص کنید که انتظار شما از کارکرد و هدف آن پروژه چیست. برای مشاهده بهترین و محبوب‌ترین زبان‌های برنامه‌نویسی در سال 2020 می‌توانید از مقالات منتشر شده در وب سایت‌های **IEEE** و **tiobe.com** استفاده کنید.

بر همین اساس برنامه‌نویسی بلاک‌چین را می‌توان در 4 حوزه کاری تقسیم بندی کرد:

- ایجاد و ارتقا یک شبکه بلاک‌چین
- پروژه‌هایپرلجر فابریک (fabric) جهت پیاده سازی دفتر کل غیر متمرکز
- راه‌اندازی یک ICO
- ساخت قراردادهای هوشمند و برنامه غیرمتمرکز (Dapp)

اولین چالش در مورد امنیت زبان برنامه‌نویسی است. از آن جایی که کدهای بلاک‌چین به صورت عمومی و قابل رویت برای همه کاربران است، هر فردی که بخواهد می‌تواند کدها را بررسی کرده و حتی در آن‌ها تغییراتی ایجاد کند، در نتیجه یک هکر می‌تواند با پیدا کردن باگ‌های امنیتی به داخل شبکه نفوذ کند و میلیون ها دلار پول یا ارز دیجیتال را به سرقت ببرد.

چالش دوم مربوط به مدیریت منبع می‌باشد. بدین معنی که توسعه بلاک‌چین باید همگام با نیازهای شبکه باشد. به دلیل اینکه که نمی‌توان پیش‌بینی‌های لازم را از ابتدای پروژه انجام داد، بهتر است سیستم برای پرسش‌هایی از سراسر دنیا و یا پرسش‌های محلی آمادگی لازم را داشته باشد. از سوی دیگر طراحی بلاک‌چین باید به‌گونه‌ای باشد تا بهترین کارایی را ایجاد کند. پس لزوم استفاده از برنامه‌نویسی منعطف که قابلیت اجرای موازی دستور العمل‌های مختلف را داشته باشد، اهمیت خواهد داشت.

در چالش سوم، کارایی و عملکرد زبان‌های برنامه‌نویسی مورد ارزیابی قرار می‌گیرد.

همانطور که گفته شد، یک بلاک‌چین همیشه باید بالاترین قابلیت‌های خود را به نمایش بگذارد. از همین رو، استفاده از امضای دیجیتالی قابلیت موازی‌سازی بلاک‌چین را بهبود می‌بخشد. در تایید امضای دیجیتال (digital signature) شما به چیزی بیشتر از یک تراکنش، یک امضا و یک کلید نیاز ندارید و با داشتن این سه تاییدیه می‌توانید وظایف را به موازات هم انجام دهید. این ویژگی در تمامی توابع یک بلاک‌چین مشهود است.

زبان برنامه‌نویسی سی پلاس پلاس (C++)
این زبان برنامه‌نویسی بیش از 30 سال پیش توسط استراستروپ ابداع شد. سی پلاس پلاس علاوه بر دارا بودن تمام ویژگی‌های کلیدی زبان برنامه‌نویسی C، نظیر انعطاف‌پذیری (flexibility)، امنیت (security) و کارایی(efficiency)، سعی کرده است که مفهوم شی گرایی آن را بیشتر کند. به همین علت است که زبان C++ به عنوان یک زبان برنامه‌نویسی شی‌گرا شناخته می‌شود اما C یک زبان برنامه‌نویسی ساخت یافته است.

در حال حاضر بسیار از توسعه‌دهندگان بلاک‌چین از زبان برنامه‌نویسی C++ برای طراحی هسته اولیه بلاک‌چین استفاده می‌کنند. البته به خاطر وابستگی زیاد C++ به نوع متغیرها و دستورات قدیمی، استفاده از آن برای برنامه‌نویسان تازه‌کار توصیه نمی‌شود. با این وجود اگر در استفاده از این زبان برنامه‌نویسی مهارت کافی را پیدا کنید، درک عمیقی از سایر زبان‌های برنامه‌نویسی بدست می‌آورید.

زبان برنامه‌نویسی جاوا (Java)

زبان برنامه‌نویسی جاوا (Java) پادشاه صفحات وب HTML/Css است و بدلیل ویژگی غیر قابل تغییر بودن (immutability) آن که مانع از هک و اقدامات خرابکارانه می‌شود، برای ایجاد بلاک‌چین‌های محرمانه با امنیت بالا مورد استفاده قرار می‌گیرد.

