SONM

Supercomputer organized by network mining

www.sonm.io

SONM

(Superordinateur organisé par les réseaux des mineurs
)

Échange pour la distribution des puissants de calculs

Système d'exploitation décentralisé pour "fog cumputing".

Technologie GRIB

13.04.2017

www.sonm.io

Google Groups	Slack	Reddit
Telegram	Twitter	BitcoinTalk
Medium	Facebook	GitHub

TABLE DES MATIÈRES

1. INTRODUCTION

1.1. Qu'est-ce que SONM?

> SONM est un supercalculateur de brouillard mondial décentralisé pour une utilisation générale depuis l'hébergement de sites vers des calculs scientifiques. Le but du projet SONM est de remplacer le "Proof-of-Work" traditionnelle de la crypto-monnaie très répandue dans la communauté "blockchain" en ce moment.
>
> Contrairement aux services cloud centralisé, le projet SONM met en œuvre une structure de calcul du brouillard [1] - un ensemble décentralisé de périphériques, tous connectés à Internet (IoT / Internet of Everything).

Les acheteurs de puissance de calcul obtiennent une solution plus rentables que les services cloud comme (Amazon, Microsoft, Google Cloud, Digital Ocean etc.) peuvent offrir.

Nous utilisons le "fog cumputing" au lieu d'une "cloud structure" donc il n'est plus nécessaire de payer à l'avance comme pour le service cloud privé et monopolisé. Étant donné que la SONM est entièrement décentralisée, il n'existe pas d'autorité qui régule la distribution des ressources informatiques.

SONM possède une architecture hybride et, par conséquent, supporte tout type de tâches informatiques sans se soucier du problème de "out of gaz" d'Ethereum.

Du point de vue technique, SONM est une couche supérieure de technologies P2P sous-jacentes – le BitTorrent pour le transfert de données, la technologie open source PaaS de Cocaine en tant que plateforme informatique décentralisée, le"smart-contract" d'Ethereum en tant que PoE (Proof of Execution) et du système de consensus, BitMessage pour la communication des nœuds, etc.
Il n'y a pas de contrôle central derrière le système et pas de backdoor ou de trappes d'évacuation. Plusieurs technologies existants ont été combinées et modifiés par nos développeurs pour créer une nouvelle technologie GRIB (GRID + Blockchain).
En ce qui concerne la valeur distribuée pour les investisseurs, SONM utilise son propre jeton SNM, basé sur blockchain de l'Ethereum.

(Cliquez ici pour ignorer la description du projet et accéder à la description du jeton SONM).

Presque tous les services en ligne ont besoin de puissance de calcul pour leur produit, y compris les sites Web, les magasins en ligne, les jeux MMORPG, les entreprises utilisant de grandes bases de données et des applications. Tous ceux qui utilisent Internet pour la business auront l'option d'utiliser les jetons SONM pour résoudre leurs problèmes de puissance informatique. D'autre part, tous les internautes pourront utiliser la SONM pour recevoir des revenus passifs en fournissant leurs ressources informatiques à louer.

Cette migration continue du cloud computing centralisé vers le "fog computing" décentralisé n'arrivera pas rapidement: ce sera une transition longue et dure, mais les résultats en valent la peine. Les calculs de prix du jeton SONM montrent un ROI décent pour les premiers utilisateurs du projet.

Le prix jeton SONM est soutenu par une demande de marché stable pour la puissant de calcul et la capacité de fournir des prix plus compétitifs que les services traditionnels de cloud computing. Les actionnaires de SONM gagnent un pourcentage des transactions et des frais d'exploitation (achat-vente-développé). C'est un analogue direct de détenir des actions et de recevoir des dividendes du bénéfice d'exploitation.

Si vous êtes un mineur ou un propriétaire de puissance de calcul, SONM est une excellente occasion d'utiliser votre équipement pour des calculs utiles et le traitement de tâches réelles.

La plateforme SONM "fog computing" est un nouveau départ pour le "mining solo". Il y a beaucoup de mineurs avec des "farms" GPU qui deviennent inutiles en raison de la difficulté accrue du mining "Proof-of-work" (même pour les altcoins). Au cours des dernières années, faire partie d'un "mining pool" a été le seul moyen de garantir le profit du "mining". Mais même en faisant cela, ce bénéfice est si petit que parfois il ne couvre même pas le coût de l'électricité consacrée au minage PoW.

> *La plateforme SONM est la solution la plus rentable pour les mineurs.*

Avec SONM vous cessez de brûler votre kilowatt pour le "mining" de PoW et commencer à servir des calculs pour tout le monde dans le réseau. Pour ceux qui sont confus par la difficulté énorme ou Ethereum (et beaucoup d'autres) PoS-migration- - chaque mineur est suggéré des applications et des tâches les plus rentables pour son matériel. CPU, GPU, ASIC, même les consoles de jeux vidéos et smartphones peut être utilisé pour SONM "fog computing". Il vous suffit de configurer le logiciel de minage et de l'exécuter.

SONM est un système multi-agents, afin que chaque utilisateur soit en mesure d'utiliser les agents intelligents et des "smart-contracts" pour maximiser le profit. Vous pouvez définir votre niveau d'automatisation en choisissant chaque projet manuellement avec les paramètres d'un seul clic. Le système SONM choisira automatiquement le projet le plus rentable pour votre équipement, travaillera avec lui et recevra des paiements sur votre adresse Ethereum personnelle.
SONM est facile à installer et à utiliser, tant pour les mineurs et les acheteurs de puissance informatique.

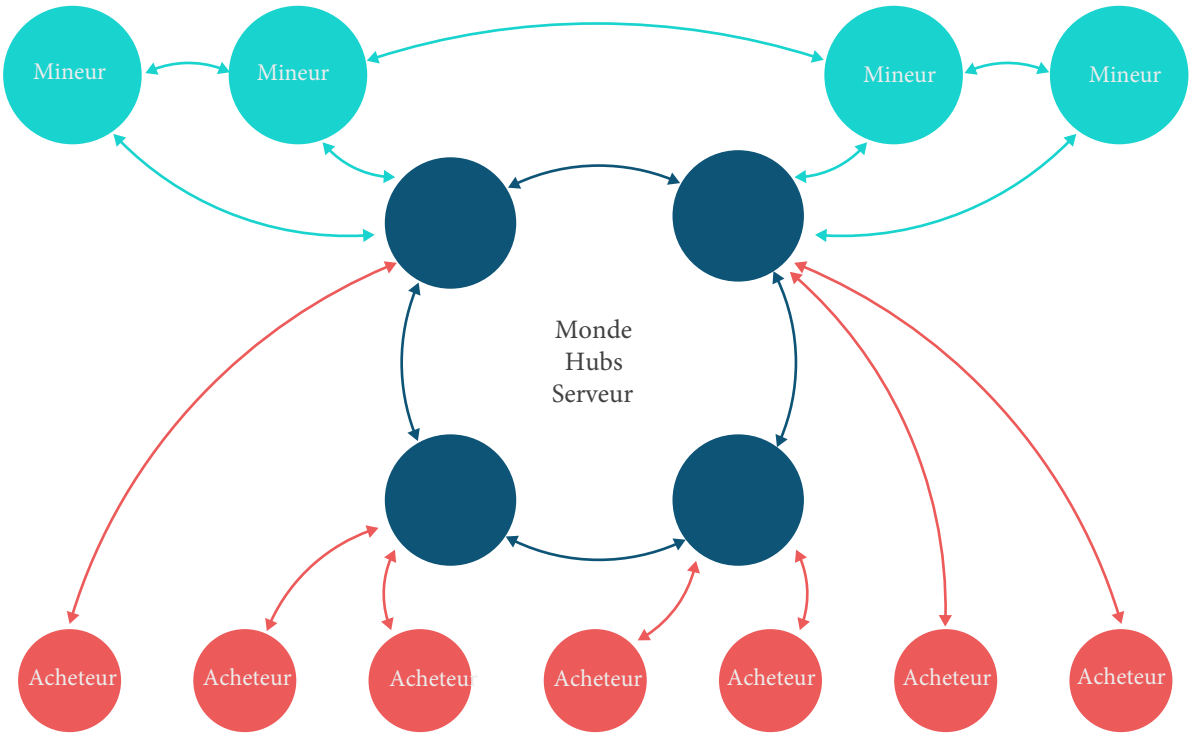
Il n'est pas nécessaire d'avoir des compétences avancées en informatiques ou d'embaucher d'un informaticien si vous utilisez SONM. Notre système d'auto-apprentissage trouve la tâche la plus rentable pour les équipements du mineur (et vice versa pour les acheteurs) et exécute cette tâche sans avoir besoin de configurer et supporte un serveur dédié.
SONM a été conçu pour être auto-apprentissage et totalement sûr pour ses utilisateurs.

Notre système prend en charge les outils d'anonymat comme le proxy, le VPN ou le TOR, mais il ne peut pas être utilisé comme un toolkit de rêve des pirates informatiques. Les agents intelligents sont capables de s'auto-éduquer en utilisant les réseaux de neurones et de garder les utilisateurs malveillants hors du système, tout en fournissant la solution de tâche la plus efficace - à la fois pour les mineurs et les acheteurs de puissance de calcul.

L'échange de puissance de calcul SONM est le marché libre, donc les centres et les utilisateurs malveillants seront ignorés par les acheteurs et les mineurs en raison de leur mauvaise réputation.

Nous nous attendons à ce que SONM soit le plus intelligent, le moins cher et le plus grand système informatique décentralisés avec des règles solides en matière de moralité et de fidélité, en raison du système de réputation du projet SONM et des agents intelligents auto-apprentissage.

SCHÉMA DU RÉSEAU



1.2. Cas d'utilisation de SONM

Nous avons de l'expérience avec les limitations de BOINC lui-même - c'est un logiciel scientifique et ne prend en charge que C ++ / FORTRAN / Python, donc ce n'est pas flexible. Nous avons commencé à utiliser des solutions plus avancées comme le conteneur Cocaine et Docker (qui supporte plus de langues, comme Java, Node.js, Go et etc.) - mais nous avons décidé que nous irons dans l'autre sens, et nous nous concentrerons davantage non seulement sur le champ de calcul distribué comme BOINC, mais plus sur le "fog computing". De cette façon, nous pouvons construire une plateforme plus universelle non seulement pour les calculs scientifiques.

1.2.1. Projets scientifiques

SONM réseau peut être utilisé pour exécuter des calculs scientifiques essentiels nécessitant une énorme puissance de calcul, par exemple :

- Statistiques sociales
- Prévisions climatiques
- Bio-informatique
- Calcul aérodynamique
- Développement de médicaments
- Modélisation
- Calcul du trajectoire des météores

1.2.2. Hébergement web

Le réseau SONM peut être utilisé pour héberger des sites Web qui ne dépendent pas des services cloud centralisés (AWS / Azure / Google Cloud, etc.) ou des fournisseurs d'hébergement. Nous utilisons la technologie open source Paas de Cocaine pour implémenter des machines virtuelles reconnues comme des serveurs, compatible avec IPFS et d'autres solutions décentralisées de stockage de données en tant que couche sous-jacente. Les propriétaires de site Web peuvent également utiliser nos extraits de code sur leurs sites Web pour collecter des paiements avec les jetons SONM ou Ether et payer automatiquement l'hébergement.

1.2.3. Cas d'utilisation du serveur de jeu

Il y a beaucoup de jeux MMO utilisant des monnaies virtuelle dans le jeu. Notre technologie offre une solution pour le déploiement de serveurs de jeu dans le réseau SONM. En outre, de monnaies peuvent être aisément remplacés par des jetons SONM et vice-versa à l'aide de notre solution d'out-of-the-box.

