A Fundamental Rethinking of RTB Advertising

Version 7.3[1]

Abstract

The purpose of this paper is to present a new approach to securing the real-time bidding (RTB) advertising environment. After a theoretical introduction to the current digital advertising ecosystem, we derive a complete product from first principles and describe it in full technical and economic details. The protocol is released as an open-source project and all documentation can be found on AdHash's website[2], while the code can be reviewed in AdHash's GitHub repository[3].

The work described in this paper achieves a leap in ad serving transparency, accountability, and efficiency through a combination of six innovations:

- Unique, immutable, and universally verifiable hash IDs issued for every publisher, advertiser, and creative which serve to enhance accountability, prevent a multitude of ad fraud variations, and eliminate double-bidding inefficiencies.
- Direct ad serving from the advertiser's server to the user's app or browser without requiring any intermediaries for hosting, auctioning, tracking, or analysing ad performance.
- Guaranteed tamper-proof account of ad serving history stored on a distributed ledger built on top of the Ethereum blockchain.
- A protocol allowing off-chain transactions to work in sync with the blockchain, enhancing the security of transactions while minimising their cost.
- A voting system designed to ensure the integrity of the network by holding all participants accountable and rewarding them for their work.
- Microtransactions enabling same-day payments.

We hope that the work described in this paper will inspire the formation of a community of like-minded developers and enthusiasts who strive to keep the internet open and free for all while eliminating tracking and preserving user privacy.

---

[1] Author: Martin Stoev, stoev@adhash.com, latest revision: August 18, 2022.

[2] "AdHash - A Fundamental Rethinking of RTB Advertising", https://adhash.com.

[3] "AdHash", GitHub, https://github.com/adhashprotocol.

Table of Contents

I Introduction

When building AdHash, we began by reasoning from first principles about the problems of the digital advertising industry and infrastructure and went through the process of developing a more efficient model. We hope that by describing the process we will help the reader derive the fundamental technology underlying our solution along with us. It is an arduous task, but by the end of it hopefully we would have enhanced the understanding of a highly convoluted industry that powers most of the open internet as we know it.

Our team has been developing products in this industry since 2014. Together we built AdTrader[4] with a mission to make advertising simple, transparent, and actionable. AdTrader grew steadily quarter after quarter, reaching hundreds of clients and being cash-flow positive. Despite our success, we were painfully aware of the multiple constrains that working in the existing ad tech ecosystem imposed on our drive to build the best possible products for both publishers and advertisers.
At our launch, in 2014, there were already an incredible number of middlemen in our industry, as shown below (we will save the reader some counting, there are approximately 1,000 logos in the figure):

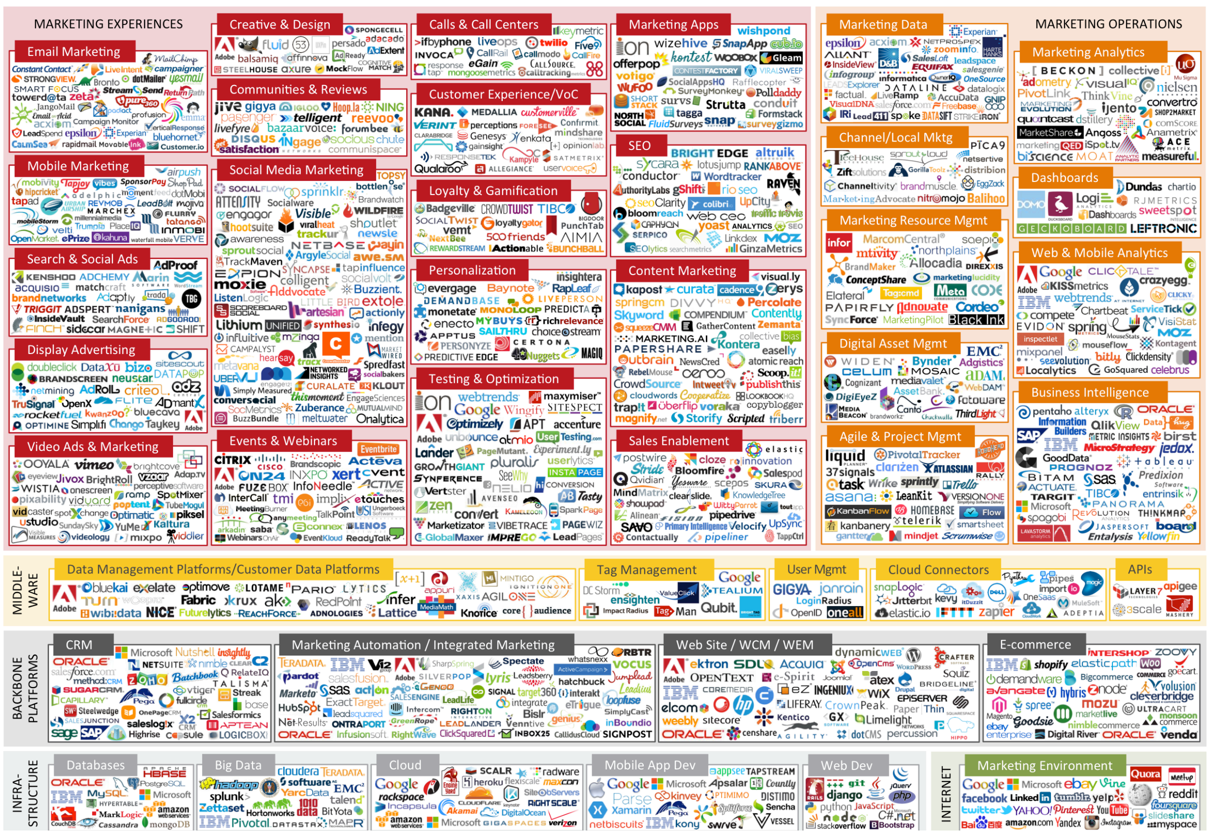


Fig. 1: The complexity of the ad tech ecosystem in 2014. Source: Scott Brinker, http://chiefmartec.com/2014/01/marketing-technology-landscape-supergraphic-2014

4 *"AdTrader", https://www.adtrader.com.*

Marketers kept relying on middlemen to host, analyse, measure, verify, track, enhance, support, optimise, customise, and A/B test their ads. This tremendous complexity led to severe inefficiencies. The process is best exemplified by the expansion of ad tech intermediaries over the past six years:

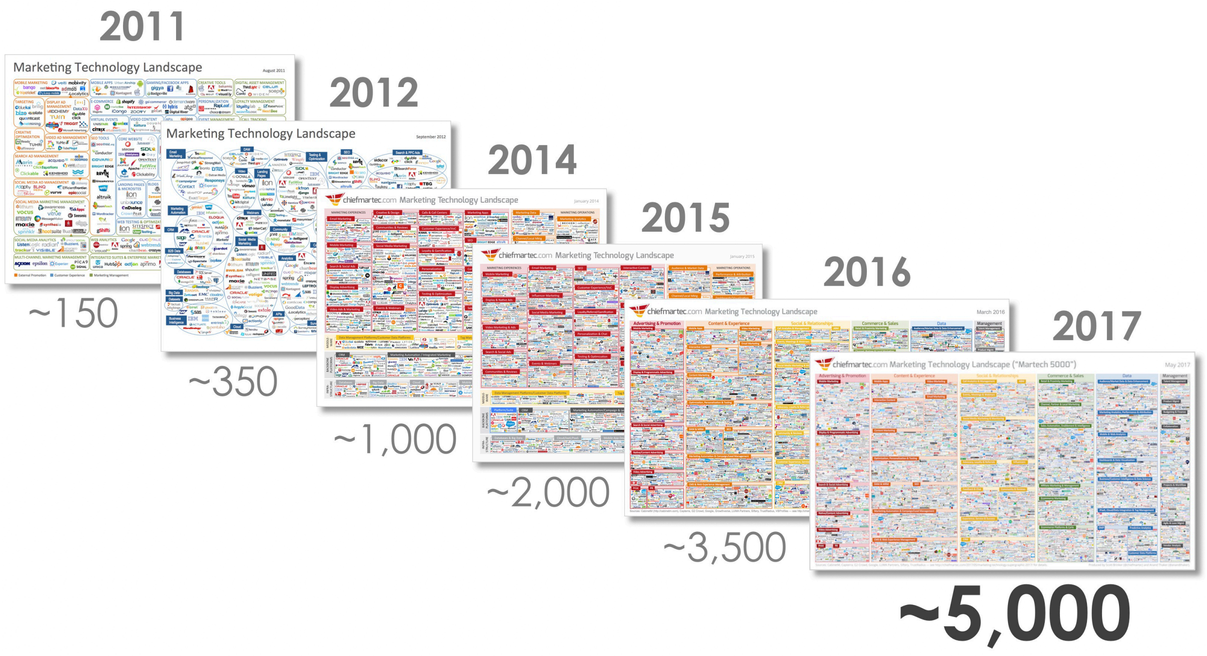


Fig. 2: The explosion of middlemen in advertising over the past six years. *Source: Scott Brinker, http://chiefmartec.com/2017/05/marketing-techniology-landscape-supergraphic-2017/.*

We were certain that things could be done in a much cleaner way. After a careful evaluation of the industry's problems, it was evident that to address them effectively, we needed to focus on the core technology and rewrite the way ads are served online. We spent over four years developing, testing, and fine-tuning a vastly improved version of the RTB protocol. One that would truly reflect our vision for the future of the advertising industry.
And with that, AdHash was born.

Before getting into the details of the AdHash Protocol, let us define the most severe problems of our industry and how they evolved historically. The following nine pages will feel painfully familiar to anyone versed in the advertising industry and perhaps confusing and outrageous to anyone new to it.

II The Failure of the Ad Tech Industry

The unfulfilled promise of real-time bidding advertising

RTB advertising began its life around 2005[5] and has been widely used for nearly a decade. Programmatic advertising's raison d'être was straight forward: rather than relying on outdated paper-based insertion orders, advertisers and publishers would reach immediate deals and would be able to serve relevant ads to specific users. They would do so in milliseconds and at an unprecedented scale. There would be no more buying of unpredictable inventory in bulk, wasted hours in human negotiations, unreliable measurements from the publishers' servers, and incalculable return on investment for the advertisers. Everything would be data-driven, transparent, and subjected to ever more efficient optimisations. All campaigns would have a much larger scale with far greater exposure and interoperability between buyers and sellers.

This began with the inception of simple ad networks, representing a group of publishers and another group of advertisers, and serving as a single intermediary between the two groups:

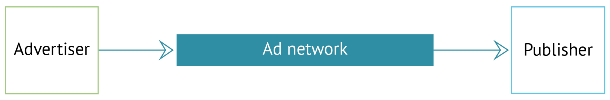


Fig. 3: A simple ad network structure.

This approach proved to lack scale which led to a fragmented marketplace in need of a unified solution. One of the first steps towards achieving this was the evolution of ad exchanges, aggregating the inventory from many ad networks:

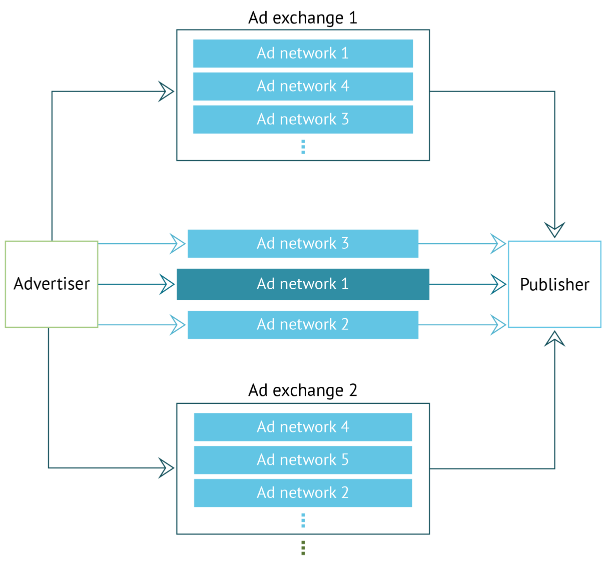


Fig. 4: Ad networks combined into ad exchanges.