زبان برنامه‌نویسی گو (GO)

زبان برنامه‌نویسی Golang یا به اختصار GO، در سال 2007 توسط شرکت گوگل ایجاد شد، اما به مرور زمان و با شناخت کارایی‌های آن در سال 2012 مورد استقبال جامعه برنامه‌نویسان قرار گرفت. زبان Go، یک زبان برنامه‌نویسی قوی و چندمنظوره است که در عین داشتن سادگی، کارایی و امنیت بسیار بالایی از خود نشان داده است. علاوه بر این، زبان Go یک زبان مفسری محسوب می‌شود و قادر است تا به صورت مستقیم با سیستم عامل ها کار کند. این ویژگی سبب شده تا از این زبان در بخش‌های مختلف توسعه یک پروژه‌ی مبتنی بر بلاک‌چین استفاده شود.

در حال حاضر اتریوم SDK پروتکلی بر اساس زبان برنامه‌نویسی GO ایجاد کرده است که برای تغییر در یک بلاک‌چین از آن استفاده می‌شود. همچنین بنیاد لینوکس از زبان Go برای توسعه پروژه‌های پرلجر فابریک بهره می‌برد.

زبان برنامه‌نویسی پایتون (Python)

Python با هدف ایجاد سادگی و خوانایی در کدها و دستورات یک زبان برنامه‌نویسی، توسط شخصی به نام Guido van Rossum ابداع شد. بسیاری از افرادی که به تازگی وارد دنیای برنامه‌نویسی شده‌اند به زبان پایتون علاقه فراوانی دارند، چراکه فراگیری آن آسان است و زبان برنامه‌نویسی مدرن و کارآمدی محسوب می‌شود.

با وجود اینکه به وسیله زبان پایتون به تنهایی نمی‌تواند ساختاری مبتنی بر بلاک‌چین را ایجاد کرد، اما باید گفت که تقریبا در تمامی بلاک‌چین‌ها، یک یا چند ابزار عمومی با پایتون و یا برای این زبان وجود دارد.

زبان برنامه‌نویسی جاوا اسکریپت (Javascript)

جاوا اسکریپت، اولین زبان برنامه‌نویسی محسوب می‌شود که برای ایجاد واسط‌های کاربری تکامل یافته و بهبود صفحات HTML، CSS به وجود آمد. امروزه تقریبا تمامی مرورگرها از جاوا اسکریپت به خوبی پشتیبانی می‌کنند.

Javascript با کمک واسطه‌هایی چون انیمیشن‌ها، منوهای کاربران، کادرهای گفتگو و نقشه‌های تعاملی توانسته است تا مسیر تکامل خود را طی کند و سبب بهتر شدن رفتارهای صفحات وب در مرورگرهای جدید و امروزی شده است. جاوا اسکریپت یکی از زبان‌های برنامه‌نویسی است که روز به روز در حال تکامل و بهتر شدن است و برای افراد تازه کار زبان نسبتا آسانی به شمار می‌رود.

استفاده از جاوا اسکریپت در پروژه‌های مبتنی بر بلاک‌چین، برای اولین بار در پلتفرم **لیسک** به کار گرفته شد.

توسعه دهندگان پروژه لیسک معتقدند که توسط جاوا اسکریپت می‌توان یک اکوسیستم کامل بر روی بلاک‌چین را پیاده سازی کرد. از همین رو پلتفرم لیسک امکان ساخت و پیاده‌سازی برنامه‌های مبتنی بر بلاک‌چین را با زبان جاوا اسکریپت برای برنامه‌نویسان فراهم کرده‌است.

انتخاب توابع کمکی

برای نوشتن این پروژه از پایتون استفاده شده‌است. دلیل این انتخاب وجود کتاب‌خانه‌های فراوان و توابع کمکی بیشتر در این زبان است که باعث سرعت بخشیدن به عمل برنامه‌نویسی می‌شود. در این بخش با مؤلفه‌هایی که به ما کمک می‌کنند آشنا می‌شویم.

رمزگذاری با sha :

SHA-256 از خانواده توابع هش رمزنگاری SHA-2 است که توسط سازمان NSA طراحی شده است. SHA (Secure Hash Algorithms) یکی از الگوریتم‌های ایمن هش است. توابع هش رمزنگاری شده یک سری عملیات‌های ریاضی هستند که بر روی داده‌های دیجیتالی اعمال می‌شوند؛ یک شخص با مقایسه هش محاسبه شده (خروجی که از درون الگوریتم هش در می‌آید) با مقدار هشی که مورد انتظار وی است، می‌تواند درستی و صحت یک داده را تشخیص دهد. داده‌ها می‌توانند در یک فرآیند یک‌طرفه به هش تبدیل شوند، اما هش تولیدشده را نمی‌توان تبدیل به داده اولیه کرد.
الگوریتم Sha-256 براساس متد ساختاری مرکل-دامگارد (Merkle-Damgard) ساخته شده است. بر همین اساس اطلاعات اولیه فورا به بلاک‌هایی تقسیم شده و پس از تغییراتی که بر روی آن صورت می‌گیرد، اطلاعات اولیه به یک خروجی 16 کلمه‌ای تبدیل می‌شود.
SHA-256 در بخش‌های مختلفی از شبکه بیت کوین مورد استفاده قرار می‌گیرد: استخراج از SHA-256 به عنوان الگوریتم اثبات کار استفاده می‌کند. به منظور بهبود امنیت و حریم خصوصی، از SHA-256 برای ایجاد آدرس‌های بیت کوین استفاده می‌شود.