En revanche, les joueurs peuvent supporter leurs serveurs de jeu favoris en fournissant leurs ressources informatiques en échange de jetons ou de la monnaie du jeu.

1.2.4. Projets de réseau neuronal

Le "neural networks" sont une puissante technologie de plus en plus répandue ces dernières années. Les projects "neuro-networking" requièrent une énorme puissance de calcul de leur déploiement, apprentissage et réglage.

Le système SONM offre une solution rentable et efficace pour la mise en œuvre du réseau neuronal.

1.2.5. Le rendering vidéo et l'imagerie par ordinateur

Le redering CGI peut être distribué sur le réseau SONM entre un grand nombre de périphériques informatiques et peut être traité très rapidement (en quelques minutes).

Nous fournissons un traitement beaucoup plus rapide pour les projets informatiques CGI des **acheteurs** (**clients**) en raison de la flexibilité infrastructurelle de SONM. Par rapport à une location d'unité K80 NVIDIA d'Amazon (par exemple, pendant 10 heures), un acheteur peut utiliser le réseau SONM pour louer 600 unités K80 NVIDIA avec un temps de traitement total de 10 minutes pour chacune d'elles. Il permet l'utilisation d'une architecture distribuée plus efficacement et d'un calcul parallèle.
Contrairement aux services de cloud computing traditionnelle, SONM peut fournir aux acheteurs tout le temps de location, toute architecture informatique et toute structure de réseau informatique.

2. TECHNOLOGIE DE SONM

De nos jours, le concept populaire "Internet of Thingd" [2] (IoT) cède la place au nouveau concept émergent appelé "Internet of Everything" (IoE).

"Internet of Everything" est l'unification de toutes les ressources informatiques de l'humanité. Il a des différences fondamentales avec la technologie de "cloud computing" centralisée qui est actuellement répandue.

Afin de développer un système mettant en œuvre cette idée perturbatrice, l'équipe SONM a utilisé les technologies les plus efficaces et éprouvées comme le P2P, l'informatique distribuée et le "blockchain".

SONM n'est pas un produit monolithique, c'est une couche supérieure construite sur des protocoles et technologies sous-jacents: Ethereum, BitTorrent, Docker, Cocaine, etc.

(Par ailleurs, le créateur de Bitcoin a également combiné les technologies existantes (cryptographie, réseau de nœuds P2P, git, concept "Proof-of-work", etc.) pour apporter un nouveau système indépendant de devises / paiement décentralisé indépendant au monde.)

2.1. IoE, IoT et "fog computing"

Avant de décrire la future architecture "Ordinateur mondial", nous devons mentionner quelques détails concernant les concepts de IoE, IoT et "fog computing".

De nos jours, le concept d'Internet of Things (IoT) est communément connu.

Selon le concept IoT, Thing est un objet naturel ou artificiel capable d'avoir une adresse IP et transférer des données sur le réseau.

Internet of Everything (IoE) représente le développement ultérieur du concept IoT:

"Cisco définit Internet of Everything (IoE) comme la connexion en réseau des personnes, des processus, des données et des choses. Le bénéfice d'IoE est dérivé de l'impact composé de connecter les personnes, les processus, les données et les choses, et la valeur que cette connexion accrue crée lorsque "Everything" est en ligne.

IoE crée des opportunités sans précédent pour les organisations, les individus, les communautés et les pays de réaliser une plus grande valeur des connexions en réseau entre les personnes, les processus, les données et les choses [3].

Cette définition met l'accent sur un aspect très important de l'IoE, qui distingue IoE de IoT: le soi-disant «network effect», formulé par James Macaulay auprès du service de conseil Cisco IBSG.

Le terme "network effect" désigne une décentralisation des organisations incluses dans IoE.

Ces types de systèmes décentralisés sont développés par des groupes appelés «crypto-anarchistes» (les personnes qui implémentent des systèmes P2P décentralisés utilisant des méthodes cryptographiques [4]).

De plus, dans ce document, nous parlons d'organisations décentralisées des ressources informatiques, et non d'organisations humaines décentralisées.

La plupart des données dans l'état de développement IoT actuel sont en cours de traitement par des cloud centralisés et privés - c'est-à-dire en utilisant des technologies cloud, comme AWS, Microsoft Azure, etc.

Les technologies de cloud centrées ont plusieurs faiblesses et ne peuvent être utilisées dans IoE.

> Certaines choses dans IoE peuvent créer des quantités massives de données. Cisco donne l'exemple du moteur à réaction, qui crée environ 10 Terabytes de ses données d'activité en 30 minutes.
> Le transfert de ces données vers le cloud et la réception des résultats du traitement de l'information nécessitent une large bande passante adéquate, nécessite beaucoup de temps et peuvent avoir des délais.

En outre, les systèmes centralisés privés peuvent être compromis, influencés par l'extérieur, attaqué ou ayant des échecs, et ont également une puissance informatique plus faible que les solutions "fog computing".

Comment ces problèmes peuvent-ils être résolus?

"Fog Computing" déplace le paradigme du "cloud computing" et le déplace au niveau inférieur du réseau. Au lieu de traiter certaines tâches à l'aide du cloud, nous pouvons utiliser tous les périphériques qui nous entourent: ordinateurs personnels, smartphones, même les cafetières et les feux de signalisation.

Ginny Nichols de Cisco a initialement inventé le terme "Fog Computing". La métaphore provient du fait que le brouillard est un nuage proche du sol et, par conséquent, le brouillard informatique traite le traitement au bord

du réseau. Dans le domaine du"fog computing", le traitement des données et les applications sont concentrées dans les périphériques au niveau du réseau plutôt que d'être presque entièrement dans le cloud. Cette concentration signifie que les données peuvent être traitées localement dans des périphériques intelligents plutôt que d'être envoyées au cloud pour le traitement [5].

Ainsi, au lieu de solutions cloud centralisées, nous pouvons utiliser des systèmes de "fog computing", obtenir la puissance de calcul de chaque périphérique connecté à Internet, avec des avantages de décentralisation comme l'indépendance de tout service centralisé, une protection complète contre les éventuelles défaillances, etc.

2.2. Ordinateur mondial

Le soi-disant «brouillard informatique» est la couche de ressources informatiques capables de traiter une sorte de tâche.

Cependant, en dehors du brouillard informatique, le système implique également que ses utilisateurs définissent des tâches de calcul et un middleware distribuant ces tâches parmi les ressources de brouillard, qui renvoie le résultat des calculs.

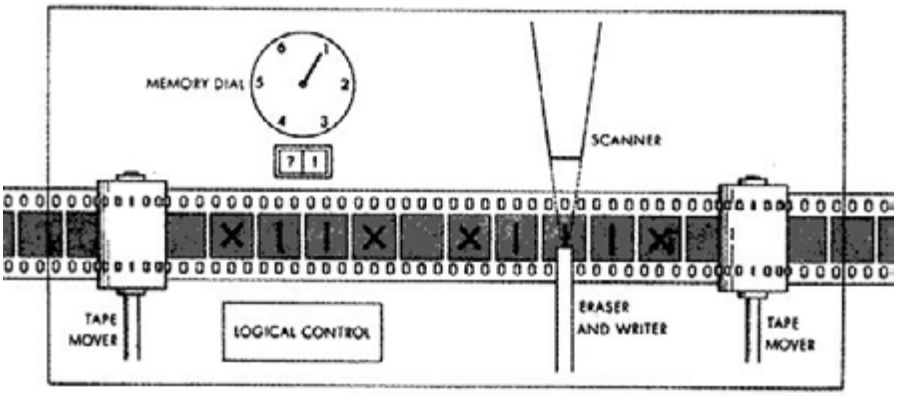
> **Ce système s'appelle "Ordinateur mondial".**

La première mention du terme «Ordinateur mondial» a été le projet Ethereum de Vitalik Buterin, qui a implémenté l'utilisation de la capacité de la chaîne de blocs pour inclure le code exécutable dans les blocs de transactions, de sorte que chaque machine de mineur exécute automatiquement ce code.

Ainsi, Ethereum est en fait l'ordinateur du monde qui fonctionne comme une machine de Turing [6], avec un "blockchain" utilisé comme ruban de registre d'état.

Cela implique également que, en raison du fait que chaque programme doit être exécuté sur chaque machine du réseau Ethereum, il est très coûteux et seules une gamme limitée de tâches peut être exécutée à l'aide de cette plateforme.

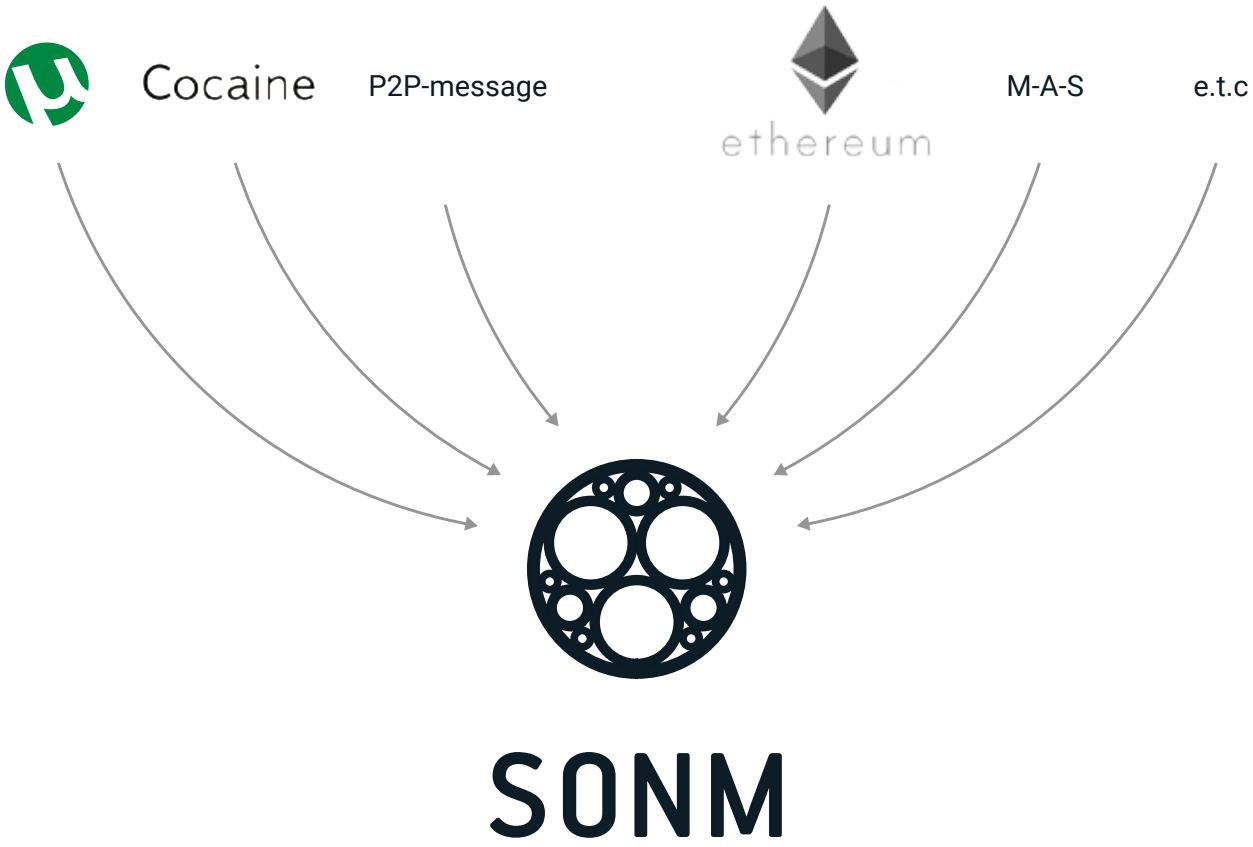
(FIG.1) MACHINE DE TURING



Il existe d'autres projets qui développent un ordinateur mondial décentralisé (Golem, iEx.Ec et autres), mais tous sont mis en œuvre en utilisant les mêmes principes que Ethereum et ont le même problème: une parallélisation excessive entraîne des coûts élevés des opérations. Ceci est dû à l'absence de tout centre de contrôle qui gère le traitement des tâches en temps réel et peut l'arrêter après avoir reçu le résultat souhaité, ce qui conduit à des processus parallèles / asynchrones.