It is the added complexity resulting from the introduction of the ad exchanges that opened the floodgates for the thousands of ad tech companies that have evolved in the last decade. Myriads of firms mushroomed and rushed to analyse consumer data, buy and sell ads, or

[5] *"Method and System for Placing Electronic Advertisements", Brian O'Kelley, United States Patent No.: US 2006/0122879 A1, 2006.*

repackage inventory and sell it to other middlemen. Eventually, an ad impression sold via RTB could change hands dozens of times before finally being bought by the advertiser. In the end, the supporting middleware that was originally supposed to glue the disparate parts of the advertising system together, was instead pushing the advertisers and publishers further apart. A growing and tremendously diversified industry would be expected to have solved by trial and error most, if not all, inefficiencies in the marketplace.

Unfortunately, as anyone acquainted with this field knows, this is not at all how things panned out. The first major problem that hurt the ecosystem was ad fraud. It did not get invented because of RTB and ad exchanges, it existed long before that, but with RTB it could truly proliferate. The more sophisticated the fraud algorithms got, the better return on investment they could generate for their creators as everything became automated.

Overall, RTB is an exceptional technological innovation unfortunately marred by poor execution. It has largely automated and accelerated processes but has also introduced tremendous complexities which have opened loopholes that fraudsters are happy to exploit.

The proliferation of ad fraud

The path of advertiser dollars to publishers is circuitous not only because of the multiple layers of middlemen but also due to various sophisticated schemes of fraud. Ranging from complex technical tricks to purely disruptive user experience, ad fraud strains the advertising ecosystem and puts pressure on all participants to safeguard the advertiser trust, the publisher revenue, and the user experience. We have compiled a list of the most harmful types of ad fraud:

- **Domain spoofing**[6] is a sophisticated technique for masking the real domains where ads appear. Fraudsters use it to cover the real sources of traffic and to falsely attribute them to reputable premium inventory. This technique is used by websites, ad networks and exchanges that take advantage of the intermediary architecture of the advertising industry and the widely accepted practice to resell or manage third-party traffic.
- **The Methbot**[7] **and Hyphbot**[8] **operations** are essentially domain-spoofing brought to an entirely new level. While technologically simple to design, those two attacks were unprecedented because of the financial scale at which they were performed. Rather than relying on malware to infect machines and execute their ad fraud, the organisations behind those attacks invested in entire data centres and in the acquisition of large batches of IPs in premium geolocations to perform custom-built domain spoofing on their own machines so that they could defraud ad networks.
- **Reselling traffic and arbitraging**[9] involve publishers driving traffic to their properties by using ad networks, often with native formats, that require less stringent checks[10].

6 *"Myspace Looked Like it Was Back. Actually, it Was a Pawn in an Ad Fraud Scheme", BuzzFeed, https://www.buzzfeed.com/craigsilverman/remember-tom?utm_term=.jwV4weJayL#.lpr7QYqrop, 2017.*

7 *"The Methbot Operation", Whiteops, https://cdn2.hubspot.net/hubfs/3400937/Resources/WO_Methbot_Operation_WP.pdf, 2016.*

8 *"How Adform Discovered HyphBot", Adform Fraud Protection, https://site.adform.com/media/85132/hyphbot_whitepaper_.pdf, 2017.*

9 *"'We Go Straight to the Publisher': Buyers Beware of SSPs Arbitraging Inventory", Digiday, https://digiday.com/media/ssp-arbitrage/, 2017.*

10 *"The Publisher of Newsweek and the International Business Times has Been Buying Traffic and Engaging in Ad Fraud", Craig Silverman, BuzzFeed, https://www.buzzfeed.com/craigsilverman/the-publisher-of-newsweek-and-the-international-business?utm_term=.uiJmErq3LA#.ymVrLokB4v, 2018.*

Publishers hope that buying the traffic will cost them less than the revenues they generate from advertising to that same traffic (hence the name “arbitraging”). Users then end up on pages with little content and multiple ads that provide very unpleasant user experience infested with malware and inappropriate ads.

- **Invisible and hidden ads** are used by malicious publishers to report impressions for ads that are actually invisible to humans. This is done in 1x1 pixel iframes, outside of the viewport area, or by stacking several ads in an iframe loaded in a single ad slot.
- **Device ID reset fraud**[11] is powered by bot farms. Device ID resets are responsible for artificial inflation of app installs that skew the cost per install and trick advertisers into wasting their budgets by paying for acquiring the same users over and over again.
- **Hijacked ads and devices** generate revenue for an attacker instead of a legitimate publisher by injecting the publisher’s content with the hijacker’s ads. It can be achieved by compromising the user’s device or the publisher’s servers.
- **Geolocation masking** affects campaign geotargeting by altering the geolocation of the inventory offered for sale. Like domain spoofing, geolocation masking exploits advertisers who are willing to pay extra for specific audiences. It defrauds them by selling them traffic from completely irrelevant locations.
- **Click farms**[12] leverage simulated or real clicks to squander advertising dollars at scale.
- **Malvertising**[13] - the advertising industry is full of platforms willing to push the boundaries of what is acceptable. There is no better illustration of this sad reality than the multitude of malware attacks carried through ads by platforms that allow third-party scripts. They lead to malicious results such as identity theft and ransomware[14] and are sometimes performed by very elaborate technological and legal structures.
- **Accidental and unintended clicks**[15] result from strategically concealed ad placements in popups and layovers. Such fake metrics plague advertising campaigns and challenge the sources of inventory to carefully monitor user activity and attribution. Usually, such activity is limited to certain bad players whose properties offer sub-standard user experience coupled with shady means of acquiring traffic.
- **Ads with unknown sources**[16] put pressure on publishers and buying platforms to better control who uses their services. Reputable advertisers are increasingly concerned that they inadvertently become party to fraudsters. Bad advertisers also disrupt the user trust in publishers and make them less reputable sources of information. This is particularly true in the context of promoted posts and native advertising, where the adverts visually appear like trusted content on platforms such as Facebook and Google.

Other well-known examples include mobile app install hijacking (claiming the reward for another publisher’s work), ad stacking (stacking multiple ads so that their call-to-action

11 *"DeviceID Reset Fraud: The New Threat to Mobile App Marketers [Data Study]", Appsflyer, https://www.appsflyer.com/resources/deviceid-reset-fraud-data-study/, 2017.*

12 *"Mobile Fraud: Marketers’ Massive Hidden Threat", Forrester Consulting, https://hub.appsflyer.com/hubfs/Mobile_Fraud_Marketers_Massive_Hidden_Threat_Forrester_Study.pdf, 2017.*

13 *"Uncovering 2017’s Largest Malvertising Operation", Confiant, https://blog.confiant.com/uncovering-2017s-largest-malvertising-operation-b84cd38d6b85, 2018.*

14 *“Ransomware”, WikiPedia, https://en.wikipedia.org/wiki/Ransomware.*

15 *"Up to 13% of Mobile Video Ad Clicks are Accidental", Pixalate, http://blog.pixalate.com/accidental-clicks-mobile-ads-data, 2017.*

16 *"Google Uncovers Russian-Bought Ads on YouTube, Gmail and Other Platforms", The Washington Post, https://www.washingtonpost.com/news/the-switch/wp/2017/10/09/google-uncovers-russian-bought-ads-on-youtube-gmail-and-other-platforms/?utm_term=.bb5071e2459a, 2017.*

buttons would be triggered simultaneously), retransmission fraud (reclaiming the reward for the same click multiple times), and others. All these various forms of ad fraud indicate a high level of sophistication of fraudsters who stay well ahead of the rest of the ecosystem. In addition, the current state of the advertising industry populated by intermediaries further favours ad fraud by making it difficult for everyone to independently check their end of the transaction. Overall, ad fraud is estimated to cost advertisers and publishers approximately 19 billion USD in 2018[17] and is projected to accelerate fast.

The ever-increasing complexity

The emergence of all these new attack vectors for ad fraud was not enough to revert the direction that the market had already taken - RTB was so vastly superior to manually making deals, that despite the negative impact of ad fraud on the emerging marketplaces, RTB kept growing at an astounding pace. As marketers geared up to combat ad fraud by throwing as much data analysis at it as possible, a whole set of problems evolved. Most advertisers started ignoring basic marketing principles and digital advertising became increasingly flooded with dull, repetitive, or even ugly and offensive creatives. The quantity of impressions or clicks was supposed to compensate for the quality of their messages. Inventing myriads of incomputable attribution metrics was intended to make up for the fraudulent traffic being bought in the first place. And working with as many trading platforms and ad exchanges as possible was supposed to overcome the inefficiencies that arose from those very same platforms.

The rise of the ad blockers

Users in turn fought back by beginning to adopt ad blockers at an increasing rate:

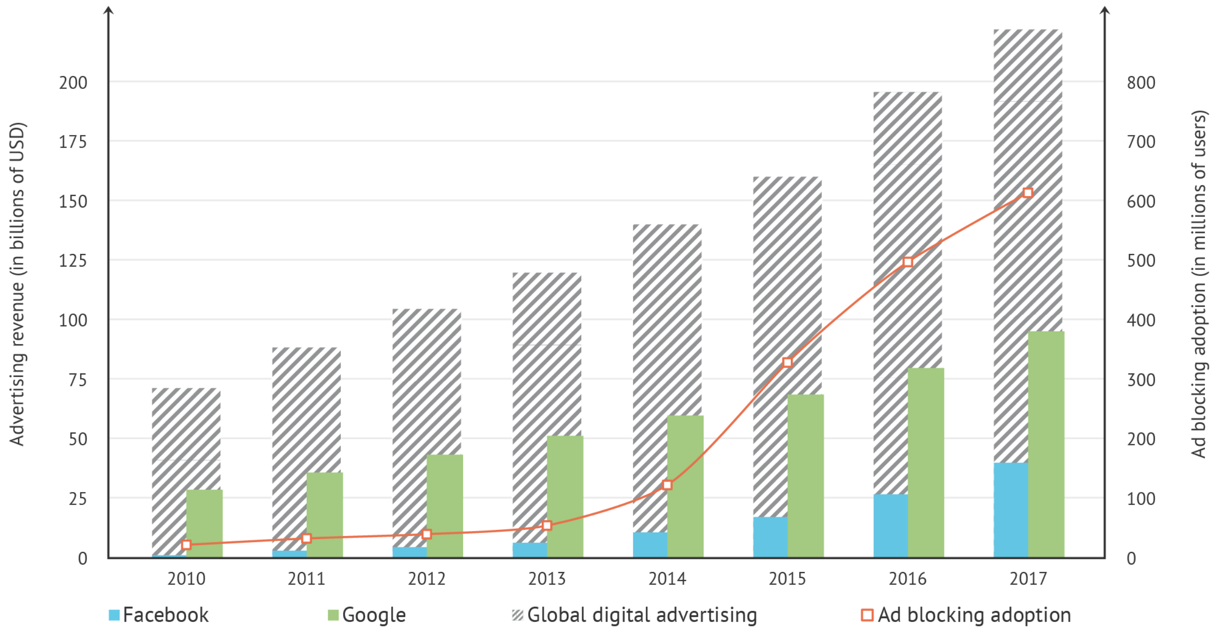


Fig. 5: The growing worldwide digital advertising spend corresponding to a rapidly increasing ad blocker adoption. Sources: https://www.recode.net/2017/12/4/16733460/2017-digital-ad-spend-advertising-beat-tv, https://pagefair.com/blog/2017/adblockreport/, https://www.emarketer.com.