تبدیل داده با Json :

JSON واژه اختصاری عبارت JavaScript Object Notation به معنای "نشانه‌گذاری شئ جاوا اسکریپت" است. البته به معنای آن توجه زیادی نکنید چون معمولا ترجمه این عبارات مفهوم دقیقی ارائه نمی‌دهند.
جیسون غالباً برای ارسال داده از یک وب سرور به یک صفحه وب استفاده می‌شود. جیسون خود توصیف (self-describing) است یعنی فهم کدهای آن به دلیل ساختار نام/مقدار (name/value) بسیار آسان است.

رمزگذاری با sha :

SHA-256 از خانواده توابع هش رمزنگاری SHA-2 است که توسط سازمان NSA طراحی شده است. SHA (Secure Hash Algorithms) یکی از الگوریتم‌های ایمن هش است. توابع هش رمزنگاری شده یک سری عملیات‌های ریاضی هستند که بر روی داده‌های دیجیتالی اعمال می‌شوند؛ یک شخص با مقایسه هش محاسبه شده (خروجی که از درون الگوریتم هش در می‌آید) با مقدار هشی که مورد انتظار وی است، می‌تواند درستی و صحت یک داده را تشخیص دهد. داده‌ها می‌توانند در یک فرآیند یک طرفه به هش تبدیل شوند، اما هش تولید شده را نمی‌توان تبدیل به داده اولیه کرد.
الگوریتم Sha-256 براساس متد ساختاری مرکل-دامگارد (Merkle-Damgard) ساخته شده است. بر همین اساس اطلاعات اولیه فورا به بلاک‌هایی تقسیم شده و پس از تغییراتی که بر روی آن صورت می‌گیرد، اطلاعات اولیه به یک خروجی 16 کلمه‌ای تبدیل می‌شود.
SHA-256 در بخش‌های مختلفی از شبکه بیت کوین مورد استفاده قرار می‌گیرد: استخراج از SHA-256 به عنوان الگوریتم اثبات کار استفاده می‌کند. به منظور بهبود امنیت و حریم خصوصی، از SHA-256 برای ایجاد آدرس‌های بیت کوین استفاده می‌شود.

تبدیل داده با Json :

JSON واژه اختصاری عبارت JavaScript Object Notation به معنای "نشانه‌گذاری شئ جاوا اسکریپت" است. البته به معنای آن توجه زیادی نکنید چون معمولا ترجمه این عبارات مفهوم دقیقی ارائه نمی‌دهند.
جیسون غالباً برای ارسال داده از یک وب سرور به یک صفحه وب استفاده می‌شود. جیسون خود توصیف (self-describing) است یعنی فهم کدهای آن به دلیل ساختار نام/مقدار (name/value) بسیار آسان است.

روتینگ با فلسک :

فلسک یک فریم‌ورک وب مبتنی بر پایتون است برای ایجاد سریع و ساده وب‌سرور که توسط آرمین‌روناچر توسعه داده‌شده است. تلاش برای ساده‌نگه داشتن طراحی فلسک و کوچکی فریم‌ورک و قائل نشدن بسیاری از پیش‌فرض ها برای برنامه‌نویسان دلیلی‌است که این بسته نرم‌افزار را یک میکروفریم‌ورک می‌نامند. فلسک رایگان و متن‌باز بوده و با مجوز آزاد BSD منتشر شده است.
بعضی از نقاط قوت فلسک که برنامه‌نویسان را به استفاده از آن ترغیب می‌کنند عبارتند از:

- یادگیری Flask بسیار آسان است. اگر کمی با زبان پایتون آشنا باشید با دیدن کدهای فلسک می‌توانید سر از کار آن دربیارید.
- هنگام کار با Flask دست شما باز است که کارها را مطابق میل خودتان پیش ببرید. یعنی این فریم ورک کاملا انعطاف پذیر است.
- یک جامعه قوی پشت زبان پایتون و فریم ورک فلسک قرار داد که می‌توانید هنگام به وجود آمدن مشکل روی کمک آنها حساب باز کنید.