En fait, ces projets ne peuvent pas fournir la fonctionnalité que n'importe quel ordinateur personnel habituel a de nos jours. L'équipe SONM a déjà passé 3 ans à développer un concept fonctionnel de l'ordinateur mondial capable de traiter tout type de tâches, jusqu'à la norme d'un ordinateur entièrement fonctionnel.

(FIG.2) SCHEMA DE SONM



2.4. Ordinateur mondial Architecture générale / Infrastructure

Quel genre d'architecture imaginons-nous lorsque nous parlons d'un PC? Un processeur, une carte mère, une batterie, un BIOS, un bus, un disque dur, un GPU, une mémoire RAM, etc.

Pour l'architecture informatique de notre "ordinateur mondial", nous avons décidé de suivre la façon modulaire de construire comme tous les ordinateurs personnels. (Figure 3)

À l'instar d'un ordinateur domestique standard, l'architecture du "ordinateur mondial" possède des éléments similaires: CPU, BIOS, bus pour l'échange de données, carte de plugins (périphériques connectables), périphériques, carte graphique, etc.

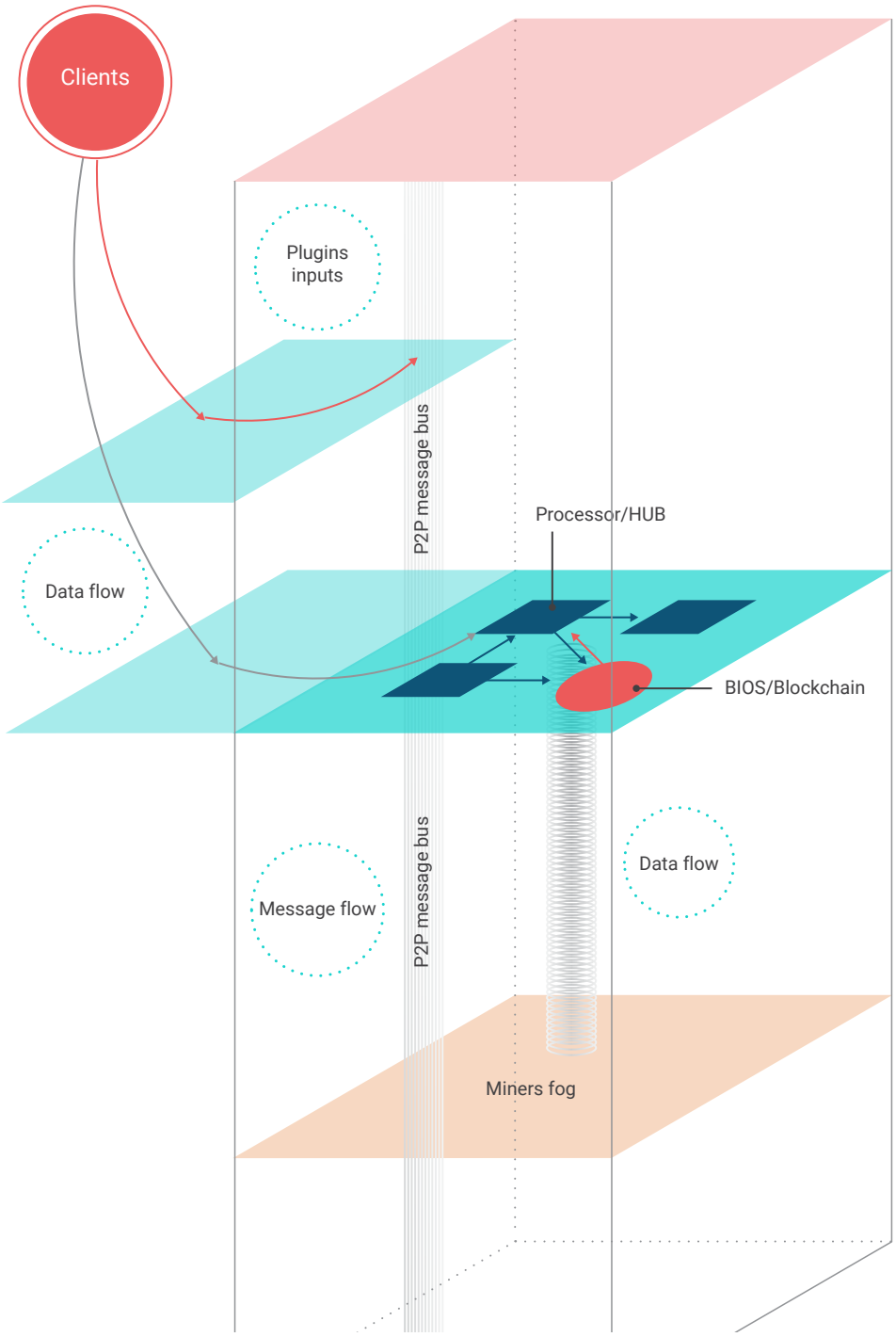
L'unité analogique du disque dur sera implémentée à l'aide de solutions décentralisées de stockage de données: IPFS (InterPlanetary File System), Storj, Sia, etc.

Le premier composant du système est le processeur. Le processeur de l'ordinateur du monde SONM est représenté par l'ensemble des nœuds de concentrateur indépendants répartissant les tâches, en assemblant les résultats des calculs, en conservant les statistiques et en assurant un fonctionnement ininterrompu du système.

Chaque noeud central sur la figure est équivalent au noyau du processeur (mais n'est pas équivalent au processeur). Il peut y avoir un nombre illimité de hubs, et ils peuvent être facilement inclus et exclus du système.

(FIG.3) SCHÉMA DE MISE EN OEUVRE DU "ORDINATEUR MONDIAL"DE SONM

Considérez les détails de cette figure de mise en œuvre de l'architecture informatique mondiale. Comme vous pouvez le voir, cette architecture comprend de nombreux éléments liés.



Les hubs ne traitent pas directement les calculs, mais ils représentent une partie très importante du système, fournissant une gestion et un support (tout comme le processeur d'un ordinateur régule et contrôle le fonctionnement du GPU, et est capable de traiter des calculs parallèles sophistiqués). Les hubs sont implémentés à l'aide des "gateway node" de Cocaine .

L'élément suivant du système équivaut au **GPU** d'un PC. Il est composé de mineur du "fog computing" et permet le traitement des calculs des tâches dans le système SONM.

Le bus de communication pour le transfert de données et de messages dans le réseau est représenté par le module de communication P2P. (Bitmessage / Slave)

Les acheteurs sont équivalents aux **périphériques PC**, généralement utilisés pour l'entrée d'informations.

La carte de plugins permet au système de se développer en permanence et d'obtenir de l'énergie en se connectant à des réseaux compatibles externes, par exemple, tout réseau Grid.

Le BIOS est une partie importante du système SONM, représenté par une "blockchain"d'Ethereum dans notre modèle informatique décentralisé. Comme nous l'avons mentionné plus tôt, les systèmes Ethereum offrent une grande fiabilité, mais ne font que des opérations de base en raison de leur architecture - c'est pourquoi Ethereum est le candidat le plus approprié pour le BIOS de l' "ordinateur mondial".

Enfin, comme nous le savons, le PC lui-même ne vaut rien sans le s**ystème d'exploitation**. Notre "ordinateur mondial" nécessite également un système d'exploitation, et nous l'avons prêt.

2.5. Infrastructure de Wwomputer en tant que service (WC IaaS)

Dans la section précédente, nous avons examiné l'architecture globale du système.

La partie infrastructure du système est gérée par un cadre de messagerie et un système de contrats intelligents (gouvernement Blockchain).

2.5.1. Le framework du messagerie Slave

Actuellement le framework du messagerie est représenté par le protocol de messagerie Slave (https://github.com/cocaine/cocaine-core/wiki/protocol)

2.5.2 API de Slave

Types communs

```
Object      ::= <Number> | <String> | <Tuple> | <Map>
Tuple       ::= ([<Object> [, <Object>]...])
```

Format général

Every message is a MessagePack-ed tuple of three fields:

```
ChannelID    ::= <Number>
MessageID    ::= <Number>
Message      ::= (<ChannelID>, <MessageID>, <Tuple>)
```

Message ID is a service slot number you're going to call. Every service has its own set of slots which can be inspected by resolving this service via the Locator. Channel ID is a way to multiplex multiple dataflows inside a single TCP session. Channel ID is generated by the caller. Tuple is a slot-specific payload.

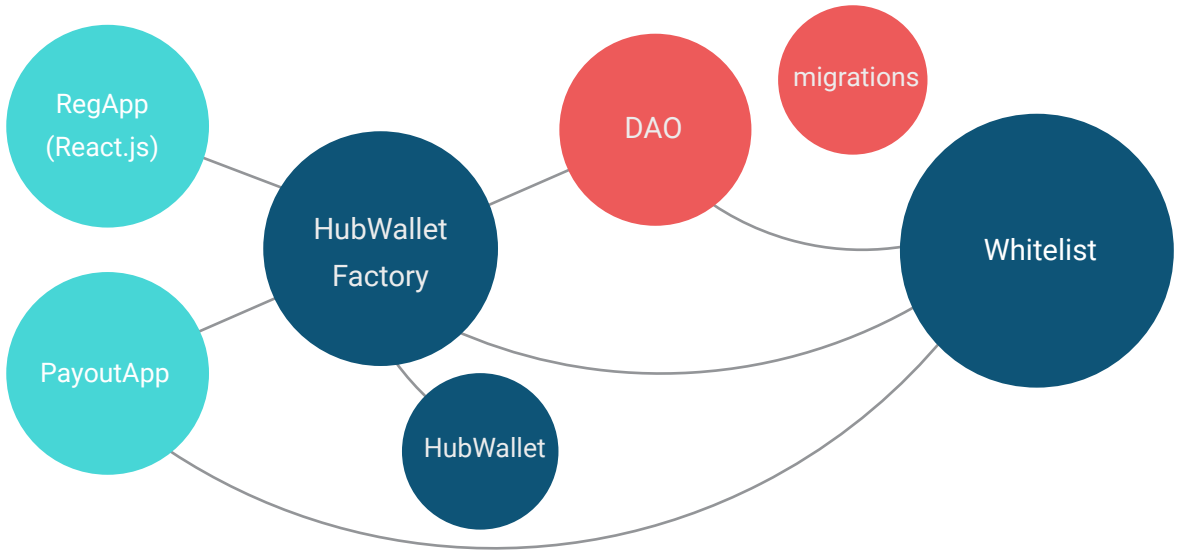
The usage of Slave will be covered more thoroughly in coming versions.

2.5.2. Le système "smart-contract"

2.5.2.1 Le gouvernement Blockchain

Le gouvernement Blockchain est une organisation (parlant métaphoriquement) composée d'un tribunal, d'un DAO, d'un registre, d'une usine d'entreprises (et d'un exemple de ladite entreprise).