[17] *"The Cost of Ad Fraud Continues to Grow", Mobile Marketing Watch, https://mobilemarketingwatch.com/cost-ad-fraud-continues-grow-73619/, 2018.*

The opposing sides of the ad blocking debate have passionate arguments whether users should ultimately be able to render content on their machines however they like or shall be responsible for contributing in exchange for the free content they are receiving from publishers.
However, it is indisputably true that at the moment, despite some formidable attempts, digital advertising remains the only considerable source of income for most online publishers. Without it the vast majority of all the great content that we take for granted and that the global internet population has free access to would most likely not exist. We, as users, need to be able to pay the creators of the content we love so that they can create more of it. It is therefore selfish and myopic to penalise publishers for the toxic environment that the actions of uninformed advertisers and greedy ad tech intermediaries have created. We need a solution that gives marketers, publishers, and users equal control; that enables self-regulation and rewards contribution; and that fosters high quality content and advertising. We need to preserve the open internet.

What real-time bidding evolved into

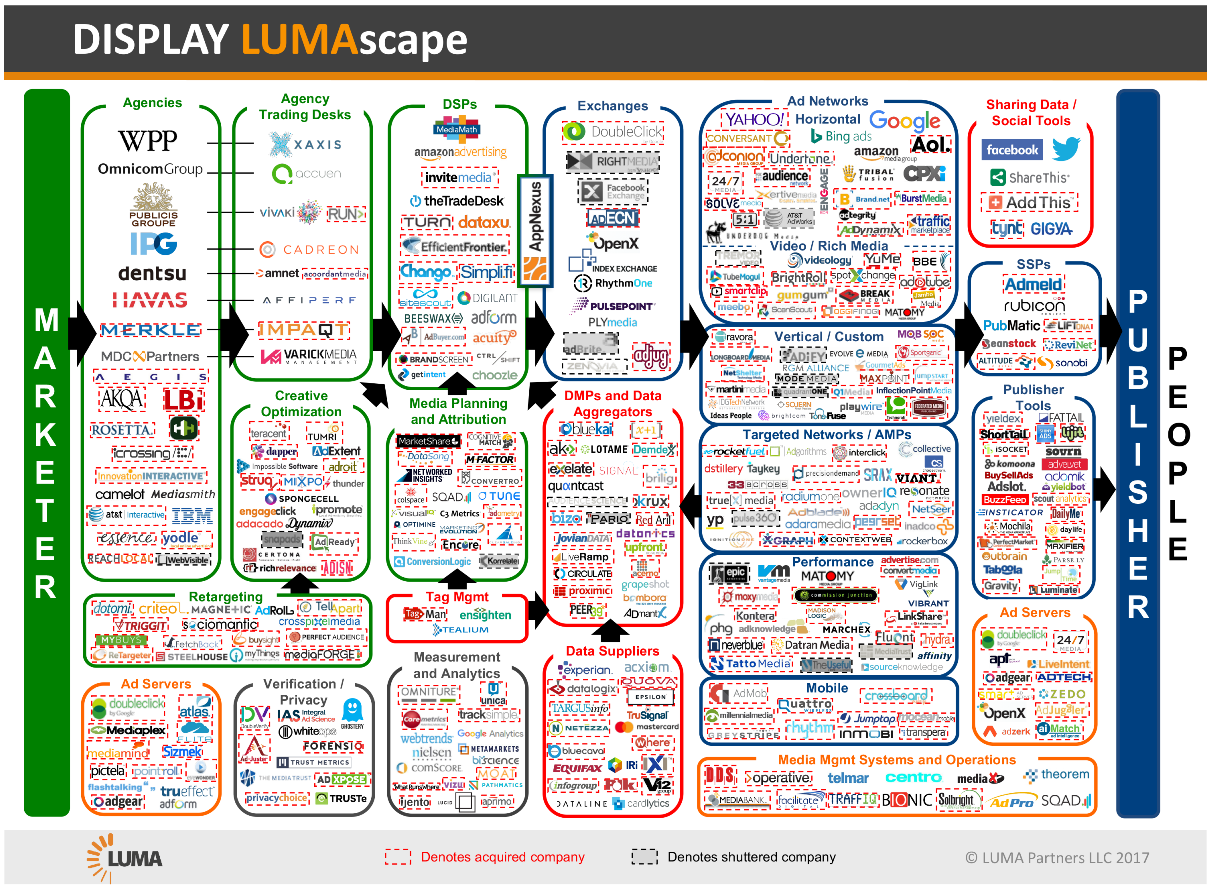


Fig. 6: The digital display advertising landscape in 2017. Source: LUMA Partners, https://www.lumapartners.com/.

We started this chapter by discussing the promises of RTB, then we explored its unintended byproducts: ad fraud, complexity, and ad blockers. It is now time to conclude with a realistic snapshot of what RTB actually became over the past fifteen years. Having established the problems in the previous sections, we would now like to dig deeper into the root cause that triggered them in the first place.

Shown on the previous page is a widely used representation of the advertising landscape. It is called a LUMA Scape. In it, there are a total of 23 different categories of ad tech companies trying to deliver a display ad from marketer to publisher. All of them have their own pricing models, relying either on commissions or fixed fees[18]. Combined, those fees can often gobble up as much as 75% of the entire advertising budget[19]. And to make matters worse, the majority of marketers and publishers do not use just one representative of each category. Because of the inherent mistrust between all parties in the digital advertising industry, most participants usually employ at least a few of the providers of services they need to make sure that they are not being taken advantage of. And all of them can thank ad fraud, greedy intermediaries, and the failure of the current digital advertising duopoly for this complicated maze.

Unfortunately, every middleman added to the chain between the advertiser's ad server and the end-user's screen introduces more latency, more tracking code, more opportunities for fraud, and of course - a larger portion of the advertising budget being absorbed by third parties. Their contribution is solely to justify their own existence by second-guessing their peers. The end result of all this are ads that become unreasonably expensive and return vast amounts of data whose sources contradict one another.

Some of the largest advertisers in the world have expressed their frustration with this situation and are looking for alternative solutions. Marc Pritchard, Procter and Gamble's Chief Brand Officer, was 2017's most widely-cited voice of criticism of the ad tech ecosystem, saying[20]:

> *"Craft or crap? That's really the big question. And technology enables both. And all too often, the outcome has been crappy advertising accompanied by even crappier viewing experiences."*

Meanwhile, publishers' margins get constantly squeezed until it becomes unprofitable for them to operate. The loudest voice on their side last year was Hamish Nicklin, Guardian's Chief Revenue Officer, saying[21]:

> *"There are so many different players taking a little cut here, a little cut there - and sometimes a very big cut. A lot of the money that [advertisers] think they are giving to premium publishers is not actually getting to us. In some instances, only 30% is making it back to the publisher."*

The users of those publishers get bombarded by tracking scripts, dozens of slow-loading ads, and their digital behaviour is under constant unregulated surveillance. Clearly, none of the parties in today's digital advertising environment are satisfied.

While the LUMA Scape shown on Fig. 6 is the industry standard for visualising the complexity of digital advertising and the roles of those intermediaries, we believe that it does

---

[18] *"Deconstructing the Anatomy of a Programmatic CPM", IAB, https://www.iab.com/wp-content/uploads/2016/03/Programmatic-Value-Layers-March-2016-FINALv2.pdf, 2016.*

[19] *"Confessions of a Media Agency Veteran: Hidden Programmatic Fees 'Must Stop'", Digiday, https://digiday.com/uk/confessions-media-agency-veteran-programmatic-margins-downward-spiral/, 2017.*

[20] *"Will February 2017 Go Down as the Month that Destroyed Adtech?", The Drum, http://www.thedrum.com/opinion/2017/02/27/will-february-2017-go-down-the-month-destroyed-adtech, 2017.*

[21] *"Where Did the Money Go? Guardian Buys its Own Ad Inventory", Mediatel, https://mediatel.co.uk/newsline/2016/10/04/where-did-the-money-go-guardian-buys-its-own-ad-inventory/, 2016.*

not fully explain all connections between the various parties. For this purpose we created our own interpretation, depicted on the next page. It is far more detailed:

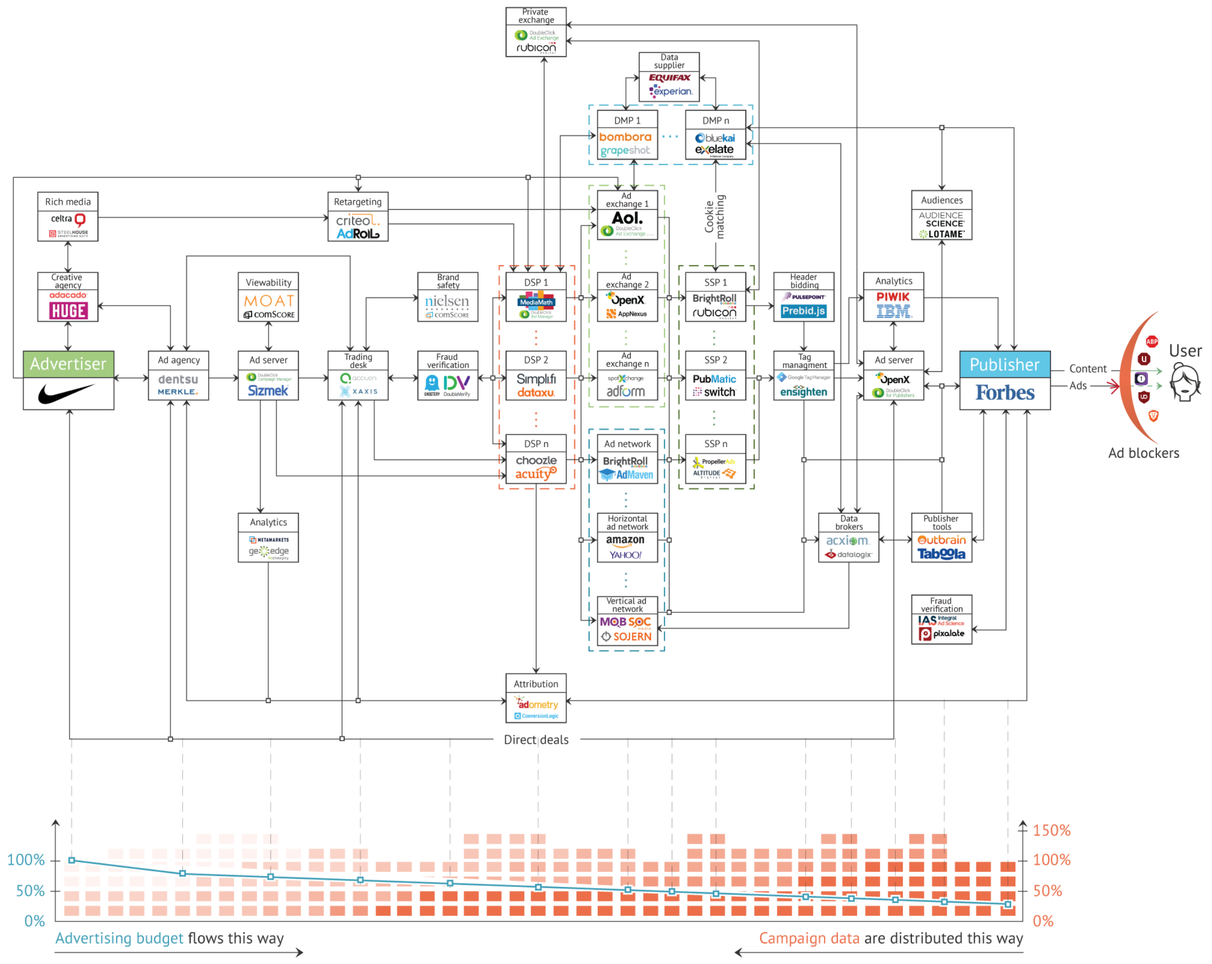


Fig. 7: The interconnected map of middlemen on an ad's convoluted journey from advertiser to user. The blue line in the lower graph explains the significant loss of revenue from the advertiser to the publisher. The orange bars provide a clue as to how the data intended for the advertiser get masked, inflated, distorted, or lost when passing through the maze of middlemen.