رمزگذاری RSA :

در تمام الگوریتم‌های رمزنگاری با کلید متقارن، فرستنده و گیرنده پیام باید کلید رمز را بدانند. وقتی فرستنده پیام از کلیدی یکتا و سری برای رمزنگاری استفاده می‌کند و گیرندگان پیام از همان کلید برای

رمزگشایی بهره می‌برند، افشای کلید رمز از طریق یکی از گیرندکان پیام، امنیت همه را به خطر می‌اندازد. در چنینی وضعیتی فرستنده مجبور خواهد بود با یکایک گیرندگان بطور مجزا بر سر یک کلید سری متقارن توافق کند تا هر یک گیرنده کلید مخصوص خود را داشته و افشای آن در امنیت دیگران خللی ایجاد نکند. در این حالت فرستنده پیام باید به تعداد گیرندگان خود کلید تعریف کرده و از آنها نگهداری کند. تعریف مثلا دهها هزار کلید متقارن برای کاربران و ذخیره و بازیابی مطمئن آنها به نوبه خود مشکل بزرگی است.

در الگوریتمهای کلید عمومی برای رمزنگاری و رمزگشایی از دو کلید کاملا متفاوت استفاده می‌شود: «کلید عمومی» و «کلید خصوصی».

کلید عمومی برای رمزنگاری اطلاعات به کار می‌رود و همه آن را می‌دانند، زیرا از این کلید صرفا برای رمزکردن اطلاعات استفاده می‌شود و دشمنان با در اختیار داشتن آن نخواهند توانست داده های رمز شده توسط دیگران را از رمز خارج کنند.

کلید خصوصی کلیدی است که داده‌های رمز شده با آن رمز گشایی می‌شوند. این کلید را هیچکس حتی معتمدین و دوستان نمی‌دانند. بدین ترتیب هر موجودیت در سطح شبکه (اعم از کاربر، ماشین یا پروسه ها) نیاز به دو کلید مستقل دارد که فقط یکی از آنها حساس و سری است و باید به دقت از آن نگهداری کرد. ماهیت الگوریتم رمزنگاری به‌گونه‌ای است که در عمل نمی‌توان با در دست داشتن کلید عمومی، کلید خصوصی را استنتاج کرد.

در سال ۱۹۷۸ سه نفر به نامهای ریوست، شامیر و اَدِلمن الگوریتمی را برای پیاده‌سازی رمزنگاری کلید عمومی با یک جفت کلید معرفی کردند که به RSA شهرت یافت و در طول سه دهه اخیر بطور گسترده‌ای مورد استفاده قرارگرفته و در گذر زمان، سخت‌افزار و نرم‌افزارهای بهینه آن به بازار عرضه شد. اگر چه بعدها الگوریتم قویتری بنام El Gamal ابداع شد اما هنوز هم روش RSA در صدر فهرست الگوریتم‌های کلید عمومی قرار دارد.

فرض کنید فرستنده پیام جفت عدد صحیح و بزرگ (e,n) را بعنوان کلید عمومی برای رمزنگاری اطلاعات خود در اختیار دارد. در طرف مقابل،گیرنده نیز جفت عدد (d,n) را برای رمزگشایی پیام به‌کار می‌برد. بدیهی است که دو جفت عدد (e,n) و (d,n) با یکدیگر ارتباط زیرکانه ای دارند ولی بگونه ای نیست که بتوان با در اختیار داشتن e و n براحتی d را استنتاج کرد.با فرض وجود چنین کلیدهایی، الگوریتم RSA در نهایت سادگی به صورت زیر است:

الف)پیامی که باید رمز شود به بلوکهای K کاراکتری (k بایتی) تقسیم بندی می‌شود.

ب)هر بلوک طبق قاعده ای کاملا دلخواه به یک عدد صحیح به نام Pi تبدیل می‌گردد.

ج)با جفت عدد (e,n) به ازای یکایک بلوکهای Pi اعداد جدیدی طبق رابطه زیر بدست می‌آیند:

$$Ci = (Pi)^e \bmod n$$

د) کدهای Ci بجای کدهای اصلی Pi ارسال می‌شوند.

روش رمزگشایی داده‌ها دقیقا مثل روش رمزنگاری است یعنی با داشتن جفت عدد (d,n) بلوک‌های رمز شده بصورت زیر از رمز خارج می‌شوند:

$$Pi = (Ci)d \bmod n$$

کل الگوریتم در همینجا خاتمه می‌یابد.

در RSA،به جفت عدد (e,n) که متن به کمک آن رمز می‌شود، اصطلاحا کلید عمومی (public key) و به جفت عدد (d,n) که متن بوسیله آن از رمز در می‌آید، کلید خصوصی (private key) گفته می‌شود. نکته اساسی در RSA آن است که جهت تضمین وارون‌پذیری روش رمز، اعداد و بایستی در رابطه $e.d \bmod n = x$ صدق کنند لذا باید در انتخاب اعداد دقت کرد.

