ORBS

PUBLIC BLOCKCHAIN INFRASTRUCTURE FOR ESTABLISHED CONSUMER APPLICATIONS

Position Paper

V.1.6

orbs.com

MOTIVATION

The Orbs Vision

By 2020, large scale consumer applications will have transitioned to the blockchain to bring decentralized services to billions of users worldwide. We see the beginning of this trend today as more and more established consumer brands, such as Kodak, Kik and Telegram, are launching new decentralized businesses and reinventing themselves in this space.

The masses of mainstream consumers present a unique combination of challenges that general-purpose blockchain solutions find difficult to adequately meet. We see a need to provide these decentralized consumer apps with an infrastructure solution that addresses the multitude of requirements for making this transition possible.

Until now, established consumer brands have been forced to either avoid blockchain technology altogether; defocus and develop their own custom infrastructure in-house; or settle for off-the-shelf solutions that don't scale, fee structures that break business models and a degree of liability that a real-world business cannot accept.

The Orbs project envisions commoditizing blockchain infrastructure for large scale consumer applications. We envision a fully decentralized public platform where consumer brands feel comfortable to operate nodes and take part in a decentralized and balanced ecosystem, one that makes this transition easier for the industry as a whole. The platform will be complete with a core product experience inspired by well-established infrastructure solutions such as Amazon Web Services (AWS) and speak in familiar terminology like Service Level Agreements (SLA) and dedicated resources.

Purpose of This Paper

We see a stark difference between a *white paper* and a *position paper*. A white paper takes a complex problem and revolves around the solution to that problem. A position paper revolves around the problem itself. It discusses the problem and provides an arguable opinion on how

it should be approached. This paper is designed to be the position paper for Orbs - discussing the problem of consumer-oriented blockchain infrastructure and how we approach it.

There's a tendency in the blockchain space to launch projects with a celebratory technical white paper. We believe that solving complex problems requires a mix of multiple solutions over time and that these solutions evolve. Accordingly, the Orbs project will publish a set of white papers, each dealing with a different aspect of the solution. These white papers will evolve; some may become obsolete and some may be replaced with different ones, as we continue to better understand the problems we're facing.

Nevertheless, we don't believe a white paper is the first piece of the puzzle. The first piece is articulating the problem, explaining exactly what we're trying to solve and why - a stable guide in the turbulent waters of momentary innovation. This position paper will carve the specific niche in blockchain infrastructure that the Orbs platform is focused on. It will then go into details regarding the main requirements we're optimizing towards and the various tradeoffs we're prepared to make. This paper will also introduce some of the solutions we're working on, with dedicated technical white papers to be published separately about each one.

INTRODUCTION

Decentralized Consensus

There's no argument that cryptocurrencies have an exciting air of disruption about them, with the potential to change core aspects of everyday life, like creating *programmable economies*. It's difficult to measure the degree of disruption objectively but it is possible to examine the indicators that historically coincide with profound technological innovation.

One of the largest disruptions of modern time was without a doubt the birth of the Internet, which created the ability to interconnect the world digitally in a matter of milliseconds. An indicator coinciding with this event was the dot-com bubble of 1997-2001, a period of excessive market speculation and wild growth. There are similarities between this period to today's crypto bubble and evidence that the crypto bubble is even greater in magnitude[1]. While we should approach bubbles cautiously, they are a requirement for extreme growth over a short period of time. Many companies did not survive the collapse of the dot-com bubble, but the giants of today's digital world, companies like Amazon and Google, emerged from the wreckage. We believe that the key for doing the same is planting firm roots in substantiated value and ignoring hype in general. These are guiding principles for Orbs and will be covered in greater depth throughout this paper.

If the Internet was the technological breakthrough driving the dot-com era, what is the underlying technology driving today's crypto era? Cryptocurrencies are not a technology unto themselves but *applications* of a technology. This technology is *decentralized consensus*.

Decentralized systems are distributed systems where a group of independent but equally privileged nodes operate on local information to accomplish global goals. These systems lack a central controller that exercises governance, supervision and control over the system, thus allowing power to be distributed over the network in a more uniform and fair manner. Distributed systems are not new, with applications such as Napster driving the peer-to-peer boom of the early 2000s.

[1] http://cnbc.com/2017/08/31/bitcons-nearly-five-fold-climb-in-2017-looks-similar-to-tech-bubble-surge

Consensus is a shared view of reality that is agreed upon between different parts of a system. Consider the most trivial example of a consumer application - an instant messenger where users can chat amongst themselves. This system requires consensus to operate, allowing every user to authenticate and speak only on their own behalf. All members must reach a shared view of reality regarding which user is which, who owns every username and so forth. The consensus property is very easy to achieve in centralized systems, where a single governing body is trusted by all members to define this shared reality.

Whereas decentralized systems are easy to build without consensus and consensus is easy to achieve in centralized systems, maintaining both properties in the same system proves difficult. This is the underlying innovation in the field of decentralized consensus. The ability to build decentralized systems where a group of independent but equally privileged nodes are able to reach a shared view of reality. Cryptocurrencies are an excellent example of an application that requires such a system, where agreement upon the ledger of transactions and balances can be reached without a governing body.

The Blockchain to Rule All Blockchains

The term blockchain originates from a core implementation construct of cryptocurrencies such as Bitcoin and refers to a continuously growing list of records, called blocks, which are linked and secured using cryptography. This chain of blocks holds the journal of all transactions in the network and forms the distributed ledger. The term blockchain has become synonymous with the core technology providing the infrastructure for such applications.

Building the next generation of blockchain infrastructure has become a fertile ground for innovation and dozens of teams are currently racing to deliver the “best” one. Many of these projects position themselves, sometimes even explicitly, as candidates for *the blockchain to rule all blockchains*. We believe that this mentality is flawed.

History shows that silver bullets are rare: complex problems are not solved by a single, simple solution. We believe that there will be no blockchain to rule all blockchains. A general-purpose blockchain can only optimize for the lowest common denominator. Therefore, the first step towards building a practical blockchain is articulating a clear use case - defining a real world need that this blockchain infrastructure is attempting to resolve and determining whether or not a market for this need actually exists.

A Requirement Driven Approach

The first question asked when presented with a new blockchain infrastructure is “what’s the key differentiator?” More often than not, the answer to this question is a groundbreaking new algorithm, a breakthrough based on years of academic research, that takes some arbitrary

narrow aspect of the problem and revolutionizes the industry by addressing it in an innovative way. That's not how we believe Orbs should be built.

We've decided early on that we're going to take a more humble approach. We're not going to start with the solution; instead, we're going to start with the problem. Our first step would be to outline a clear need that current blockchain infrastructure solutions fail to adequately meet. Next, find actual businesses, real customers for this missing infrastructure. Ideally, the next step would be to work side by side with some of these businesses. In the beginning, attempt with them to rely on existing blockchain solutions for their production use case; we need this process to understand the practical limitations of these solutions, where the actual challenges stem from, which features solve real problems and which are merely nice to have.

Our mission, apparently, has become to find *design partners*.

This would make Orbs a requirement-driven blockchain. Consider this as a key differentiator. The initial design should emerge in an iterative manner with each piece added only to resolve the most pressing requirement of our design partners. Our experience from building production systems in the pre-blockchain age teaches us that this approach is more likely to deliver superior and pragmatic solutions. It is definitely more efficient than concocting a solution first, designing a system around it and then scrambling to find a market fit.

A good relevant example is the emergence of the modern cloud, the de facto standard for infrastructure as a service today. AWS, the first and leading cloud provider, emerged[2] from Amazon's pressing e-commerce needs for solid internal systems to deal with the hyper growth it was experiencing.

The Era of Decentralized Consumer Apps

The crypto space is growing rapidly with more and more businesses joining the ecosystem every day. The first wave of businesses included mostly early adopters, newly formed crypto-first businesses or small existing businesses looking to experiment with the technology. The consumer apps among them, products like Steemit, Gnosis and Augur, achieved very mild success[3]. This is not surprising since consumer markets are notoriously difficult to penetrate, especially when dealing with complex technologies that even the tech savvy find challenging to master.

We're starting to see the next wave of businesses join this maturing ecosystem. This wave consists of established businesses making the transition to blockchain, whether it's established venture capital firms joining the crypto funding landscape or established companies looking to tokenize. The most interesting among these are mature consumer brands - companies like Telegram, LINE or Kik Interactive, with millions of end users, companies that have successfully penetrated consumer markets outside the blockchain

[2] https://techcrunch.com/2016/07/02/andy-jassys-brief-history-of-the-genesis-of-aws/
[3] https://steemit.com/statistics/@arcange/steemit-statistics-20171117-en

world. The transition for these companies is slow and far from straightforward, mostly because they have so much to lose if the foundations turn out to be inadequate for their needs.

It is estimated that the total number of active cryptocurrency end users in the world is 6 million[4] as of 2017. This figure is mind boggling in proportion to the total market cap of cryptocurrencies in 2017, which has surpassed 600 billion USD[5]. It is up to these established consumer brands to make the next giant leap for cryptocurrencies and decentralized technologies and deliver them to the masses.

Why Do Consumer Apps Decentralize

Traditionally, established consumer brands have followed a very strict centralized model. The question of their motives comes to mind as more and more of these applications take parts of their business and decentralize them over the blockchain. Critics may claim that the opportunity to raise funds provides a strong enough incentive for these brands to join this space. Nonetheless, we have a less cynical perspective which reveals a multitude of different reasons.

Tokens are becoming the standard for the transfer of value in the digital world. They are immune to many of the drawbacks that payments in fiat currencies face in a digital setting. Tokens can be shared instantaneously across borders, removing the need for separate payment solutions per geography. Businesses can receive tokens directly with minimal integration, cutting out middlemen such as payment providers and the hefty fees they charge. Tokens are easier to secure, reducing fees that are traditionally high to offset chargebacks and fraud. Tokens are also well-suited for flexible payment models such as microtransactions and can be incorporated seamlessly into larger systems through their programmable interfaces.

Tokenization also provides the means to create handcrafted micro economies that obey a set of predefined rules. This allows to design controlled environments where consumer apps can monetize effectively. Monetization has always been difficult for consumer apps, many of which must resort to selling the attention and data of their consumers to advertisers and marketers. The ad-based approach has resulted in advantages almost exclusively for the largest applications, concentrating wealth and power in the hands of the few.

Token economies today hold billions of dollars in value. Securing so much value on centralized infrastructure is costly because criminals have huge incentive in breaking into it. Decentralized ledgers are easier to secure because no single entity can manipulate the ledger, and in particular, nobody with access to the company's servers can use this access to steal

[4] https://jbs.cam.ac.uk/faculty-research/centres/alternative-finance/publications/global-cryptocurrency
[5] https://coinmarketcap.com/charts/

money from consumers. In addition, multiple parties constantly audit the integrity of the ledger and can identify discrepancies from the agreed upon protocol.

Tokenization also allows consumer apps to distribute the economic value created in them, particularly to their users. This ensures that users are fairly compensated for the value they create within the app. This model resonates well in an environment where monetization is normally not in the best interest of the consumer.

Another benefit of decentralization for consumer apps is the ability to band companies together as equals. The consumer space is slowly consolidating into the hands of the giants like Facebook and Google. Decentralization provides the means for the long tail of companies to align together and create combined ecosystems where no single entity can skew the balance of power in its favor. These ecosystems can even carry on after some of the individual companies cease to exist, allowing consumers to have direct ownership of the value they hold, without relying on the services of others.

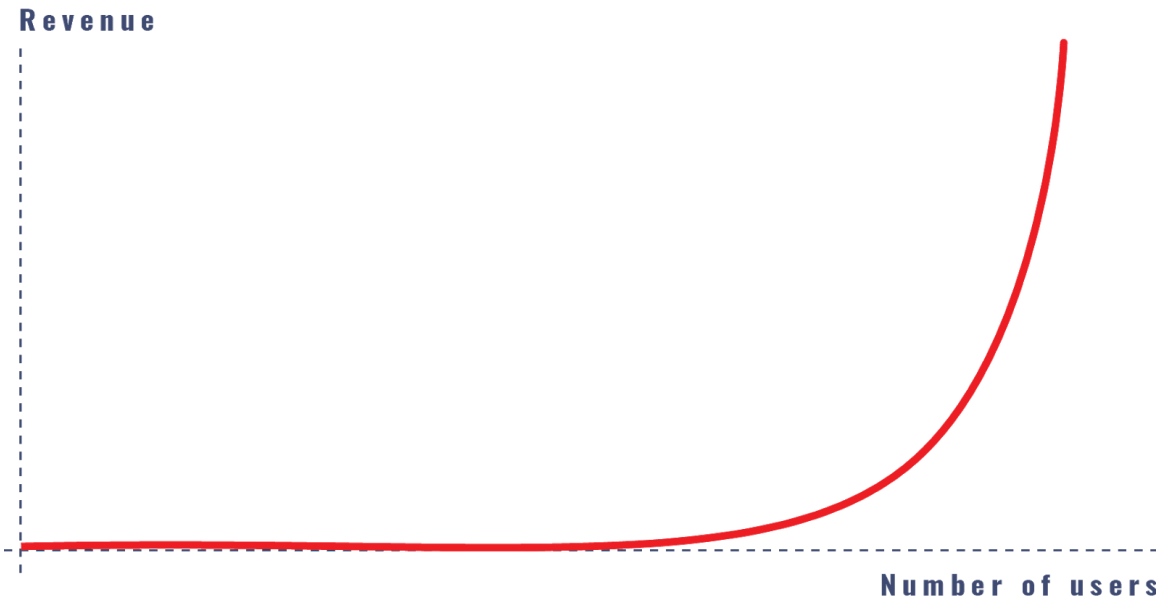
Kin by Kik Interactive - a Case Study

Kik Interactive Ltd.[6] is an established consumer brand based in Canada and the US with the mission of connecting the world through chat. Kik Interactive has over 150 employees in 4 offices around the world and is a member of the Fortune Unicorn List of private companies with valuations of over one billion dollars. Its main consumer product is Kik Messenger[7], a popular mobile chat app and the fifth most-searched term in the iOS App Store[8]. Kik has been operating for 9 years and has raised $120 million to date from private venture capital.

Kik is the classic example of a consumer product created with the optimistic vision of a fully connected world, brought on by the technological disruptions of the dot-com and mobile eras. It's a world where a group of friends, at the University of Waterloo in this case, can band together and create an app in their garage that will eventually find its way to 100 million users.

Over the years, Kik has been searching for a sustainable monetization model that does not compromise user experience or privacy. Nevertheless, it is operating in a world where digital services have been organized largely around an attention-based economy and monetization through advertising. The problem is, that the online advertising market is structured in a very unfair way: bigger players have more data on their users, and are therefore able to sell advertisements at much higher prices. This means that smaller apps don't enjoy revenues that are proportional to their size. Instead, revenue resembles a pattern of exponential growth:

[6] https://www.kik.com/
[7] https://en.wikipedia.org/wiki/Kik_Messenger
[8] https://www.kinecosystem.org/static/files/Kin_Whitepaper_V1_English.pdf



When the market is structured in this way, a very large share of the ads market go to the few big platforms who also enjoy the highest profit margins. They then use their market dominance to set the market prices to a level that ensures smaller platforms don't make a profit, but the bigger ones still do. This creates a long tail of small products unable to sustain themselves and a small number of extremely profitable giants.

This revenue graph means that Kik will eventually struggle to become profitable. Despite the odds and reaching an impressive number of users, Kik was struggling to sustain itself. This problem is not unique to Kik. Almost every digital service that sits on the "wrong" side of the exponent - which unfortunately includes almost everybody due to the way exponents behave - struggles to monetize. This is not surprising, as the 1% usually have 99% of the power. That's just how the world works.

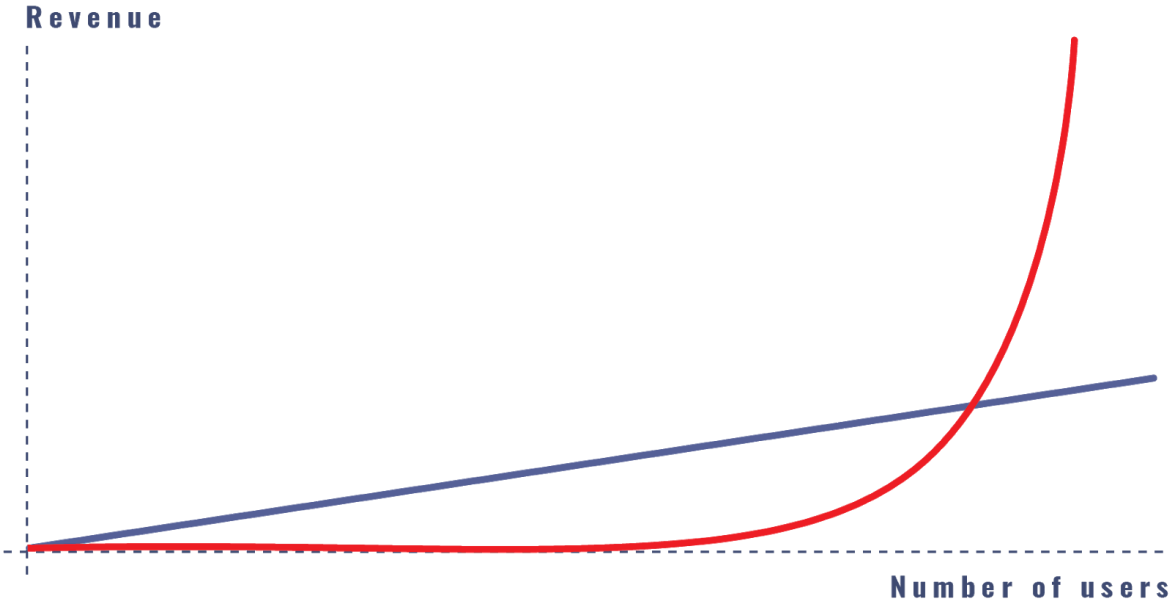
This problem worsens every year. One could say that the exponent is getting steeper. The world is consolidating, making it harder and harder to compete with the giants and less likely that a group of university friends trying to change the world will be successful. The result is disadvantageous for consumers as consolidation reduces market competition and hurts innovation.

Kik Interactive has taken on the challenge to change this reality. Many have tried and failed to solve the monetization problem of consumer apps in the last decade, but Kik believes that only a disruption in the magnitude of the one that created the Internet will suffice. This disruption is finally upon us: Cryptocurrencies, or the technology of decentralized consensus, to be exact.

With this mission in mind, Kik Interactive has launched Kin, a decentralized ecosystem of digital services for daily life based on a new token named KIN.

This token makes a new economy possible where digital services can monetize without direct reliance on advertising. The Kin white paper[9] sets forth the vision for the Kin Ecosystem, of aligning incentives of developers and consumers through the Kin Rewards Engine (KRE)[10]. Kik's Token Distribution Event (TDE) was held in September 2017, selling almost $100 million in KIN tokens.

We can use our graph example from before to demonstrate the general motivation behind Kin:



The red line in the graph above designates present economic behavior. The blue line is what a fair economy would look like. If I am a digital service developer with a service that is 1000 times less popular than Facebook, I expect to make 1/1000 of Facebook's revenue. This would mean that I am still able to sustain myself because my company is also 1/1000 of the size of Facebook Inc. While it is true that the 1% usually have 99% of the power, in this case, the 99% would prefer to move to a world that behaves like the line in blue.

Through the Kin Reward Engine, developers of digital services are incentivized in relation to their contribution to the overall adoption of Kin. Kin thus constitutes a powerful tool to change the natural behavior of the economy that results in a *linear* relationship between the number of users and revenue.

This also explains why consumer apps favor decentralization. The KIN token economy could generate substantial value. Securing a central ledger from theft, hacking or embezzlement is difficult and expensive. With a decentralized ledger, no one can manipulate the ledger by gaining access to the company's servers. This provides a greater degree of protection to its users. In addition, the success of the Kin ecosystem depends on the number of digital

[9] https://www.kinecosystem.org/static/files/Kin_Whitepaper_V1_English.pdf
[10] https://kinecosystem.org/static/files/Kin_Rewards_Engine_RFC.pdf

services that band together. Cooperation between competing digital services would have been next to impossible without creating a decentralized equilibrium where all parties are equal.

Target Audience for Orbs

The Orbs platform is focused on providing blockchain infrastructure for large scale consumer applications. Successful digital brands in the consumer space have access to millions of end users, usually through websites and mobile apps. The majority of these consumer brands are already established and have amassed their user base before the blockchain era. The Orbs platform provides the infrastructure for these consumer brands to build emerging decentralized businesses for the benefit of their users, such as the ability to utilize blockchain technology, tokenize, integrate smart contracts and more.

Working with a distinct and well-defined target audience also means the requirements and goals are clear. Orbs doesn't try to be a general-purpose blockchain, it's built from the ground up for the consumer app market and the very specific set of requirements it entails.

Building Through Design Partners

Orbs is being designed using a strict requirements-driven approach. The founding team has been working closely with some of the leading consumer brands that are in the process of transitioning to the blockchain and has been relying on them as design partners. By choosing partners representing different business domains, we are ensuring that the requirements we are exposed to cover the essentials for a truly practical solution.

In the domain of social/messaging, we are working with Kin by Kik Interactive. The Orbs team has been advising the Kin TDE from the beginning and some of its key members have been employed by the Kin engineering team on its journey to implement Kin using existing infrastructure solutions. Kik has been vocal about the challenges identified throughout the process[11] which became a cornerstone to the Orbs design. Orbs maintains a close relationship with the Kin engineering team, cooperating both on architecture and development.