In conclusion, we would like to emphasise one specific issue of the current advertising ecosystem that is especially severe and will be mentioned regularly in the coming chapters of this paper:

Double-bidding

Double-bidding is not a problem isolated to RTB advertising. It is a common threat to the fairness of most auction designs. RTB's standout feature in this context, unfortunately, is how utterly helpless the current infrastructure is in dealing with the issue. While individual platforms might occasionally be able to prevent advertisers from bidding against themselves (as has been explained by the inventor of RTB and CEO of AppNexus Brian O'Kelley[22]), the actual use case for a typical advertiser is to bid through several platforms simultaneously. The complete lack of universal cross-platform ad IDs means that an intermediary platform would most likely not be able to label two bids arriving through different channels but

22 *"DSP User: In What Situations Am I Bidding Against Myself?", Response by Brian O'Kelley, https://www.quora.com/DSP-User-In-what-situations-am-I-bidding-against-myself, 2013.*

submitted by the same advertiser as identical. Thus, advertisers keep bidding against themselves and end up paying more than should be necessary in order to win the auction:

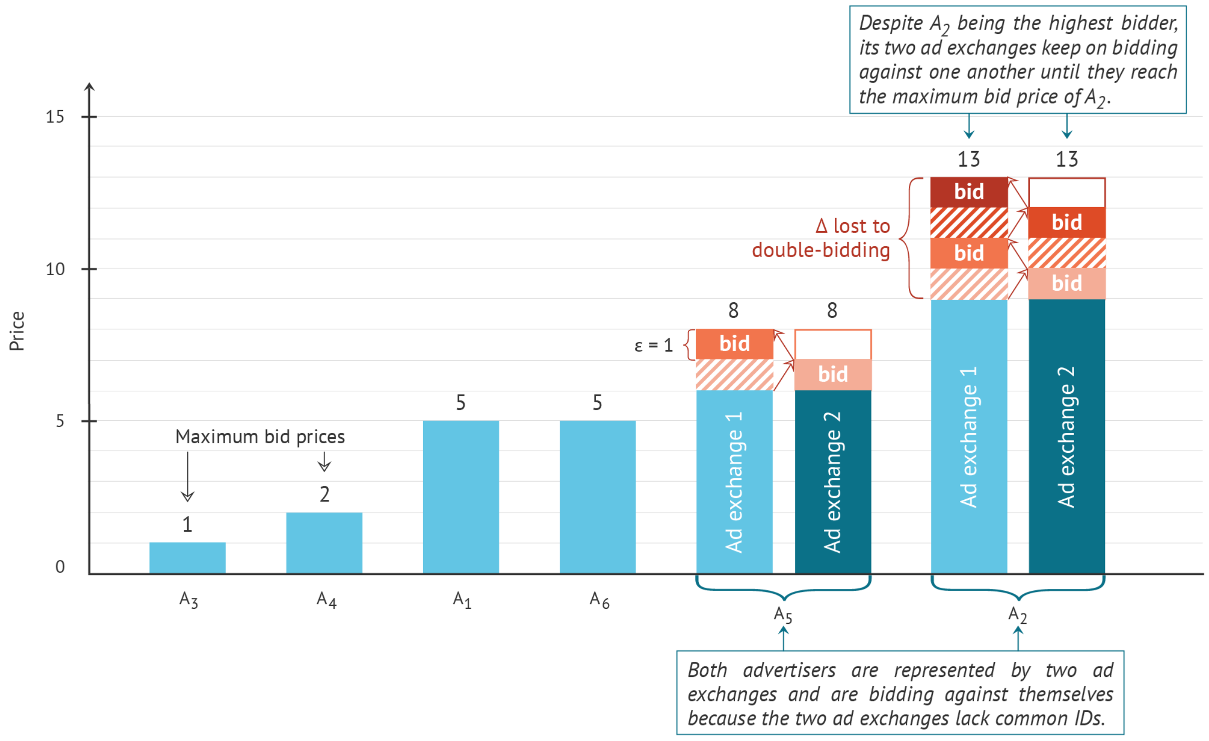


Fig. 8: Six advertisers (A_1 to A_6) are offering $A_1 = 5$, $A_2 = 13$, $A_3 = 1$, $A_4 = 2$, $A_5 = 8$, $A_6 = 5$ as their maximum bid prices. In a second-price auction, a minimal bidding step $\varepsilon=1$ is supposed to allow the winner of the auction to pay the second highest price + ε. However, the lack of a universal ad IDs among the ad exchanges results in advertisers bidding against themselves (A_2 and A_5 in our example are using two ad exchanges each). This artificially inflates the winning bid price, leading to a total loss of Δ. In the case of A_2, instead of winning with a bid price of $A_5 + \varepsilon = 9$, A_2 ended up paying 13, realising a loss of $\Delta = 4$.

While the complexity of advertising technology is increasing, the transparency into its processes is decreasing. In such an opaque landscape, it is easy to forget what the purpose of these technologies was in the first place. Hundreds, if not thousands of ad tech companies are building more tracking services[23],[24] to an online space that is already bombarded by tracking scripts. Industry players are frantically rushing to build solutions that would fix the leaks of a system that is fundamentally broken. What we have today is a product of constant and desperate retrofitting.

Digital advertising is in need of a revolutionary change of course, not just an evolutionary step forward. And that is exactly what AdHash has introduced to the market. Our product will be described in detail in Chapters III through VIII of this paper. Chapters IX and X will outline how our product is superior to those that currently exist and Chapters XI till XIII will explain how we intend to successfully scale it and why we are the right team to do so.

---

[23] *"Browser Fingerprints – The Invisible Cookies You Can't Delete", Mark Stockley, https://nakedsecurity.sophos.com/2014/12/01/browser-fingerprints-the-invisible-cookies-you-cant-delete/, 2014.*

[24] *"Canvas Fingerprinting", Wikipedia, https://en.wikipedia.org/wiki/Canvas_fingerprinting.*

III Overview of Our Solution

Problems to solve

The purpose of this chapter is to outline the approach that we propose for re-engineering a healthy and transparent advertising environment from the ground up. When describing problems in the current advertising ecosystem, we will refer to digital advertising, as those are characteristic of the way ads are programmatically being traded on computers, smartphones, and other digital devices. When outlining our product, we will simply refer to advertising, as a transparent solution would bring many benefits to the way insertion orders are fulfilled in other advertising models as well.
Our main objective here is to work along with the reader in discovering potential solutions to the following problems:

- **Excessive complexity** - with too many layers of intermediaries leading to huge proportions (50 - 75%) of ad spend never reaching publishers.
- **Ad fraud** in its various forms described in the previous chapter.
- **Ad measurement inaccuracies** - the lack of standardisation across all intermediaries leading to discrepancies and data loss.
- **Double-bidding** - advertisers bidding against themselves for the same inventory through multiple ad exchanges and bidding platforms.
- **Lack of accountability** - for all its centralisation, the digital advertising business has been largely unpoliced and unregulated. It has turned into a dizzyingly complex and opaque contraption with unprecedented capacity for abuse.
- **Lack of trust** between advertisers, publishers, and the ad tech intermediaries, largely triggered by the inverse correlation between profits and losses that defines the relationships between the different parties.
- **Non-transparent media buying** - with so many middlemen and unknown commission fees, advertisers rarely know how much of their budgets is actually spent on purpose and how many hands their funds pass through before reaching the publishers.
- **Security vulnerability** - all ad platforms currently suffer from single points of failure because their centralised databases are honey pots for malicious hackers to target.
- **Slow payments** - it usually takes months for earnings to reach publishers' accounts.
- **Slow ad serving** - it often takes days for an ad creative to get approved and ads, on average, take multiple seconds to load.
- **Lack of user control** over the way information about them is gathered and used. The current ad control solutions are largely ineffective.
- **Ad blockers** - reducing significantly publishers' earnings.

Discovering our solution

We decided to approach the task by reasoning from first principles. The aim was to reduce all of the problems listed in the previous section to a single primary cause. Addressing this core problem would allow us to set up the groundwork for solving the more complex issues. The first ten problems listed above all stem from essentially the same nucleus - the lack of a unified identification system. The last two problems would not exist without the first ten.
For all participants to work in sync, there needed to be a standardised method of administering IDs on an industry level, not just a platform level. While there are many ways to generate a unique ID, the important distinction is to make it universally verifiable and tamper-proof. Rather than going through the evolution of unsuccessful attempts we made to get here, we will go straight to the point and synthesise the deduction process of the final solution into ten steps:

1. To create a tamper-proof universal identification system we needed **unique and easily verifiable IDs**. Cryptographic hashes were a perfect fit as they are easy to verify by anyone, yet impossible to falsify. Assigning unique hashes to every publisher, advertiser, and creative in the system would guarantee transparency. All creatives ever served would be known, as would be their sources, guaranteeing accountability. Double-bidding inefficiencies from bidding against oneself would be eliminated.
2. **To further enhance the efficiency of the bidding process and reduce losses and slowness in ad serving**, we introduced the AdHash Bidder. It allowed us to redesign the auction mechanism to only perform one auction per ad size and eliminate ad repetition. Supporting only static creatives further reduced ad serving latency. The end result was an order of magnitude improvement in speed compared to today's ad serving technology and a more pleasant user experience.
3. The bidding process would accumulate vast amounts of data which had to be measured accurately and stored securely. To ensure this, **we had to remove the middlemen that were causing tracking discrepancies, inaccuracies, and data leakages**. For that purpose, we built the AdHash Server-Side Platform. The platform allows advertisers and publishers to store all data on their own servers without ever exposing them to any third parties, not even the AdHash Bidder. Most importantly, it also enables advertisers to self-host their creatives, eliminating the need for any ad serving, targeting, verification, tracking, analytics, and attribution middleware, resulting in unparalleled cost reduction and enhanced security.
4. While privacy was important, transparency was paramount. **We needed an immutable record of all unique IDs, payment transactions, and voting decisions** in order to introduce trust between the transacting parties. Distributed blockchains brought for the first time this level of security and transparency by stripping away power from corruptible centralised databases and redistributing it amongst thousands of network nodes. Instead of relying on dozens of observing middlemen, honest behaviour in the AdHash ecosystem is guaranteed by the incorruptibility of the blockchain.
5. However, existing blockchains could not record enough transaction data to support RTB advertising, so **a solution that offered orders of magnitude more transaction capacity was required**. We developed a way to store all detailed transactions as off-chain transactions. This would provide sufficient capacity and minimise transaction costs, while adding a layer of privacy for publishers and advertisers by keeping their detailed bidding strategies and performance metrics a secret. Thus, the combination of on-chain and off-chain usage enables us to utilise the security and transparency of the blockchain technology in a way uniquely suited to the high-speed transactions of digital advertising.
6. The so far described Server-Side Platform, blockchain, and off-chain applications facilitate safe record keeping. They do not, however, guarantee the trustworthiness of those records. For this **we needed a protocol that would enforce honest transactions**. We designed a voting system supported by an incentivised community that would react much faster than indifferent employees at big corporations. It would bolster security by allowing participants to proactively identify and block abusers.
7. Preventing abuse of the voting system made us **search for a way to make voters submit deposits as collateral for the accuracy of their voting decisions**. Those deposits are governed by smart contracts and foster a self-regulating community in which all disputes are resolved through public votes - a level of transparency impossible to achieve by centralised platforms.
8. Having designed the system for penalising network abuse, **we then needed a rewards system for incentivising contributions**. The prosperity of the entire ecosystem would be dependent on the quality of the voting community so its members needed to be rewarded for their work. To issue rewards, we needed a currency that could support micropayments worldwide, which has proven to be an impossible task for traditional banks using fiat currency. Fortunately, an economic model based on a digital currency

capable of minting rewards would allow us to sidestep this problem. We created the AD token to serve a number of functions in the context of the rewards system. It ensures the security and integrity of the network by rewarding voters for contributing to keep the network healthy and fostering its growth. It enables the network to scale by rewarding publishers for quality traffic, which incentivises initial publisher participation. Most notably, rewards are minted independently of where the publishers or voters earning them are located and how much advertisers are willing to pay for the traffic. This levels the playing field and rewards anyone who delivers high-quality traffic equally. It also decouples the inverse correlation between publishers' gains and advertisers' expenses, minimising the inherent mistrust between the two parties.

9. **Finally, we needed to make sure that all these technologies could interoperate and work in sync**, so we designed a protocol acting as the nervous system of our framework. The AdHash Protocol is easiest to understand when divided into three separate temporal layers serving the different needs of the advertising, payment, and voting components of our framework.
10. **In an effort to ensure the long-term prosperity of the proposed solution**, we open-sourced the protocol we built to allow anyone to contribute, enhance its functionality, and build on top of it. This allows publishers and advertisers to access a universal hash-based identification system. We believe that no single centralised organisation could prompt the entire advertising industry to accept their proprietary identification system and open-sourcing the technology is the only way forward.

Before diving into the building blocks of our product in the following chapters, let us first answer a very important basic question: what is AdHash's business model?

Introducing the AdHash Bidder

The AdHash Bidder is the single intermediary between publishers and advertisers. It performs real-time bidding, matching advertisers' demand with publishers' supply and allocating each impression to the highest bidder. While publishers rely on a CPM pricing model (Cost Per Mille, meaning cost per one thousand impressions) to predict how much revenue they will make, marketers are more interested in the number of visitors they are driving to their sites. Therefore, they often choose to be billed using a CPC model (Cost Per Click). The relationships that tie CPC, CPM, and CTR (Click-Through Rate) together are:

$$CPC = \frac{price}{clicks}; \qquad CPM = \frac{price}{impressions} * 1000; \qquad CTR = \frac{clicks}{impressions}$$

$$=> \qquad CPC = \frac{1}{1000} * \frac{CPM}{CTR}$$

To predict the CTR of a given ad, the bidder can use various statistical methods that assess the ad's past performance, some attributes of the actual creative, its load time, the category of the advertiser, average viewability metrics for the auctioned ad slots, and other data.

Unlike the established CPM model in the current RTB environment, bidding in the AdHash ecosystem happens on a CPC basis. There are four further distinctive features which considerably enhance the efficiency of the auctions of the AdHash Bidder:

User data privacy

When running an auction, the AdHash Bidder is not exchanging user data with anyone, not even with the advertisers bidding for the ad slot. The bidder's capability to directly match advertisers' criteria to the data supplied by the user's browser or mobile SDK, without actually revealing any of this data, offers privacy for the consumer while still enabling the best possible decision-making. We do not let user data be subject to constant probing, collecting and repackaging by various unknown third parties. Our method enables effective targeting without the risk of any data leakage. And we do this not by relying on behavioural advertising that is becoming increasingly unpopular with both users and regulators but by doing purely contextual targeting.

Hard data targeting for advertisers

The AdHash Bidder targets technical parameters, locations, semantic contextual data, real-world contextual data about financial instruments, weather forecast, sport results, and other data that are directly measurable and verifiable. We call these *hard data*. The bidder does not consider any approximated, probabilistic, or purchased third-party data that are not verifiable or directly measurable. We call these *soft data*.

Our conscious decision to exclude any dubious data from our targeting was aimed at protecting advertisers from the low-quality aggregated audience data often found in Data Management Platforms (DMPs). According to research conducted by Nielsen[25], the on-target percentage for most targeting criteria rarely exceeds 50%, and in certain cases can often fall below 20%. Yet, DMP fees can easily amount to 50% of the total advertising spend.

Those figures create more than the typical concern about the accuracy of third-party data and their return on ad spend. Unfortunately, this is not just symptomatic of the third-party DMP industry. Even the highly-rated first-party data Facebook proudly advertises its platform for can grossly overestimate its reach. One analyst discovered that Facebook claimed to have 41 million users in a specific US demographic - 32% more than were actually alive according to US census data[26]. Similar discrepancies can be found in the numbers of unique cookies provided by Google's DoubleClick Bid Manager availability reports.

These are just a few examples in the large pile of evidence stacking up against the reliability of audience targeting. That approach is not only inefficient for advertisers but also unprofitable for publishers, as all gains from it end up in the hands of middlemen who are either delivering no value, or actually bring negative value. As Carnegie Mellon University professor of IT and Public Policy Alessandro Acquisti stated at an FTC hearing on the economics of big data and personal information:

> *"What we found was that advertising with cookies - so targeted advertising - did increase revenues but by a tiny amount. Four per cent. […] Simultaneously we were running a study, as merchants, buying ads with a different degree of targeting. And we found that for the merchants sometimes buying targeted ads over untargeted ads can be 500% times as expensive. […] How is it possible that for merchants the cost of targeting ads is so much higher whereas for publishers the return on increased revenues for targeted ads is just 4%?"*[27]

---

[25] *"Nielsen Digital Ad Ratings", The Nielsen Company, http://www.nielsen.com/content/dam/corporate/us/en/reports-downloads/2016-reports/nielsen-digital-ad-ratings-us-benchmarks-report-q2-2016.pdf, 2016.*

[26] *"Facebook Ads Figures Don't Match Census Data - Analyst", Nasdaq, http://www.nasdaq.com/video/facebook-ads-figures-don-t-match-census-data---analyst-59b05b0144a64b1ec55c0b49, 2017.*

[27] *"The Case Against Behavioural Advertising is Stacking Up", TechCrunch, https://techcrunch.com/2019/01/20/dont-be-creepy.*

Recalibrating our targeting approach exclusively around hard data allows for more reliable campaign strategies and predictable outcomes. With the proliferation of data-driven algorithmic decisioning, it is more important than ever to utilise accurate and verifiable data.

Efficient auctions

The unique ad hashes allow us to further bolster the efficiency of the bidding process when running second-price auctions[28]. Publishers typically have two or three different ad sizes, each used several times on a single page. Instead of running a separate auction for each ad size on that page, the AdHash Bidder uses a single auction that distributes the slots in a descending order to the highest bidders. This reduces the number of auctions performed by the bidder, allowing quicker ad serving. It also eliminates any ad repetitions on the same page, boosting both ad efficiency and user experience. These gains are additionally amplified by using strictly static creatives that are much lighter to load.
The bidding efficiency will be subject to continuous refinements not just from within our team, but from everyone willing to contribute or build their own bidder on top of our framework. More information on integrations can be found in Chapter IX.

A transparent and verifiable commission

The AdHash Bidder is at the core of our business model. It collects a commission from the price of every verified click. The commission can be verified through the bidder's public node. It is deducted from the price of each winning bid, meaning that it comes out of the publisher's end and its value can easily be checked through the blockchain records. This is the only fee in an ad's journey from the advertiser's server to the publisher's page, making AdHash an order of magnitude cheaper than any existing solution on the market.

A simplified code explaining how the auction model works is presented below:

```
/*
The bidder receives an ad request from the user visiting the publisher's website.
The list is passed in the POST parameters (example on the next page).
getRequestedCreatives() returns the list of requested ad sizes as a JSON object.
*/
var requestedCreatives = getRequestedCreatives();

/*
The bidder receives the publisher wallet ID in the POST request.
If no publisher wallet ID is provided, then the bidder will serve no creative.
There is no point in providing a fake ID as the funds will be transferred to a
wallet ID not controlled by the publisher.
getPublisherID() returns a wallet ID, e.g.:
0x17E1E918A113331ECCca9FA51f64a0B1187608D6.
*/
var publisherID = getPublisherID();
if (!publisherID) {
   return {};
}

/*
The bidder goes through each requested ad size and tries to find the highest
bidding advertiser.
All results are stored in the creatives array to be used in some of the operations
below.
*/
var creatives = [];
for (var key in requestedCreatives) {
   var currentCreative = requestedCreatives[key];

   /*
   The bidder first checks if the minimal requirements to start the bidding process
```