Le point d'un gouvernement de Blockchain est de fournir un processus de travail simultané pour toutes les entreprises regroupées dans un tel système, les motivant à payer des «taxes» à la DAO d'un ordre supérieur, en recevant la protection judiciaire pour leur entreprise en échange, ainsi En tant que protection contre les partenaires injustes sur le marché. SONM utilise le schéma suivant pour les contrats intelligents pour réaliser le modèle d'un "gouvernement blockchain" : https://github.com/sonm-io/Contracts-scheme



Des prototypes de "smart-contract" peuvent être trouvés ici: https://github.com/sonm-io/Forge

Structure des contrats:

1. Migrations(Standard)
2. Sonm Token
3. DAO(Standard)
4. Hub wallet factory
5. Hub wallet
6. Whitelist prototype
7. RegApp (Simple React/Webpack App to work with hub registrations)
8. PayOut App (already implemented for DD@H project) https://github.com/sonm-io/drugdiscovery-token

Abstrait

La description du système de c"smart-contract" qui sera mis en œuvre dans le réseau SONM est présentée. Vous trouverez plus d'informations sur l'interaction entre les réseaux et les contrats dans le whitepaper.

Flux de données simple

HUB

Avant que le hub commence à payer des jetons aux mineurs et à recevoir des paiements auprès des acheteurs, il doit créer un portefeuille de hub - un contrat simple avec un montant fixe de fonds gelés. Si le hub est pris sur la triche, DAO peut lancer le processus de liste noire de ce concentrateur et exproprier ses fonds gelés.

Ces fonds expropriés seront également gelés au compte DAO pendant un certain temps. Il s'agit de se protéger contre les décisions malveillantes du DAO: les jetons peuvent diminuer le prix pendant le "frozen", donc il n'y a aucune motivation pour exproprier les hub.

HUB FACTORY

Le portefeuille Hub ne peut être créé que par la portefeuille Hub FACTORY(une usine de réplication simplifiée), ensuite le HUB FACTORY va crée un nouveau contrat pour la portefeuille HUBentral et l'enregistre dans un contrat «Whitelist».

WHITELIST

Le contrat "Whitelist"est un contrat de registre contenant des informations sur les hubs et leurs statuts. Tous les portefeuilles de hub créés par HUB FACTORY sont enregistrés dans ce contrat. Il est censé être un registre simple avec un mappage spécial pour les hubs 'de confiance'. Au départ, les hubs «de confiance» seront vérifiés par les développeurs SONM manuellement / les hubs SONM officiels. Plus tard, il est censé être également une liste de notation - tout le monde pourrait vérifier les HUB et l'évaluer (pariant une quantité de jetons SONM pour éviter la fraude de notation).

REGAPP

En tant que REGAPP, nous utilisons l'application React.js qui est une application Web simple (page Web) dans le but de rendre le processus d'enregistrement plus convivial.

PAYOUT APP

Payout App est une application pour traiter les opérations de mécanisme de paiement de jetons de mineurs. Pour l'instant, il est mis en œuvre pour fonctionner avec le mécanisme statistique BOINC.

2.5.2.1 Exemple d'utilisation d'un contrat «hub-wallet»

Abstrait

Avant que hub commence à payer des jetons aux mineurs et à recevoir les paiements des acheteurs - il doit créer un portefeuille de hub - un contrat simple avec un montant défini de fonds gelés. Si le hub triche - DAO pourrait initier un processus de "blacklist" de ce hub et exproprier des fonds gelés.

Ces fonds expropriés seront également gelés au compte DAO pendant un certain temps. Il s'agit de se protéger contre les décisions malveillantes du DAO: les jetons peuvent diminuer le prix pendant le "frozen", donc il n'y a aucune motivation pour exproprier les hub.

Logique

La logique des contracts

Le contrat existe en 4 états - Created, Registered, Idle, Suspected (+Punished)

Lorsque le contrat est créé, la fonction constructeur désigne les adresses du DAO, de l'usine, de la whitelist, du propriétaire du portefeuille et de quelques autres variables, comme la durée de la période de paiement (qui est actuellement réglée à 30 jours). La période de paiement est une période pendant laquelle le hub peut effectuer des paiements aux mineurs, mais ne peut pas prendre tout son solde.

Dans l'état Created, le contrat peut être enregistré dans la whitelist, en gelant un montant défini sur son solde (1 jeton SONM). Ceci est conçu pour contourner une situation comme celle-ci: le hub dépose d'abord 0,00000001 SNM, enregistre le contrat, puis dépose la somme principale de 100 SNM - le premier montant est fixé. En outre, le moment de l'enregistrement est enregistré lorsque le contrat est enregistré dans la whitelist.

Une fois que le contrat a été enregistré dans la whitelist, il devient Registered, dans cette état il a accès au transfert, au jour de paie, aux fonctions suspectes. Examinons-les de plus près dans l'ordre.

Fonction transfert

Cette fonction permet au contrat de mener des paiements aux mineurs du hub. Il fonctionne comme suit: d'abord, un lockFee est désigné, un pourcentage du paiement qui sera verrouillé pour la période de paiement. La valeur par défaut est de 30%. Ensuite, une limite est fixée (le montant total des fonds gelés + le montant gelé de l'inscription + le pourcentage pour cette transaction particulière) et le solde est coché - si le solde est inférieur à la limite, cette transaction particulière n'est pas effectuée, si tout Est en ordre - le pourcentage congelé est ajouté au montant total des fonds gelés et le contrat invoque la fonction Approuver (les détails ci-dessous) vers le mineur. L'explication de la raison pour laquelle le processus est effectué de cette façon est donnée dans la partie PayDay de la description.

Fonction Approve

Cette fonction ne déplace pas les jetons sur le portefeuille du mineur, mais permet au mineur de mener cette transaction sur son propre portefeuille. Cela empêche le hub d'enregistrer un portefeuille dans le système tout en effectuant les paiements via un portefeuille séparé car le mineur attend l'approbation de ce portefeuille particulier. Approuver est une fonction standard. (Norme ERC20).

Fonction Payday

Cette fonction définit l'état du contrat de Registered vers Idle. Cette fonction vérifie le temps d'enregistrement par rapport à la date actuelle et ne peut donc être invoquée qu'à la fin de la période de paiement. Si cette condition est remplie, elle transfère 0,5% des fonds gelés sur le portefeuille DAO, après quoi il déverrouille tous les fonds gelés et met l'état en Idle. Dans cet état Idle, le contrat peut ramener tous les fonds au portefeuille du propriétaire ou enregistrer à nouveau le contrat dans la whitelist. Pendant l'état Idle, le hub ne peut pas effectuer de paiements ni être démantelé.

Ainsi, si le propriétaire peut déplacer les fonds du hub vers son portefeuille personnel, il peut le faire de deux façons: faites-le conformément aux règles, attendez jusqu'à la fin de la période de paiement, payez le DAO 0,5% des fonds gelés Et déplacer le reste sur son porte-monnaie; Ou il peut tricher et déplacer tous les fonds en utilisant la fonction de transfert sous le couvert de payer les mineurs, mais dans ce cas, 30% de tous les fonds resteront gelés +1 SNM. Un tel système motive le hub à agir conformément aux règles.

Le contrat comporte également les conditions Suspected et Punised. Dans l'état Registered - l'état où le contrat peut être enregistré dans la whitelist - le DAO et seulement le DAO peuvent invoquer la fonction Suspected, ce qui permet de soupçonner les statistiques du contrat - soupçonné d'être malveillant. Cette fonction bloque tous les fonds sur le portefeuille du contrat pendant 120 jours. Dans l'état Suspected, les fonctions suivantes peuvent être invoquées exclusivement par le DAO:

Fonction rehab

Cela réhabilite le hub, supprime tous les gisements de fonds et met l'état du contrat en Idle. Peut être invoqué à tout moment.

Fonction Ban

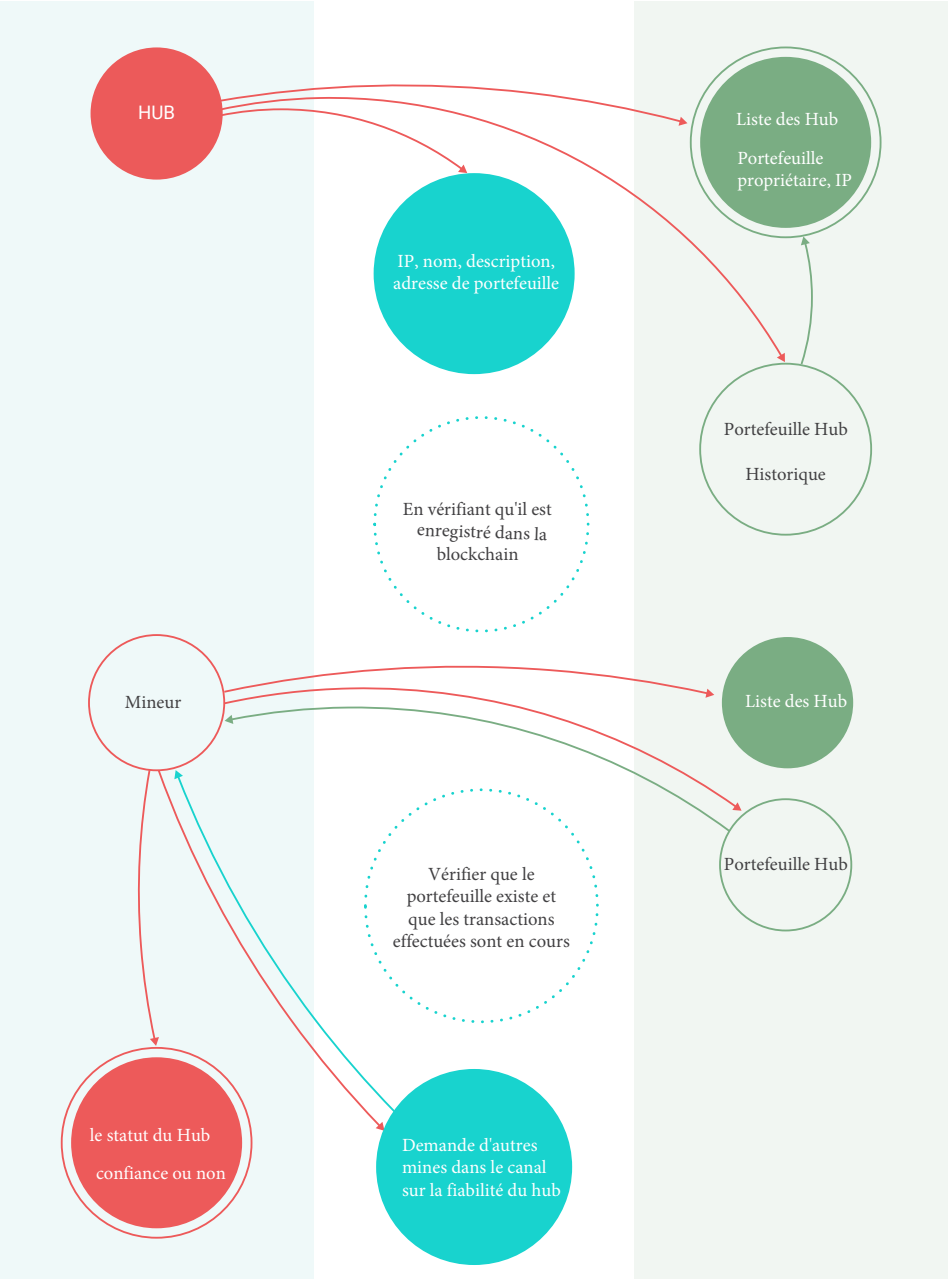
Cela ne peut être invoqué que par le comité DAO après que 120 jours se sont écoulés depuis que l'état du contrat a été suspecté, alors tous les fonds gelés des contrats sont envoyés au portefeuille DAO, l'état du contrat est définitivement puni et le propriétaire du contrat est bloqué lors de la conduite d'autres opérations utilisant ce portefeuille.