In the domain of payments/settlements, we are working with Zooz Labs and PumaPay. Founded in 2010, Zooz is a leading solution provider for cross-border payments with customers such as Wix.com, Burberry and Gett. Zooz Labs is introducing a protocol for blockchain-based cross-border payments in prevailing consumer-oriented mobile wallet apps. PumaPay is working on a blockchain-based pull-payment protocol - a decentralized alternative to PayPal. Among its launch partnerships are businesses interacting with hundreds of millions of users and processing billions of dollars worth of payments, such as lifestyle media company FashionTV and adult media giant Vivid Entertainment.

[11] https://medium.com/kin-contributors/kins-blockchain-considerations-ebd0b60aebd5

In the domain of online advertising, we are working with Zinc - a blockchain-based advertising protocol that improves efficiency in the ad industry by increasing user data transparency, delivering a better user experience, and removing the impact of bad actors. Zinc is working with IronSource as a design partner, one of the largest advertising technology providers in the world. IronSource, founded in 2011, has a global reach of over 1 billion users a month, giving Zinc the potential to exponentially scale to become the next advertising protocol standard.

Another notable domain is business intelligence. Our team has been cooperating closely with Endor, a business intelligence technology vendor, with customers such as Coca-Cola, Walmart and MasterCard. While traditionally not a consumer-oriented domain, Endor is bringing predictive business analytics to end users, developing a decentralized platform with a Google-like interface for AI queries.

Together, these companies and their partners represent a network containing hundreds of millions of consumers, billions of app installs and billions of USD in annual revenues. Our team has been working closely with these projects since their inception, helping design their solutions both on and off the Orbs network.

INFRASTRUCTURE FOR DECENTRALIZED APPS

First Generation (Bitcoin)

Bitcoin is considered the first generation of blockchain technology. The building blocks used to create Bitcoin, such as the concept of Proof of Work, have predated Bitcoin by several years. Nevertheless, Bitcoin is unique in its combination of these elements to solve the problem of consensus on a decentralized ledger in an open, streamlined, elegant and secure way. The public success of Bitcoin should not be attributed to its underlying technology, interesting as it may be, but to its product. After all, Bitcoin is primarily an *application* and not an *infrastructure*. This application has been so successful, in fact, that cryptocurrency has become *the* killer app for blockchain technology.

One of the greatest feats of Bitcoin is alleviating any doubt as to whether blockchain technology is indeed working or that it is secure and solid enough to hold assets of great value [12]. Furthermore, it can be trusted to a greater degree than any infrastructure solution previously offered by any authority. It did so in a remarkable way by creating an environment free of central governance, where all nodes are equal but without assuming they are all honest. And still, it successfully manages an entire financial system worth billions.

Bitcoin naturally has its flaws as a first generation technology. Problems like excessive fees, long confirmation times and difficult upgrade politics are preventing the system from being used in production for any purpose other than as an electronic alternative to stored gold.

Second Generation (Ethereum)

Ethereum is considered the second generation of blockchain technology. Its primary innovation was the separation between the *application* and the *infrastructure* layers. Ethereum has little application on its own, but it provides a solid base for application developers to create their own decentralized applications over the blockchain. This is done by writing smart

[12] https://coinmarketcap.com/currencies/bitcoin/

contracts[13], immutable pieces of software running in a decentralized manner over all the nodes in the network under the promise of full consensus upon the execution of its terms.

Before the age of Ethereum, decentralized applications like Bitcoin were implemented as a monolith, mixing both the application and the custom-fitted infrastructure for running this specific application. The boom of decentralized application development in recent years can be largely attributed to this separation of application from its infrastructure, as the barriers for their implementation have been greatly reduced.

Naturally, as the first project to make this giant leap forward, significant parts of Ethereum remain as working proofs of concept but ultimately unable to run actual businesses on production[14].

Third Generation (Orbs)

Now that the foundations of the technology have been firmly set, the race is on to create the next generation of blockchain infrastructure that finally allows real-world businesses to decentralize. The third generation of blockchain infrastructure projects will go beyond the smart contract technology proof of concept and into practical business applications in real world production environments. The question is no longer whether this is possible, but what's the most efficient way.

We've witnessed how wide the gap is between proof-of-concept and production-ready after working with the Kin engineering team on bringing Kin to production over Ethereum[15]. Ethereum itself is going through transformations to evolve into a third generation project, with core changes like the gradual switch to Proof of Stake (PoS). Other than that, the number of third generation candidates is impressive, with excellent projects focusing on different aspects of the challenge, like EOS[16] and Cardano[17].

There isn't one blockchain to rule all blockchains. We're hopeful that many of these projects will reach fruition and each provide a best of its class solution for a distinct scenario.

Centralized Infrastructure as a Service

Solutions for Infrastructure as a Service (IaaS) for centralized apps have matured significantly over the last 10 years. Leading cloud providers like Amazon Web Services (AWS)[18], Microsoft Azure or Google Cloud Platform have become the de facto standard in the industry. There's a

[13] https://en.wikipedia.org/wiki/Smart_contract
[14] https://medium.com/kin-contributors/kins-blockchain-considerations-ebd0b60aebd5
[15] https://medium.com/kin-contributors/ethereum-challenges-while-launching-iplv2-8a33e1ba5a64
[16] https://eos.io/
[17] https://whycardano.com/
[18] https://aws.amazon.com/

great deal that can be learned from these services when designing our IaaS solution for *decentralized* apps.

Almost all established consumer apps use cloud services as their infrastructure for centralized IT services. Just as companies like Airbnb[19], Netflix[20] and Lyft[21] rely on AWS to provide the infrastructure for their centralized businesses, we expect companies with decentralized applications to rely on the Orbs platform to provide infrastructure for their decentralized businesses.

Another observation: emphasis on having a well-rounded *product* is sorely lacking in existing blockchain infrastructure solutions. What exactly is a well-rounded product for a blockchain? This question is often asked in emerging fields which have yet to produce concrete business applications. Consider, for example, the question of fee structure. Should blockchain fees be paid by the sender of a transaction or by its recipient? Or maybe by a third party? Should the fee be paid per transaction or per month with a subscription? Should the fee be predictable and constant or market-determined? These are all questions that have strong influence over the product - how to design the user experience when working with the infrastructure.

It seems that so far, the answers to most of these questions in existing solutions did not start from a product discussion, but from security, fairness or game theoretical considerations. What would a real business using this infrastructure prefer? A consumer app may want to subsidize its users' infrastructure costs. In addition, it may probably want to plan the budget for these expenditures in advance.

There's rarely a need to reinvent the wheel: the product for decentralized infrastructure is not materially different from that of centralized infrastructure like AWS. Mature platforms like AWS provide us with practices that are proven after years of calibration to fit the needs of actual businesses.

Lessons in Pragmatic Systems Design

We've made several observations that are important from a pragmatic standpoint from working closely with our design partners on architecting an expansive decentralized system. First, a decentralized system doesn't need to be decentralized end-to-end. Depending on your goals, significant parts of the system can remain centralized. This has a great effect on efficiency since centralized infrastructure is usually significantly faster and cheaper than a decentralized counterpart.

Consider an obvious example: a system that transacts video content securely over the blockchain. A naive approach would be to store the video data on blockchain storage, but doing so would be inefficient and immensely expensive. The system will perform better and be

[19] https://aws.amazon.com/solutions/case-studies/airbnb/
[20] https://media.netflix.com/en/company-blog/completing-the-netflix-cloud-migration
[21] https://aws.amazon.com/solutions/case-studies/lyft/

cheaper to operate by orders of magnitude if the raw video data is stored on cheaper and faster centralized storage and served through a CDN, while the blockchain is only used to distribute hash codes and keys.

Another important observation is that real-life systems often span over more than one blockchain. We've mentioned before that there's no one blockchain to rule all blockchains. Different blockchain implementations are better suited for different goals, even in the same application.

Immediately after its TDE, Kin launched as an ERC20 token over Ethereum. After understanding that Ethereum will not be able to hold the transaction throughput and fee efficiency required for a consumer app with millions of transactions per day, the project analyzed migrating to an optimized infrastructure solution - such as the Orbs platform. This migration would have a negative effect on some existing users as ERC20 is currently better integrated into the ecosystem of exchanges, wallets (including availability of hardware wallets) and so forth. Whereas Orbs is a great consumer platform, Ethereum is currently better suited for professional asset management and exchange. An ideal solution would be to enjoy the benefits of both by making the token available on both platforms with two-way portability using atomic swaps[22].

[22] An example of atomic swap mechanism: https://en.bitcoin.it/wiki/Atomic_cross-chain_trading

THE ORBS PLATFORM

Overview

The Orbs platform is a decentralized, open and transparent network providing public blockchain Infrastructure as a Service (IaaS) for large scale consumer applications. As the trend for decentralized businesses continues to rise, we expect more and more established consumer brands to take advantage of the new opportunities decentralization presents and start making the transition to blockchain. Orbs provides the cloud infrastructure to run these decentralized applications and facilitate this transition.

The three primary infrastructure offerings of the Orbs platform include consensus-based decentralized compute services, consensus-based decentralized storage services and Consensus as a Service (CaaS).

Decentralized-Consensus Compute

Compute services enable developers of decentralized applications to run their apps over the network and execute their code on the various nodes. Applications are executed in a fully decentralized and secure way over multiple independent nodes. The results of execution undergo consensus to make sure a single coherent outcome emerges. Compute services are similar in spirit to centralized IaaS services like AWS EC2 or even AWS Lambda, but use blockchain technologies. In the blockchain world, compute services are similar in principle to the execution of smart contracts over Ethereum.

Decentralized-Consensus Storage

Storage services enable developers of decentralized applications to store data over the network. Data is replicated and sharded between multiple independent nodes and stored securely in a fully decentralized manner. Storage enforces data to be in consensus, making

[23] https://aws.amazon.com/ec2/
[24] https://aws.amazon.com/lambda/
[25] https://en.wikipedia.org/wiki/Shard_(database_architecture)

sure there are no inconsistencies between nodes. Storage services are similar in spirit to centralized IaaS services like AWS S3[26] or even AWS DynamoDB[27]. From a blockchain perspective, storage services are similar in principle to storing data on the blockchain itself in Ethereum, or on IPFS[28].

Consensus as a Service

Since the Orbs platform is completely decentralized and comprised of independent nodes owned or operated by different organizations, the ability to reach consensus between these nodes is a core underlying capability of the network. The consensus layer of Orbs is architected to be modular and allow additional infrastructure layers, such as compute and storage to be built on top of it. It also allows its users to reach consensus independently of the other services. For example, one might use CaaS to notarize documents, or inputs from decentralized oracles. Using Orbs' inherent cross-chain connectors, CaaS can even be used to notarize states in other blockchain platforms, or execute atomic swaps between platforms.

Design Principles

Orbs is designed in a requirements-driven approach. By working with design partners that are currently operating mass-market applications serving hundreds of millions of users, we've been able to distill the following four areas as our focal points:

Service Level Agreement (SLA)

SLA is a commitment between the service provider and a service user. Particular aspects of the service - like minimum quality, availability, responsiveness - are agreed upon in advance with detailed mechanisms to guarantee the SLA in practice or compensate the user if the SLA is not met. SLAs are an industry standard in the traditional consumer infrastructure space and are widely used by centralized IaaS platforms like AWS[29]. In Orbs, SLAs are much more than these traditional written agreements between parties, since they are implemented as a direct part of the protocol and are *integral to the network design*.

SLAs are important for consumer apps in order to increase the predictability of the service level. Consumer expectations are high and even small disruptions in the availability of an app causes users to leave. A blockchain platform will bring little value to its applications if it fails to perform when congested, even if it provides magnificent performance and ultra-low fees.

[26] https://aws.amazon.com/s3/
[27] https://aws.amazon.com/dynamodb/
[28] https://github.com/ipfs/ipfs/blob/master/papers/ipfs-cap2pfs/ipfs-p2p-file-system.pdf
[29] https://cloudiqtech.com/aws-sla-summary/

Consumer Scale

Consumer apps serving millions of end users normally have aggressive scaling demands - primarily for message throughput, message latency and the number of supported users. As a reference, Lyft handles over a million rides per day[30]. If handled over blockchain technology, each of these rides should consist of several transactions: ordering the ride, driver taking the ride, payment, review etc. Each of these transactions needs to be processed almost immediately to accommodate an efficient transaction between the rider and the driver. Using blockchain terminology, message throughput and latency translate to transaction throughput and confirmation time.

Consumer scale does not only apply to raw network properties like throughput and latency. Scale influences almost every aspect of platform design. Consider the scalability of the fee model for example. Infrastructure fees with a fixed cost per transaction that scale linearly with the number of transactions provide a poor framework for consumer apps to grow, especially for mass usage patterns like microtransactions.

Consumer Protection and Regulation

There is a common misconception, that decentralized networks are unregulatable. In practice, regulation may lag behind fast-paced technological innovation, but its users are ultimately subject to the ruling of regulators and seek its protection when in need. Consumer apps are normally developed by well-defined organizations or companies with clear legal identities, even if they are decentralized. They are subject to regulation, both by governments and by industry bodies (e.g. app stores), and must operate within legal and contractual limits. For established consumer brands, that have much to lose if found noncompliant, regulatory uncertainty is a risk rather than an opportunity.

Our experience with design partners shows that compliance often becomes the critical path of the roadmap. Technical and infrastructure choices may impact the requirement for government licenses and compliance to app store rules.

Modern Deployment Paradigms

Painful lessons in history, such as Bitcoin SegWit2X[31], have shown how governance plays a critical role in the practical success of a decentralized system and its ability to keep improving over time. We consider governance and the orderly process for the continuous evolution of the protocol a first-class citizen in the design of Orbs. Incentives for evergreen nodes are constructed into the very base of the ORBS token economy.

[30] https://blog.lyft.com/posts/one-million-rides-a-day
[31] https://www.coindesk.com/2x-called-off-bitcoin-hard-fork-suspended-lack-consensus/

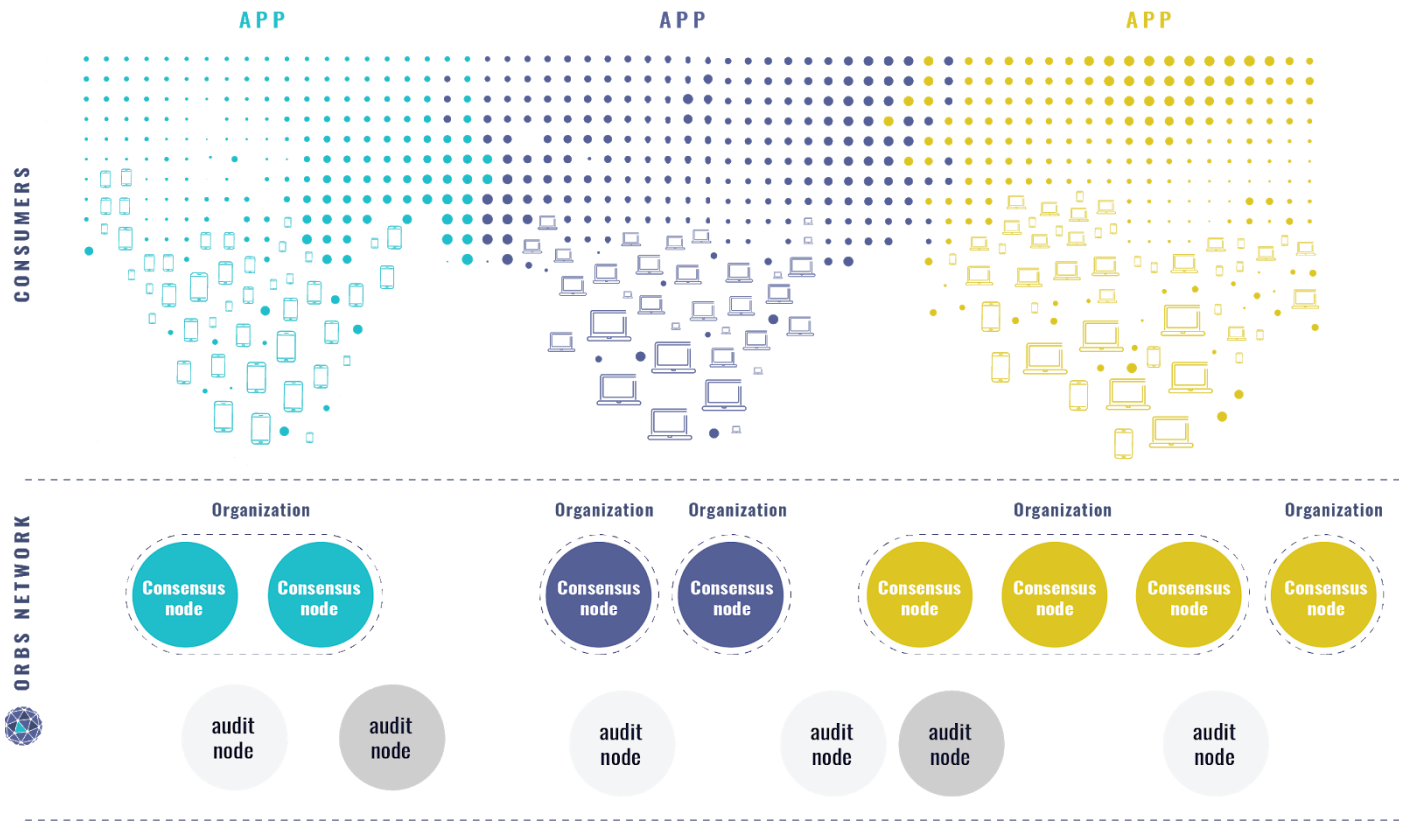
The challenge of smart contract immutability has also become important with ordeals such as the aftermath of the Parity multisig bug on Ethereum[32]. Smart contracts could use some local governance, to be able to adapt to extreme situations without requiring intervention of the governing bodies (or the governing DAOs).

Furthermore, our experience with real-world production at scale also shows that it is impractical for a production system to deploy major changes to the network as forks, going from 0% to 100% at the flick of a switch. Modern deployment strategies require the ability to deploy changes gradually (5% to 20%, then to 50%, etc.) with constant monitoring and the ability to rollback. The system must also be allowed to deploy multiple orthogonal changes side by side as independent experiments or else development will have to be serial and slow down significantly.

[32] https://blog.comae.io/the-280m-ethereums-bug-f28e5de43513

Network Entities

Orbs is a blockchain infrastructure for large scale consumer apps. As such, we can categorize several distinct types of entities participating in the network:



Consumers

These are the end users in the network who use the apps that are running on top of it. They would make up the vast majority of wallet holders for currency-based products. In general, we can assume they number in the hundreds of millions. Consumers will likely gain access to the network by using a mobile app or a website. Consumers do not have direct access to the Orbs network and do not operate nodes like in alternative blockchain implementations such as Bitcoin. Consumers' access to the Orbs network is always provided through an *app*.

Usage profiles on mobile apps and websites are characterized by extremely low availability of resources - low computational ability, low memory and low persistent disk storage. Network connectivity is extremely intermittent without any guarantee of how often consumers are online and for how long. Consumer apps in general are also suffering from high churn - meaning users are prone to abandon the app. The number of active users is usually significantly lower than the number of registered users (this number is often as low as 5%). From a regulatory perspective, most of the effort made by regulators is aimed at the protection of consumer interests.

Apps

These are the products running on top of the blockchain infrastructure, performing transactions for the benefit of consumers. We can assume a low number of popular consumer apps in the network, in the area of several hundreds. This is mostly because consumer apps are extremely competitive and consumer attention is very difficult to attain. The average mobile consumer does not use more than a dozen mobile apps, although millions of apps exist in app stores. The Orbs platform is naturally optimizing for the high-volume, popular apps.

Consumer apps are normally developed by well defined organizations or companies with clear legal identities. They are prone to regulation and must operate within legal limits. The majority of consumer brands targeted by the Orbs project are well established and have existed as centralized entities prior to the blockchain era. They currently rely on centralized channels to reach consumers - mobile apps in the app stores, as well as websites with branded domains are fully centralized.

Resources should be plentiful for consumer apps. Outside of the blockchain world, consumers are known to have high expectations of network scale and responsiveness. We can assume nodes operated by apps will have higher computational ability, more abundant memory and persistent disk space. Network connectivity can also be assumed to be steady and performant.

Consensus Nodes

These are the servers that participate in the consensus process and provide the actual compute and storage resources to execute the decentralized apps on top of the blockchain infrastructure. Nodes are owned and operated by various companies and organizations. Each of the organizations may operate multiple nodes.

Nodes participate in the network by following the protocol, for example by running the Orbs software stack[33] - a freely distributed open source reference implementation introduced by the Orbs project and maintained by the open source community. The collection of nodes in the network does not have any centralized point of governance and is not owned or controlled by the Orbs project. The number of nodes in the network could be in the same order of magnitude as the number of apps. In fact, app developers are encouraged to operate nodes in the network and a large similarity between the two groups can be expected. If we account for node redundancy (a single organization spawning several nodes for robustness) and ecosystem apps like Kin (where an ecosystem of organizations operate an app together), we can expect the number of nodes to be one or two orders of magnitude larger.

[33] https://github.com/orbs-network

Audit Nodes

These are also servers in the network whose primary purpose is adding another layer of security by public audit of the blockchain. Audit nodes do not take an active part in the consensus process itself and do not have the capability to write data to storage or close blocks. As such, unreliable behavior by audit nodes, such as intermittent network connection or downtime, does not have a direct negative effect on the overall performance of the network.