28 *"Generalized Second-Price Auction", Wikipedia, https://en.wikipedia.org/wiki/Generalized_second-price_auction.*

```
are fulfilled:
- are there advertisers who are bidding for the particular ad size;
- are there advertisers who want to bid in the specific time slot.
If no advertisers are interested at that time, then an empty object is returned.
getBids() returns an array of possible bids sorted by bid price (example on
p.27).
*/
var bids = getBids(currentCreative.width, currentCreative.height);
if (bids.length == 0) {
   creatives.push({});
   continue;
}

/*
The bidder filters all interested bids based on the targeting criteria, such as:
- blockclists of users, publishers, and advertisers;
- targeting based on device, model, browser;
- targeting based on screen resolution;
- targeting based on geolocation.
filterBids() returns a list similar to bids but filtered only for the relevant
criteria.
*/
var filteredBids = filterBids(bids);
if (filteredBids.length == 0) {
   creatives.push({});
   continue;
}

/*
The first element in filteredBids is the highest valid bid for the ad position.
getExpectedHashes() gets a list of the expected hashes from the blockchain.
*/
var expectedHashes = getExpectedHashes(filteredBids[0].advertiserID);
/*
The bidder has to calculate the winning bid for the ad position which is equal
to the second bid price plus a minimal bidding step ε.
If there is only one bid, then the price of the click is equal to ε.
If there are several bids, the bidder takes the second best price + ε.
getMinimumStep() returns the minimum ε.
getCommission() returns the commission derived from the winning bid.
getSecondPrice() returns the second highest price. If the first and the second
bids are the same, no ε is added so that the final price is not higher than the
advertiser's bid. The winning advertiser is chosen at random.
*/
var price = 0;
var commission = 0;
if (filteredBids.length == 1) {
   price = getMinimumStep();
} else {
   price = getSecondPrice(filteredBids) + getMinimumStep();
}
commission = getCommission(price);
price -= commission;

/*
After the bidder has acquired all information for the ad position, the creative
is added to the list supplied to the publisher, containing the following
information:
- advertiserID is the wallet ID of the advertiser;
- advertiserURL is the URL where the user can request the creative;
- commission as calculated above;
- maxPrice as calculated above.
*/
creatives.push({
   advertiserID: filteredBids[0].advertiserID,
   advertiserURL: filteredBids[0].advertiserURL,
      expectedHashes: expectedHashes,
      commission: commission,
      maxPrice: price
   });
}

/*
Now that the bidder has everything ready the final result object is served:
- creatives is the list of creatives to be served calculated above;
- period is the current period as set by the bidder ;
- nonce is a random integer used to prevent duplicate requests.
*/
return {
   creatives: creatives,
   period: getCurrentPeriod(),
   nonce: getRandomNonce()
};getRequestedCreatives() example output:
```

```
[
       {width: "300", height: "250"},
       {width: "300", height: "600"}
]
```

```
getExpectedHashes() example output:
```

```
["091f0aadd17687e1f326108581f2caedce5ea9e8",
"17d6883bcec9b022396e1722e10bd0da5d3cc677",
"94ae6f3e7825d1a3e681f2009e09d9fdb949750c"]
```

```
getBids() example output:
```

```
[
       {
              "advertiserID": "0x61f2d8c7003045D589D93E0e2Ab932E6121239a3",
              "advertiserURL": "http:\/\/example1.com\/serve.php",
              "maxPrice": 2.6
       },
       {
              "advertiserID": "0x401715FB0728dcB8aD80a775F4446A32fd69a1eD",
              "advertiserURL": "http:\/\/example2.com\/serve.php",
              "maxPrice": 2.05
       },
       {
              "advertiserID": "0xFFd721B3e47a37bb01dF091ee8bA7437ca8319EF",
              "advertiserURL": "http:\/\/example3.com\/serve.php",
              "maxPrice": 1
       }
]
```

Our aim will be to continuously maintain the minimum bidding step ε close to the equivalent of 0.01 EUR by adapting it based on the exchange rate of AD to EUR.

Generating unique ad hashes

Now that we have explained how the bidder efficiency is increased, let us cover the topic of unique ad hashes. They offer a diverse set of benefits for many of the processes described in this paper: from eliminating double-bidding and disarming ad fraud, to introducing accountability and facilitating a more efficient bidding process. Each of these benefits is presented within context in the appropriate sections. This paragraph will focus solely on the process of generating the unique ad hashes.

When a marketer uploads a new ad creative, its hash is calculated, recorded on the blockchain, and distributed across all its nodes. This makes the history of the ads served, the advertisers who served them, and any changes made to these ads accessible to anyone at any time. A bad actor hosting a malicious advert would therefore be immediately known by the entire network and the associated campaigns would be stopped almost instantly, followed by confiscation of the bad actor's advertising deposit.

A simplified version of the code that calculates the unique ad hashes and publishes them on the blockchain is presented on the next page. It is important to note that the AdHash Bidder only allows static creatives as a solution to clutter, slow ad loading, intrusive creatives, and malware.

```
*/
The Server-Side Platform extracts the following information from  the file system:
*/
var creativePath = './creatives/creative1.png';
var creativeFilename = 'creative1.png';
var creativeFiletype = '.png';
var creativeWidthPx = 300;
var creativeHeightPx = 250;

/*
The actual creative (JPG, JPEG, or PNG) is then converted to Base64 and stored.
*/
var base64Path = './creatives/creative1.b64';
transformToBase64(creativePath, base64Path);

/*
A SHA1 checksum is generated from the Base64 file. fileChecksum() returns a 40
digit SHA1 string, e.g.: "a5f2ed2240b6f13b0f8ea0aebae856105ffc2b07"
*/
var SHA1Checksum = fileChecksum(base64Path);

/*
The Base64 file can be later served directly as static content via JSON instead of
calculating it each time which saves processing time.
storeCreativeInDatabase() updates the advertiser's database and returns true if
successful.
*/
var storeSuccess = storeCreativeInDatabase(
    creativeFilename, creativeFiletype,
    creativeWidthPx, creativeHeightPx, SHA1Checksum
);

/*
Finally, the new creative must be recorded on the blockchain.
When the creative is announced, an event is triggered and the bidder is notified.
From there on, the bidder serves the checksum in the "expected hashes" to all
publishers.
They do their own SHA1 checksum and validate that the creative is expected.
If even the smallest detail in the creative is changed, the SHA1 checksum becomes
different and the publishers refuse to display it.
announceCreative() creates a transaction in the blockchain containing the SHA1
checksum.
*/
if (storeSuccess) {
  var yourOwnWallet = getWalletID();
  announceCreative(yourOwnWallet, SHA1Checksum);
}
```

Benefits of our solution

We will explain how our framework works for the twelve problems listed at the beginning of this chapter and use this opportunity to list four additional benefits brought by it. This list is not exhaustive and there are some minor further improvements that are not mentioned here but can be found on our website.

- **Excessive complexity** - offering advertisers the opportunity to self host their creatives removes the need for the vast majority of intermediaries currently present. Here is a modified LUMA Scape to demonstrate this effect:

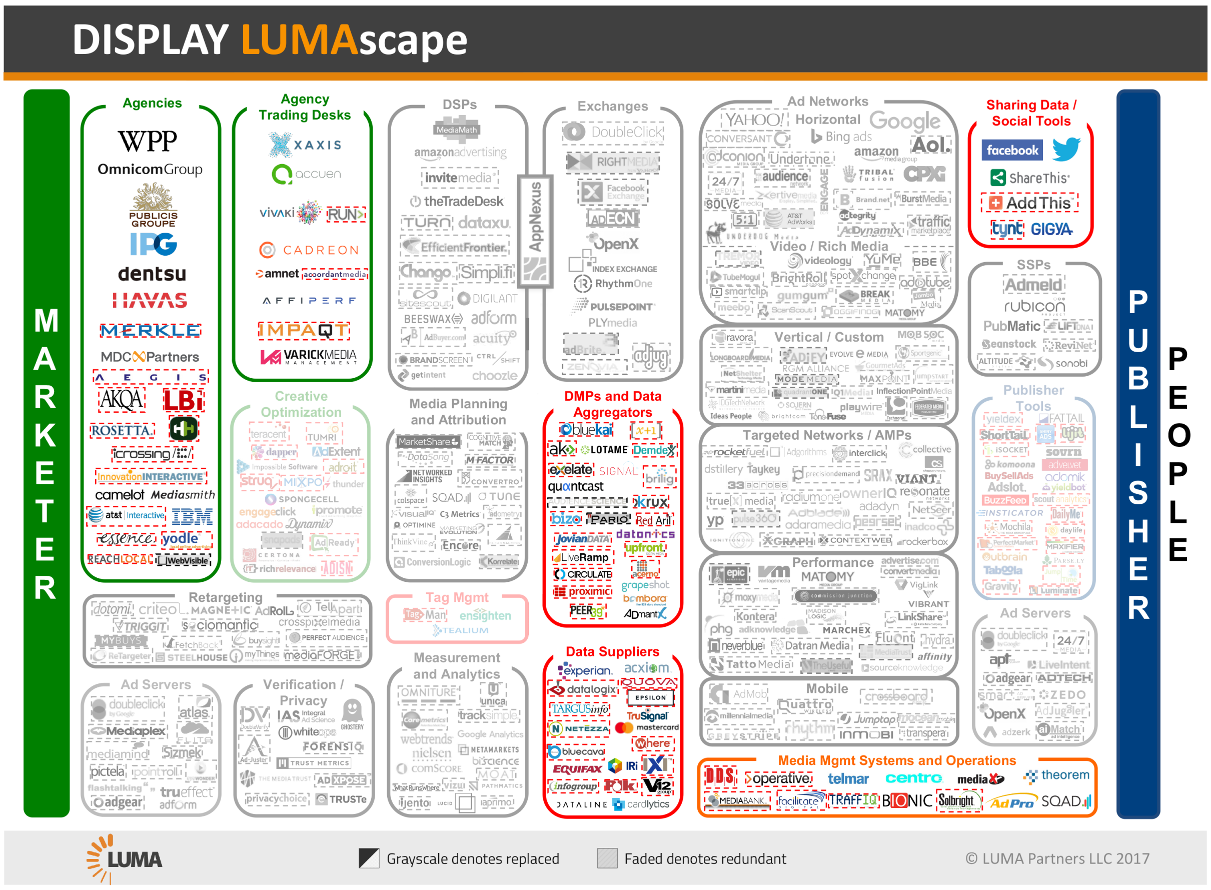


Fig. 9: The LUMA Scape from Fig. 6 reimagined for our ecosystem. All fields in greyscale represent intermediaries who have been replaced by our technology; the ones in faded colours have been made redundant by it.

- **Ad fraud** - over ten different types of ad fraud were presented in the previous chapter. Domain spoofing and operations similar to Methbot and Hyphbot are rendered helpless thanks to the unique ad hashes and smart contracts. Fraudsters would still be able to spoof the domains the ads are showing on, but they would not be able to collect any payments from advertisers since they would go to the wallet IDs of the publishers they are pretending to be and not their own. Device ID reset frauds would not be applicable because we are not using the CPI/CPA model of payment those fraudsters take advantage of. Invisible or low viewability ads, on the other hand, will not incur any damage to advertisers because they are only paying for clicks and not for impression that are easier to manipulate. Geolocation masking cannot be completely eradicated until internet service providers can ensure the integrity of their data. However, the encryption of our creatives described in Chapter VII provides a second layer of verification and security for advertisers since the creatives would only decrypt if the user's details match the advertisers' first-party targeting data. And for that, they can use any geolocation database they choose, including integrating their own. Encryption also helps in the case of hijacked ads because they would not be decrypted when shown on another user's website. Click farms, accidental clicks, and bot traffic (even when mixed with real traffic) would be weeded out by the combination of transparency brought by on and off-chain transactions and the heatmap analysis tool, described in Chapter IV. Malvertising is rendered impossible by the use of static image creatives with no scripts. Ads with unknown sources are a thing of the past thanks to the immutable ad serving history stored on the blockchain.