2.5.3 Solution d'interaction SONM Miner-Hub

Considérons le processus des mineurs et les hubs de la SONM qui communiquent lorsqu'ils doivent établir une coopération mutuelle (c'est-à-dire la première phase, lorsque le mineur n'a pas encore décidé de participer aux calculs et de recevoir des tâches à partir du ub ou non).

Tout d'abord, l'administrateur du hub SONM configure un contrat intelligent Ethereum contenant des jetons SONM utilisés pour les mineurs pour les calculs. Ensuite, l'adresse éthérée de ce contrat intelligent, l'adresse de l'administrateur de pool et de l'IP du hub sont enregistrées dans un "smart-contract" de SONM "Hubs Pool List"

DIAGRAMME DE L'ÉCHANGE DE MESSAGES "MINER-HUB":



La liste des pools de concentrateurs comprend les hub non confirmés et les hubs vérifiés (c.-à-d. Listés dans la whitelist des hubs). Au début, la whitelist sera gérée par l'équipe SONM, et elle ne sera formée que par les mineurs. Dans tous les cas, les informations des hub dans les contrats intelligents de SONM comprennent l'adresse du propriétaire du hub, l'adresse du portefeuille du hub et l'adresse IP du hub. Dans le cas d'une modification de l'adresse IP ou du wallet, le propriétaire du hub peut modifier l'enregistrement du hub.

Par conséquent, le hub SONM enregistre l'adresse des contrats intelligents contenant les fonds utilisés pour payer les mineurs pour les calculs (afin que les mineurs puissent vérifier l'existence de ces fonds) et enregistre des informations de base sur lui-même, y compris l'adresse du propriétaire et de la propriété intellectuelle.

Ensuite, l'agent de hub SONM commence à diffuser sur le réseau en utilisant le protocole P2P messenger, en envoyant un message diffusé sur le format: «IP, adresse du propriétaire du hub, adresse du portefeuille, nom du hub». L'agent du côté mineur écoute le canal, reçoit des messages de données des hubs, puis demande au contrat intelligent de la liste des pools de hub pour comparer les données de la whitelist des hubs avec les données dans les hubs. Le mineur peut personnaliser les paramètres de l'agent pour accepter les messages de tous les serveurs ou uniquement des listes prouvées listées dans la liste des pools des hubs.

Après cela, l'agent de blockchain du mineur demande des informations sur le portefeuille du hub, le montant des fonds dans le portefeuille du hub et les transactions récentes du portefeuille. Un agent intelligent vérifie les données reçues pour le comparer avec les conditions établies par le mineur: y a-t-il des fonds suffisants dans le portefeuille du hub, les paiements globaux aux régimes de mineurs

, quel est le montant moyen des jetons payés aux mineurs par ce hub. Ensuite, l'agent de messagerie P2P envoie un message direct au hub pour demander des méta-données supplémentaires et enregistre des informations complètes sur le hub dans sa liste de hubs avec une marque "non confirmée".

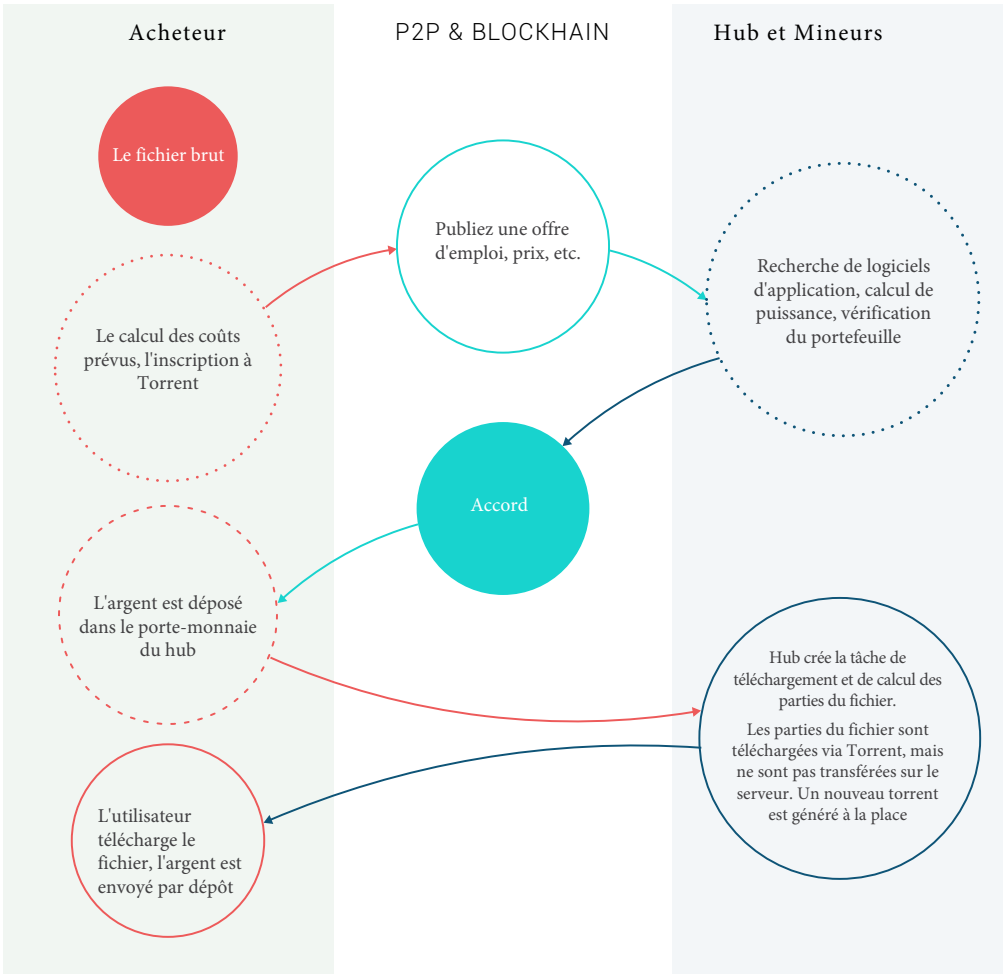
Dans le même temps, l'agent de messagerie P2P diffuse constamment des messages de questions sur le canal de données des mineurs communs pour obtenir des informations sur le hub, le montant moyen de la récompense qui leur est versée, etc. Les autres agents des mineurs diffusent des messages de réponse positifs au canal si les informations du hub dans le message de question sont corrélées avec leurs informations ou leurs réponses négatives, si elles estiment que ce hub est malveillant ou pas fiable.
Si un agent de mineur reçoit une quantité suffisante de confirmation du réseau, le concentrateur reçoit l'état «vérifié» dans la liste des centres de mineurs. Si la transaction reçue par le mineur de ce hub correspond à l'accord original, l'état de ce hub passe à "fiable". Ensuite, en fonction des paramètres du logiciel du mineur, un mineur peut soit sélectionner manuellement un hub pour se connecter et effectuer des tâches informatiques, soit un agent de mineur peut sélectionner automatiquement un hub offrant un profit maximal et se connecter à lui.

2.5.4 Solution d'interaction SONM Client-Hub

L'interaction des clients (acheteurs) avec les hub SONM est similaire à l'interaction de l'agent mineur-hub, avec une différence dans l'analyse des résultats des agents intellectuels, qui pour les acheteurs préfère les hubs avec le prix de calcul le plus bas (et vice versa pour les mineurs). Les acheteurs utilisent plus probablement le «Application Poo», que le contrat intelligent «Hub Pool».

L'acheteur crée une tâche et dépose des fonds sur le portefeuille de contrat intelligent du hub pour payer le travail. Lorsque l'acheteur reçoit le résultat des calculs, il confirme le transfert d'argent à l'aide d'une fonction contractuelle intelligente (similaire à Multisignature Wallet).

DIAGRAMME DU PROCESSUS D'INTERACTION DU CLIENT-HUB:



2.5.5. Politique d'expansion de SONM 'Blockchain-government'

Auparavant, nous avons examiné les moyens de mettre en œuvre le «blockchain-government» pour travailler avec le système SONM en utilisant les pôles de calcul en tant qu'entreprises et mineurs en tant que «travailleurs», mais que faire si nous allons au-delà du modèle de calcul et considérons que le système de contrat intelligent actuel dans un sens plus large?
Que faire si nous prenons une entreprise aléatoire et essayons de l'appliquer au système actuel? Supposons que vous soyez propriétaire d'un restaurant, auquel cas vous pouvez également déployer un contrat de hub dans le blockchain et vous inscrire dans la whitelist et effectuer vos transferts d'affaires réguliers - recevoir le paiement des clients et payer vos travailleurs, mais votre comptabilité sera relativement transparente Pour n'importe qui, vous serez sous la protection d'un DAO (un groupe de personnes partagées de personnes ordinaires qui résoudra les problèmes par le biais du vote) et votre entreprise sera enregistrée dans la whitelist, semblable au registre gouvernemental, donnant à votre entreprise une certificat d'«honnêteté» de sorte et vous donner un avantage concurrentiel.
La création du système "blockchain-government" n'est pas la priorité pour SONM, mais comme vous vous le rappeler, SONM est un assemblage. Nous supposons que ceux qui s'intéressent au système décrit ci-dessus s'inscriront dans la whitelist de la SONM, en exécutant ainsi le plan d'expansion du «blockchain-gouvernment» sur d'autres marchés et mises en œuvre.

2.5.6. Méthode de livraison du contenu SONM Client-Hub

La méthode de livraison du contenu est la seule différence significative entre les interactions client-hub et mineur-hub

Comme vous pouvez vous attendre, il n'y a pas de différence entre le rendu d'une vidéo de 6 heures à l'aide de l'ordinateur local et le téléchargement de cette vidéo vers le serveur en attendant le rendu vidéo sur le serveur distant, car la plupart du temps sera dépensé lors du téléchargement.

Nous avons développé une solution pour ce problème:

> *Lorsqu'un client souhaite charger un grand fichier de données brutes sur le serveur, SONM crée automatiquement un torrent et envoie un message au concentrateur sélectionné. Ce hub reçoit le message et crée une séquence de tâches pour le téléchargement de torrent, le travail de calcul avec les fichiers téléchargés et la création d'un nouveau fichier torrent pour les résultats de calcul.*
>
> *Après avoir traité les calculs et créé un torrent pour les données résultantes, le hub envoie un message à l'acheteur, qui n'a qu'à télécharger le fichier reçu des mineurs.*

Nous nous attendons à ce que cette solution soit la plus rapide de tous ceux qui existent actuellement.

2.6. Plateforme d'ordinateur mondial en tant que service (WC PaaS)

En tant que plateforme pour SONM, nous proposons d'utiliser SOSNA - Système d'exploitation Superglobal par / pour Architecture de réseau

.

2.6.1 Qu'est-ce que SOSNA

SOSNA est un système d'exploitation mondial basé sur le principe de la poupée de nidification. Pour mieux comprendre ce concept, passons par la structure de SOSNA du bas à la couche supérieure, allant de l'application de l'utilisateur final à l'infrastructure de couche externe.