Audit nodes also participate in the network by running the same software stack as consensus nodes. In fact, consensus nodes perform a similar audit process among themselves in parallel to their additional responsibilities. Like consensus nodes, they do not have any centralized point of governance and the collection of audit nodes is not owned or controlled by the Orbs project.

The Orbs platform is designed to be public and open for inspection. Any entity and individual is encouraged to operate audit nodes to help contribute to the network general security. Node operation is also incentivized by the token economic model. We can accommodate a large number of audit nodes in the network which doesn't necessarily correlate to the number of consensus nodes.

THE ORBS ECOSYSTEM

Core Infrastructure

The nodes of the network running the Orbs protocol software stack provide the core competence of the Orbs platform as an infrastructure layer for the developers of consumer-oriented decentralized apps. The core offering includes consensus-based compute, consensus-based storage services and CaaS. The reliance upon Turing-complete[34] languages as the basis for applications to build upon has been proven as vital by projects like Ethereum. We consider it unwise for an infrastructure layer to limit the abilities of the applications it carries, as they are the main driver of innovation in the field and their exact requirements cannot be fully anticipated in advance.

Specialized Infrastructure

The core competence of the platform only holds the most basic of the building blocks for decentralized apps. These blocks are the most flexible, but experience shows that complex applications require specialized infrastructure as well, that is often built by third parties.

Decentralized analytics is a concrete example of specialized infrastructure in the Kin use case. Most of the partners in the Kin ecosystem would need a data and business intelligence layer (BI) but don't have the resources to develop or maintain their own solution. Reliance on a centralized BI solution offered by an established software vendor, such as Sisense[35] or Tableau[36], would be problematic: partners should base their most strategic decision-making on this data, but they cannot trust that the data was not manipulated by whoever controls the centralized account. A decentralized solution is a better alternative, but currently Kin would have to develop it from scratch. The best option in this case is for an established BI software vendor to develop a decentralized version of its analytics platform, given that it already

[34] https://en.wikipedia.org/wiki/Turing_completeness
[35] https://www.sisense.com/
[36] https://www.tableau.com/

possesses the domain expertise. The vendor can interface with the Orbs platform to provide the infrastructure to Kin developers via APIs directly accessible from Orbs smart contracts.

Of course, use-cases for specialized infrastructure are not limited to BI and can also include tools and integration points such as back-office software, oracles, integration ERP/CRM platforms, other types of storage and databases that go beyond a naive key-value store, and much more.

Infrastructure Marketplace

Orbs will encourage the formation of an open infrastructure ecosystem of specialized third party solutions, whose participants will provide a *one stop shop* for decentralized application developers. Such a platform will promote decentralized technologies in general and empower specialized complementary infrastructure providers. This ecosystem will establish an economy for services based on the ORBS token where third parties can receive payment using the token for infrastructure solutions they have created.

The benefit for decentralized application developers is clear - a single integration channel and a common language for all of their decentralized technology needs. This model has been proven to be successful on centralized infrastructure platforms such as AWS Marketplace[37], where third parties are upselling specialized services to AWS customers.

[37] https://aws.amazon.com/mp/

THE ORBS TOKEN

Overview

As the native token for the network, the Orbs platform relies on the ORBS token to fuel network operation and provide the means to pay the fees involved with operation of the consensus layer, execution of smart contracts and consensus-based storage - as these are the three primary services provided by the platform. The fee model serves as incentive for nodes to allocate the necessary resources for operating nodes and ensures an SLA consistent with consumer expectations in term of predictable and stable service, availability, performance and degree of security. In other words, nodes are paid for their services to consumer applications that utilize them.

The ORBS token is not only the driver for the Orbs core infrastructure, but for the entire ecosystem built around the Orbs platform. It fuels an *infrastructure marketplace* and serves as the means of payment for third party infrastructure providers which choose to upsell specialized decentralized infrastructure solutions on top of the platform. The billing models for third party decentralized infrastructure solutions must evolve side by side with the infrastructure itself. They also must adapt to the new token economy and rely upon the new dimensions of billing flexibility and the programmability it provides.

Billing Subsystem

The Orbs platform makes a clear distinction between *billing* and *accounting*. Drawing inspiration from centralized infrastructure counterparts such as AWS, where billing uses local fiat currencies in monthly intervals and accounting is tabulated separately per use and on demand with domain-specific metrics (CPU minute for compute or GB for storage and bandwidth). Current-generation blockchain platforms, like Ethereum, do not make this distinction and pair between transactions and their fee payments by requiring that the fee itself be explicitly attached to every transaction.

The Orbs billing subsystem is based on the ORBS token and provides the flexibility to be performed in monthly intervals as well. Accounting on the Orbs platform is performed

separately per transaction and on demand with domain-specific metrics (transaction throughput or storage used on chain). This separation adds a degree of flexibility compared to the rigid per-transaction billing and fee model used in most blockchain solutions today.

Programmable Fee Models

The Orbs billing subsystem takes infrastructure fee models a step further. Our experience with design partners taught us that different applications prefer to collect fees for infrastructure costs in different ways. Microtransaction-oriented peer-to-peer marketplaces hosted on digital services prefer that the digital services will subsidize the fees for infrastructure and hide them from their end users. Imagine for example a dating app like Tinder where the free model is crucial for user adoption. In these scenarios, expecting the end user to pay upfront for infrastructure costs makes as much sense as asking end users of an instant messenger to pay for the costs to deliver their own messages.

This is achieved on Orbs through the introduction of subscriptions. Subscriptions are designed for the developers of decentralized consumer applications which are often responsible for payment for the infrastructure services. This fee model is closer to the pricing model of AWS than to that of Ethereum where the end users pay for infrastructure costs by themselves.

The Orbs platform is designed to support alternative models as well. Consumer applications where larger sums change hands in a less frequent manner, such as in a decentralized Airbnb for example, may prefer that the party that originates the transaction will pay its fee. They may even go one step further and make the fee proportional to the amount paid despite the actual cost of infrastructure being constant. In other products, it makes more sense for the recipient to fund the fees.

The elegant way to deal with these varying demands is to move elements of this decision from the infrastructure layer to the application layer. By adding a degree of programmability to the fee model, with a smart contract so to speak that specifies how the fee is paid, applications will maintain the freedom to adjust the fee model to their needs. This also resolves another common challenge on such systems, where the fee is paid with one token and the transaction is performed with another, requiring users to hold balances in both tokens in order to operate.

Economy Incentives

One of the main benefits of basing economies on a token instead of traditional fiat currency is the ability to design an inherent system of incentives that will govern the system and steer it towards a global set of predefined goals. Bitcoin’s goal, for example, is relying on the native token to incentivize the security of the network and rewards nodes for securely closing blocks with Proof of Work. As of January 2018, Bitcoins with value of approximately $150,000 are

distributed every 10 minutes in average for this purpose[38]. This mechanism creates an economy in Bitcoin where predefined inflation in supply, in addition to the fees paid by users, provide the funding for this self professed global goal. The reward is scheduled to decrease gradually until fees comprise most of the incentive for miners.

On the Orbs platform, compensation to validators (consensus nodes) is designed to be based entirely on fees (no inflation to the token supply) from the very beginning. As opposed to where Bitcoin stood, we believe the blockchain industry is mature enough to correctly price transactions and validation without the "training wheels" of block rewards. Moreover, whereas fees distribute the burden relative to the amount of actual use of the validation services, generating rewards by inflation levies the cost of the reward on holders of the token relative to their holding amount. Like usage of property taxes to subsidize trade which creates a skewed market, using inflation to subsidize validation will lead app developers to make uneconomical choices.

The fees for the Orbs platform are designed with several aims in mind:

- Incentivize nodes to maintain a high SLA
    - High server availability with no downtime
    - Secure servers against hacking and protect their private keys
    - Fast server hardware with a fast network connection
    - Uphold its commitments to other nodes (like dedication of resources)
    - Regular server maintenance

- Incentivize nodes not to fork the official ORBS token
    - Take part of the official ecosystem and not split apart
    - Participate in the consensus process with other network members
    - Incentivize new consumer brands and organizations to join the network

- Incentivize nodes to evaluate protocol updates regularly and to adopt them
    - Participate in examining protocol changes proposed by the open source community, other federation members and the Orbs project
    - Run the same protocol as everybody else
    - Enable fast end-of-life cycles for outdated versions, reducing maintenance costs

- Incentivize public audit of the network
    - Public validation that the network is secure
    - Operation of audit nodes that verify conformity to the protocol in real-time
    - Incentivize security researchers to report vulnerabilities instead of exploiting them

[38] https://www.anythingcrypto.com/guides/bitcoin-mining-block-rewards-2018

Other goals for the economy include handling excessive demand gracefully and determining who gets service in this scenario, allowing applications to pay for dedicated resources such as throughput or storage and providing the necessary friction to prevent applications and users from spamming the network and paying for actual node server costs.

Besides the specific implementation details of the billing subsystem, which controls how collected fees are paid to node operators that provided services across the network, two other core aspects of the Orbs platform are used to implement the economy incentives. The first is a *reputation system* for nodes calculated during the consensus process and facilitated by the platform consensus algorithms such as the *Helix Consensus Algorithm*. The second is encouraging consumer applications to join the ecosystem and maintain a consensus node. This creates alignment between the incentives of the users of the platform and those of the nodes operating it.

In addition, in order to bootstrap the platform, the Orbs project is budgeting tokens in reserve for offsetting the entry costs of high profile consumer brands in joining the ecosystem.

Token Implementation

On the initial launch of the Orbs platform, we plan to start implementing the billing subsystem over the Ethereum blockchain. We believe that Ethereum is a great initial choice for a practical decentralized billing system mostly due to the extensive amount of third party integrations available for ERC20 tokens in the industry today. These ecosystem integrations with exchanges, third party wallets and hardware wallets are important for the target audience of the billing subsystem, companies and professionals, as they are often required to manage large amounts of money in the form of cryptocurrencies. The current scale limitations of Ethereum are not likely to pose a problem for a billing product since the rate of billing transactions is low - once per month, and the amounts transferred are high - making the significant fees of Ethereum negligible in this case. These parameters are not very different from that of a wire transfer which is a common means of payment for centralized infrastructure solutions like AWS.

Implementing the ORBS token on launch as an ERC20 token will also provide a production use case for core platform features of Orbs like *Polyglot Cross-Chain Contracts*, as the subscription smart contracts running over the Orbs platform will access billing information found on the Ethereum blockchain. This is a great example where a multi chain hybrid makes sense for practical reasons, where two blockchains can be used side by side, focusing on their relative advantages. Towards launch, the resources of the Orbs project are better allocated towards the primary differentiating factors of the platform such as consumer scale and blockchain virtualization. Producing an ecosystem of integrations to exchanges, third party wallets and hardware wallets is not an immediate priority. However, the Orbs project ultimately intends to invest efforts in those as well.

ARCHITECTURE

To Fork or Not to Fork

One of the first questions that comes to mind when designing a new blockchain infrastructure is whether to fork an existing system as a starting point from which to build. As some measure of dominance in the industry, we can look at the top 20 tokens on CoinMarketCap[39], excluding Bitcoin and Ethereum, and see that *over half* of the tokens are forks of other popular tokens. Forking has become a popular method to bootstrap new systems quickly due to the permissive nature of intellectual property in the field.

Our conclusion, ultimately, was that this decision should depend on what the system is trying to achieve. Let's assume that we have adequately mapped the space and found an existing blockchain solution. The solution closely implements our conceived ideas, so we consider carrying out a fork. If we suspect that our final result will differ in 30% from said system, we should fork. If we suspect that our final result will differ in 70%, we shouldn't. Using this as our guiding principle, and the lack of maturity of the consumer application sector over blockchain in general (with the first billion dollar consumer brands starting transitioning to blockchain only recently) we have decided not to fork. We are willing to take the short term penalty of delaying our time to production for the luxury of architecting a system from scratch and designing it for our distinct use case. We want to free ourselves from the potential burdens caused by using an architecture emerging from a different set of requirements.

Polyglot Microservices

In the evolution of software systems architecture, traditional systems grew gradually from the initial single program to complete systems. Initially, systems were very simple with all functionality in a single program - what is considered a *monolith*. As functionality was added to the system, both its codebase and group of contributing developers grew. Gradual growth then lead to decomposition of the project to separate modules following the *separation of*

[39] https://coinmarketcap.com/

concerns principle[40]. Over the years, well-architected modular systems proved to work well for scaling functionality and for scaling development teams.

The Internet revolution brought about new challenges to systems design, as modern systems commonly need to work at massive scales - interfacing with other systems and millions of end users. This leads to changes in both ends of the engineering process. On the development end, the complexity of such systems requires new development paradigms such as refactoring, agile development, continuous deployment and build-measure-learn cycles. On the operational end, it leads to dependence on intricate infrastructure to enable scaling the throughput of overflowing interfaces. Both changes prove to be hard to apply in modular systems that suffer from delicate module interdependencies. This hardship lead to the evolution of *service-oriented architecture and microservices*[41] design methodology, where each functional component is implemented as a separate, simple and focused product.

It is easy to observe that most current-generation blockchain platforms are built as monoliths. This not only shows the immature state of their development, but also hampers the ability to evolve and extend the functionality of systems based on these platforms. Moreover, in high-complexity open source projects, the reliance on well-acknowledged libraries and frameworks becomes limited when monoliths are constrained to design choices optimal for just some of their functions. The optimal environments for developing high-performance cryptography are different from those optimal for decentralized storage or from those for high performance networking and so on. The microservices architecture enables a system to be *polyglot*, i.e. use different programming languages and frameworks for the different services. Such an approach allows for higher quality services, and better usage of resources such as open source solutions and engineering talent with expertise in relevant domains.

Specification as Code

As many software engineers know, a specification document grows stale the minute it gets published - if not before. Therefore, we strive for *executable* specifications that will trigger conspicuous alerts upon failing. By using executable specifications, we ascertain that at no point in time does a codebase diverge from its specification, thus assuring that backwards compatibility and correctness are never compromised.

The distributed and decentralized nature of a blockchain network makes utilizing executable specifications even more important as the developers have little control over the deployment lifecycle of the services, so rolling back a deployment that broke an API or introduced a bug is not a viable option. Therefore, our workflow makes extensive use of executable specifications

[40] https://en.wikipedia.org/wiki/Separation_of_concerns
[41] https://martinfowler.com/articles/microservices.html

in two major categories: using Protocol Buffers[42] for generating our API schemas, and using Test-Driven Development (TDD) for achieving highly testable code.

Protocol Buffers (or protobuf) is an Interface Definition Language (IDL) developed at Google that is programming-language-agnostic and allows defining APIs with backward and forward-compatibility in mind. This creates a clear distinction between the code that defines an API specification and the language and code used to implement it. If a developer changes an implementation in a way that breaks an API, static type-checking will fail the build and an alert will be sent to the developer immediately upon failure. An added value of using a language-agnostic IDL is as an enabler for writing polyglot microservice, as the API between each pair of microservices is not defined in any specific language.

Test-Driven Development (TDD) is a methodology in which each required behavior is coded into unit tests before coding the behavior itself. In practice, it means the developer starts by defining the missing behavior, thus creating a test that fails and making sure the failures are as expected. Only then can he go on to implement the code that makes that test pass. Pursuing this methodology assures that no untested code ever gets into the source code repository. Next, the code is reviewed, but unlike regular code review, the reviewer focuses on validating the correctness of the test (representing what the code does) rather than that of the code itself (how it does it). The tests are written in a semantic language describing the business domain (for instance, transferring some funds between two addresses) rather than the specific implementation; changing the implementation does not affect the test. Practice shows that TDD leads to shorter, more concise code and that the coding process comprises of more cycles of refactoring, thus reducing technical debt.

Meta Programming

As the distributed and decentralized nature of a blockchain network imposes engineering challenges that are not present in a traditional deployment, the need arises for creative solutions allowing to circumvent some of these limitations. The Orbs platform makes extensive use of *meta programming* for critical components that will need to support Over The Air (OTA) deployment. Other blockchain networks such as Ethereum offer the concept of smart contracts as a way to execute user-deployable code. The Orbs platform borrows from these ideas and extends them for engineering-facing problems such as deployment and provisioning.

One interesting area where we make use of meta programming is resource management and provisioning. This is implemented as hot-deployable code not unlike smart contracts, running on an instance of the network itself (which can be thought of as a meta network), controlling the way resources are provisioned. For example, new virtual machines may automatically be instantiated to increase network capacity when a new member joins the ecosystem - particularly when this member is paying for dedicated resources. Since it is difficult to foresee

[42] https://developers.google.com/protocol-buffers/

the types of limitations and challenges we might face when onboarding new members to the ecosystem, making this area of the management code OTA-deployable makes a lot of sense. An added value of this approach is that the developers will have immediate visibility into the platform's runtime, as they will be users of the platform themselves, "eating their own dog food".

Another example is a public DNS-like service which enables the resolution of a public address in a more user friendly format (alternatively, it can be used to shuffle between multiple addresses). Implementing such an address resolution mechanism as a smart contract makes it easier to maintain than making it part of the platform's native core - just as Ethereum chose to implement ENS as a smart contract.

Universal Addressing

Addressing is an important topic in blockchain infrastructure controlling the scheme of how various resources are labeled and referred to across the system. Logical entities that have a distinct address include token accounts, smart contracts and their stored variables.

Different blockchain implementations have adopted different addressing schemes. We believe that no single scheme is superior over all others, furthermore, different addressing schemes have different qualities and are suitable for different applications. For example, some addressing schemes, such as Schnorr public key based, enable native support for multiple signatures. Other addressing schemes have wider ecosystem presence and are supported by more hardware devices and HSMs. Moreover, an addressing scheme that is compatible with the one used by another blockchain implementation will allow end users the convenience of using the same public address across multiple platforms.

In order to promote interoperability between blockchain implementations and allow for easier migration onto the platform, Orbs supports a universal signature and addressing scheme. This method allows applications and users to use a range of popular addressing schemes side by side by specifying the type of address next to the address itself. From an architecture perspective, addressing schemes are managed by a dedicated module which can be upgraded to support additional future schemes as they become popular across the industry.

Network Owned Secrets

In centralized systems, secure operations are commonly based on secrets owned by the governing service which can be used to sign, encrypt or decrypt data. As decentralized networks are comprised of independent nodes in a trustless setting, applying a similar method is not as straightforward. Secrets can only be held by individual nodes. The network as a whole is usually unable to hold a shared secret and use it for secure operations, as due to the open and decentralized nature of the system, this secret will easily leak.

This limitation often causes blockchain implementations to rely on trust for scenarios where trust should not have normally been required. A good example is how the client of an end user communicates with the network and performs queries on its state, such as checking the user's balance. Assuming the client cannot run a full node that synchronizes with the network and performs the complete suite of resource intensive validations, some compromises must be made. A common workaround is for the client to communicate with the network through a specific gateway node and delegate some of the validations to it. This would mean a client query for network state needs to trust the gateway node to provide an honest response. Some improvements over this approach include redundancy tactics of querying multiple gateway nodes at once, choosing the gateway node randomly, fraud-proofs[43], etc.

Network owned secrets is a cryptographic protocol introduced by the Orbs platform, that provides the ability to hold a shared secret securely in a decentralized network. None of the consensus nodes have direct knowledge of this secret, and only a combined effort of a specified majority among them can facilitate this secret to perform a secure operation like signing, encrypting or decrypting data. The mechanism relies on a cryptographic primitive called *threshold encryption* and described in detail in the technical white paper of *Helix Consensus Algorithm*. The benefit of this method is that the combined signature is only produced after reaching a threshold amount of signatures by individual nodes, each using their own individual secret that no other node knows. Thus, we have created a combined signature that can effectively be considered as the signature of the entire network. When the network as a whole has the parallel of a private and public key, many useful secure operations become easy to implement.

Network owned secrets provide the ability for secure interaction with the network without the need to trust specific nodes. Consider a client that desires to perform a smart contract calculation and is unable to run a full node. Instead of querying one of the network nodes as a gateway and trusting its response, by using a network secret to sign the combined response of multiple nodes, the client can verify the response simply by checking the signature.

Another interesting capability gained from the network's ability to own secrets is holding assets or accounts on the network level. Consider the requirement to implement a smart contract over the platform that controls an account on a different blockchain like Ethereum. Normally, this requirement would not have been possible to implement because smart contracts cannot hold secrets. Using a network owned secret, though, a private key can be generated by the collective for nodes after consensus. This private key is unknown to any of the individual nodes and can be used to access an external Ethereum wallet securely. Moreover, smart contracts can be used to provide key services such as key management, DRM, etc.