- **Ad tracking inaccuracies** - allowing advertisers for the first time to self-host their ads enables them to track and measure every single ad request to their servers using their own technology, our analytics tools, or chosen third-party providers. It is a widely agreed-upon fact that first-party data are far more accurate[29], but until now, advertisers had no way of using them.
- **Double-bidding** - unique creative, advertiser, and publisher hash IDs completely eliminate the problem which was described in the previous chapter.
- **Lack of accountability** - our voting system combined with deposits executed through smart contracts guarantee that bad actors would be held accountable.
- **Lack of trust** - shared off-chain transactions verified by an independent third party (the AdHash Bidder) and summarised in an immutable blockchain should eliminate most if not all concerns regarding trust among parties. Publishers and advertisers will no longer need to wonder whether to trust one-another.
- **Non-transparent media buying** - all commissions are publicly declared and most importantly - completely verifiable through the blockchain. Thanks to the superior quality of first-party data measured directly by the advertisers when self-hosting ads, the amount of third-party tracking services is brought to a minimum. 97% of the advertisers' budgets therefore go directly to the publishers within 24 hours.
- **Security vulnerability** - all data are securely stored on their owners' servers, thus creating a vast distributed storage network that is resistant to attacks. On top of that, competing bidders would be built by independent teams, decentralising the infrastructure of our solution and preventing the existence of a single point of failure.
- **Slow payments** - launching a native digital currency for trading ads ensures that payments would be executed daily.
- **Slow ad serving** - restricting creatives to just static images greatly reduces the average creative size. Additionally, HTML 5, CSS, and most importantly - Javascript are not allowed in the creatives, making them far less demanding on the devices loading them. This would also exclude any risk of seeing the worst offenders in digital advertising joining the AdHash ecosystem - ads running mining scripts for Monero[30] and abusing the resources of unsuspecting users.
- **Lack of user control** - unique ad hashes provide vastly improved accountability and reduced overall complexity that will translate in superior user control.
- **Ad blockers** - we are making the advertising experience more acceptable and meaningful by removing intrusive, distracting, and annoying advertising formats and focusing on contextual rather than personally targeted advertising. For users who do not wish to see any ads at all, we present an alternative: to participate in the bidding process with AD tokens and outbid any advertisers currently reaching out to them. This way, users who do not want to see ads can use the internet comfortably without hurting the publishers who bring them the content they enjoy. If all websites a user visits

[29] *"Age, Gender Targeting are Broken; Publishers' First-Party Data Can Fix Them", Adweek, http://www.adweek.com/digital/matt-bruch-pch-media-guest-post-first-party-data/, 2017.*

[30] *"Crypto-Jackers Slip Coinhive Mining Code into YouTube Site Ads", The Register, https://www.theregister.co.uk/2018/01/27/cryptojackers_slip_coinhive_mining_code_into_doubleclick_ads/, 2018.*

participate in the AdHash ecosystem, this should not cost more than a few euros a month. This increases publishers' independence from advertisers, further minimising the mistrust between the two. Ad blockers, on the other hand, would not be able to block ads without entirely blocking access to the websites of the advertisers, which would not be accepted by users.

- **Reduced impact of human errors** - having a dedicated community minting rewards for their work guarantees that mistakes will be fixed much more quickly and reliably.
- **No ad repetition** - having unique ad hashes for all creatives would ensure that no user will ever have to see the same ad repeated endlessly on the same page.
- **Transparent policies** - no central authority can ban or regulate specific types of ads or content, all decisions are proposed and voted into action by the community.
- Your advertising strategy is not for sale - "similar audiences" and other such tools effectively resell a advertisers' core audiences to their competitors. AdHash makes this impossible by enabling all advertisers to keep the data about their campaigns on their own servers and not exposing them to anyone, even the AdHash Bidder.

In order to arrive at a solution that would fit all requirements, we did our research thoroughly. We got into the ad tech industry knowing that it was flawed and that we needed to work on a fix. We carefully tested all inefficiencies and tried to solve them within the confines of the existing framework. With time, it became obvious that such a gentle approach was not going to work. We then designed an entirely new framework, built and tested a minimum viable product, described our work in this paper, and went on to build a fully-functional product.

The next three chapters will reveal the three core building blocks of AdHash's solution.

IV Server-Side Platform

Basics

We start the technical description of the three main components of our product with the one that would feel most familiar to those operating in the current advertising ecosystem. The Server-Side Platform provided by AdHash is developed for both publishers and advertisers. It acts as the connective tissue between all parties: the advertisers, publishers, users, voters, the AdHash Bidder, and the blockchain. The Server-Side Platform handles off-chain transactions and their communication and verification. And importantly, it is the software that stores and analyses all advertising data for both publishers and advertisers. Analysing and visualising data have been our team's strengths for years and we have used our experience to build a product that is exceptionally functional.

The current chaos in ad measurement and verification often leaves marketers no other option but to compare data from their ad servers and various platforms to the data measured directly on their web properties by Google Analytics[31], Mixpanel[32], and others. The latter data are usually considered the reference point. The marketers have to compare datasets provided by different third parties using independent technologies and attribution metrics that barely match at all.
As mentioned in the previous chapter, AdHash's innovation in allowing ad serving to be self-hosted makes it possible for all measurements to be performed directly from the advertisers' servers - the same servers used to deliver the content of their websites or apps. This dramatically reduces the discrepancies and essentially makes any external measurement, analytics, and verification services completely unnecessary. It also allows accurate data to be collected in real-time.

The communication between the server-side platform and the bidder is compatible with the OpenRTB 3.0 protocol[33]. While we believe that our solution performs best when used with the AdHash Server-Side Platform, there are no implied lock-in restrictions. Any individual publisher, advertiser, or competitor of AdHash can build and integrate an alternative platform that would just as easily communicate with the rest of the infrastructure.

Below we outline briefly what the Server-Side Platform does. Anyone interested in integrating their own solution can download our full documentation regarding the integration with our bidder, adding new bidders to the ecosystem, and executing the blockchain and off-chain transactions. More information about the integrations can be found in Chapter IX of this paper.

Data storage

The Server-Side Platform has to be installed on the publishers' and advertisers' servers. The platform allows clients to control the amount of storage designated for their databases and the type of data recorded in it. This determines how much historical information about the ad serving will be recorded. All detailed advertising information is stored exclusively on the owner's servers. Neither the AdHash Bidder, nor the counterparty in any given transaction has access to that information. This ensures a high level of security and great network

[31] *Google Analytics, https://analytics.google.com/analytics/web/.*

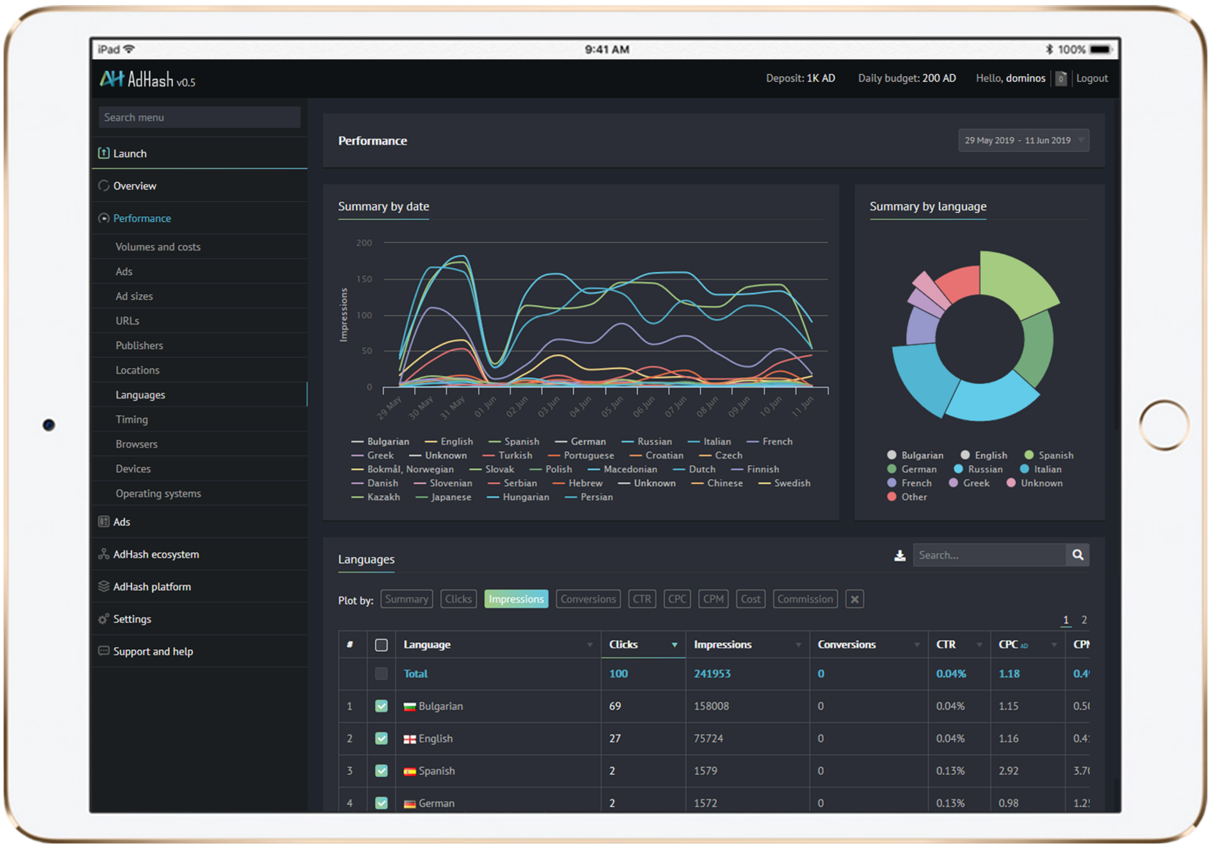
[32] *Mixpanel, https://mixpanel.com.*

[33] *"OpenRTB 3.0 Framework", IAB Tech Lab, https://iabtechlab.com/wp-content/uploads/2017/09/OpenRTB-3.0-Draft-Framework-for-Public-Comment.pdf, 2017.*

scalability. Each participating node performs data aggregation and analysis for its own dataset, making sure that there is no single point of failure. All publishers, advertisers, and bidders are required to store their data for a minimum of 60 days in order to comply with any off-chain transaction requests. Failure to provide all data when called through a sync request will result in a temporary suspension from the ecosystem and potential withholding of the deposit. This is done to ensure maximum accountability when disputes arise.

Analytics dashboard

On top of providing the storage services listed above, the Server-Side Platform also offers a user interface for setting up campaigns and analysing their results.
The analytics part is similar to what publishers and advertisers have come to expect from advertising platforms. As already explained in the paragraph about our philosophy regarding user data, measurements only include hard data that is directly measurable. Here is a screenshot of our analytics dashboard for advertisers:



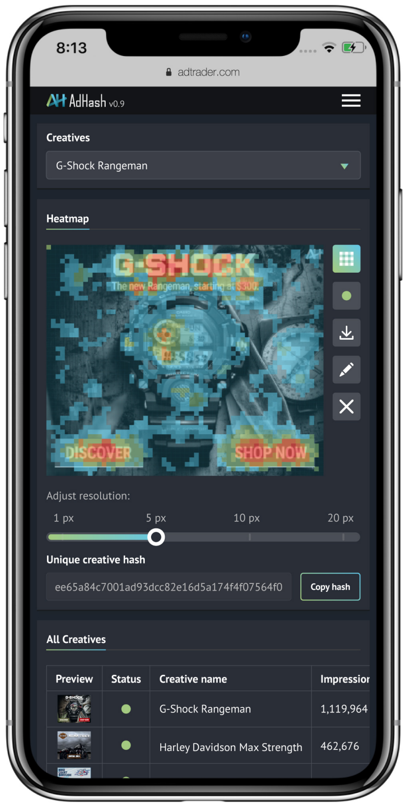
#		Language	Clicks	Impressions	Conversions	CTR	CPC AD	CP
		Total	100	241953	0	0.04%	1.18	0.4
1	✓	Bulgarian	69	158008	0	0.04%	1.15	0.5
2	✓	English	27	75724	0	0.04%	1.16	0.4
3	✓	Spanish	2	1579	0	0.13%	2.92	3.7
4	✓	German	2	1572	0	0.13%	0.98	1.2

Fig. 10: A preview of AdHash's analytics dashboard for advertisers which includes true real-time data analysis that is not supported by any platform on the market today.