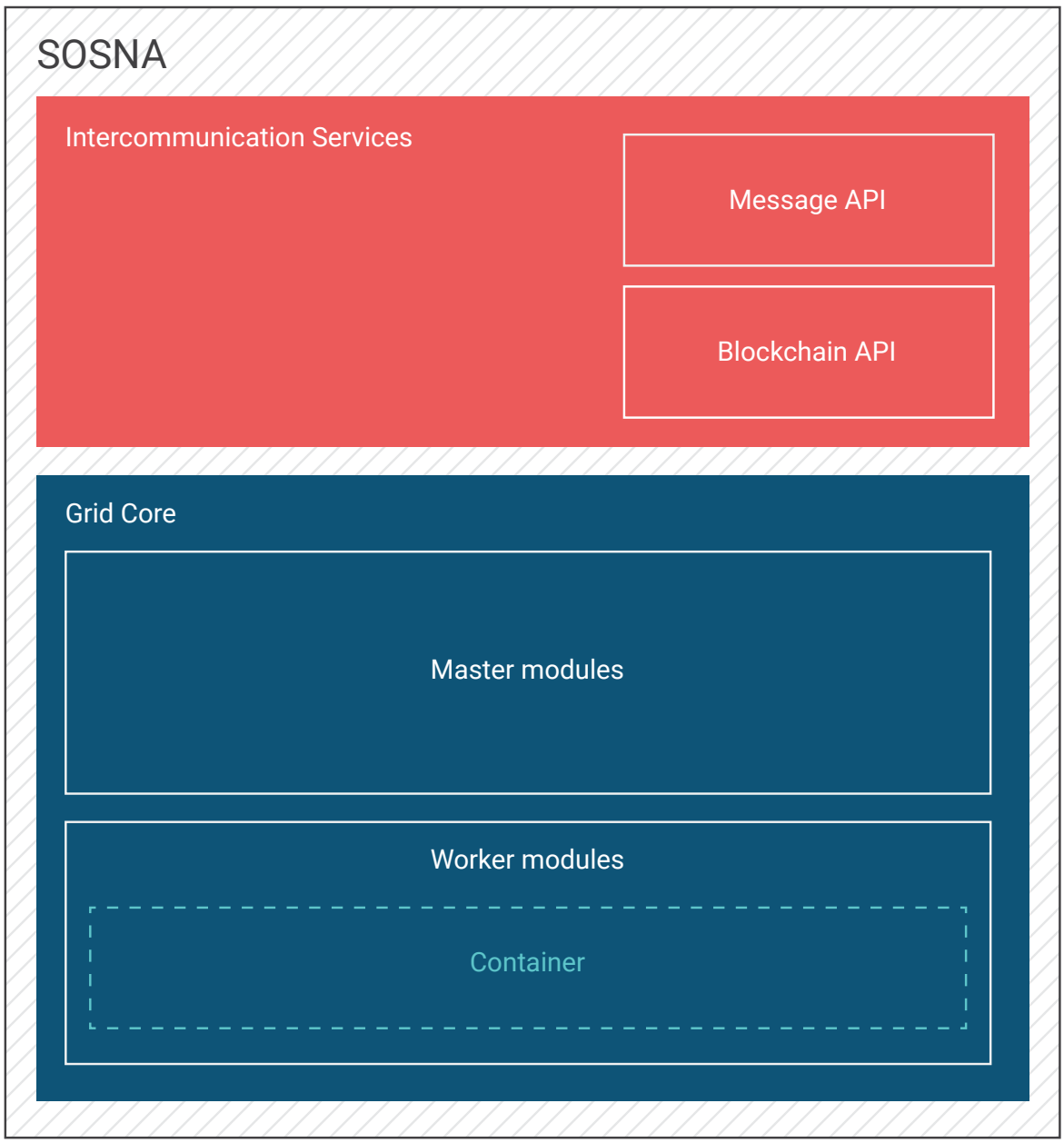
2.6.2 Applications et conteneurisation

Lorsque vous développez une application, vous devez vous assurer que pour l'utilisateur final, cela fonctionnera correctement. Mais l'ordinateur de l'utilisateur final ne possède pas la même quantité de bibliothèques que celui que vous avez utilisé pendant le développement, ou il se peut qu'il ne soit pas à jour avec des vulnérabilités, cela peut entraîner des résultats inattendus.

Existe-t-il un moyen de forcer le programme à fonctionner exactement comme il était prévu, tout en la rendant en toute sécurité pour l'utilisateur final? Pour cela, il existe des conteneurs.

Les conteneurs nous permettent d'exécuter * n'importe quel logiciel * dans un environnement sécurisé et isolé. En soi, un tel contenant est une machine virtuelle miniature, rempli de toutes les bibliothèques de dépendance de votre système, de sorte que le problème de compatibilité et les dépendances sont relativement résolus. De plus, un tel système est isolé par rapport au système hôte, de sorte que personne ne peut nuire à l'ordinateur du mineur.

SYSTÈME DE FONCTIONNEMENT SUPERGLOBAL PAR / POUR L'ARCHITECTURE DE RÉSEAU



2.6.3 Slaves & leurs service

Avançons d'un niveau.

L'hôte de mineur dans cette architecture est un nœud simple, un travailleur. (Dans l'architecture du cloud, un tel système s'appelle Slave ou Minion). Toutes les applications effectuées dans des contenants sont appelées services. Nous parlerons plus sérieusement de ce que les conteneurs sont dans le chapitre SaaS. L'hôte du mineur lui-même peut être définitivement représenté comme un ensemble de services et un système de localisation des services.

Service

Service est un acteur, un code RPC, qui accepte un certain nombre de messages. En termes techniques, chaque service diffuse un protocole de service - c'est-à-dire une liste de méthodes et leurs SlotID respectifs que vous pouvez appeler en envoyant des messages au service juste après qu'une connexion a été établie. Cette description de protocole peut être obtenue dynamiquement (avec d'autres éléments) en résolvant un nom de service via le localisateur.

> *La partie importante ici est que, conformément au modèle d'acteur, le client est également un acteur. Donc, après avoir envoyé un message à un service pour faire quelque chose pour vous, il répond en envoyant des messages aussi. Mais contrairement aux services côté serveur avec des protocoles spécifiques au service, chaque client dispense le protocole du service de diffusion, principalement pour la compatibilité et la facilité d'utilisation.*

Chaque connexion entre un client et un service est multiplexée à l'aide de ChannelIDs, et les deux extrémités d'un canal donné diffusent certains protocoles spécifiques, éventuellement différents. Par exemple, la session habituelle entre un client et un service se déroule comme suit:

- Un client se connecte à un service et choisit n'importe quel canal au hasard (par exemple, canal 1) parce que tous ne sont pas utilisés au début. Initialement, le côté service d'une chaîne diffuse le protocole spécifique au service et le côté client distribue le protocole de diffusion.

- Le client envoie un message marqué avec le ChannelID choisi pour appeler l'un des services cela indique le début d'une session.

- Le service passe son côté du canal au protocole null, de sorte que le client ne pouvait pas appeler une autre méthode dans le même canal pendant que les processus de service que vous demandez.

- Le client commence à recevoir le protocole de diffusion Chunk messages avec le service réponse .

- En fin de compte, le service envoie un message Choke pour indiquer que la session a été complétée. et remplace son côté du canal au protocole spécifique au service.

- Si c'était la seule requête, le client se déconnecte.

Notez que certains services fournissent des méthodes pouvant être transmises: dans ce cas, le service passera au protocole de diffusion en lieu et place du protocole null, afin que vous puissiez diffuser des données sur le service.

Localisateur

Lorsqu'un nœud démarre, il lit son fichier de configuration, qui contient une liste de services à exécuter. Cette liste spécifie uniquement les noms et les types de services, mais pas les propriétés liées au réseau, car la couche E / S et la couche RPC sont complètement séparées. En outre, les services eux-mêmes n'ont pas de code à communiquer sur le réseau, seul le code d'envoi des messages.

Afin de permettre à ces services de recevoir et d'envoyer des messages sur le réseau, le nœud lance un service spécial appelé localisateur. Tous les autres services sont attachés au localisateur, qui les enveloppe dans une boucle d'événement, les lie à certains points d'extrémité du réseau et les annonce dans le cluster. Le localisateur lui-même fonctionne toujours sur un port bien connu.

Ainsi, un client doit effectuer les étapes suivantes pour se connecter au service demandé:

- Connectez-vous au localisateur de service sur un port bien connu.

- Envoyer un message Résoudre avec le nom du service requis à l'aide de n'importe quel canal.

- Recevoir un message Chunk avec les informations sur le point final du service, sa version de protocole et ses cartes d'envoi (qui est un mappage des numéros de message sur les noms de méthodes).
- Recevoir un message Choke indiquant que la demande a été complétée.

- Connectez-vous au point d'extrémité spécifié et travaillez avec le service demandé.

Les services peuvent empiler des protocoles. Par exemple, le service Elliptics implémente le protocole de stockage générique et son propre protocole spécifique, ce qui signifie qu'un client qui demande un service de stockage peut être acheminé vers l'instance du service Elliptics. C'est bon, car l'empilement permet au client de travailler avec l'instance Elliptics sans même connaître les détails du protocole spécifique au service - les messages de protocole ont les mêmes SlotIDs quel que soit le service implémentant le protocole donné et s'il utilise un empilement de protocoles ou non.

2.6.4 Masters et Gateways

Avançons une couche de plus en haut. Ici, vous pouvez voir que, en plus de la machine minière elle-même, il existe une machine maîtresse, c'est-à-dire Hub, dont la fonction approximative a été considérée dans le paragraphe sur IaaS.

Master

Master gère l'exécution des services sur les machines des mineurs, maintient les statistiques, équilibre la charge, réalise la validation des résultats, dirige le planificateur de tâches, etc. - c'est-à-dire se comporte comme un pool de crypto-monnaie conventionnel.

Master est aussi appelé noeud Gateway

Gateway

En option, le localisateur peut être configuré pour agréger les annonces de multidiffusion des autres localisateurs (ou utiliser une liste fournie de noeuds distants) et servir de point d'entrée de cluster pour les clients. En d'autres termes, le travail de localisation agrégé consiste à configurer une passerelle en se connectant à tous les nœuds distants et à surveiller leurs mises à jour de santé et de service.

Les passerelles sont des modules de localisation enfichables qui fournissent des fonctionnalités de localisation à distance. Par exemple, une seule passerelle adhoc intégrée choisit au hasard un nœud distant pour chaque client et IPVS Gateway fonctionne sur un équilibreur de charge IPVS du noyau pour configurer un service virtuel local pour chaque service disponible dans le cluster.
Les clients peuvent utiliser ces localisateurs d'agrégation pour accéder à tous les services du cluster indépendamment de leur emplacement physique de manière équilibrée en charge.

2.6.5 Grid - Core

Deux machines - Master & Worker forment une implémentation de base de Grid standard - un réseau informatique couplé. Une caractéristique essentielle de la norme Grid est la condition préalable à la décentralisation et à l'éloignement géographique des maîtres des travailleurs. Par exemple, nous considérons le produit https://github. com/cocaine/cocaine-core comme exemple de Grid-Core.

2.6.6. Services d'intercommunication

Les services d'intercommunication SOSNA sont un bus de messages p2p commun, avec lequel les mineurs, les hubs et les clients communiquent, ainsi que le service API Blockhain, qui permet à SOSNA de communiquer avec Blockchain.

2.6.7. SOSNA en bref

SOSNA lui-même est une enveloppe de couche supérieure qui fonctionne avec Grid-core (BOINC, Yandex.Cocaine / Autre PaaS compatible avec le réseau) et l'infrastructure des contrats intelligents SONM.

2.7. Ordinateur Mondial SaaS et son API

Example of the simple application that can be run on SOSNA

```
#!/usr/bin/env python

from cocaine.services import Service
from cocaine.worker import Worker

storage = Service("storage")

def process(value):
    return len(value)

def handle(request, response):
    key = yield request.read()
    value = yield storage.read("collection", key)


    response.write(process(value))
    response.close()

Worker().run({
    'calculate_length': handle
})
```

2.8. Vérification des résultats

Le problème de la validation des calculs exécutés par un tiers est un sujet de recherche approfondi [7] [8], mais manque encore de solutions prêtes à l'emploi, car la plupart d'entre elles sont très coûteuses en pratique (au moins dans un paramètre HPC).

Les solutions plus pratiques sont basées sur la répétition des calculs - la vérification par réplication. Cette approche requiert des nœuds (hubs) désignés pour distribuer des unités de travail, agréger les résultats et les vérifier. Docker utilise cette approche et a une mise en œuvre hautement testée.