اصل اساسی دیگری که باید در رمزنگاری RSA حتما رعایت شود آن است که کدهای Pi که به هر بلوک نسبت می‌دهیم باید در شرط $Pi< n \geq ۰$ صدق کند. بنابر این اگر بلوک‌ها بصورت رشته‌های k بیتی مدل شوند، باید شرط $۲K< n$ برقرار باشد. دلیل این امر آن است که براحتی بتوان گزاره $Pi \bmod n = Pi$ را نوشت در غیر این‌صورت، در حالت کلی این گزاره درست نمی‌باشد و در این صورت رمزگشایی صحیح داده‌ها تضمین نخواهد شد.

روش انتخاب e وd که توسط ابداع کنندگان RSA پیشنهاد شده، عبارت است از:

الف)دو عدد دلخواه (اما بزرگ) p وq را انتخاب می‌شود.

ب)اعداد n وz را طبق دو رابطه زیر محاسبه می‌گردد:

$$n = p * q$$

$$z = (p-۱) * (q-۱)$$

ج) عدد d طوری انتخاب می‌شود که نسبت به z اول باشد یعنی هیچ عامل مشترکی که هر دو بر آن بخش‌پذیر باشند یافت نشود.

د)بر اساس d،عدد e طوری انتخاب می‌شود که رابطه زیر برقرار باشد: (به‌عبارتی معکوس ضربی d در پیمانه z محاسبه شده وe نامیده می‌شود)

$$(e * d) \bmod z = ۱$$

آنچه که مشخص است در کاربردهای عملی، اعداد pو qحداقل صد رقمی (صد رقم در مبنای ده) انتخاب می‌شوند یعنی این دو عدد حداقل از مرتبه ۱۰۱۰۰ هستند. در این حالت عدد صحیح متناظر با بلوک‌های Pi که طبق شرط فوق باید کمتر از n باشند، نبایستی از ۸۳ کاراکتر بیشتر باشند، زیرا:

$$p , q \approx ۱۰۱۰۰ \rightarrow n = p * q \approx ۱۰۲۰۰ \rightarrow (Pi <(۲٦٦٤\approx ۱۰۲۰۰)) \rightarrow Pi <۲٦٦٤$$

پس هر بلوک متن بایستی حداکثر ٦٦٤ بیت یا ۸۳ کاراکتر هشت بیتی باشد.

در اینجا توجه به این نکته ظریف لازم است که برای محاسبهAe mod n لازم نیست که A به تعداد e بار در خودش ضرب و سپس باقیماده اش بر n پیدا شود زیرا با استفاده از برخی خواص ریاضی نتیجه محاسبات هیچگاه از n فراتر نمی‌رود.

حال فرض کنید یک نفوذگر بخواهد با در اختیار داشتن کلید عمومی (e,n)، را بدست آورد .در این صورت باید در وهله اول n را به دو عامل اول آن یعنی p و q تجزیه کند تا بتواند z را محاسبه کرده و سپس d را به‌دست آورد. برای تجزیه اعداد به عوامل اول آن هیچ راهی بجز جستجو و آزمون وجود ندارد و با توجه به این که n حداقل دویست رقمی است،تجزیه چنین اعدادی حتی به کمک کامپیوتر هزاران سال طول خواهد کشید.

اگر چه تحقیق بر روی مسئله تجزیه اعداد بزرگ به عوامل آن هنوز ادامه دارد اما هنوز الگوریتم کارآمدی که بتواند اعداد بزرگ را با هر طولی در زمان ثابت یا در حد متعارف کوچکی به عوامل اول آن تجزیه کند، یافت نشده است، لذا با گذشت ۳۰ سال از معرفی RSA هنوز از ارزش آن کاسته نشده است بلکه فقط کلیدها به جهت محکم‌کاری بزرگ‌تر شده‌اند.

از آنجا که اعداداول هیچ نظم شناخته شده‌ای ندارند، لذا انتخاب اعداد اول بسیار بزرگ p و q یکی از چالش‌های بزرگ RSA است زیرا برای اثبات اول بودن عددی مثل p باید محدوده اعدادکمتر یا مساوی $\sqrt{p}$ بررسی و بخش‌ناپذیری p بر آنها مطالعه گردد که هرچه p بزرگتر باشد محدوده جستجوی $\sqrt{p}$ بزرگ‌تر خواهد بود. برای مثال محدوده جستجوی عددی ۵۱۲ بیتی از مرتبه ۲۲۵۶ می‌شود که جست‌وجوی چنین فضایی عملا غیرممکن است. بنابراین تنها راه چاره استفاده از یکسری از قضایای ریاضی است که به ما کمک می‌کنند محدوده جستجو را کوچکتر نموده و مراحل حدس زدن کمتر شود.