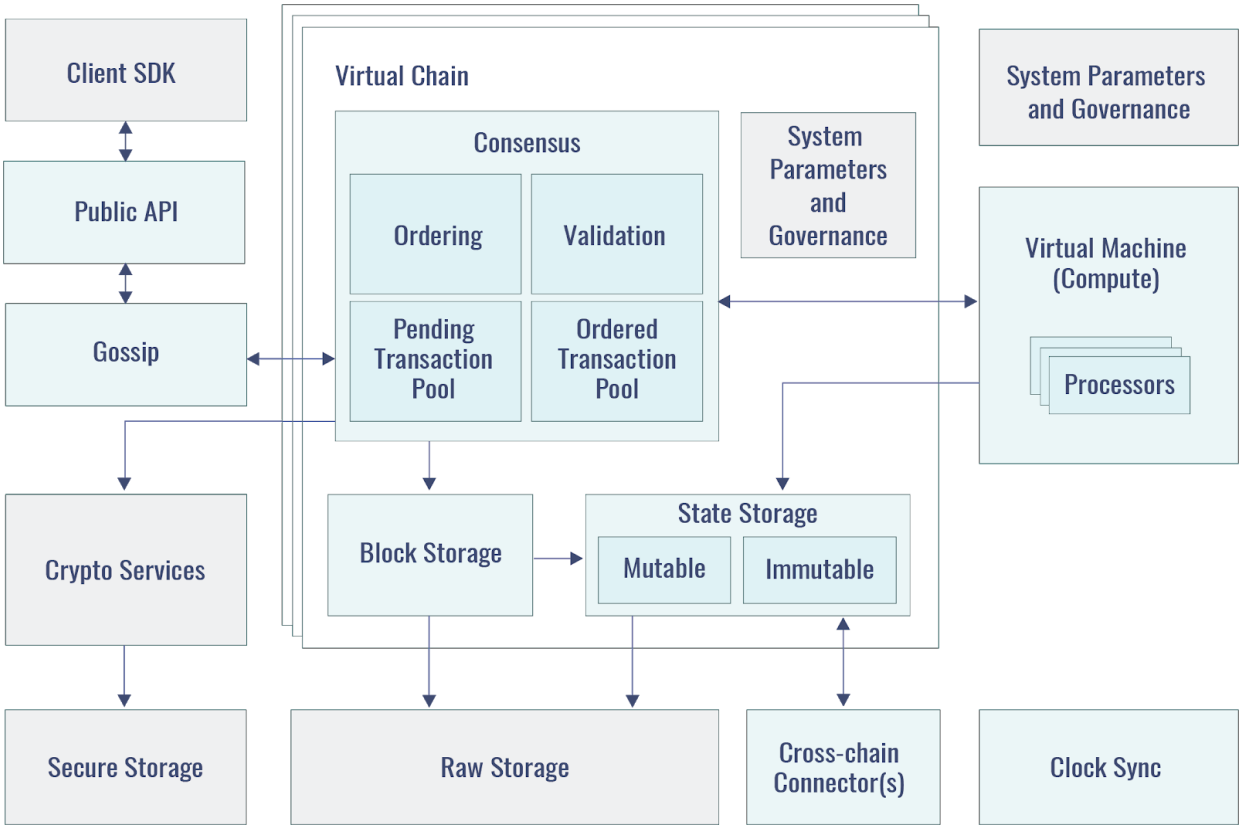
[43] https://gist.github.com/justusranvier/451616fa4697b5f25f60

Orbs Architecture

The detailed architecture of the Orbs platform will be outlined in a set of technical architecture white papers. In general, the Orbs architecture consists of multiple layers, responsible for different functions in the system. The layers and services are built such that they can operate on different machines and scale independently as needed. As a design goal, the architecture attempts to separate services and modules as much as possible within a layer in order to enable flexibility, upgradability and interoperability.

A key component in the architecture is the support of virtual blockchains - multiple parallel instances of the consensus, state and storage layers; that provide the illusion of a separate dedicated blockchain over a shared physical node environment. The concept of blockchain virtualization is discussed in greater detail in the following sections.

Infrastructure Layers and Services



Client SDK - A client-side library for mobile and web apps that connects end users to the network. The SDK can sign and encrypt requests and verify responses without relying on trusted nodes.

Public API - A microservice that exposes all public web API to clients (such as REST or JSON-RPC). Provides the endpoint handling all end user transactions and queries.

Gossip - A microservice that provides efficient one-to-many and one-to-one communication between nodes in the network. The novel communication scheme of the Orbs platform is discussed in detail in the Helix white paper.

Crypto Services - A library and service providing the low-level cryptographic routines and services like signatures, hashes and encryption. Has both native and non-native fallbacks.

Secure Storage - A library and service that store secrets like private keys in a secure manner. Uses HSM[44] when available, relying on dedicated hardware and tamper-resistant enclosures.

System Parameters and Governance - Holds infrastructure configuration parameters and handles updates and provisioning.

Virtual Machine (Compute) - A microservice that owns the execution of transactions and smart contracts, serving all virtual chains. The compute layer holds transient state for non-final execution and cross-chain data.

Processors - A set of microservices providing the actual runtime environments needed for the execution of smart contracts in various languages (EVM, Python, Java, JavaScript, etc).

Raw Storage - A layer responsible for storing and retrieving raw data on local machines.

Cross-chain Connector(s) - A set of microservices providing cross-chain interoperability with third party blockchains like Ethereum. Provides access under consensus of the third party.

Clock Sync - A microservice responsible for synchronizing clocks between different machines, nodes and services. Provides a consistent frame of reference for absolute time. Global clock synchronization is not a requirement for the *Helix Consensus Algorithm* but other system services may benefit from this ability.

Consensus - A microservice instantiated per virtual chain that is responsible for achieving agreement among nodes on the order of transactions and their validity. Implements the consensus algorithm. Consists of the sub-layers: Ordering, Validation, Transaction Pool.

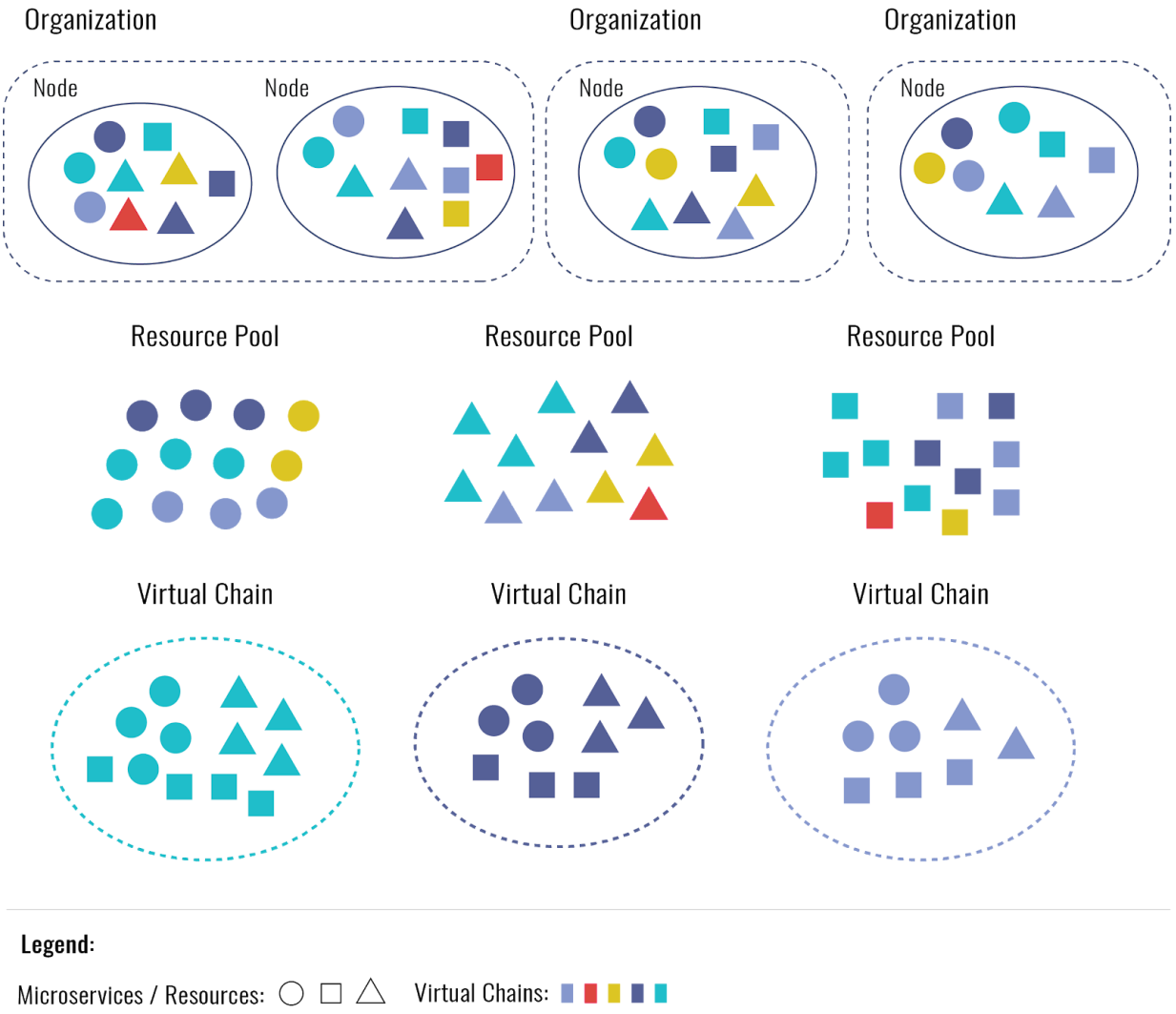
State Storage - A microservice instantiated per virtual chain that holds the mutable and immutable state that is under consensus. Manages efficient caching, sharding and redundancy for the state data. Accessed by the Virtual Machine and Public API.

Block Storage - A microservice instantiated per virtual chain that holds the incremental long-term journal of all previous closed blocks. Manages efficient sharding and redundancy for the blocks' data. Used to generate and validate the state.

[44] https://en.wikipedia.org/wiki/Hardware_security_module

Virtual Chain Parameters and Governance - Instantiated per virtual chain and holds the virtual chain specific configuration parameters and handles updates and provisioning.

Resources Pools vs. Virtual Chains



The Orbs architecture is comprised of a set of microservices providing different types of resources to the network. Each type of resource can be scaled separately according to the actual requirements of the network. A service like Public API can be scaled aggressively by launching multiple instances as more concurrent end users connect to the network, compared to the Consensus service that is instantiated per node based on the number of virtual chains.

Different applications running on top of the Orbs platform may have different requirements for different resources. For example, a compute intensive application may require a large capacity of decentralized compute resources while a database application may require a large capacity of decentralized storage. Moreover, the amount of resources is expected to change over time with the introduction of new applications and the evolution of their needs.

In order to efficiently utilize the resources and microservices available on each node, they are regarded as a shared pool of resources that is then allocated to different virtual chains based on their SLA requirements. Some resources may be dedicated to a virtual chain whereas others are allocated dynamically based on demand. An architecture based on resource pools effectively utilizes the varying capabilities of heterogeneous nodes and supports an elastic resource capacity. Each node is paid for the sum of the services it provided to the network, and penalized if it failed to deliver on its commitments. This gives node operators the incentives to add resources as needed, and to maintain the committed quality of service.

CONSENSUS

Pragmatic Decentralization and Trust

Consensus is one of the core subsystems required for a blockchain infrastructure and the choice of consensus model is without a doubt one of the first decisions we have to make. The question of consensus, above any other, is riddled with preconceived notions and strong opinion. Debates between the camps of Proof of Work (PoW) and Proof of Stake (PoS) often borderline philosophy[45]. Our position, as usual, will have to emerge from a careful analysis of our requirements.

Consensus comes to solve the problem of agreement in a decentralized system, where the selection between different possible agreements could create profits or losses to different parties. In a centralized system, there are no different parties, thus no conflict in selecting an agreement. Before diving into the discussion, we should examine how decentralized each part of the network really is.

Looking at the *network entities* diagram, let's begin with end users, our first anchor. Consumers, users of apps like instant messengers, are generally unaware of the benefits of decentralization, unlike the early adopters of Bitcoin. In the near future, it's also safe to say that the vast majority of consumers will not grasp the difference between a decentralized system and a centralized one. Contrary to the trustless ideal behind Bitcoin, end users will never go over the source code of a product before compiling it themselves, nor will they verify the signature to make sure they've downloaded an authentic copy. Consumers, as always, will place their trust in a brand.

Brands are almost always centralized entities: they have one leadership and one policy. They usually rely on centralized delivery channels to reach consumers, like the Apple or Google app stores and websites with branded domain names managed centrally and hosted on centrally-managed servers. Consider a consumer typing in their secret passphrase into a mobile app wallet. The consumer must trust that the developer of this mobile app isn't stealing their private key, abusing it or transmitting it outside. When interacting with

[45] https://download.wpsoftware.net/bitcoin/old-pos.pdf

blockchain, it will be code created and signed by brands that will ultimately interface, on behalf of their users, with decentralized apps.

This observation fundamentally changes how we evaluate open consensus models: it means that any user's voting power is essentially delegated to the brand whose code they are using. Even in models like Delegated Proof of Stake, the voting power in the delegate election is implicitly delegated to the brand.

Healthy Distribution of Power

Before we discuss political power in a decentralized network, we need to be clear on what this power is useful for. In most decentralized networks, political power is used for two types of decisions: real-time validation of transactions, and governance of the network itself (agreeing on protocol upgrades, parameter changes, forks to the blockchain, etc).

Real-time Validation Rights

The effect of political power in real time validation is limited when the protocol is well-defined, because the fundamental rules are axiomatic to the operation of the network. For example, a decentralized ledger would not enable any verifier - no matter how powerful - to approve transactions that are not signed by the payer. If it did so, it violated the rules of the protocol, so the allegedly-approved transactions will either be ignored by the network, or the network will halt because the consensus broke. It is clear that egalitarian distributions of power, which make it harder to stage such attacks, leads to more robust and sustainable platforms.

Manipulations that do not break the protocol are limited, then, to undetectable violations of the protocol; for example, deliberately failing to propagate transactions, manipulating the order of transactions within a block, selfish mining[46], and so on. To some extent, the ability to abuse power in this real-time validation can be further limited if the protocol is designed in a way that limits validators' ability to apply such manipulations.

Governance Decision Rights

An efficient mechanism for agreeing on network governance issues is critical for its long-term relevance. As an industry that relies on state-of-the-art technologies in cryptography, distributed systems, network and software infrastructure, we can expect a steady current of innovations that can further improve and extend the abilities of a given blockchain platform. Furthermore, as blockchain solutions enter mainstream industries, new uses and new contexts of use expect to be catered by these platforms. Adaptability is important for any

[46] https://www.cs.cornell.edu/~ie53/publications/btcProcFC.pdf

good blockchain platform, and the mechanics of network governance should not be a barrier to it.

Whereas most technological advancements are easily agreed upon, conflicts arise where interests are not aligned. In order to analyze healthy power distribution patterns, we must first understand what interests are in play, and who the actors are.

With a bird's eye view on blockchain governance, we can see three archetypes of stakeholders:

- End users
- Decentralized app developers catering to end users
- Network infrastructure operators (e.g. miners)

As designers of a platform that aims to serve consumer applications, our approach is to try and maximize the utility for users.

It is easy to see that the interests of infrastructure operators (such as Bitcoin miners) are normally misaligned with those of the end users. As a result, networks are slow to adopt changes, even when those are urgently needed. A notable example of the damages of such conflict is the 20-month debate in the Bitcoin community over the switch to SegWit (BIP141[47]) that was designed to resolve a technical issue in the protocol, but had several side-effects expected to impact the revenues of miners. Ultimately, following months of great uncertainty for users and app developers, the network forked on August 2017.

Developers' interest in infrastructure choices are usually well aligned with the end users', except for situations where a proposed change could be used by an incumbent to fend off competition. App vendors may also be divided on their preferences between newer and field-proven technologies. Vendors of mature and established apps tend to be risk-averse and prefer waiting for technologies to ripen, whereas fledgling apps see value in adopting avant-garde technologies that have greater potential to disrupt incumbents. *Platform virtualization* has great potential in mitigating many of these conflicts, as it allows each application to govern many aspects of its own infrastructure independently from others.

As for the end users, we believe their vote is well-aligned to their utility, as long as they are well informed as to the practical implications of their decisions. However, involving end users in governance decisions is extremely hard in decentralized systems: user identities are usually not connected to real-world identities, and digital identities can be easily forged. This issue is usually mitigated by weighting user votes with their stake in the system's currency, at the risk of concentrating power in the hands of a few rich users.

[47] https://github.com/bitcoin/bips/blob/master/bip-0141.mediawiki

Proof of Work

If most people are honest, a majority vote on the integrity of a public ledger is a straightforward consensus that is decentralized, permissionless and open. But majorities are an elusive concept when relying on digital identities, as the cost of generating new identities is negligible (Sybil Attack[48]). An ingenious solution to this problem is PoW, proposed by Dwork and Naor in 1992[49], in which suffrage is subject to spending computer resources and energy on solving a cryptographic puzzle. PoW is widely used today in permissionless distributed ledgers.

Commonly, a PoW ledger arrives at consensus over time: anyone can be the public verifier for a set of transactions (usually, a "block"), if they are first to solve a PoW puzzle; their efforts on the puzzle will be compensated by a cryptocurrency reward. Consensus on the validity of a block builds gradually, as future blocks refer to our block as their predecessor. A perpetrator will be at loss for trying to publicly validate an invalid block, because his expenditure on the PoW will not be reimbursed if his (invalid) block is not referred to by future blocks.

Although PoW ledgers achieve the utopian ideal of a decentralized, permissionless and open consensus, their application is far from creating utopia. For PoW to be adequately secure, the cost of solving the puzzles must be proportional to the value of the underlying assets; the global impact of this on mass-market ledgers is frightening. To illustrate, as of January 2018, Bitcoin mining is estimated to be taking nearly 1/5000 of the world electricity consumption[50]. As cryptocurrencies reach mass market audiences and significantly grow in value and transaction volumes, we expect that PoW ledgers will become unsustainable due the amount of emissions involved with their operation. Naturally, these costs are imposed on the users of the Infrastructure as fees and inflation; costs of using PoW networks are higher than those of the alternatives. In order to reach the massive scale required for consumer apps, lowering infrastructure costs is also a core requirement of the Orbs platform.

In the near-term, PoW poses several immediate and practical challenges to mass-market adoption. One is with the governance of such networks; in addition to the inherent complexities of governing a decentralized movement, a side effect of the proliferation of PoW is that mining has become a business of specialists, adding another powerful, heavily invested stakeholder in the system whose interests aren't aligned with those of users or app vendors (as discussed above). Developers of consumer apps are the political group which the Orbs platform is optimized for, as their involvement in the network is paramount to its purpose, yet nobody expects companies like Kik to become competitive PoW miners.

Another critical barrier to using PoW in mass-market applications is the inherent latency associated with eventual consensus. The acceptance of a block is determined by the depth or

[48] https://www.microsoft.com/en-us/research/wp-content/uploads/2002/01/IPTPS2002.pdf
[49] http://www.wisdom.weizmann.ac.il/~naor/PAPERS/pvp.ps
[50] https://digiconomist.net/bitcoin-energy-consumption

weight of other blocks referring to that block; this weight accumulates slowly as blocks cannot be created too fast without raising the probability of conflicts, which in blockchain systems come as forks to the chain. Newer protocols can significantly reduce this latency by using data structures that embrace forks as a valid state of the blocks' formation. Significant work has been done in this field by Sompolinsky, Lewenberg and Zohar, including the design and validation of the SPECTRE protocol[51] that implements a block-DAG (Directed Acyclic Graph) to replace the conventional block-chain. Such work shows a lot of promise in advancing PoW decentralized payment ledgers beyond their current limitations, but at the moment do not offer a good solution for more abstract systems such as smart contracts, as their use entails very high complexity of calculating the system state.

The Orbs research group has invested significant efforts based on the original SPECTRE research and had discussions with Prof. Zohar in attempt to develop a practical production implementation of the protocol. Ultimately, due to the difficulty in attaining the finality required for the efficient execution of smart contracts in DAG-based systems, and inherent limitations of PoW in general, we have decided to move away from the SPECTRE research as the base for the Orbs platform consensus strategy. We remain supportive of this promising concept and watchful as its research matures with advances like PHANTOM[52].

Proof of Stake

An alternative means for countering Sybil attacks is to tie suffrage to ownership of a digital asset, usually a cryptocurrency. On the face of it, PoS provides an elegant alternative to PoW, without the costs and energy waste associated with it. But the fact that PoS schemes don't require verifiers to risk any exoteric resources requires dealing with a whole new set of challenges.

One major challenge is that participation in the validation process is not a direct concern of most users of the platform. Someone using a blockchain for transferring value, or a user of an application that makes use of smart contracts, would usually not be aware of the mechanics of the decentralized ledger, and is likely to have little motivation to participate in the process. This is especially problematic for real-time validation of transactions, in which the validator is required to be constantly online and to allocate network and processing resources to a nonstop feed of transactions.

Some platforms try to replicate the ecosystem of specialist miners that are commonplace in PoW systems. This includes a wide variety of models, ranging from purely permissionless models in which would-be verifiers need to stake or burn tokens as a form of earnestness, to semi-permissioned models such as delegation, where stake-owners trust their voting rights to a specialist. Though the models greatly differ, all face the same fundamental challenges of a

[51] https://eprint.iacr.org/2016/1159.pdf
[52] https://eprint.iacr.org/2018/104.pdf

lack of incentives for validators to act honestly and an increased risk of formation of a dishonest majority.

There are attempts at creating a full-participation, direct voting PoS systems. One notable example is AlgoRand[53] by Chen and Micali, which very cleverly weighs a random sortition by user stake, thus requiring only a small sample of users to participate in validation at any given moment as some kind of jury duty. We have high hopes for such models, but it is important to note that at the moment there are significant practical barriers to their implementation in mainstream applications. Mass-market users cannot be expected to be active in the technical process of transaction validation, nor to verify the software they are using for such validation; they merely download an app from an app-store. In practice, this gives the app developers total control over a user's voting power, making the system just as good as a delegated PoS system with all its disadvantages. We hope that newer advances in cryptography, such as proof bearing code and other innovations, can make these systems practical for mainstream applications.