Pour certains types de problèmes de calcul, il est possible de décharger la tâche d'agrégation et de vérification sur un contrat intelligent. Le processus se déroule comme suit: le mineur calcule une unité de travail et publie le hash de racine de merkle-tree dans le contrat intelligent. Un autre mineur calcule la même unité de travail et les différences que les résultats diffèrent. Dans ce cas, il est possible de calculer une preuve compacte de triche. La preuve peut être vérifiée par le contrat intelligent et le tricheur puni.

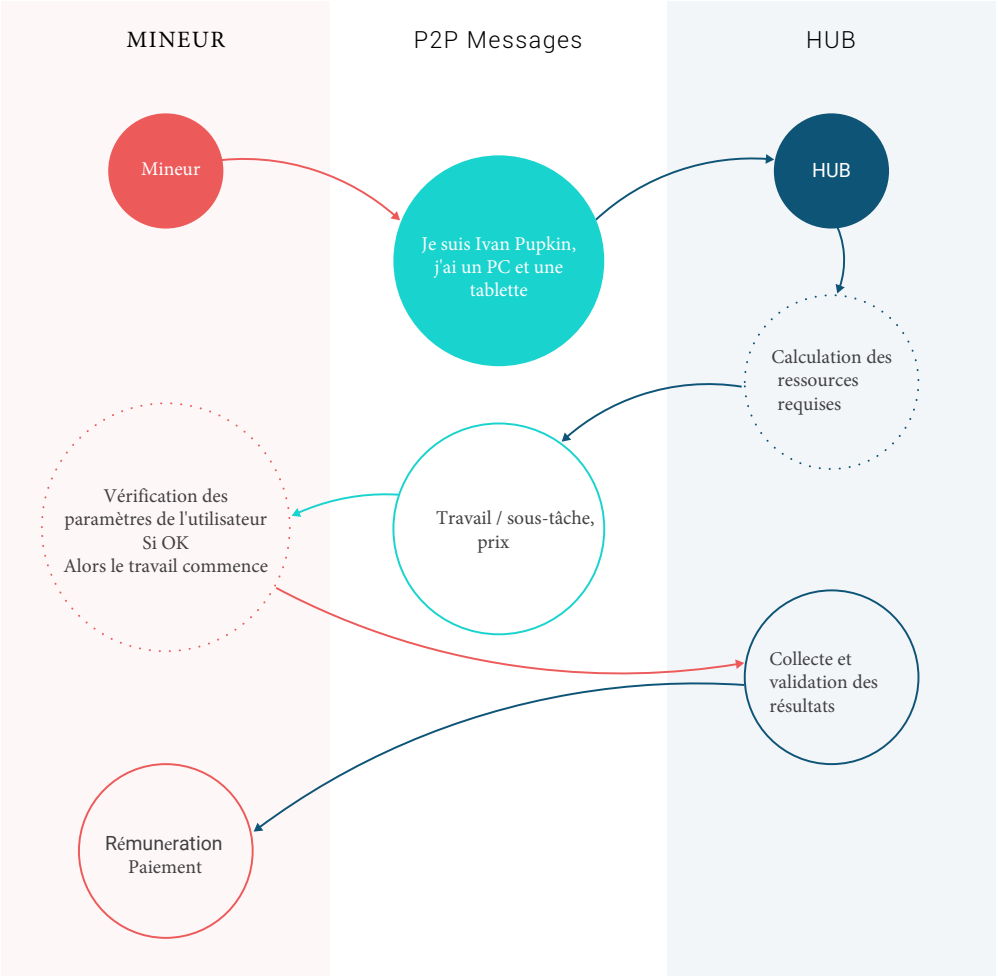
La motivation économique est utilisée pour promouvoir ce comportement de double vérification: les mineurs déposent une quantité fixe de jetons et ce dépôt sera retourné après un délai d'attente si aucune preuve de triche n'a été publiée. D'autre part, il est possible de gagner des jetons en vérifiant les calculs et les tricheurs révélateurs.

La vérification par contrats intelligents est activement recherchée [9] [10] et présente des avantages:

- *ne nécessite pas de tiers de confiance pour agréger et vérifier les résultats;*

- *n'impose aucun frais généraux dans le cas de mineurs honnêtes;*

- *a des frais généraux limités et supportables en cas de mineurs malhonnêtes.*



Notes:

Nous aurons un système entièrement fonctionnel, capable d'être utilisé pour tous les calculs à usage général, à partir de v.2.0. Très probablement, par cette étape, la plate-forme SONM aura des projets de calcul à grande échelle déployés avec un volume de roulement élevé.

En outre, par la v.2.0, nous espérons que la SONM attire beaucoup de membres de la communauté open source, ce qui signifie:

- *La communauté va créer de nombreuses applications décentralisées compatibles avec le réseau.*

- *Des marchés et des équipes tout neufs vont potentiellement apparaître, ainsi que de nombreux outils communautaires pour l'interaction avec la plate-forme SONM, probablement mieux que les applications d'origine, développées par l'équipe SONM. Par exemple, le client Ethereum officiel créé par Ethereum Foundation comparé à Parity par EthCore ou Windows Media Player par rapport à WinAmp ou Internet Explorer comparé à Mozilla Firefox. Nous le comprenons et nous l'accueillons.*

Cela signifie qu'à partir de ce point, nous devrons réduire nos efforts pour l'élaboration d'outils et céder la place au marché libre et à la communauté.

Nous nous concentrerons sur la création de nouvelles formations pour l'interaction avec ce marché:

- une équipe dévouée développant un échange de puissance décentralisé; - les équipes

fournissant des services d'hébergement de serveurs basées sur la plateforme SONM;

- logiciels pour les marchés de niche;

- divers projets d'intégration;

- Formes xternes pour le développement d'outils (comme Metamask.io par

ConsenSys)

C'est-à-dire que, à ce stade, nous aurons une division distincte des domaines de développement de la SONM. Par exemple:

- les développeurs principaux de SONM créent des protocoles de base du système;

- d'autre équipe vont crées des applications dans des solutions intelligentes;

- L'équipe SONM ExChange développe des outils conviviaux pour l'interaction avec les acheteurs et la

gestion de l'échange décentralisé;

2.9. Fiablilité et sécurité

Isolation des dockers.

L'un des logiciels de docker est un daemon - qui consiste en un serveur conteneur, lancé via la commande "docker -d"), des outils clients qui permettent à l'utilisateur de contrôler le modus et les conteneurs directement via l'interface de ligne de commande et une API qui Permet à l'utilisateur de contrôler les conteneurs via un programme REST.

Le daemon fournit une isolation complète pour les conteneurs lancés sur le noeud au niveau du système de fichiers (chaque conteneur possède sa propre racine), au niveau du processus (les processus ont accès uniquement au système de fichiers du conteneur et les ressources sont divisées Usign libcontainer), au niveau du réseau (chaque conteneur a accès exclusivement à la gamme des noms de réseau directement liés à lui et aux interfaces de réseau correspondantes).

2.10 Mise en œuvre d'AI

Notre système résout les problèmes d'optimisation combinatoire, par exemple, le problème du Knapsack et le problème du vendeur ambulant. Ces problèmes sont incomplets, donc nous avons une implémentation de base d'une IA de classe faible. Le problème du Knapsack est résolu dans le contexte de «mineurs briefcase» - comment diviser les ressources entre différents projets / hubs, avec un profit maximal et une diversification des risques? En d'autres termes, il s'agirait de «Quelles pièces de monnaie faut-il mener si btc diminue et dans quelle proportion pour chacune d'entre elles?» Le problème du vendeur ambulant est résolu dans le contexte de la répartition des ressources et du retour aux normes du réseau GRID (cette fonctionnalité n'est pas entièrement mise en œuvre pour l'instant).

2.11. Répertoire de SONM Github

github.com/sonm-io

3. FEUILLE DE ROUTE DU DÉVELOPPEMENT

3.2. La feuille de route de mise en œuvre des modules:

Smart-contracts	Platform	SOSNA core	Messaging	Ver.
PresaleToken, Presale, “Forge”	Yandex.Cocaine	-	Slave Protocol	0.1
ICO, Token	PayoutProto	-	-	0.2
DAO	Debug + Cutting off Yandex pitfalls. Payout Dapp	Waiting for contracts’ deployment. Interaction protos	Bitmessage ‘slave’ protocol implementation	0.3
“Forge” debug	DCFS (etcd, Swarm, IPFS) integration	Business logic implementation (including price API)	Sonm hub DNS reconstruction, additional messaging types and channels’ specification, debug	0.4
Whitelist, Hub wallet, Hub Factory	Locator service improvement.	Interaction with p2p message bus & ethereum blockchain API	Debug and feedback	0.5.
BugFix + Escrow	BugFix	Graphic UI	Global channels and Global DNS improvements.	1.0
Debug & feedback	Debug & feedback	Debug & feedback	Debug & feedback	1.1
				1.n
	CoreOS(https://coreos.com/)			2.0

v.0.1 Version actuelle

Yandex.Cocain comme plateforme, Docker comme

isolation. Langues prises en charge:

- C++
- Go
- Java
- Node.js
- Python
- Ruby
- [En développement]Racket

Nous avons les services suivants:

- Logging
- Node-local ile storage
- MongoDB storage
- Elliptics storage
- Node-local in-memory cache
- Distributed in-memory cache
- URL Fetch
- Jabber
- [En développement] Notifications
- [En développement]Distributed time service

Prototypes du système de contrats intelligents ("Forge"), protocole Slave pour la communication entre les noeuds.

Toute personne pourrait créer son propre hub et essayer de collecter des pouvoirs auprès des mineurs, ou créer son propre cluster (à partir de nombreuses machines appartenant). Toute personne pourrait utiliser n'importe quel conteneur docker habituel ou créer votre propre application dans le cadre de Cocaine (voir les sections ci-dessus ou github).

v.0.2 - 1 mois de dev time

Main Token Contract et ICO application. Payto proto (déjà implémenté pour la plateforme BOINC "Drug-Discovery@home.com")

v.0.3 - 2-3 mois dev time (environ un mois sera consacré aux problèmes organisationnels).

Sur cette version (si nous obtenons suffisamment d'argent), nous nous concentrerons sur les éléments les plus importants du système. Pour l'instant, le protocole esclave est littéralement «protocole» - il n'a pas de bibliothèques propres ou API, c'est juste un accord dans le système de modélisme. Nous devons le réécrire à l'aide de messages électroniques modernes p2p, comme bitmessage, pour obtenir une norme de messagerie agréable.

Sur le plan de la plateforme, ce sera également PayOut dapp - un DApp simple qui permet aux jetons de paiement de l'administrateur central aux mineurs, en fonction de leur travail - il est déjà fait pour les plates-formes BOINC comme "DrugDiscov-ery @ home", et nous avons besoin Pour l'adapter simplement à notre nouveau contrat de jeton déployé et à l'architecture du noeud de passerelle de cocaïne.

Sur le plan des contrats intelligents, il fonctionnera avec notre contrat DAO.

v.0.4 - 5 mois dev time

Dans cette version, nous ajouterons de nouveaux types de messagerie pour de nouveaux systèmes de messagerie, afin de régler la communication entre les mineurs et les hubs. Nous réécrivons probablement également le service DNS interne de reconnaissance par les pairs (il permet de trouver des pairs lors de l'écoute de la chaîne générale dans le système de messagerie).
En ce qui concerne la plateforme principale, nous travaillerons à la mise en œuvre du business modèle (marché et IA) et au réglage des messages et à l'API blockchain.

Au niveau de la plateforme, nous implémentons l'intégration avec DCFS comme IPFS ou Swarm.