Binascii

ماژول binascii شامل تعدادی روش برای تبدیل بین نمایش‌های باینری و مختلف رمزگذاری شده ASCII باینری است. به طور معمول، شما از این توابع به طور مستقیم استفاده نمی‌کنید بلکه از ماژول‌های بسته بندی استفاده می‌کنید.

نتیجه گیری:

با توجه به مفاهیم، مفروضات و بررسی همه‌ی الگوریتم‌های رمزگذاری مورد نیاز نیاز به زبانی است که در الگوریتم‌های موجود را با ساده‌ترین و سریع‌ترین روش ممکن طراحی کرد. پایتون زبان برنامه‌نویسی است که با کتاب خانه‌های غنی و گسترده‌ای که دارد سرعت طراحی پروژه را بالاتر و قدرتمندتر می‌کند.

فصل ۵:

پیاده سازی

مقدمه

در این بخش شروع به ساخت یک ارز رمزگذاری شده بر بلاک‌چین می‌کنیم. این ارز باید قابلیت تراکنش‌پذیر بودن، قابلیت حفرشدن و همین‌طور غیر قابل هک و دستکاری باشد. بعضی از برنامه کلاس های گفته شده مانند sha-256 قبلا نوشته شده‌اند و ما نیازی به نوشتن دوباره آن‌ها نداریم و فقط آن ها را ایمپورت می‌کنیم.

```
import hashlib as hl

import json
import pickle
import requests
```

در قسمت شروع برنامه فانکشن‌های هش‌لیب برای رمزگذاری، جی‌سان برای تبدیل داده‌ها، پیکل برای ساخت لایه محافظت از کد و درخواست برای ساخت فرستنده به کمک فلسک را ایمپورت می‌کنیم.
اولین مرحله ساخت یک بلاک‌چین، تعریف بلاک برای آن است. برای شروع این کار اولین فایل را به نام block.py ذخیره میکنیم.

هر بلوک دارای یک «اندیس» (index)، «برچسب زمان» (timestamp) (به زمان یونیکس)، یک لیست از تراکنش‌ها، یک proof و هش بلوک قبلی است. در ادامه مثالی از چگونگی یک بلوک مجرد آمده است.

```
:class Block(Printable)
:def __init__(self, index, previous_hash, transactions, proof, time=time())
self.index = index
self.previous_hash = previous_hash
self.timestamp = time
self.transactions = transactions
self.proof = proof
```

حال فایل دیگری به نام blockchain.py می‌سازیم. این فایل جایی است که همه ی اتفاقات اصلی و توابع نوشته می‌شوند.

```
:class Blockchain
:def __init__(self, public_key, node_id)
```

```
"""نحوه تعریف کلاس در بلاک‌چین."""
# اولین بلاک بلاک چین
genesis_block = Block(0, '', [], 100, 0)
# ساخت لیست خالی تراکنش و اولین بلاک که بلاک جنسیس نام دارد
self.chain = [genesis_block]
#تراکنش خالص و خام
[] = self.__open_transactions
self.public_key = public_key
()self.__peer_nodes = set
self.node_id = node_id
self.resolve_conflicts = False
()self.load_data
```

کلاس Blockchain زنجیره بلوک‌ها و همچنین معاملات باز و گره‌ای که در آن اجرا می‌شود را مدیریت می‌کند.

ویژگی ها:

chain : لیست بلوک

open_transactions (خصوصی): لیست معاملات باز

hosting_node: گره متصل(که blockchain را اجرا می‌کند.

در اینجا باید کانستراکتور کلاس بلاک‌چین را تعریف کردیم. یک کانستراکتور در واقع عملی است که قبل از ساخت یک شی جدید در جاوا انجام می‌شود. زمانی که برنامه جاوا شما یک شی جدید از یک کلاس را ایجاد می کند در ابتدا کانستراکتور کلاس بررسی می‌شود و در صورت وجود کدهایی که در آن وجود دارند اجرا می شوند.

```
def load_data(self):
"""مقدار اولیه دادن به بلاک چین+ باز کردنو خواندن  اطلاعات از فایل."""
try:
with open('blockchain-{}.txt'.format(self.node_id), mode='r') as f:
file_content = f.readlines()
blockchain = json.loads(file_content[0][:-1])
# به دلیل هش شدن اطلاعات آن هارا باید آن هش سپس مرتب سازی شده و بعد در اختیار کاربر قرار داد
updated_blockchain = []
for block in blockchain:
converted_tx = [Transaction(
```

```
tx['sender'], tx['recipient'], tx['signature'], tx['amount']) for tx in block['transactions']]
updated_block = Block(
block['index'], block['previous_hash'], converted_tx, block['proof'], block['timestamp'])
updated_blockchain.append(updated_block)
self.chain = updated_blockchain
open_transactions = json.loads(file_content[1][:-1])
updated_transactions = []
for tx in open_transactions:
updated_transaction = Transaction(
tx['sender'], tx['recipient'], tx['signature'], tx['amount'])
updated_transactions.append(updated_transaction)
self.__open_transactions = updated_transactions
peer_nodes = json.loads(file_content[2])
self.__peer_nodes = set(peer_nodes)
except (IOError, IndexError):
pass
finally:
print('Cleanup!')
```