Permissioned Models

The idealistic nature of the blockchain community has traditionally pulled towards consensus designs that are decentralized, permissionless, open and transparent. By relaxing the constraint on being permissionless, Sybil attacks are no longer a concern, and faster, more efficient consensus algorithms can be used. Furthermore, permissioned validation also entails that the identity of the validators is known to all. Not hiding behind the veil of anonymity, validators may be required to make public commitments to abide by the rules of the protocol; in such case even the rules that cannot be enforced using technical measures, may be enforceable in commercial lawsuits.

There are two forms for setting up a public permissioned network in decentralized context. The first, a *consortium*, in which a central body governs the network thus distributing the validation permissions. Real-time validation permissions may still be considered decentralized, although the question of whether the governing body carries liability for operating the network remains. The second, a *federation*, leaves the governance decentralized: permissions are not common to the network but are rather specified by each user or app developer. Different users may be seeing different projections of the ledger, when they don't share the same set of permissioned validators. In some architectures, this adds significant complexity to the consensus protocol and ledger structures, however it remains very simple in platforms that provide blockchain virtualization.

Federated models for public blockchains are well-established in the industry with projects like Ripple[54] and Stellar[55] steadily gaining ground. These projects maintain a high degree of

[53] https://arxiv.org/pdf/1607.01341.pdf
[54] https://ripple.com/
[55] https://www.stellar.org/

decentralization, openness and transparency, where any party can set up a node and verify the history for transaction integrity. The permissioned aspect of the model comes into play in the validation of new transactions being written onto the chain. Every node can specify the list of nodes it trusts to participate in the validation of its transactions, thus creating a combination of groups with differing consensus quorums.

A Layered Approach

How should we choose a consensus strategy best suited for the use case of large scale consumer apps? The first question to ask is how should political power be distributed in the network. Consumer apps are already a primary and certain stakeholder as they drive traffic to the network and bring their user base. We established that consumers trust consumer brands through which they make transactions in decentralized apps. Is adding additional stakeholders as validators in a permissionless way beneficial for consumers?

We believe that today, federated blockchains offer the best solution for mass market consumer applications, both in performance and in alignment with the interests of the consumers. We do not expect consumers to be directly involved in governance of the blockchain anytime soon; any practical system that wishes to align with the long-term interest of the consumer can either empower app developers or interested 3rd parties such as miners. App developers are already the trusted, dominant stakeholder in the app market, entrusting them to this power maximizes the benefit for the consumer. Distribution of power between developers should be such that limits the individual power held by each one separately. All these requirements are best met by federated consensus models.

Are federated models open enough? We can try to answer this question in several ways. From the practical perspective, the answer is clearly "yes," it puts the trust in parties that have a clear stake in the success of the platform and that proved to collaborate well in most scenarios (with the exception of blocking competition). From a strategic perspective, it is unclear whether these models are stable over time, especially in comparison to PoW models which are considered very robust. From the legal and regulatory perspective, there is no ruling currently on the subject, either for or against. Our analysis (see *Decentralized Ledger Security*) and the de-facto prominence of federation models among today's leading blockchain platforms provide some indication that these models are seen as sufficiently decentralized.

As the market matures, new insights or new regulatory approaches could tip the scales against federated governance. In such circumstance, we anticipate that the governance of the federation itself (accepting and rejecting of federation memberships; changes to federation members' permissions; changes to the federation rules) will transition to a permissionless model such as delegated PoS, similar in principle to the governing model of EOS[56] or TON[57].

[56] https://eos.io/
[57] https://techcrunch.com/2018/01/08/telegram-open-network/

This structure preserves most advantages of the pure federated model, but remains our second choice because it lacks the simplicity and elegance of the federated model.

Helix Consensus Algorithm

As blockchain technology reaches mass-market apps, we realize that classic forms of decentralized consensus are simply inapplicable. The volumes of transactions in a mass-market setting would cause PoW consensus to be too expensive, too slow and cause too much environmental damage. The fact that the apps themselves, rather than the mass-market users, are in control of the users' voting power make PoS too risky of a choice for such a platform. Moreover, the typical pattern of app popularity is that of extreme inequality: at any given time and for almost any segment, a handful of popular apps overpower the infinitely long tail of less-popular apps. Because we think the ideal power distribution should be one that avoids significant inequalities in power, we prefer to create a system that assures an upper limit to any one stakeholder's power.

With the birth of the Orbs platform, we will be introducing Helix, a decentralized, open and transparent consensus algorithm tailored to the ideals of mass-market apps. Our fundamental assumption is that app vendors hold most of the power in a platform that caters to apps, and that a consensus protocol must be designed from ground up to ensure that vendors' interests are aligned with each other and with the interests of their users. For network governance, this means the protocol has to work natively with chain virtualization, as it allows the governance of each app to be separate from that of other apps. Beyond that, voting power is distributed between known verifiers that are members of a federation, thus limiting the power held by a single voter. For real-time validation, we require the protocol to be fast, immediately-final and for it to be impractical for validators to manipulate the selection and ordering of transactions.

The full details of the algorithm are published in a separate peer-reviewed technical white paper. The primary requirements that dictated the design of the algorithm include:

Finality of Consensus Results

The Helix consensus algorithm provides immediate transaction finality. In business applications, transaction finality is a highly desirable property - enabling stakeholders to provide intended services immediately once the transaction is executed. Unlike the probabilistic finality of transactions in systems like Bitcoin, stakeholders are not required to wait for multiple blocks in order to gain sufficient confidence that a transaction will not be reversed.

The finality property also enables efficient usage of the state database. The state database can be updated under consensus at the closure of each block and its authenticity can be easily validated by its root hash. Keeping a state database that is always under consensus

prevents the need for high bandwidth access to a large transaction journal and the need for additional checkpoint mechanisms.

Ordering of Opaque Transactions

An important property of consensus algorithms that rarely gets the attention it deserves is fairness. Many algorithms rely on full-nodes or miners to decide the fair order in which new transactions are inserted into blocks, without defining rules or method to enforce fairness. Furthermore, some networks give miners the freedom to choose which transactions to include in a block, thus creating preference for high-fee transactions and enabling censorship.

The Helix consensus algorithm uses ordering of opaque transactions to ensure fairness and censorship resistance. Transactions are encrypted by end-users before transmission and are only decrypted after consensus on their ordering was reached. This mechanism guarantees clients receive fair service without a need to trust nodes or to provide them with direct incentives.

Separation of Ordering From Validation

Pending encrypted transactions are first ordered, and only once consensus on the ordering is achieved, are the decrypted transactions executed, achieving consensus on their validation. Separation of ordering from validation enables higher scalability and a higher transaction rate. Moreover, it allows the system to optimize the properties for each stage, such as committee size or the use of encrypted data.

Fast Consensus by Committees

The amount of communication in consensus protocols is highly dependent on the number of nodes participating in the consensus. On one hand, we desire to increase the number of nodes in order to increase the decentralization and security. On the other hand, we want to control the amount of inter-node communication in order to reduce confirmation time and increase throughput. Using randomly selected, unpredictable committees that take active part in the creation of each block allows the system to increase the overall number of nodes while maintaining an upper bound on communication overhead.

Efficient Leader Election by Randomized Sort

Many Byzantine Fault Tolerance algorithms, such as PBFT[58], are based on rotation of a primary or a leader node. In order to ensure liveness, these algorithms require mechanisms to identify a faulty leader and act for its deposition. These mechanisms are typically complex, rely on timeouts and result in a slow transition in cases where a new leader election is

[58] http://pmg.csail.mit.edu/papers/osdi99.pdf

required. The transition overhead discourages frequent leader rotation resulting in imbalance and suboptimal fairness.

In order to efficiently and randomly elect different leaders for each consensus round, Helix uses sortition for committee and leader election. For each block, the consensus nodes are sorted based on a hash of the decrypted data of the previously ordered block, which provides a random and consistent selection. Using the decryption that is only available after the previous block has reached consensus prevents a leader from manipulating the block data in order to control the next block committee members. The availability of a sorted list of the committee members enables efficient fault tolerant communication protocols. These can reduce the amount of network traffic and maximum propagation time, thus improving transaction rate and scalability.

Node Reputation System

The consensus algorithm operates in a Byzantine environment, where some of the nodes may be faulty or act maliciously. Moreover, not all the consensus nodes are homogeneous and their performance or responsiveness may vary. In order to rapidly identify faulty nodes, balance resources and incentivize nodes to behave according to the protocol, the Helix algorithm maintains a decentralized reputation system where every node is scored by its peers. Node reputation affects the political power of a node, such as its chance of being included in committees. Reputation also assists in economic incentivization, such as payment of fees to operators. For example, a node behaving in a manner discouraged by the protocol will have its reputation score reduced accordingly and will be able to charge less for its services.

SERVICE LEVEL AGREEMENT (SLA)

Industry Standard

A Service Level Agreement (SLA) is a contract that describes the official commitment between a service provider and a client. It describes the expected level of service, the metrics used to measure it and the penalties in case the agreed service level is not achieved. SLAs are widely used for services provided between organizations or even within an organization. It is an industry standard in telecommunication, Internet providers, online services and also - cloud Infrastructure as a Service (IaaS) providers such as AWS.

A separate SLA is typically defined for each of the provided services. For example, in an IaaS platform, each of the core services (compute, storage, networking) would have their own SLA. Users may have a choice between different SLAs, enabling different customers to plan according to their needs. For example, an online consumer application may focus on availability and consistent performance. Alternatively, offline applications may prioritize average performance over consistency.

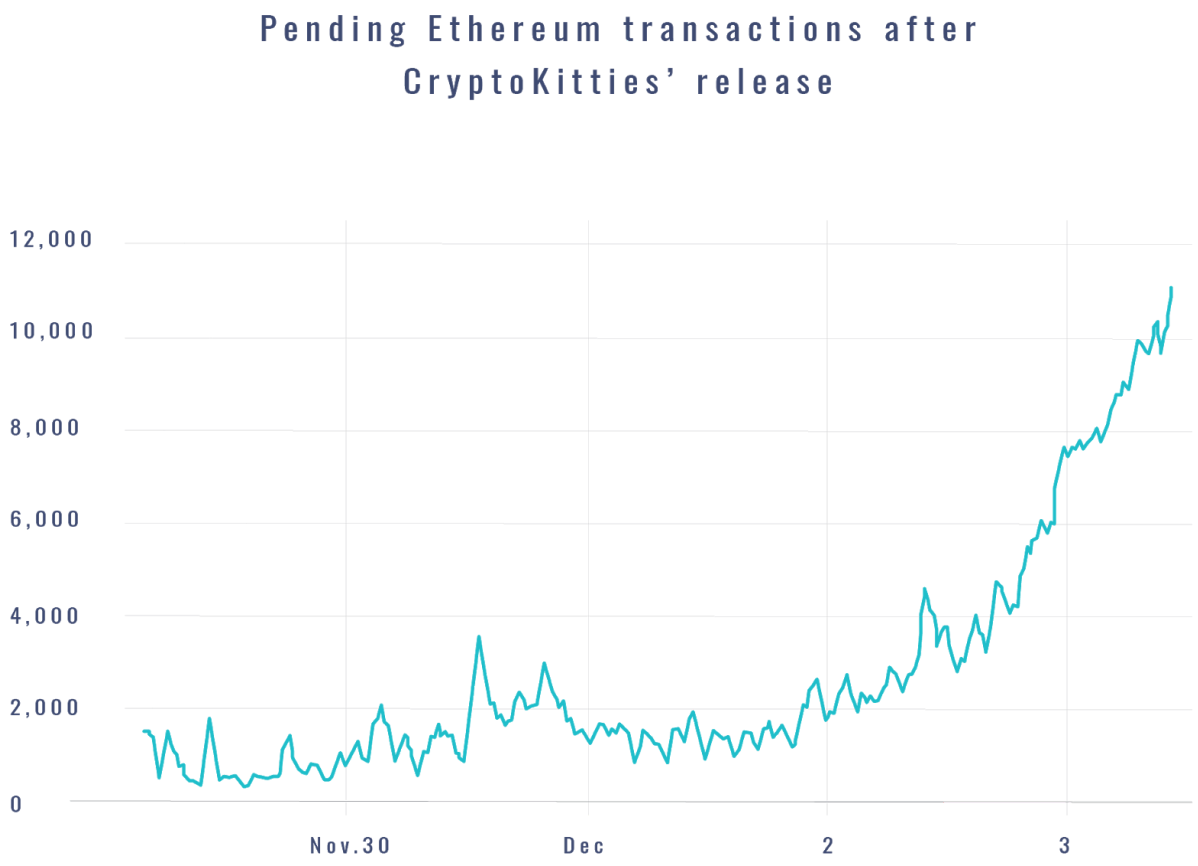
An SLA protects both the customer and the service provider. It prevents misunderstandings and misinterpretations by setting expectations explicitly. Moreover, an SLA helps customers predict the level of received service in advance and budget accordingly. For service providers, an SLA provides a means to estimate required resources and plan ahead.

While SLAs are prevalent for applications running on centralized IaaS platforms, they are lacking for decentralized ones. The inability to predict performance and costs reliably creates a challenge for consumer brands to transition to decentralized businesses.

CryptoKitties - a Case Study

CryptoKitties is a social game operating on top of the Ethereum platform that allows players to adopt, raise, breed and trade virtual cats. While the kitties themselves are cute and delightful, the surge in popularity of the game and the amount of traffic it generated have managed to bring the Ethereum network to severe congestion exceeding its capacity.

CryptoKitties was launched at the end of November 2017 and immediately received high interest from users, pouring game-related transactions to the Ethereum network. In the days that followed the launch, game related transactions were almost 20% of the total transaction volume on Ethereum. As a result, the amount of unprocessed transactions rose about six-fold [59] and transaction fees climbed accordingly.



The resulting congestion on Ethereum has drawn attention from the crypto community and the media. Many of the articles covering the phenomena raised questions about the general scalability of blockchain technology[60]. There's no argument that highly scalable solutions are desired, but focusing on scale alone misses the main issue.

Decentralized applications running on top of blockchain infrastructure cannot base performance and fee estimations on whether or not a popular game will pop up. Popular apps like CryptoKitties should not be viewed as a potential problem; popular applications showcase the potential of blockchain technology and the impact it can have on everyday life.

Kin by Kik interactive launched their Kin IPLv2 to production at the apex of the CryptoKitties craze. As a result, their launch suffered from significant side-effects of the impact CryptoKitties had on Ethereum[61]. Their main conclusion was that SLAs are sorely needed.

[59] https://www.theatlas.com/charts/rkt8jKMZz
[60] http://www.bbc.com/news/technology-42237162
[61] https://medium.com/kinfoundation/insights-from-kin-initial-product-launch-441c458a4ece#479b

Consumer applications require an environment with predictable performance, transaction rate, confirmation time and fee cost. A spike of 20% increase in traffic is difficult to handle by any infrastructure, centralized or not. However, an infrastructure solution that applies SLA rules and isolation among applications would have had minor impact on existing applications and would be more likely to contain this impact in agreed upon boundaries. For example, isolation would assure a surge in CryptoKitties traffic would not cause congestions in other applications.

SLA in a Decentralized Context

One challenge in providing an SLA in an open and decentralized platform is that platform users don't have direct agreements with infrastructure providers, so they don't have a natural counterparty with which to make such an agreement.

Basic service on the Orbs platform is provided by shared resources, but as shared resources get overloaded, the service level cannot be guaranteed. The platform introduces a service level mechanism by allowing application developers using the platform to acquire dedicated resources.

By default, the Orbs platform can provide users interested in dedicated resources with a minimal quality-of-service enforced by smart contract code. In case the service provided falls below the requested service level, the user will be automatically compensated at the expense of network nodes that did not contribute the processing performance expected from them. A smart contact based scheme should be sufficient for most consumer applications, alleviating the need for direct legally binding agreements between parties.

Some users of the platform may require legally binding agreements, for example applications that need to provide such SLA to their own users. If such requirement arises, the Orbs platform could provide a mechanism in which application developers purchase dedicated resources directly from operators of network nodes. By acquiring sufficient resources to support the throughput required for the operation of their application, developers can guarantee a minimal service level for their users, backed by a user agreement. This can be conducted in a marketplace where app developers and node operators can trade directly. Such external agreements can be facilitated by the protocol by indicating the usage agreement on a transaction alongside the signature used to authorize it.

Of course, acquisition of dedicated resources does not come in place of decentralization. The acquired capacity will not be provided directly to the purchaser, but rather added to the shared resource pool. So, in practice, acquiring the resources establishes an SLA between the node operator and the shared pool back-to-back with an SLA between the shared pool and the purchaser. In case the node operator fails to deliver the required capacity when the system is under load, a compensation amount will automatically be deducted from their fees.

Regardless, it is still possible that other providers on the shared pool have vacant resources and the quality of service with the purchaser can be maintained.

Predictable Fee Models

As discussed in the CryptoKitties case study, one of the main challenges witnessed during the launch of Kin by Kik Interactive on top of Ethereum is unpredictability of fees. Kin took it upon itself to subsidize infrastructure costs for Kik power users, in order to encourage end users to adopt the token with minimal friction. The main problem, as it turned out, wasn't that the fees were high, but that they were impossible to plan and budget for in advance.

Budgeting is crucial for the success of consumer applications: product development is expensive, and entrepreneurs or companies want to know in advance, that if their app succeeds, they will get a return on their investment. To know that, they need to weigh in their operating costs.

During the first few months of the Kin launch, Ethereum fees changed in a full order of magnitude. First, the USD price of Ether, notorious for being volatile, increased by 200% during the launch period. Since fees are paid in Ether, this exchange rate fluctuation translated directly to an increase in costs. Second, Ethereum fees are determined by market forces, where transactions offering higher fees will be processed first. During the Kin launch, the Ethereum network became congested, resulting in higher bids from users across the network. To get its transactions across, the Kik app had to offer significantly higher fees.

The Orbs platform is designed to provide a pricing and fee model that is predictable and can be calculated in advance. We believe that such certainty is a fundamental requirement from a platform. The industry standard from centralized infrastructure solutions, such as AWS, is to provide accurate pricing calculators[62]. Prices of on-demand services and subscriptions are listed in ORBS tokens.

The fact that services are listed in ORBS, a token with floating exchange rate, creates risks for both service sellers and buyers. In case of exchange-rate fluctuations, ORBS tokens may increase in value, effectively raising the infrastructure costs to Orbs platform users, or decrease in value, possibly making the operation of a validating node non-economical. Additionally, operation expenditures are exposed to price fluctuations of the underlying IT infrastructure (storage, processing, network access etc), but their prices have tended to decline gradually over time. As for exchange-rate fluctuations, we expect them to be slow-paced. Economic models predict that high rate of economic activity in a cryptocurrency attenuates exchange rate fluctuations[63], this should be the case with ORBS tokens, thanks to the amount of activity expected from the launch partners.

[62] https://calculator.s3.amazonaws.com/index.html
[63] https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2842557

To accommodate potential price fluctuations and provide service cost stability to users, on-demand service prices will be normalized periodically to match changes in IT service costs. This is done by pegging the on-demand tariffs to an index of storage and compute unit prices, as published by major third party cloud service providers such as AWS. In the future, it may be possible to replace the cloud services index with an index of dedicated capacity prices, as these are determined by supply and demand in free market settings. Such a solution would be more sustainable over time than a cloud services index, because it is directly linked to de-facto costs of node operators, but its implementation requires some experience in terms of how the marketplace for dedicated capacity behaves.

Dedicated, Reserved and On-Demand Resources

One of the challenges in dynamic resource management is the inherent trade-off between the ability to share resources and the ability to guarantee their availability. In general, we differentiate between three main schemes of resource allocation:

Dedicated resources - A physical resource is dedicated to the application, providing maximum isolation, high predictability and visibility. Dedicated resources are guaranteed to always be available for the customer. These resources must be paid for, even when unused, making this scheme of allocation the most expensive of the three.

Reserved resources - An amount of resources is provisioned in advanced. Reserved resources may be guaranteed under some restrictions or prioritized over on-demand resources. As reserved resources are provisioned in advance and allow the service provider to plan better, they are typically provided with a significant discount over on-demand resources.

On-demand resources - Resources are shared between applications and are allocated on-the-fly based on availability. Payment is normally based on actual usage. On-demand resources are recommended for low cost applications or for application with unpredictable workloads.

A common strategy for an application looking to optimize is to allocate a mix of resources. For example, an application may allocate dedicated resources to guarantee the minimum performance required for basic operation, reserved resources to meet a typical workload and on-demand resources to accommodate peak usage.