Sur le plan des contrats intelligents, nous allons terminer le travail sur "Forge".

v.0.5 - 2 mois dev time

À ce niveau, tous les nouveaux contrats de "Forge" - Whitelist, HubFactory, HubWallet seront déployés. Ce sera le début de la formation d'une véritable homéostasie du système. Après cela, nous pensons que quelques versions de "débogage" seront nécessaires avec différentes propositions communautaires.

v.1.0 - Post-production (éliminer toutes les contraintes pour la version de version) - 1 mois de temps de développement si aucun problème majeur ne survient

a toute première version commerciale de cette plateforme à usage public.

Les améliorations globales du DNS et du localisateur de services nous permettent de créer un nouveau navigateur Internet, qui permettrait à tous de trouver et d'exécuter des services comme https://servicename.

Les améliorations graphiques de l'interface utilisateur pour chaque partie du système nous permettent d'améliorer l'expérience de l'utilisateur et de commencer à étendre largement parmi les «non-bitcoiners».

Nous croyons également que d'autres entreprises utiliseront notre organisation de contrats intelligents (Forge), ce qui leur permettrait d'utiliser un contrat-enregistrement et une protection équitable du système contre les utilisateurs malveillants et la fraude.

v.1.1

Amélioration UX, propositions communautaires, commentaires, débogage, etc.

v1.n

Le développement de la nouvelle version SOSNA est lancé, qui sera basé sur CoreOS (un système que vous pourriez littéralement exécuter partout - micro-ondes et machines à laver). Lisez sur CoreOS - c'est génial!

v.2.0

Sortie de SOSNA 2.0. Imaginez si vos montres intelligentes d'Apple pourraient vous gagner de l'argent? C'est ce dont nous parlons - Lorsque "Le temps est de l'argent!" N'est pas seulement des mots.

Notes:

Nous aurons un système entièrement fonctionnel, capable d'être utilisé pour des calculs à usage général, à partir de v.1.0. Très probablement, par cette étape, la plateforme SONM aura des projets de calcul à grande échelle déployés avec un volume de roulement élevé.

De plus, par la v.2.0, nous espérons que SONM attirera de nombreux membres de la communauté open source, ce qui signifie:

- La communauté va créer de nombreuses applications décentralisées compatibles avec le réseau.

- Beaucoup de nouveaux marchés et de nouvelles équipes vont potentiellement apparaître, ainsi que de nombreux outils communautaires pour l'interaction avec la plateforme SONM, probablement mieux que les applications d'origine, développées par l'équipe SONM. Par exemple, le client Ethereum officiel créé par Ethereum Foundation comparé à Parity par EthCore ou Windows Media Player par rapport à WinAmp ou Internet Explorer comparé à Mozilla Firefox. Nous le comprenons et nous l'accueillons.

Cela signifie qu'à partir de ce point, nous devrons réduire nos efforts pour l'élaboration d'outils et céder la place au marché libre et à la communauté.

Nous nous concentrerons sur la création de nouvelles formations pour l'interaction avec ce marché:

- une équipe dévouée développant un échange de puissance décentralisé;

- les équipes fournissant des services d'hébergement de serveurs basées sur la plateforme SONM;

- logiciels pour les marchés de niche;

- divers projets d'intégration;

- formations externes pour le développement d'outils (comme Metamask.io par ConsenSys)

C'est-à-dire que, à ce stade, nous aurons une division distincte des domaines de développement de la SONM. Par exemple:

- les développeurs principaux de SONM créent des protocoles de base du système;

- une autre équipe crée des applications dans des solutions intelligentes;

- L'équipe de SONM ExChange construit des outils conviviaux pour l'interaction avec les acheteurs et la gestion de l'échange décentralisé;

3.3. Diffusion de l'information sur le processus de développement

- L'équipe du projet est responsable de rendre les résultats ouverts au public et d'utiliser toutes les ressources disponibles pour diffuser des informations sur le projet.

- Au moins une fois par semaine, nous publierons un rapport sur les résultats et les problèmes de développement actuels.
- Le rapport contiendra les besoins et les problèmes actuels du projet.

- Toutes les percées majeures seront communiquées avec tous les médias de masse intéressés et se propageront dans les principaux forums communautaires tels que BitcoinTalk et CryptoCoin Talk.

4. SONM EN COMPARAISON AUX AUTRES PROJETS INFORMATIQUES GRID

4.1. SONM par rapport au réseau Golem

SONM dispose les avantages suivants par rapport à Golem:

Le réseau Golem n'a pas encore démontré de les "proof-of-concept". Le réseau Golem n'est actuellement pas en mesure de traiter les calculs à usage général. Leur réseau n'est disponible qu'en mode test que pour le rendu CGI.

De plus, SONM utilise la plateforme open source PaaS de Cocaine qui est compatible avec la plateforme BOINC plus commune et standardisée, également utilisée dans beaucoup de projets distribués existants. SONM est donc compatible avec plusieurs d'entre eux. En outre, contrairement à BOINC, la plateforme de cocaïne prend en charge SaaS (Logiciel en tant que service), des langages de programmation étendus, modernes et standard,isolés, des conteneurs sûrs et standard (Docker).

En addition,alors que nous développons SONM en utilisant de nombreuses technologies open source, nous avons déjà la base, la plateforme réseau et la plupart des autres fonctionnalités importantes du projet, et en fait, nous sommes en avance sur le projet Golem au moins par deux ans de développement.

La gamme d'applications de Golem est encore très limitée. À l'heure actuelle, les tâches testées efficacement dans Golem sont limitées uniquement au rendu de CGI dans Blender.

Golem dispose moins de fonctionnalités. Golem représente le «marché du peer-to-peer» pour les ressources informatiques. SONM est un protocole cryptographique sûr fournissant la distribution des tâches, la validation des résultats et le paiement correct proportionnel aux puissances de calcul utilisées.

Validation des résultats informatiques. Le système de validation des résultats est l'un des points faibles du projet Golem. Tous les résultats des calculs ne sont pas validés, de sorte que Golem dépend de leur système de réputation pour empêcher les utilisateurs de payer des mineurs malveillants pour de mauvais résultats de calcul. Ce système est potentiellement vulnérable et peut être exploité.

SONM utilise le système de vérification du Docker permettant de vérifier tous les résultats reçus pour une correction correcte.

4.2. SONM par rapport au projet iEx.Ec

IEx.Ec utilise son propre protocole XtremWeb-HEP. C'est un middleware similaire à BOINC, mais il est moins testé et a une communauté plus petite qui le soutien derrière lui.

Par rapport à Golem et iex.ec, nous nous attendons à accélérer le marché grâce à l'utilisation des technologies et des protocoles open source. Nous utilisons des technologies largement répandues dans le temps, donc nous avons déjà implémenté le noyau du système SONM, la plupart de ses fonctionnalités importantes et nous avons le prototype fonctionnel disponible pour les tests communautaires alpha.

4.3. SONM par rapport au projet Elastic

L'équipe élastique est anonyme. Le projet est en cours d'élaboration par la communauté élastique, et il n'y a pas de membres de l'équipe qui montrent leur identité réelle. De cette façon, en cas d'échec du projet, il n'y a personne qui assume la responsabilité

Elastic est un projet expérimental non commercial. En fait, les développeurs Elastic sont de bons ingénieurs, mais ils manquent de marketing et de relations publiques, et ne pensent pas au côté commercial de la plateforme et à la monétisation. Ils n'ont pas de modèle financier ou un plan de marketing clair, de sorte que le prix futur des tokens Elastic et la capitalisation boursière des projets n'est pas très clair. De plus, Elastic avait terminé le financement de l'ICO il y a plus d'un an, mais leurs jetons ne sont toujours pas répertoriés sur des crypto échanges, et les investisseurs de l'ICO n'ont toujours pas accès aux jetons.

Elastic utilise un pool de transactions pour les tâches, un mécanisme similaire à celui utilisé par les systèmes de "blockchain" traditionnels, tels que les cryptomonnaies. Cela entraîne un problème sérieux: un bloc de transactions doit être confirmé dans un certain temps, donc une tâche doit être traitée dans ce délai. Dans le cas de calculs à usage général (par exemple, pliage de protéines), nous ne pouvons pas savoir avec certitude combien de temps prendra-t-il pour traiter la tâche.

Elastic utilise son propre langage de programmation pour résoudre ce problème avec un mécanisme similaire àEthereum, qui conduit à un parallélisme excessif et à la nécessité d'exécuter le code sur toutes les machines du réseau.
SONM utilise un protocole BOINC modifié, qui a d'abord été développé exclusivement pour les calculs informatiques GRID et est beaucoup plus efficace pour la distribution et le traitement des tâches.

4.4. Différences avec GridCoin, FoldingCoin et CureCoin

Il existe des projets de cryptomonnaie tels que CureCoin, FoldingCoin et GridCoin déjà impliqués dans l'informatique distribuée scientifiquement.

Cependant, ces projets utilisent la sélection de la liste blanche pour les projets informatiques scientifiques. Ils n'ont pas et ne créent pas de marché de puissance informatique.

Dans notre projet, tout **acheteur** peut acheter un pouvoir informatique pour une tâche de toute taille et tout **vendeur** peut louer une puissance informatique. Par conséquent, notre principal avantage par rapport à ces projets et que SONM n'est pas limitée par une liste de projet spécifique. SONM sera un marché ouvert et décentralisé du marché de la sécurité informatique disponible pour tous les besoins. Cependant, SONM est entièrement compatible avec ces platesformes, donc, une fois que ces projets commencent à utiliser l'infrastructure SONM, on peut gagner du SNM et des jetons correspondants (FoldingCoin, GridCoin, CureCoin etc.). En outre, SONM est compatible avec les applications Grid, de sorte que toutes ces applications peuvent être exécutées dans notre système.

4.5. Compatibilité et intégration avec d'autres services informatiques décentralisés à la demande

Bien que nous ayons mentionné les différences entre Golem, Elastic Project, les réseaux iEx.Ec et SONM, soulignant certains avantages de notre projet, nous considérons la compatibilité et la possibilité de l'intégration de ces systèmes comme un grand avantage du réseau SONM et comme un moyen prometteur de Optimisation de l'utilisation de l'alimentation informatique. Par conséquent, l'objectif est de créer une plateforme informatique globale intégrée, où SONM,

Golem, iEx.Ec et des systèmes similaires peuvent être intégrés ensemble et le calcul informatique évoluera vers le système le plus rentable et le plus efficace.

L'un des principaux objectifs de notre projet est le développement d'un système intelligent, axé sur l'apprentissage profond, efficace pour les calculs informatiques et pour la résolution de tâches spécifiques.

5. LES RÉFÉRENCES

[1] *https://en.wikipedia.org/wiki/Fog_computing*

[2] IoT, https://en.wikipedia.org/wiki/Internet_of_things

[3] http://internetofeverything.cisco.com/sites/default/files/docs/en/ioe_value_at_stake_public_sector%20_analysis_faq_121913final.pdf

[4] https://en.wikipedia.org/wiki/Crypto-anarchism

[5] http://internetofthingsagenda.techtarget.com/definition/fog-computing-fogging

[6] https://en.wikipedia.org/wiki/Turing_machine

[7] Verifying computations without reexecuting them: from theoretical possibility to near practicality. Walfish, Blumberg.

[8] Making Argument Systems for Outsourced Computation Practical (Sometimes). Setty, McPherson, Blumberg, Walfish.

[9] Practical Delegation of Computation using Multiple Servers. Canetti, Riva, Rothblum.

[10] An Intro to TrueBit: A Scalable, Decentralized Computational Court. Simon de la Rouviere.

Google Groups

Slack

Reddit

Telegram

Twitter

BitcoinTalk

Medium

Facebook

GitHub