با این کلاس می‌تواند همیشه بلوک های جدید ساخت و بلاک‌چین را آپدیت کرد.
حال باید از بلاک‌چین موجود یک اسنپ شات تهیه کنیم. اسنپ‌شات تقریبا عمل بکاپ را انجام می‌دهد ولی لفظ درستی برای آن نیست. زیرا اسنپ‌شات فقط بر روی سیستمی که ساخته شده بود بازگردانی می‌شود و اگر برای سیستم مشکلی پیش بیاید اسنپ شات نیز بلا استفاده خواهد بود.

```
def save_data(self):
"""ذخیره بلاک‌چین + اسنپ شات در یک فایل"""
try:
with open('blockchain-{}.txt'.format(self.node_id), mode='w') as f:
saveable_chain = [block.__dict__ for block in [Block(block_el.index, block_el.previous_hash, [
tx.__dict__ for tx in block_el.transactions], block_el.proof, block_el.timestamp) for block_el in self.__chain]]
f.write(json.dumps(saveable_chain))
f.write('\n')
saveable_tx = [tx.__dict__ for tx in self.__open_transactions]
f.write(json.dumps(saveable_tx))
f.write('\n')
```

```
f.write(json.dumps(list(self.__peer_nodes)))
except IOError:
print('Saving failed!')
```

حال نیاز به ساخت اثبات کارداریم. اثبات کار در شبکه ارزهای دیجیتال با بلاک چین جدا، مانند بیت کوین، اتریوم، لایت کوین و ... یک پروتکل امنیتی است که با هدف بازدارنگی از حملات سایبری مانند حملات دیداس طراحی شده است. در این حملات که یک یا چند کامپیوتر اصلی مورد تهاجم قرار می‌گیرند، با ارسال درخواست های تقلبی و حملات گسترده، شبکه از دسترس خارج می‌شود. طبق این پروتکل ماینرها با در اختیار قراردادن کامپیوتر خود برای شبکه، نسبت به انجام کاری که انجام می دهند کوین جدید استخراج می شود و به ماینرها تعلق می‌گیرد.

```
def proof_of_work(self):
last_block = self.__chain[-1]
last_hash = hash_block(last_block)
proof = 0
# Starts guessing  to find the write anwere.
while not Verification.valid_proof(self.__open_transactions, last_hash, proof):
proof += 1
return proof
```

اثبات کار را برای معاملات باز ، هش بلوک قبلی و یک عدد تصادفی ایجاد می‌کند.

```
def get_balance(self, sender=None):
if sender == None:
if self.public_key == None:
return Noneparticipant = self.public_key
else:
participant = sender
# لیستی از کل مقادیر سکه ارسال شده برای فرد داده شده (لیست های خالی در صورت بازگشت
شخص ارسال کننده برگشت داده می شود) را فهرست می کند
# این قسمت واگذاری مبلغی از معاملات را که قبلاً در بلوک های زنجیره ای گنجانده شده بودند ،
ارسال می کند
tx_sender = [[tx.amount for tx in block.transactions
if tx.sender == participant] for block in self.__chain]
# لیستی از کل مقادیر سکه ارسال شده برای فرد داده شده را بدست می آورد
# این کد مبلغ معاملات باز شده را ارسال می کند (برای جلوگیری ازدابل اسپندینگ)
open_tx_sender = [tx.amount
```

```
for tx in self.__open_transactions if tx.sender == participant]
tx_sender.append(open_tx_sender)
print(tx_sender)
amount_sent = reduce(lambda tx_sum, tx_amt: tx_sum + sum(tx_amt)
if len(tx_amt) > 0 else tx_sum + 0, tx_sender, 0)
# تعداد سکه ها و تراکنش هایی که در بلاکـچین تاکنون انجام شده اند را محاصبه میکند.
# جلوگیری کننده از اتلاف و کم شدن سکه تا زمانی که تراکنش موفقیت آمیز ارسال شود
tx_recipient = [[tx.amount for tx in block.transactions
if tx.recipient == participant] for block in self.__chain]
amount_received = reduce(lambda tx_sum, tx_amt: tx_sum + sum(tx_amt)
if len(tx_amt) > 0 else tx_sum + 0, tx_recipient, 0)
# باز گرداندن  مقدار سکه در حساب
return amount_received - amount_sent
```