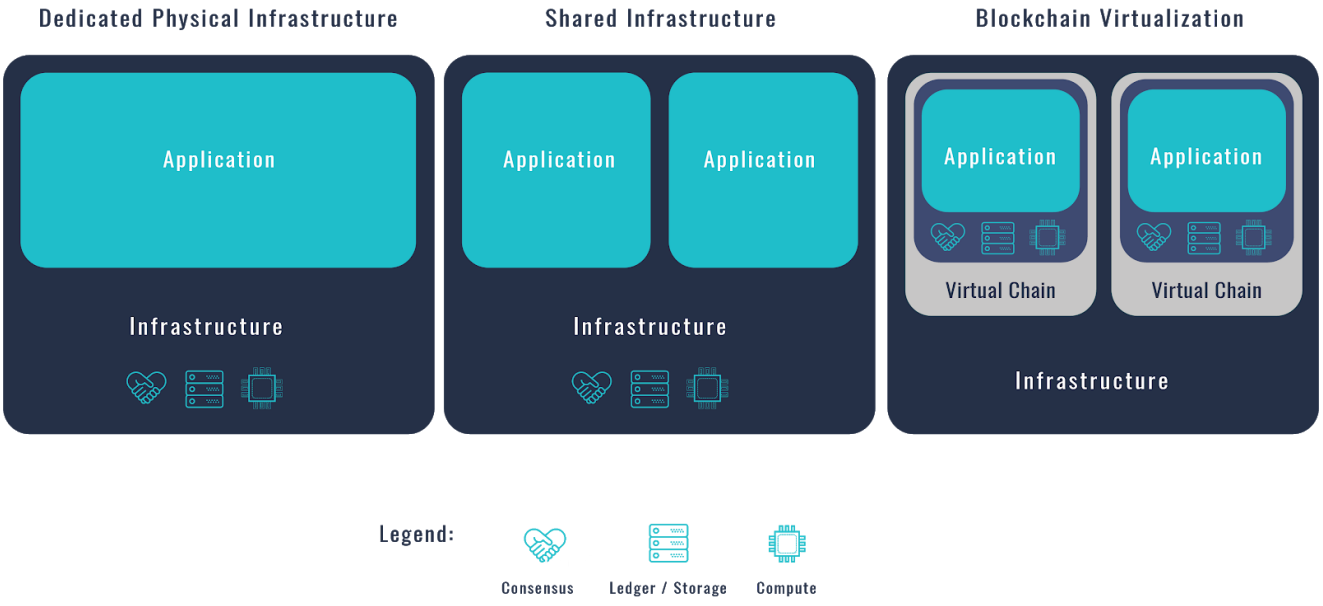
Virtual Chains and Blockchain Virtualization

Blockchain virtualization implemented over the Orbs platform provides the illusion of a dedicated blockchain while running on top of a shared physical node infrastructure, thus enjoying the same security and decentralization provided by the shared environment. Virtualization separates the resources available to applications from the underlying physical ones. The properties of blockchain virtualization include isolation, quality of service, SLA,

control, governance and elastic resource capacity. The majority of blockchain implementations today, like Ethereum, are shared, where multiple decentralized applications run side by side without isolation, suffering from unpredictable performance. Blockchain virtualization allows us to overcome these limitations without compromising on the risks of a centralized or private infrastructure.

Almost two decades ago, the industry started to shift towards server virtualization. Today, almost every consumer application runs on virtual machines. A similar shift started a decade ago in networking, where virtualization enabled large networks with flexible topologies to run as virtual networks over an underlying shared infrastructure, providing the look and feel of a dedicated private network. We expect the same industry shift to take place in the field of blockchain.

The term "virtualization" broadly describes a layer of abstraction that provides separation of a logical resource from the underlying physical delivery of its function. With blockchain virtualization, each component of the blockchain infrastructure, such as the consensus, the state and block storage, and the virtual machine (compute) layer are virtualized. This allows virtual consensus instances to be allocated with a desired transaction confirmation rate that can differ across virtual chains. Moreover, different virtual consensus instances can operate concurrently, scale gracefully and provide better utilization of resources. Unlike private blockchains, virtual consensus instances enjoy the security, resilience, decentralization and compliance benefits of the underlying shared infrastructure.



Dedicated physical infrastructure - First generation (Bitcoin). Each application runs over dedicated infrastructure and has its own separate blockchain.

Shared infrastructure - Second generation (Ethereum). Multiple applications run on top of shared infrastructure. The consensus, storage and compute services are shared across applications without isolation or SLA commitments.

Blockchain virtualization - Third generation (Orbs). Each dominant application runs on a separate virtual blockchain, relying on virtual instances of the consensus, storage and compute services, but sharing the same physical infrastructure.

Design Principles

Blockchain virtualization addresses some of the challenges that decentralized applications are facing and provides properties that resemble the familiar operation over centralized IaaS or cloud platforms. The full architecture details are published in a separate technical white paper. The design principles of the solution include:

Service Level Agreement (SLA) - Each virtual chain may guarantee a service level to meet its needs. Adhering to an SLA is a commitment to reduce the performance impact of other applications that share the same physical infrastructure.

Isolation - The separation of block storage and state of each virtual chain creates isolation from faults and errors that occur on other chains. For example, a bug in an application's smart contract may lead the virtual chain to fork but will not impact other virtual chains on the network.

Sharding and scalability - Virtualization enables inherent sharding of the consensus, and a first level of sharding for compute services and state storage. As there is no synchronization dependency among different virtual chains, their consensus and storage can be handled separately and run concurrently.

Governance - While some configuration parameters have to align across all virtual chains, many can be controlled independently. This allows every virtual chain to optimize and cater to its application's needs and reduces governance conflicts between stakeholders.

Elastic capacity - Separation between physical and virtual resources allows a virtual chain to add resources on demand in order to meet evolving usage patterns. Moreover, elastic capacity allows temporary allocation of resources during unexpected bursts.

Security and decentralization - While virtual chains can be dedicated to a single application, the multitude of physical nodes operated by independent organizations and applications are used in practice to process its consensus, capitalizing on the security in decentralization.

Cross virtual chain smart contracts - While isolation is important for transactions within an application or virtual chain, simple cross-chain interoperability proves useful. This requires synchronization of all involved chains. Such operations are slower and require more resources than a standard operation.

CONSUMER SCALE

Throughput and Latency

The first challenge when designing a blockchain infrastructure for consumer scale is meeting consumer expectations regarding throughput and latency. Successful consumer products have the potential to reach millions of end users performing billions of interactions. This massive scale regularly pushes traditional centralized infrastructure to its limit, making the challenge for consensus-based decentralized infrastructure that much greater.

Throughput is defined as the number of messages per second the network can accommodate. In the field of blockchain, the number often taken into account is the number of *transactions per second* that the network can confirm. Traditional blockchain implementations, such as the current production version of Ethereum, can confirm about a dozen transactions per second[64]. The gap is significant, but this isn't surprising since decentralization comes at a price. For example, transactions over blockchain are notoriously difficult to parallelize, because the result of one transaction may depend on another. Performing transactions synchronously adds a significant constraint and makes the implementation much harder to scale out. Moreover, contrary to centralized systems, the consensus process involves a number of independent nodes that must reach agreement over every transaction. This process incurs significant overhead that centralized systems are not required to deal with.

Latency is defined as the amount of time it takes to process a single message over the network. In the field of blockchain, the number often perceived by users is *confirmation time*. If, for instance, Netflix was a consumer product on the blockchain, then a request to stream a video must be confirmed immediately so that the user would not have to wait to start enjoying the video. Traditional blockchain implementations, such as the current production version of Ethereum, take dozens of seconds to confirm a transaction[65]. This number often grows to minutes or even hours when the network becomes congested. The gap here is not surprising as well. First, performing transactions synchronously means a transaction must wait in line

[64] https://blog.ethereum.org/2018/01/02/q4-roundup/
[65] https://etherscan.io/chart/blocktime

and will only be processed once every previous transaction has been processed. Second, the consensus process between a group of independent nodes usually requires several roundtrips and is constrained by propagation time of the network which increases as more nodes participate. Third, long block intervals are essential for security in some consensus algorithms. In models that rely on eventual consensus, actual confirmation is only reached when an arbitrary number of subsequent blocks are generated.

Failure to meet consumer expectations in both of these regards threatens the very success of the product. Consumers are notorious for having low tolerance for bad user experience. Their expectations are determined by the experience they are used to getting from current, centralized applications. It is reasonable to expect that most consumers will not be aware of whether the application they use is decentralized or not.

Scalable Fee Models

Scalability of a system goes beyond raw network parameters like throughput and latency. According to Kin by Kik Interactive, the main barrier to scale when launching on Ethereum was infrastructure fees[66]. The CAC (Customer Acquisition Cost) rose above $10 in transaction fees alone[67]. It is quite clear that a successful consumer app cannot flourish in this environment. Reaching ten million users would cost over $100 million - beyond the total funding of the project.

An obvious solution is to reduce infrastructure fees dramatically. The high costs of Ethereum are closely linked to its reliance on PoW consensus; in PoW, the operation costs grow proportionally to the total value of assets on the network. In case the value grows, the process becomes more wasteful to maintain the proportion. In addition, the number of nodes in the Ethereum network that validate every transaction is more than 20,000[68], orders of magnitude more than reasonably required in a distributed system. Both of these cost factors can be eliminated by moving away from PoW and using committees to reduce the number of participants in consensus.

There's more to fee scalability than reducing absolute amounts. Network usage peaks may cause fees to spiral out of control. In general, although market pricing is seemingly the most efficient for determining fees, it is problematic when markets are too volatile. For instance, it may cause entire apps to experience outages. Consider two popular consumer apps with millions of users running side by side when demand exceeds the network capacity. Once the first app modifies its client and increases the transaction fee bid over the other, at one instant its millions of users will have precedence over those of the other app, causing an overwhelming outage for that app.

[66] https://medium.com/kin-contributors/kins-blockchain-considerations-ebd0b60aebd5#2340
[67] https://medium.com/kinfoundation/insights-from-kin-initial-product-launch-441c458a4ece
[68] https://www.ethernodes.org/network/1

In Orbs, app developers can purchase reserved capacity in advance, protecting themselves from price fluctuations; they can procure dedicated resources, isolating themselves from peaks in demand; and they can use monthly subscriptions, reducing the overall exposure to price volatility.

Another fee-related decision that takes a significant toll over scalability is charging fees per transaction - the common approach taken by general-purpose blockchains such as Ethereum. This imposes a large overhead on the network operation. Processing of each transaction requires writing to the native token's ledger, making transactions harder to shard (since even unrelated contracts need to be synchronized). In addition, per-transaction billing adds an overhead in processing and storage, when compared with bulk subscriptions.

A different fee model that is used to reduce per transaction costs in the industry is minimum balance per wallet. This model is used for example in blockchain platforms like Stellar. The system is designed to provide the necessary friction to reduce spam and fake transaction abuse. Once a user has proven to the platform that their wallet is indeed "real", by placing a certain amount of tokens in this wallet in this case, the platform lifts usage limitations and allows the user to place a large number of transactions from this wallet for a negligible fee.

The problem with this approach is that consumers are not likely to commit up front for a service that they're not sure about. Expecting end users to pay the minimum balance fee will probably drop conversion below a level that consumer products can accept. What happens usually, is that consumer-facing digital services, the providers of the decentralized apps, attempt to subsidize these fees to attract customers. Once these fees are subsidized by a 3rd party, they lose their edge in countering Sybil attacks. Even worse, they create a new target for Sybil attacks, enabling attackers to pocket the subsidy in addition to any other gains. By enabling subscription payments as alternative to per-transaction fees, the Orbs platform limits the bounty one can find in such a Sybil attack, and the costs can only be laid on the digital service - who is the only party in power to mitigate such attacks.

Ever-Growing Storage

A big cost factor in current generation blockchain platforms is the fact that resources often scale without correlation to actual technical requirements. On Ethereum, for example, there would be as many full copies of the blockchain storage as the number of full nodes, and about as many processors running every piece of smart contract code. While distributed and decentralized systems do require a certain level of redundancy in both execution and storage, the proper redundancy for them would likely be constant, and in most cases not more than a dozen or two. Though fees are only paid to the solver of the PoW puzzle, all miners expect the mining operation to be cash-flow positive, so in equilibrium the fees would offset the total

costs for all miners. The architecture of the Orbs platform assures redundancy of any component is within determined bounds.

Storage is a cost generator in another sense. Orbs storage APIs natively define expiration for data, as well as explicit expiration blockchain history. This ensures that data has a defined (rather than infinite) time to live, and greatly reduces storage costs to a magnitude expected in cloud services. In addition, relying on consensus that provides finality allows the Orbs platform to maintain the state under consensus and removes the need for high bandwidth access to block storage.

Light Clients

As indicated in our discussion about *network entities*, consumers will access the network primarily by using mobile apps and websites. These usage patterns are characterized by extremely low availability of resources, thus requiring a thin client implementation that is commonly referred to in the industry as a *light client*. These clients do not synchronize over the entire blockchain history like a full fledged node and usually must maintain some relationship of trust with the node serving them as gateway.

The need to trust a gateway node creates a dependency of the client on the node's honest behavior and enables vulnerabilities such as man in the middle attacks. In order to mitigate the risks, some clients perform a partial validation of the state by validating block headers. Another common strategy is validating data by querying multiple nodes, which of course doesn't scale well. The problem is even more significant when a client needs to query a smart contract, as it requires the client to trust a node that runs this smart contract on its behalf.

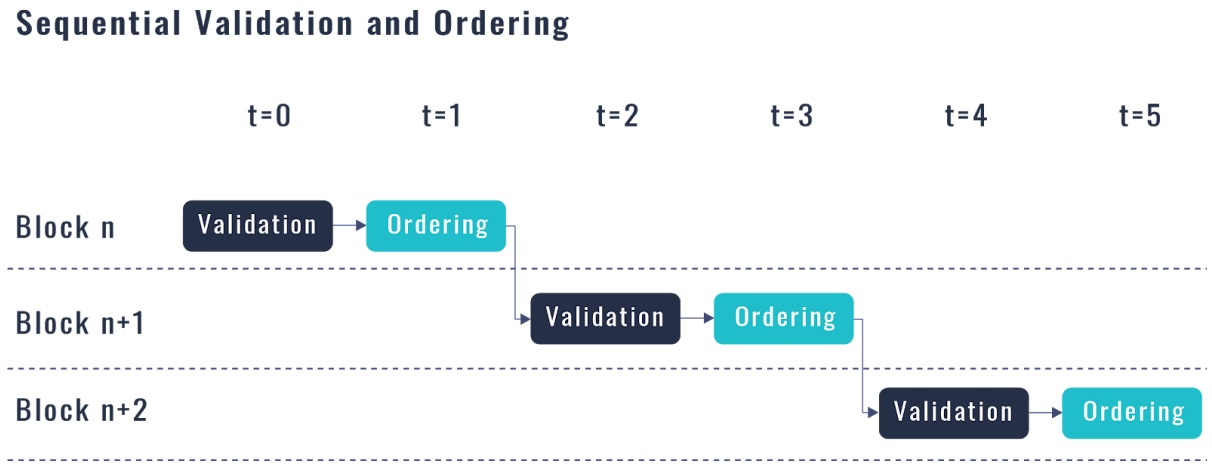
We see value in providing light clients that can operate with a low level of trust in a gateway node. This capability is provided over the Orbs platform using *network owned secrets*. Going back to the smart contract example, the client will provide the contract address and arguments to a gateway node, which will perform the query and return a signed response. The light client will be able to validate the signature of the network as a whole, thus reducing the level of trust needed from the specific gateway. An additional aspect of the Orbs platform that reduces the need for a light client to trust a node is that the protocol guarantees fairness in transaction ordering. By transmitting encrypted pre-consensus transactions that are opaque to consensus nodes, we can ensure that transactions will be ordered for execution without censorship or bias.

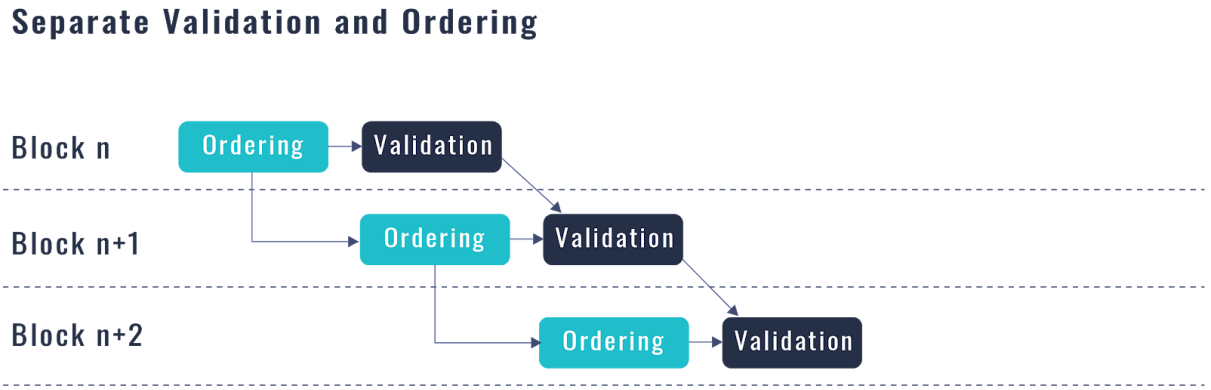
Separation of Ordering and Validation

The Orbs platform relies on multiple strategies to increase scalability by several orders of magnitude in order to meet the requirements of mass market consumer apps. Beyond the careful selection of a consensus strategy that favors professional nodes that are incentivized towards maintaining high SLAs regarding connection speed, uptime and processing power,

several other strategies are incorporated into the platform. The first strategy is separating consensus ordering and validation.

The validation process of transactions and smart contracts is expensive and incurs duplicate computational cost since it has to run over multiple nodes. When validation and ordering are performed sequentially, transaction throughput is limited by the overall time required for the completion of both. Separating the two creates a *pipeline*, thus increasing overall throughput.





In addition to improvements in throughput, validation of ordered transactions is an easier problem which permits simpler schemes for concurrent computation. Moreover, it allows the network to reduce the amount of consensus nodes required for validation, thus promoting better resource utilization overall.

Many existing blockchain implementations perform validation and ordering sequentially. Nodes normally execute all transactions first to validate their output and only then propose a block comprised entirely of valid transactions. Separation techniques are used by few state-of-the-art implementations like Hyperledger-fabric[69], where a transaction is first sent to a group of endorsers that execute it and return proposed responses. When enough endorser responses are collected, the transaction is forwarded to the consensus ordering service. Performing validation before ordering works well for some applications. However, for

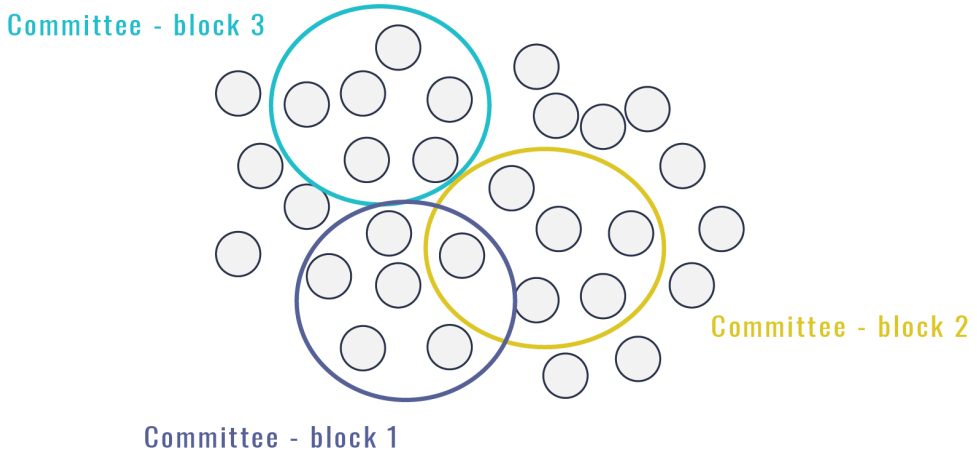
[69] http://hyperledger-fabric.readthedocs.io/en/release/arch-deep-dive.html

consumer applications we see a benefit in performing ordering first on encrypted transactions, in order to guarantee fairness.

Efficient Consensus via Committees

The second strategy used by the Orbs platform to increase scalability involves reducing the number of nodes directly participating in the consensus process. This is important since in most consensus algorithms, message complexity grows quadratically with the number of nodes. If we plan to scale the network to a large number of validators, an important goal for security and decentralization, it is certainly desirable that performance will remain within reasonable bounds.

An efficient method for reducing dependency on the total number of nodes is relying on smaller *committees* for consensus. If we randomize committee members between consensus rounds, e.g. on every new block, we can prevent an attacker from knowing which nodes to attack. The randomization process must fulfill several properties to make sure membership cannot be manipulated in advance or else we risk compromising the security of the entire model. The process is discussed in detail in the technical white paper for the *Helix Consensus Algorithm*.



Sharding via Blockchain Virtualization

In systems engineering, scaling systems cannot always be achieved by adding more resources, due to bottlenecks that cannot grow easily. Sharding is a technique for scaling out systems by dividing them into smaller, nearly-independent parts called *shards*, each small enough to work well despite bottlenecks. Decentralized consensus is an unavoidable bottleneck; the Orbs platform decouples tenants to different shards allowing the network to scale out horizontally as new tenant apps are added. Contrary to centralized infrastructure solutions like AWS, simply adding resources is usually not enough to increase capacity. When the number of products that use AWS grows, increasing infrastructure capacity by adding hardware like servers and network connections, is usually enough to meet demand since separate products run completely separately.

This normally isn't the case with blockchain. Transactions on Ethereum, for example, even by different contracts, may affect one another and therefore must be performed in sequence. To make matters worse, the number of verifiers in PoW blockchains is related to the level of security the blockchain enjoys; reducing unit sizes by sharding is reducing the level of security in the same proportion. Significant efforts are invested in research and development of effective sharding techniques in Ethereum. This problem is apparently much simpler to solve in the proposed architecture for Orbs. Permissioned consensus models, especially when employing consensus committees, do not weaken its security properties when sharded. Different decentralized consumer apps are independent and work with different unrelated assets. This is particularly true if fees are paid in bulk and not per transaction. By sharding decentralized apps by default, the architecture can revolve around parallelization.

The Orbs infrastructure is designed to support a large number of independent consumer applications running on top. While applications are independent from one another by design, when running on top of a shared infrastructure they enjoy the benefits of sharing resources, but still allow the system to scale out thanks to the natural sharding of virtual chains.