مصاحبه و بازگشت اطلاعات حساب افراد شرکت کننده در بلاکـچین

```
def add_transaction(self, recipient, sender, signature, amount=1.0, is_receiving=False):
transaction = Transaction(sender, recipient, signature, amount)
if Verification.verify_transaction(transaction, self.get_balance):
self.__open_transactions.append(transaction)
self.save_data()
if not is_receiving:
for node in self.__peer_nodes:
url = 'http://{}/broadcast-transaction'.format(node)
try:
response = requests.post(url, json={
'sender': sender, 'recipient': recipient, 'amount': amount, 'signature': signature})
if response.status_code == 400 or response.status_code == 500:
print('Transaction declined, needs resolving')
return False
except requests.exceptions.ConnectionError:
continue
return True
                                                    return False
```

مقدار جدید و همچنین آخرین مقدار blockchain قبلی را به blockchain اضافه می کند.
آرگومان ها:
sender : فرستنده سکه ها.
Recipient : گیرنده سکه ها.
Amount : مقدار سکه های ارسال شده در معامله (پیش فرض = 1.0)

```
def mine_block(self):
بلوک جدیدی ایجاد کرده و تراکنش جدید را به آن اضافه می کن
# آخرین بلاک بلاک چین را واکشی میکند.
if self.public_key == None:
return None
last_block = self.__chain[-1]
# ابلاک قبلی را هش میکند (=> برای مقایسه صحت بلاک قبلی و جلو گیری از دست کاری آن
hashed_block = hash_block(last_block)
proof = self.proof_of_work()
# ماینرها باید پاداش بگیرند ، پس بیایید یک برای کارشون پاداش ایجاد کنیم
reward_transaction = Transaction(
'MINING', self.public_key, '', MINING_REWARD)
# این تضمین می کند که اگر بنا به دلایلی معدنکاری از کار بیفتد ، تراکنش پاداش را در معاملات
باز ذخیره نمی کنیم
copied_transactions = self.__open_transactions[:]
for tx in copied_transactions:
if not Wallet.verify_transaction(tx):
return None
copied_transactions.append(reward_transaction)
block = Block(len(self.__chain), hashed_block,
copied_transactions, proof)
self.__chain.append(block)
self.__open_transactions = []
self.save_data()
for node in self.__peer_nodes:
url = 'http://{}/broadcast-block'.format(node)
converted_block = block.__dict__.copy()
converted_block['transactions'] = [
                    tx.__dict__ for tx in converted_block['transactions']]
                    try:
response = requests.post(url, json={'block': converted_block})
```

```
if response.status_code == 400 or response.status_code == 500:
print('بلاک رد شد، مشکل را برطرف کنید.')
if response.status_code == 409:
self.resolve_conflicts = True
except requests.exceptions.ConnectionError:
continue
return block
```

گاهی گره ها در بلاک چین دچار نا هماهنگی می‌شوند و باید سینکنرایز یا همگام شوند. حل اختلاف راه‌حل معمول برای این مشکل است. بلاک چین همه گره های همتا را بررسی می‌کند و آن‌ها را با اعتبار به روز شده دیگر جایگزین می‌کند.

```
def resolve(self):
winner_chain = self.chain
replace = False
for node in self.__peer_nodes:
url = 'http://{}/chain'.format(node)
try:
# درخواست میفرستد و پاسخ را باز میگرداند
response = requests.get(url)
# جی-سان را تبدیل به دیکشنری میکند
node_chain = response.json()
node_chain = [Block(block['index'], block['previous_hash'], [Transaction(
tx['sender'], tx['recipient'], tx['signature'], tx['amount']) for tx in block['transactions']],
block['proof'], block['timestamp']) for block in node_chain]
node_chain_length = len(node_chain)
local_chain_length = len(winner_chain)
زنجیره های گرفته شده را ذخیره میکند.
if node_chain_length > local_chain_length and Verification.verify_chain(node_chain):
winner_chain = node_chain
replace = True
except requests.exceptions.ConnectionError:
continue
self.resolve_conflicts = False
# زنجیر طولانیتر یا برنده را با زنجیر کوتاه تر جایگزین میکند
                                                    self.chain = winner_chain
```

```
if replace:
self.__open_transactions = []
self.save_data()
return replace
```

مراجع

1) Blockchain: Challenges and Applications; Pinyaphat Tasatanattakool; 2018

2) A Critical Review of Blockchain and Its Current Applications; Bayu Ashi et al; 2017