The 3 types of cost factors in blockchain applications are consensus rounds, reads and writes of state storage, and compute operations. Consensus on transactions of different virtual chains can run independently as long as there are no ordering dependencies among them. Therefore, consensus of different virtual chains can be sharded and run concurrently on separate resources. As there is no ordering requirement, the ledgers of virtual chains can also be maintained independently. Compute scheduling requires that dependent transactions will be executed in order. As virtual chains have independent ordering, their compute can be performed in parallel. Moreover, the isolation of state for each virtual chain reduces the memory requirements of its virtual machine.

Elastic Capacity

Consumer applications require an ever-growing increase in transaction rate, accounts and storage. Moreover, as additional applications use the infrastructure, there's a need for higher resources capacity. No transaction rate, compute, or storage resources, as large as they may be, that are allocated initially, can meet future requirements. In order to meet future capacity recruitments, there is a need for elastic capacity.

Elastic capacity requires that the architecture will enable blockchain components - such as consensus, compute or storage - to scale with the addition of resources. Moreover, on-the-fly updates in resource allocation should be performed without interruption to the operation of decentralized applications.

When an application requires additional resources in a centralized system, their capacity can be adjusted by a system administrator by providing additional resources, virtual or physical. For a decentralized infrastrastructure, there is a need for a decentralized mechanism for resource allocation. In addition, there is a need for an incentive mechanism for the nodes to provide the resources that are required by the applications.

CONSUMER PROTECTION AND REGULATION

Regulatory Evolution

In his 2006 book Code v2[70], Lawrence Lessig observes a pattern in which societies that emerged into a free, unregulated and anarchic state, grow to create the institutions, the structures and the constraints that rule them. Lessig shows how this pattern applies to the evolution of the Internet, and how invisible powers shape these institutions; left to their own devices, those powers may end up stifling the freedom and progress that could exist.

Bitcoin was introduced 3 years after Lessig published Code v2, and the crypto state has followed precisely the path he laid out. At first, blockchain presented itself as an architecture of freedom: unregulatable, self-ordering, free from control. As it matures, new structures emerge, forming the architecture of the ever-growing blockchain society. But the invisible hand in forming this architecture is not neutral. Established companies and distinguished and important members of the crypto community are the drivers of much of the innovation in the industry; this may create a force that will steer progress towards constructs that enable monopolistic control of markets. Governments, frequently out of genuine desire to protect users, use the crude power of law to shape the interfaces of blockchain with the "old world"; this may steer progress towards constructs that enable more control of the platform. It is up to the designers of blockchain protocols to assure the architecture we're forming is one of freedom and progress.

Established Consumer Brands

Through our work with design partners that have established brands outside of the blockchain domain, we observed the stark difference between their preferences to those of companies operating entirely inside of it. Typically, the former have millions of existing users and ongoing business operations that are expected to maintain their performance. As a result, they are particularly concerned with the risks of regulatory uncertainty associated with blockchain operations, perceiving the possibility of an encounter with regulators as one that puts their

[70] http://codev2.cc/download+remix/Lessig-Codev2.pdf

existing business at risk. Blockchain companies, on the other hand, tend to see this uncertainty as a natural state for a new industry - let alone one that introduces disruptive business paradigms - and ignore it as an industry risk.

Not derogating from the importance of pure blockchain companies and their remarkable contribution to the ecosystem, we believe it is wrong for platforms to ignore the newcomers' concerns in this case. For blockchain technologies to become mainstream, it is vital that established companies enter the field with their existing user bases. These companies cannot afford the legal uncertainty, and it is up to platforms to enable protocols that can comply with existing regulatory requirements.

Consumer Protection

Suppose an app developer wants their app to enable users to store or transfer valuable assets. This could be peer-to-peer payments, trade of virtual goods, payment for services and so on. The product engineering challenge is not big: the developer can choose from a plethora of industrial grade transactional databases, and put together a straightforward implementation of a ledger. But once the assets transferred are convertible to cash, new types of problems arise: this ledger becomes a big target for thieves, crackers and embezzlers, and anybody with access to it is at risk of being liable for damages or criminal accusations. Good protection against these risks is an elusive goal. In some contexts, someone in control of the ledger can even make profit in subtle and indirect methods, such as delaying or deleting transactions.

For a business whose core competency is consumer applications, properly protecting its users is a big burden. In many cases it could force the company to alter its procedures and structure in ways that will make it harder for it to go about its usual business. Most prefer to avoid such features if they're not their main product focus, or integrate 3rd-party solutions to provide such risky services.

To some extent, blockchain technologies have the potential to lower the barrier to provide such risky services, thanks to its reliance on secure cryptographic protocols and decentralization that mitigate or completely neutralize that risk.

Decentralized Ledger Security

Decentralized implementations for a ledger are easier to secure because the burden of securing the ledger is shared between multiple independent entities. When there is no centralized ownership or governance, no single entity can control the ledger and jeopardize it if it becomes compromised. Simply put, if nobody is in control of the ledger, nobody is able to steal off of it. In addition, multiple parties constantly audit the integrity of the ledger and can identify discrepancies from the agreed upon protocol.

In established PoW platforms such as Bitcoin or Ethereum, in a typical transaction, Alice sends some value to Bob by transmitting a transaction (signed with Alice's private key) in which ownership registration of a certain amount of token will be modified on the shared ledger to reflect the transfer. The signed transaction is propagated across the peer-to-peer network and reaches miners who verify its validity, and if valid, include it in a block candidate. Each miner then looks for a solution to the PoW puzzle on the block candidate. Eventually, one miner will solve the puzzle and publish the closed block. Other miners receive the closed block, check its validity, and, if valid, use it as the previous block for their future block candidates.

Let's see whether validators are in control of the virtual currency, which means the validators have sufficient credentials or authority to execute unilaterally or indefinitely prevent transactions for a user[71]. Mallory, a malicious miner, can simply prevent a transaction for users by not including their transactions in the block candidates; however she cannot block it indefinitely because other miners should eventually include it in their blocks. Could she execute a transaction unilaterally? Without Alice's private key, Mallory cannot create a valid proof that she has the authority to move value to Bob. But Mallory can create a block candidate that includes an invalid transaction moving Alice's funds to herself, and with some work can solve the PoW puzzle and get that block closed and published. Miners of future blocks are supposed to validate Mallory's block and disqualify it from being included in the blockchain because it contains an invalid transaction. Note that in both cases, what limits Mallory from wrongdoing is knowing that an undefined, random group of future miners will not confirm her blocks' validity.

In the case of Bitcoin and Ethereum, we see two properties that together assure the validators are not in control of the ledger: the cryptographic protocol makes it impossible for validators to send transactions unilaterally, and the openness of the network makes it impossible for validators to indefinitely prevent transactions.

The cryptographic protocol defines what valid transactions are, in a deterministic and universally accepted fashion. This means that if invalid transactions are included in a block, and the network consensus is to accept the block, then this consensus is not following the protocol and in essence it is not a consensus of the Bitcoin or Ethereum network. If we allow ourselves to use some circular logic, we could say that if Bitcoin accepts invalid blocks, then it's not Bitcoin anymore.

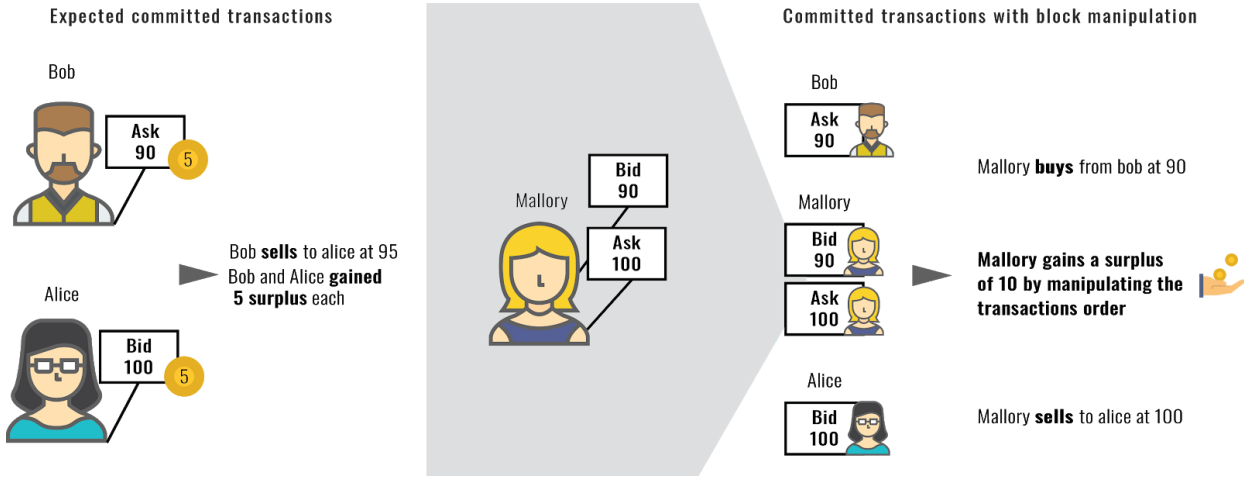
The openness of the network is provided by the ability of anyone to join the network as a validator. If Alice suspects her transactions are censored by the network's validators, she may join the network as a validator and approve her own transaction. Because her block is valid, future block validators should include it in their chain (as said before, if they're not following the protocol, it's not the same network).

[71] https://coincenter.org/entry/when-does-a-company-actually-control-customer-bitcoins

Censorship and Front-running

Relying on the openness of the protocol for combating censorship of transactions is not ideal in some real-world uses. Though miners cannot indefinitely prevent a user from transacting, a miner closing a block can arbitrarily choose to leave transactions out, delaying them to a future block closed by another miner. This power is significant in the hands of large mining pool leaders.

Moreover, miners can choose the order in which transactions are placed in blocks, and even craft transactions that will manipulate them for financial gain. For example, suppose we have a smart contract implementing a two-sided trading market in which a certain class of assets are traded for a currency. At a certain time, Alice posts a "bid" for an asset, for a maximum price of 100 tokens. Bob posts an "ask" for the same asset, for a minimum price of 90 tokens. A fair trading contract will split the spread and commit the trade for 95 tokens, leaving both Bob and Alice with surplus of 5 tokens. But Mallory, the miner who will close the block, can then add two more transactions: prepend Bob's transaction with a "bid" for a maximum price of 90 tokens, and append it with a transaction containing an "ask" with a minimum price of 100 tokens. The new sequence of transactions will cause the smart contract to sell Bob's asset to Mallory for 90 tokens and then sell it to Alice for 100, leaving the entire surplus of 10 tokens in her hands. A similar scenario, albeit in a more complex setting, was proposed by Emin Gun Sirer as theoretical attack against the Bancor smart contract[72].



However, we maintain that the challenge of combating manipulation by miners can be tackled with more adequate tools than openness. A consensus protocol in which validators agree on the order of transactions without knowing their contents could assure that validators don't have knowledge they can use for manipulations such as censorship and front-running. Such

[72] http://hackingdistributed.com/2017/06/19/bancor-is-flawed/#front-running

protocols, like HoneyBadger BFT[73], were proposed. A similar scheme is used in the Helix protocol.

Compliance Protocols

A common problem in dealing with blockchain assets is conformance to the requirements of regulation on ownership, custodianship and transfers of assets.

One class of international regulation that current cryptocurrency facilities struggle to comply with is Anti-Money-Laundering regulation (AML). AML regulation has been adopted globally in the past two decades and is seen as a systemic measure to combat crime and terrorism, by eliminating the financial incentives for criminal actions and the funding of terrorism. AML regulation that applies to financial institutions requires that asset owners be identified and their identity recorded, that sources of money transfers be verified, that large or anomalous transfers of money be reported to enforcement agencies, etc. Value transfers on cryptocurrency platforms do not conform with AML standards of financial institutions, which sometimes prevent the financial institution's ability to register the transaction. For example, some institutions require deposits to be associated with a payment and for the source of the payment to be proven legal. In virtual currencies in which payers are anonymous, it may be hard or even impossible to prove the legitimacy of the transaction.

Another asset class whose management entails regulatory requirements are securities. In many jurisdictions, securities law requires ownership of securities to be properly registered; if ownership of an asset by a single individual exceeds a threshold, it may also require reporting. Some assets may be limited in the ability to be owned by individuals, such as private companies whose equity can only be shared by a small number of unaccredited investors.

We intend to develop a framework of interfaces and smart contracts to allow representing common asset classes on the blockchain, with the intention of it being compatible with local regulation, including protocols that allow trading of assets between localities when the legal framework allows for that.

Privacy and AML

An interesting challenge is brought about when designing new protocols for a payment ledger that can be used in normal circumstances and interface with traditional financial institutions. On the one hand, consumers expect a high level of privacy from financial platforms; in blockchain platforms that aim to be a major payments method for their users, this will require a very high level of privacy. In many countries, this expectation is made mandatory by privacy laws. On the other hand, existing AML regulation requires significant compromises on user privacy. In traditional financial institutions, records are kept discreetly and so the institution

[73] https://eprint.iacr.org/2016/199.pdf

can maintain users' privacy towards the public eye, while exposing all information to enforcement agencies.

It is important to make the distinction between existing regulation and long-term enforcement strategy. Many enforcement agencies acknowledge that the transparency that a shared global ledger provides is a powerful tool for identifying anomalies, suspicious transactions and problematic accounts. Regulations that force virtual currencies to comply with procedures that make sense in traditional banking diminish both the commercial and the enforcement advantages of newer technologies. We believe that in time, regulators and enforcement agencies will prefer protocols that are more suited to the nature of virtual currencies and blockchain and are better at safeguarding users' privacy.

Our intention is to make an effort on both fronts: design our payment protocols to comply as much as possible with any existing regulation which may apply; and to design novel, forward-looking protocols, that can provide users with superior privacy and law agencies with sufficient tools to combat crime.

The White Chain

As one can expect, not all transactions and not all use cases will see value in conforming to the protocols. For example, apps that only use small payments that are exempt, or apps that predate the release of such protocols, are likely to choose not to adopt them. A payment ledger may contain both types of accounts and a mix of conforming and non-conforming transactions. And naturally, some businesses will limit their transactions to be only on the "white-chain" of strictly conformant accounts.

Network Governance

In the past two decades, software engineering methods shifted away from long cycles of design, implementation and testing to ever-shorter cycles, up to the point where high-frequency of fine-grained releases are the industry standard in development of back-end servers for mass-market apps. Google[74], Facebook[75], Amazon[76] (and in particular AWS) are the most prominent examples of a near-unanimous agreement that frequent software releases enable higher quality of deployed software, fewer deployment problems, faster response to bugs in production and enable faster development.

For decentralized apps, backend servers are replaced by blockchain. However, governance models of current-generation blockchain platforms are incompatible with methodologies that rely on frequent, fine-grained software release cycles. Mass-market applications developed for or migrating into decentralized architecture must compromise on their development methods, forcing them to a position of disadvantage in their competition with their centralized opponents.

We're designing the Orbs platform to aim for continuous integration and continuous delivery on the application back-end. That applies both to backend endpoints, to smart contracts, and to the platform core. Naturally, decentralization creates barriers to quick acceptance of changes. Our approach combines fine-grained definitions of the deployment procedures of each component with economic incentives for quorum members on testing and deploying changes rapidly, and disincentives for lingering, as well as for carelessness. Fine-grained definition of the procedures refers both to the form of the procedure (for example, implementation optimizations to an endpoint may be tested and deployed by every quorum member separately; changes to protocols need to be in consensus before any implementation is deployed; and so on) and the participants. The participants involved in different governance procedures may vary based on the nature of the procedure. Some changes need to be

[74] http://eclipsecon.org/2013/sites/eclipsecon.org.2013/files/2013-03-24 CI at Google Scale.pdf
[75] https://code.facebook.com/posts/270314900139291/rapid-release-at-massive-scale/
[76] https://www.youtube.com/watch?v=dxk8b9rSKOo

accepted by Orbs federation members, some by a virtual chain's parties of interest; some changes, mostly in smart contracts relating to decentralized applications, may require a referendum of the affected users, etc. The economic incentives for involvement are applied by altering a node's reputation scores, which determine the pricing it could charge for its services.

Evergreen Nodes

An important aspect of high-frequency deployments is that each change is small enough to be quickly reviewed and tested, and perhaps more important, to gain full confidence in its robustness soon after it is used in production. As changes propagate quickly across the network from nodes that are risk-takers to nodes that are relatively risk-averse, outdated code can be deprecated early, reducing security risks and system complexity.

This behavior is not simple to implement in consensus-based decentralized systems where nodes are operated by independent organizations. Examining historic behavior of systems like Bitcoin is not encouraging, where several proposed changes like SegWit2X[77] have shown that consensus among participants is not always simple to achieve. The danger of a lack of consensus in this case is that the network is under risk of splitting, where some nodes reject a proposed protocol change while the others adopt it and fork outside.

Projects like Tezos[78] have been discussing the question of governance extensively and proposing several mechanisms to make its process of consensus more streamlined. We believe that these mechanisms are important, but they are not enough if the fundamental cohesion of the network is fragile. This is usually the case when network politics create different pressure groups whose interests are misaligned and guided by opposing incentives. Taking the question of fees as an example, miners are usually in favor of keeping fees high, naturally because fees provide their means of compensation. Users, on the other hand, are usually in favor of reducing fees as much as possible, as long as they retain the same quality of service. The most important design decision towards easing the consensus process is eliminating opposing interests by assembling the network from similar players that share the same general motivations and view of the world. In the case of the Orbs platform, as the target audience of the network is consumer applications, choosing a consensus algorithm that allows these same consumer applications to become efficient miners, goes a long way.

Further incentivization towards fast resolution of governance questions can also be applied through economic means. The node reputation system maintained by the *Helix Consensus Algorithm* allows for simple implementation of incentives, since reputation score controls the voting power of a node in the consensus process and the pricing it could charge for its share of the fees thereafter. We want to incentivize nodes to vote quickly on making a new protocol version official. This can be done by reducing the reputation of those who linger. We want to

[77] https://www.coindesk.com/2x-called-off-bitcoin-hard-fork-suspended-lack-consensus/
[78] https://www.tezos.com/static/papers/white_paper.pdf

incentivize nodes to upgrade to the latest version of the protocol which is under consensus. This can be done by reducing the reputation of those who fail to do so - slowly at first, when the new protocol versions are still backwards compatible, but more aggressively when the older versions approach end of life. The voting mechanisms for governance purposes in the Orbs platform are implemented as smart contracts.

Another important mechanism for reducing friction is providing an outlet for changes that are important only for a small part of the network. Consider a modification that is only required by one of the consumer brands participating in the ecosystem. If this change is not important to anyone else, or even worse, impacts performance in a negative way for those who don't need it, its contributor may find it difficult to bring it to consensus. This issue is resolved on the Orbs platform by allowing protocol modifications to apply to a specific virtual chain only. In this case, the consumer app requiring the modification would limit its effects to a voluntary configuration that is only enabled in the dedicated virtual chain it is renting from other nodes. This would eliminate most of the reasons for the other participants to oppose to the change.

Gradual Migrations

Once a core protocol change has been agreed upon, the question of its methodology of deployment still exists. Migrating the entire network at once, a process often seen in Bitcoin protocol modifications, holds substantial risk. What happens if unforeseen problems arise only after deployment to production? It is almost impossible to guarantee that all malfunctions will be identified in advance by testing and simulation.

Almost all changes can be deployed gradually, enabling the developers and chain administrators to gain confidence in its correctness based on actual performance rather than estimates (reviews, simulations, tests, etc). Using the method of Blue/Green Deployments[79], when changes are entering the production system, the entire network clones and directs traffic to both the unchanged (blue) and changed (green) code. Processed traffic can then be considered either live or in testing; essentially every transaction is processed on both environments, but the "live" transactions are committed to the permanent storage and the "test" transactions are only tested to verify that they are correct and maintain a set of predetermined key performance indicators (KPIs) as compared to the live environment. The deployment process starts with a period in which the entire "green" traffic is in testing, then split 90%-10%, 50%-50% and 0%-100%. Each such change can be approved by its developer or by consensus, after checking performance KPIs of both systems. Another benefit of this methodology is the ability to rollback and return to the blue system in case a major malfunction is discovered.

This gradual process of migration can also be used in order to migrate from a previous blockchain solution like Ethereum to Orbs in a risk free manner. Consider a secondary token,

[79] https://martinfowler.com/bliki/BlueGreenDeployment.html

TOK, that was originally launched on top of the Ethereum blockchain. Let's assume that the time comes and TOK is ready to migrate from Ethereum to Orbs. Naturally, performing a full migration at once holds risk. Instead, we propose a softer, risk-free migration. The TOK client SDK used by consumer apps that support the token will start *mirroring* all transactions performed by end users to Orbs in parallel to Ethereum. All writes will take place to both blockchains, creating a duplicate ledger for TOK on top of the Orbs platform. This ledger can constantly be audited and compared to the Ethereum ledger which is considered the source of truth. Initially, TOK clients perform business decisions based on the Ethereum reads, but this can be changed per user with a runtime feature toggle. To start the migration, this feature toggle is switched for 5% of users. If everything is alright, 50% and finally 100%. If a problem is discovered, the feature toggle can be switched back and all users will return to making their business decisions based on Ethereum. Since all writes are duplicated, there is no risk of losing the ability to rollback.

Upgradable Contracts

Once deployed, smart contracts are immutable in nature and cannot be updated. After all, what's the point of a contract if one of the parties can change it single-handedly after it has been signed? Although this behavior is a feature of smart contracts, immutability poses a lot of practical concerns. Developers are known to make mistakes and every piece of software always has another bug yet to be discovered. What happens if such a bug is discovered in an immutable smart contract?

This problem is similar in nature to the governance problem of updating protocol versions of the node codebase. We've resolved the questions of protocol updates using consensus. Once a majority of parties has agreed to upgrade the protocol to a new version, we can start enforcing the upgrade throughout the network. It makes sense to resolve the upgrades of smart contacts in a similar fashion. Smart contracts over the Orbs platform are encouraged to employ an upgrade strategy. This strategy is implemented as another smart contact and controls the process through which the contract can be upgraded. Contracts for secondary tokens can elect to upgrade based on a stake-weighted vote of all holders of the token.

Multi Chain Hybrids

Practical design concerns of real-world decentralized applications often require that we base the complete solution on multiple blockchain infrastructures running side by side. Consider the following challenge we've met while collaborating on Kin with Kik Interactive. The Kin token was launched on the Ethereum blockchain, which was at the time the de facto standard for raising funds as part of an ICO (Initial Coin Offering). Ethereum has a great ecosystem and secondary tokens based on its ERC20 standard are easily integrated into exchanges, third party wallets and hardware wallets such as Trezor. On the other hand, Ethereum also has severe limitations on transaction scale caused by high fees, network congestion and low

transaction throughput. The inability to scale Kin on top of Ethereum has led the project to consider a migration of the tokens to another blockchain infrastructure.

The difficulty with this plan was that the scalable blockchain solutions considered as an alternative to Ethereum did not have the same well integrated ecosystem. Performing a one-way migration to them would jeopardize the ability of some parties to integrate with the token. Instead, a better strategy would be to base the Kin token on a hybrid solution of two blockchains. The first blockchain would be Ethereum, used primarily for its extensive ecosystem of integrations. The second blockchain would be a scalable solution like the Orbs platform. These would be two different implementations of the *same* token. Users would be able to perform a 1:1 swap of tokens between the two implementations and the total amount of circulating tokens on both implementations would always equal the original number of tokens created during the Kin ICO.

Polyglot Cross-Chain Contracts

It seems that often a better strategy to a one-way migration between token implementations is the addition of a new implementation in parallel. This would provide us with a hybrid solution relying on multiple different blockchains at the same time. Strategy such as this introduces a significant technological challenge - how would we combine multiple different blockchains into one solution? How would they communicate?

Traditionally, smart contracts are limited in their ability to access external sources and can only rely on data existing on the blockchain itself. Such is the case with Ethereum contracts for example. This limitation is not surprising because smart contracts operate within a closed system of trust. External data, provided by an entity often labeled as an *oracle*, simply cannot be trusted like the data stored on chain.

The Orbs platform overcomes this inherent limitation of smart contracts by introducing *cross-chain contracts*. These smart contracts running on top of Orbs can read data from other blockchains in a secure and trusted way. Just like a smart contract can read variables from the secure on-chain Orbs storage, the smart contract can also read a variable from Ethereum. This expansion opens up an exciting new class of decentralized applications. Applications that can span multiple blockchains and choose the most appropriate one to hold every piece of its data. Another interesting ability made possible with this technology is to seamlessly import smart contracts developed originally for other blockchains directly into the Orbs platform. Consider a system of smart contracts originally designed for Ethereum. Normally, to migrate it to another blockchain infrastructure would require a complete rewrite. The Orbs platform is able to run these existing smart contracts almost as is.

Accordingly, the Orbs platform is designed to support the execution of multiple languages of smart contracts. The most popular smart contract language today for smart contracts is Ethereum Solidity[80]. As decentralized applications become mainstream, forcing engineers to transition to specialized languages for smart contract development creates undeniable friction. Orbs' design supports the development of smart contracts using common and widespread languages such as Python, Java and JavaScript, thus lowering the barrier of development for established brands even further. Because Orbs has lower redundancy of smart contract execution it is okay for the accounting to be loose compared to platforms like Ethereum. This means that it is not limited to languages that compile to custom bytecodes such as Solidity and EVM, which can bill code execution accurate to the opcode level.

[80] https://solidity.readthedocs.io/en/develop/

DESIGNING FOR CONSUMERS

Brands and Trust

In our discussion about consensus we have observed that we cannot practically design a decentralized system for consumers that is completely trustless. This is different from the decentralized ideal behind systems like Bitcoin, where end users are theoretically expected to verify that the client software they use conforms to the protocol by reviewing its source code and compiling it by themselves. Even if an end user downloads a precompiled binary, they are at least theoretically expected to validate that the signature of this binary matches the community consensus.

We can assume that the typical consumer is uneducated about cryptocurrencies and these extreme security measures. They are probably not looking to use decentralized consumer products for the idealistic benefits that decentralization and trustlessness provide. Decentralization is a design choice made by applications. Consumers that are using an application are normally unaware whether the application is decentralized or not.

We can therefore assume that the relationship between a consumer and the brand providing the product that the consumer is using involves *trust*. If such a brand would abuse the trust placed in it by its users, for example by leaking secrets like private keys that users must provide to client apps, its image and reputation will suffer the consequences in addition to any legal ramifications.

Mobile and Web Clients

The delivery mechanisms that can bring a consumer product today to the hands of millions of end users are agnostic as to whether this product is decentralized or not. Consumers are using mobile and web clients almost exclusively. This is different from the primary vehicle for systems like Bitcoin, where end users are theoretically expected to run a client that is also a full fledged node. Only after synchronizing the entire block history can the client truly trust that its perception of state is accurate.

Mobile and web clients do not have this privilege. Storing the entire block history requires significant storage space and the synchronization process takes a long time and ample bandwidth[81]. The resource constraints of mobile and web clients are too severe to make this practical. Therefore, when designing a blockchain system for consumers, we can only rely on what the industry refers to as *light clients*. Such clients connect to one or more *full nodes* which they can query to read blockchain data or send transactions through. Though they employ some heuristic proofs of the validity of the data they are getting, they do expect to have a certain degree of trust with the nodes they connect to. Orbs mechanisms such as *Ordering of Opaque Transactions* and *Network Owned Secrets* simplify the light client protocol and can significantly reduce the required level of trust.

Consumer Patterns of Network Access

Consumers also have a typical pattern of network access that may influence infrastructure design. Consumers are likely to access a single account from multiple devices concurrently. For example, on the go they may rely on their mobile phone to access an app, in the office rely on their laptop and at home use a tablet to access the same app.

Certain design decisions on the infrastructure layer can make the platform incompatible with such patterns. Consider the use of *nonce* in transactions on Ethereum. The purpose of the nonce is to assure uniqueness of a transaction and make sure the network does not process the same transaction twice. Ethereum clients need to put sequential numbering in the nonce field that increments on every transaction sent from an account. Ethereum relies on this numbering and will not process a transaction until its predecessor is confirmed. This mechanism is not suited for use by multiple devices concurrently, as the sequence numbering needs to be synchronized across devices. We are working to avoid this complication entirely by using a non-sequential mechanism to add uniqueness to transactions. This comes at a cost of limiting client transactions to an explicit time window of execution, which is an important feature by itself, giving end users a bound to how long a transaction can wait before being either processed or discarded.

Another common access pattern by consumers is the transmission of multiple requests in parallel instead of one after the other. Consider a peer-to-peer advertising platform where a group chat user shares an advertisement with the entire group. If the user needs to reward all group members for receiving the content, they would probably transmit multiple transactions in parallel. On Ethereum, confirmation time for each transaction can take 15 seconds[82]. How is the nonce calculated in this case? A standard implementation would be to increment the nonce on the client-side and send all transactions in parallel with sequential nonce numbers. Now, assume that one of the transactions failed for some reason. The platform would not process the subsequent transactions because a transaction having the first unused nonce

[81] https://ethereum.stackexchange.com/questions/143/what-are-the-ethereum-disk-space-needs
[82] https://etherscan.io/chart/blocktime

was not confirmed. This edge condition is once again circumvented on the Orbs platform by using a non-sequential mechanism.

Infrastructure Implications of Churn

Churn, short for *churn rate*, is a measure of the number of individuals or items moving out of a collective group over a specific period. When applied to consumer applications, churn refers to the proportion of end users who leave and stop using the product during a given time period. This may be to due to a variety of reasons like customer dissatisfaction, better alternatives from competition, disappearance amongst the noise and a generally shrinking attention span.

Churn is a fact of life in the consumer space. It is not uncommon for consumer products to lose the vast majority of their user base to churn. Figures like 5% of all registered users are active in a given month are within the norm and illustrate just how severe the phenomenon is.

When the decentralized app being developed involves a token that consumers use, churn is likely to dictate how distribution behaves over time. As the number of users lost to churn increases monotonously, so does the number of tokens left inaccessible inside the wallets that they possessed. Since the supply of such tokens is often limited, we may discover eventually that the majority of tokens end up locked forever out of circulation. This is problematic from another aspect as well, since the bootstrap of such token economies usually relies on stimulus in the form of subsidies - where large amounts of tokens are distributed to groups of consumers in order to spark a critical mass of use. We may discover that the bulk of the amount used for subsidies ends up in the hands of users who will never use it.

Careful design of the token protocol layer can go a long way towards dealing with churn gracefully. We'll show this with a simplified example. Suppose that the token smart contracts allow recycling of introductory subsidies back to the subsidizing body, in case the wallet was not used in the 12 months following the subsidy. This behavior, although crude, will eliminate the problem of churn quite effectively. Normally, giving someone the permission to reclaim funds out of a user's account creates a risk of embezzlement. But in this case the rules for recycling are clear and transparent, and enforced by a decentralized smart contract, so nobody is in control of the users' funds.

Consumer Apps and Open Source

Blockchain solutions, such as the Orbs platform, are normally open source and have a permissive intellectual property policy. Nevertheless, there are dozens of different open source licenses and the particular choice of license may affect how applicable it is for the consumer app use case.

Consumer brands can be very particular about their use of open source licenses. Any open source technology used by these brands must be careful to allow them to protect their assets,

particularly, reliance on the GPL[83] family of open source licenses may impact a brand's ability to incorporate this code in a closed source asset such as its mobile app. This family of licenses is popular in many blockchain projects like Ethereum. GPL licenses are *copyleft*, meaning that software that is derived from GPL-licensed code is required to adopt the same license. If a closed source app, for example, is using a GPL-licensed library, it is in risk of being required to open source itself - a legal liability that most brands will not accept.

The Orbs platform has a clear open source policy and relies within its ecosystem only on the MIT license[84]. This is one of the most permissive open source licenses available which explicitly makes no impact on consumer apps and their commercial assets. Unlike GPL, MIT-licensed software can be used in commercial closed source applications without any limitation.

The Problem of Private Keys

Cryptocurrencies have yet to penetrate the mainstream consumer market. For comparison, Facebook, one of the top consumer products today, has a reach of over 2 billion users worldwide[85]. The total number of users worldwide who have ever operated a cryptocurrency wallet is only estimated under 20 million[86]. This gap stems from a multitude of reasons, one of which is a severe technical barrier of entry.

Access to a cryptocurrency wallet is synonymous with knowing a single secret - the wallet's *private key*. This key is immutable and cannot be recovered if lost. In order to protect the wallet against attempts to guess the key with brute force, a high degree of entropy is commonly used; Bitcoin wallets, for example, use 256 bit keys.

Safeguarding such private keys from being stolen or lost is not an easy task. Consumers are not capable of maintaining the procedures required to manage and protect strong cryptographic keys as large organizations do, and do only moderately well with simple authentication methods, such as passwords, PIN codes, recovery questions, biometrics etc. Simple authentications by themselves are not safe for protecting valuable assets, such as identities or large funds, and are usually used as part of an authentication protocol that includes actions in the "real world", such as delays, fallback between different authentication challenges (for example, try PIN code, or fall back to security questions), multi-factor authentication, access notifications, timelocks, and more. Current technology only enables such protocols to be executed on centralized repositories (i.e. a bank's computer system can enforce a delay between failed PIN code entry attempts, but decentralized blockchains cannot).

[83] https://en.wikipedia.org/wiki/GNU_General_Public_License
[84] https://en.wikipedia.org/wiki/MIT_License
[85] https://newsroom.fb.com/company-info/
[86] http://jbs.cam.ac.uk/faculty-research/centres/alternative-finance/publications/global-cryptocurrency

Decentralized Secret Bearing

One of the unique novelties in the Orbs platform is that it enables the platform to store and use secrets, particularly secret cryptographic keys. This is done using novel protocols that make use of secret sharing and threshold cryptography, enabling operations like signing or decrypting a document with the decentrally-stored key. This functionality opens the door to a plethora of interesting business cases: signing a blockchain's state with a single canonical signature, signing transactions to be executed on other blockchains, notarizing the state of other (connected) blockchains, signing API calls for 3rd-party infrastructure, and many more.

An interesting use-case for decentralized secret bearing is saving cryptographically-strong private keys in a decentralized way and enforcing a secure authentication protocol which may include delays, multi-factor authentication and other methods popular with consumers - on a decentralized platform.

Full details about this mechanism are published in a separate technical white paper.

THE ORBS FEDERATION

The network built around the Orbs platform is envisioned as a community of ecosystem participants, primarily comprised of organizations that make the target audience of Orbs - decentralized consumer businesses and established consumer brands transitioning into blockchain. For example, organizations like PumaPay, Zinc or Kin and companies like ironSource and Kik Interactive. This collection of organizations and companies is aligned as independent but equal members of a *federation*.

The primary roles of members in the Orbs federation include:

- Operating *consensus nodes* in the network and actively participating in the consensus process.
- Contributing to the open source development of Orbs and evolving the platform over time as their own requirements come to light.
- Providing decentralized governance for the platform such as reaching consensus over core updates to the protocol.

Pre-launch Design Partners

The Orbs platform is initially designed in a requirements-driven approach through close work with a set of *design partners*. The production requirements of these design partners shape the iterative process by which the system is designed. The initial design partners also act as the founding members of the Orbs federation and operate the first set of consensus nodes. This first set also acts as the first customers of the platform and the developers of the first decentralized applications that run on top of it.

The collaboration process with design partners is designed to be completely open and transparent. All insights learned in the process are published on public channels for the benefit of the Orbs community. The Orbs reference implementation codebase is an open source project maintained publicly on GitHub[87]. The code is provided with the permissive MIT license

[87] https://github.com/orbs-network

for free public use. New members of the Orbs ecosystem are encouraged to contribute to the codebase and become active members of the Orbs federation by running nodes.

The role of the Orbs project is kicking off this new ecosystem. The initial design is proposed to the Orbs community by publishing this position paper and a set of technical white papers, and by providing the first reference open source implementation for the community to build on. When the protocol is well defined, the platform can launch. This initial design also provides the scaffold for decentralized governance which will steer the platform once the federation is formed. True to its decentralized nature, the platform will not be governed by a single entity like the Orbs project and its founding team. Beyond a leadership role in the early days of pre-launch, the Orbs project will not be involved in ongoing governance but will continue research and development in the field, like other participants, and propose protocol modifications for the federation to evaluate.

Governance

The model of a federation has been established in the industry and demonstrated in principle in projects like Stellar and Ripple. Contrary to a *consortium*, where a centralized governing body adds each chosen member to the collective, a *federation* has no point of centralized governance. Organizations and companies can join the collective by approaching some of the existing members and asking them for sponsorship. Once a new member is designated by other members, they gain their initial status.

One of the primary roles of a federation member is operating a consensus node. A consensus node is elected to participate in the consensus process of a virtual chain at random times; one consensus node may find itself participating in the consensus of various virtual chains at any minute. The measure of a node's influence over consensus and its ability to validate transactions is governed by the protocol. The protocol incentivizes several behaviors for nodes, such as maintaining a high SLA, which manifests during the consensus process as the node's *reputation* and is defined in a decentralized manner by the consensus of nodes in the network. The reputation also takes into account each node's participation in evaluating and agreeing on protocol upgrades. This enables the Orbs platform to keep on evolving and provide its users with state-of-the-art blockchain technologies.

NOTE: THIS POSITION PAPER PROVIDES AN INITIAL SUMMARY OF CERTAIN BUSINESS AND TECHNOLOGY ESSENTIALS UNDERLYING THE ORBS PROJECT AND PLATFORM. THIS DOCUMENT IS EXPECTED TO EVOLVE OVER TIME, AS THE PROJECT AND THE DEVELOPMENT OF THE ORBS PLATFORM PROCEEDS, AND THE ORBS PROJECT MAY POST MODIFICATIONS, REVISIONS AND/OR UPDATED DRAFTS FROM TIME TO TIME.

Definitions

This position paper uses certain defined terms, as follows -

- “Orbs ecosystem” - a general term referring to combination of the Orbs platform providing core services, together with third party infrastructure developers providing services over the infrastructure marketplace; as described in the section labeled “The Orbs Ecosystem”.
- “Orbs federation” - a collection of ecosystem participants providing decentralized governance to the platform and operating consensus nodes; as described in the section labeled “The Orbs Federation”.
- “Orbs platform” / “Orbs” - a decentralized network governed by the Orbs federation providing infrastructure as a service to applications; as described in the section labeled “The Orbs Platform”.
- “Orbs project” / “we” - Orbs Ltd. and its shareholders, officers, employees and consultants, constituting together the founding team writing this paper and publishing the initial design of the protocol, as well as the scaffold for decentralized governance which will steer the platform once the federation is formed.
- “ORBS token” / “Orbs token” - the token fueling the platform which is used primarily by application developers to pay for infrastructure fees; as described in the section labeled “The ORBS Token”.

Legal Disclaimer

This position paper is for information purposes only and may be subject to change. We cannot guarantee the accuracy of the statements made or conclusions reached in this position paper and we expressly disclaim all representations and warranties (whether express or implied by statute or otherwise) whatsoever, including but not limited to:

- any representations or warranties relating to merchantability, fitness for a particular purpose, suitability, title or non-infringement;
- that the contents of this document are accurate and free from any errors; and
- that such contents do not infringe any third party rights.

We shall have no liability for losses or damages (whether direct, indirect, consequential or any other kind of loss or damage) arising out of the use, reference to or reliance on the contents of this position paper, even if advised of the possibility of damages arising.

This position paper may contain references to third party data and industry publications. As far as we are aware, the information reproduced in this position paper is accurate and that the estimates and assumptions contained herein are reasonable. However, we offer no assurances as to the accuracy or completeness of this data. Although information and data reproduced in this position paper are believed to have been obtained from reliable sources, we have not independently verified any of the information or data from third party sources referred to in this position paper or ascertained the underlying assumptions relied upon by such sources.

The information contained in this position paper is intended for informative purposes only and shall not form the basis of, or be relied upon in connection with, any offer or commitment whatsoever in any jurisdiction, including (without limitation) the United States or the State of Israel. The information contained herein shall not constitute or form part of, and should not be construed as, any offer for sale or subscription of, or solicitation of any offer to buy or subscribe for tokens issued by Orbs Ltd. or any products or services offered by Orbs Ltd. Any offer to acquire ORBS tokens will be made, and any customer should make his or its purchase decision, solely on the basis of the information that will be contained in the applicable agreement made between Orbs Ltd. and eligible token purchasers.

No promises of future performance or value are or will be made with respect to the Orbs platform and/or ORBS tokens, including no promise of inherent value, no promise of any payments, and no guarantee that the Orbs platform and/or ORBS tokens will hold any particular value. Unless prospective participants fully understand and accept the nature of the Orbs platform and the potential risks associated with the use of the Orbs platform and the acquisition, storage and transfer of ORBS tokens, they should not use the Orbs platform or purchase, acquire or otherwise obtain or use any ORBS tokens.

This position paper does not constitute a prospectus or disclosure document and is not an offer to sell, nor the solicitation of any offer to buy any investment or financial instrument in

any jurisdiction. ORBS tokens should not be acquired for speculative or investment purposes with the expectation of making an investment return.

No regulatory authority has examined or approved any of the information set out in this position paper. No such action has or will be taken under the laws, regulatory requirements or rules of any jurisdiction. The publication, distribution or dissemination of this position paper does not imply that applicable laws or regulatory requirements have been complied with.

The Orbs platform and/or ORBS tokens could be impacted by regulatory action, including potential restrictions on the ownership, use, or possession of such tokens. Regulators or other competent authorities may demand that we revise the mechanics of the token allocation and/or the functionality of Orbs platform in order to comply with regulatory requirements or other governmental or business obligations. Nevertheless, we believe we are taking commercially reasonable steps to ensure that the operation of the Orbs platform and the token allocation mechanics do not violate applicable laws and regulations.

This position paper contains forward-looking statements or information (collectively "forward-looking statements") that relate to our current expectations of future events. In some cases, these forward-looking statements can be identified by words or phrases such as "may", "will", "expect", "anticipate", "aim", "estimate", "intend", "plan", "seek", "believe", "potential", "continue", "is/are likely to" or the negative of these terms, or other similar expressions intended to identify forward-looking statements. We have based these forward-looking statements on current projections about future events and financial trends that we believe are relevant to the operation of the Orbs platform.

In addition to statements relating to the matters set out here, this position paper contains forward-looking statements related to the Orbs platform's proposed operating model. The model speaks to our objectives only, and is not a forecast, projection or prediction of future results of operations. The platform operation and development is reliant on the formation of the Orbs federation. We are unable to guarantee that sufficient members will join the federation to support and realize the intended design in its entirety.

Forward-looking statements are based on certain assumptions and analysis made by Orbs project team in light of its experience and perception of historical trends, current conditions and expected future developments and other factors we believes are appropriate, and are subject to risks and uncertainties. Although the forward-looking statements contained in this position paper are based upon what we believe are reasonable assumptions, there are risks, uncertainties, assumptions, and other factors which could cause our actual results, performances, achievements and/or experiences to differ materially from the expectations expressed, implied, or perceived in forward-looking statements. Given such risks, you should not place undue reliance on these forward-looking statements.