Bot traffic and accidental clicks prevention

Building an advertising platform from the ground up gave us the opportunity to rethink some of the core functionalities in the currently existing systems. We knew that simulated clicks by bots and the incentivisation of accidental clicks were some of the aspects of ad fraud that were hardest to detect and wanted to offer advertisers the best tools to combat them.
Fortunately, there is a simple visual representation that can help detect fake and random clicks on any ad creative - a heatmap overlay. It makes the click density on various parts of the creative easy to compare. Unintended clicks produce random noise. Click bots, designed

to defraud advertisers, typically produce either random noise, or a concentration of clicks in certain areas (usually in the top left pixel). By reviewing all click locations, a human can easily assess the quality of acquired traffic - clicks should be concentrated around buttons, logos, and attention-grabbing parts of the creative.
This system is not a guarantee against fake and random clicks, but it is one of the most effective ways to discern between good and bad traffic. Heatmap overlays are currently being sold as very expensive third-party services requiring additional scripts from companies like MOAT[34].
The AdHash Server-Side Platform offers a free heatmap functionality. Coordinates are recorded for every click and their visualisation is configurable to provide the best insights:



Creatives: G-Shock Rangeman

Adjust resolution: 1 px · 5 px · 10 px · 20 px

Unique creative hash: ee65a84c7001ad93dcc82e16d5a174f4f07564f0

Preview	Status	Creative name	Impressior
	●	G-Shock Rangeman	1,119,964
	●	Harley Davidson Max Strength	462,676

Fig. 11: A preview of the mobile version of the AdHash analytics dashboard illustrating how heatmap analysis helps advertisers distinguish between real and invalid traffic. Heatmap data are collected in real-time as well.

Platform management

The AdHash Server-Side Platform interface equips both publishers and advertisers with a set of control tools for their bidding preferences and targeting criteria. Advertisers can set CPC caps for each creative along with frequency and recency capping to avoid annoying users with repetitive ads. Publishers can set up new domains and ad unit tags and implement price floors. The platform allows all participants to set up some general parameters, such as storage allocations, maximum gas prices, geolocation databases, and to register to participate in the voting system.

An important management functionality is the option to classify counterparties into two lists. Examples are given from the point of view of advertisers:

34 https://moat.com

- Whitelists could be used so that no matter how many new publishers join the ecosystem, only a pre-approved group would be included in the targeting of campaigns. That also means that any unwanted publishers would never participate in their campaigns.
- Blocklists offer the same functionality for the individual advertiser as blacklists do. However, they are also communicated to the voting community and if approved, they get updated for the entire ecosystem. If a publisher's unique hash ID gets added on a blocklist, that ID would be permanently banned unless another vote is called to remove it from the blocklist.

Communication between parties

The Server-Side Platform plays an integral role in the communication between the user, publisher, advertiser, bidder, and the blockchain. The advertiser's platform needs to push all newly uploaded or edited creatives' hashes to the blockchain. It has to update the bidder about any changes to the bidding strategy. It must record all clicks off-chain and as soon as a request comes from the bidder, needs to synchronise those transactions with the blockchain. The platform also facilitates bidirectional communication between the transacting parties, allowing them to compare and verify each others' metrics. All communication with the blockchain regarding blocklists and voting is also handled by the Server-Side Platform.

Real-time data feed

Another exciting feature enabled by AdHash's decentralisation is something that is not offered by any major advertising platform at the moment: real-time data analysis. All ad interactions are measured and displayed within five seconds of occurring. This means that not only are ad creatives approved within minutes rather than days in the AdHash ecosystem, but also that data can be collected, analysed, and acted upon in seconds rather than hours or days.

Dynamic click exits

As described on page 27, click interactions with any ad creative through AdHash contain the precise coordinates of the click. This creates the opportunity to define multiple click exit actions on a static ad creative, allowing the flexibility of HTML 5 ad creatives without the slow loading, security vulnerabilities, and user fatigue typically associated with those. In terms of flexibility, having different buttons on an image lead to different landing pages is an innovation unmatched by any of the ad platforms (most notably Facebook's) that serve just static image ads.

Speed and scalability

It could be argued that any sufficiently small advertising platform could provide real-time data while the load on its servers is still small enough. In the current advertising ecosystem the biggest platforms are often the slowest ones and the bigger an advertising campaign is, the longer it takes to generate reports for it. The beauty of distributed solutions, however, is that there is infinite scalability built into them. The speed of data recording and processing for each client depends entirely on that client's hardware. Even if some machines start running out of resources and struggle to process all the necessary information (or get temporarily disconnected from the network), that would not affect anyone else's ability to process information.

An added benefit of the distributed nature of ad serving through AdHash is the fact that there will be no more discrepancies between the connection speed to an ad server and the connection speed to the advertiser's server. Currently, ad serving is performed by centralised ad servers and there are often significant inconsistencies between data measured by the ad exchange, the publisher's ad server, and the corresponding data on the advertiser's server. The most frequently cited source of this discrepancy is the fact that sometimes the server of the advertiser loads too slowly and users who clicked on the ads gave up before the page would even load.
In our distributed solution, the ads are hosted on exactly the same machines as the content they lead to. Therefore, if an advertiser's server is slow at any given moment, the ad will most likely not load and the advertiser will not be charged.

Requests load

Discussing the topic of scalability cannot be complete without considering the requests load in our ecosystem and comparing it to that in the existing model of advertising. Here we will offer an approximate numerical value of just how significant the load reduction in our solution is.

The following use cases provide an overview of how an ecosystem of a certain number of advertisers and a number of publishers functions. We demonstrate the load on all the servers involved and provide comparison with the traditional centralised programmatic ecosystem.

The AdHash Bidder establishes a direct connection between the advertiser's server and the user. The number of server requests during this auction is proportional to the number of different ad sizes on the page that is being loaded. Note that even in the case when the publisher has several identical ad unit sizes on a page, there is only one auction per ad size. This is achieved because in the AdHash Bidder's auction mechanism the second best position of each ad size goes directly to the second highest bid, the third best position to the third best bid, and so on. An important additional detail is that the unique ad hash prevents the same creative from bidding multiple times even if there are multiple bidders involved. In such a scenario, the verification for duplicate bids coming from the same advertiser is performed by the publisher's Server-Side Platform.

The single auction per page mechanism described on page 18 is just a minor technical improvement on the current programmatic auction model but it has a considerable impact on reducing the volume of requests. The real strength of AdHash's technology is that no repeated requests are issued to multiple ad exchanges as it is currently being done between DSPs, ad exchanges, and SSPs. The presence of the asynchronous layer of the AdHash Protocol described in Chapter VII ensures that all advertisers push their updated bidding criteria to the bidder. Thus, the bidder does not have to pull them with requests at each auction. This results in a much faster bidding process that happens entirely inside the bidder without the need to issue any requests to outside parties. This equates to just one request for each auction. In today's advertising industry, a publisher's ad server usually reaches out to dozens of supply-side platforms and ad networks and does so for each ad slot it needs to fill. On top of that, arbitraging means that those networks are often reselling the same traffic. The end result is having hundreds of requests flowing in-and-out of the publisher's ad server. This gives AdHash a two orders of magnitude advantage in terms of number of requests while handling the exact same bidding process and delivering more transparent and detailed results.

Even more dire is the current situation for publishers running their header-bidders on the client side. They typically execute hundreds of requests for each ad auction, but the burden of those requests falls on the users' machines[35].

In this section we focused on the first of the three crucial components of our solution - the Server-Side Platform. It is the one that shares the most similarities with the familiar advertising infrastructure of today and that stores the largest amount of data. It is the part of our solution that provides security to both publishers and advertisers. In the next chapter, we discuss how the transaction records are made immutable to provide accountability.

35 *"Server-Side Header Bidding vs. Browser-Side Header Bidding", Sovrn, https://www.sovrn.com/hub/learn/server-side-header-bidding-2/.*

V Storing Information on the Blockchain

A brief introduction to blockchains

A blockchain is a digital ledger containing immutable information and distributed amongst all participating nodes in a network. Distributed systems benefit from far superior transparency than centralised ones. They also eliminate the risk of interference by a singular authority. The problem that blockchain technologies solve is to introduce trust between parties who do not necessarily trust one another by enforcing a shared version of history authenticated by mass collaboration and powered by collective self-interest.

The blockchain is composed of consecutive linked blocks, hence its name. Each block contains a hash[36], or a unique identifier, a list of recent valid transactions, the hash of the previous block, and a nonce. The previous block hash links consecutive blocks together and prevents any of them from being altered or new ones being inserted between two existing blocks. In this way, each subsequent block strengthens the verification of the previous one, and therefore the entire blockchain. The correct version of the blockchain is thus always assumed to be the longest existing one. Here is a simple visualisation:

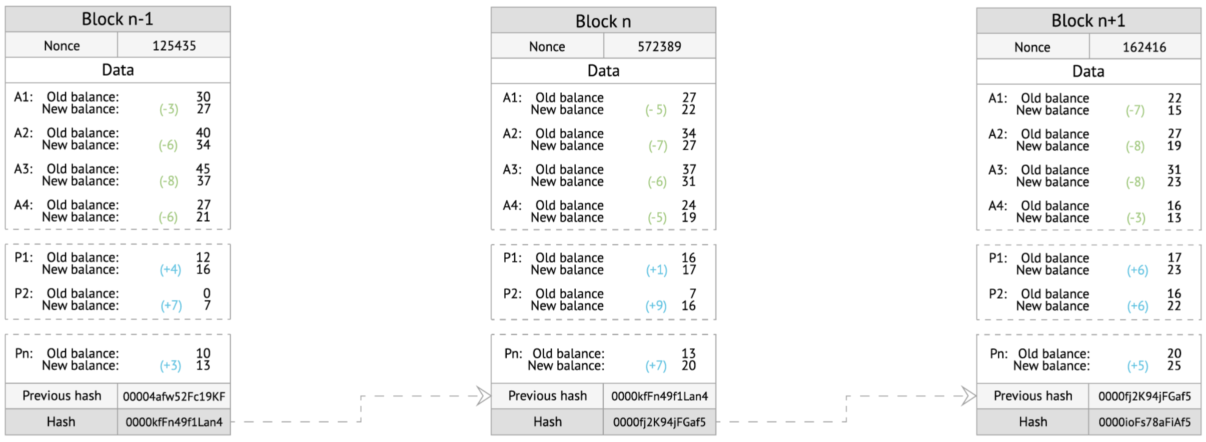

Block n-1
Nonce 125435
Data
A1: Old balance: 30 New balance: (-3) 27
A2: Old balance: 40 New balance: (-6) 34
A3: Old balance: 45 New balance: (-8) 37
A4: Old balance: 27 New balance: (-6) 21
P1: Old balance: 12 New balance: (+4) 16
P2: Old balance: 0 New balance: (+7) 7
Pn: Old balance: 10 New balance: (+3) 13
Previous hash 00004afw52Fc19KF
Hash 0000kfFn49f1Lan4
Block n
Nonce 572389
Data
A1: Old balance 27 New balance (- 5) 22
A2: Old balance 34 New balance (-7) 27
A3: Old balance 37 New balance (-6) 31
A4: Old balance 24 New balance (-5) 19
P1: Old balance 16 New balance (+1) 17
P2: Old balance 7 New balance (+9) 16
Pn: Old balance: 13 New balance: (+7) 20
Previous hash 0000kfFn49f1Lan4
Hash 0000fj2K94jFGaf5
Block n+1
Nonce 162416
Data
A1: Old balance 22 New balance (-7) 15
A2: Old balance 27 New balance (-8) 19
A3: Old balance 31 New balance (-8) 23
A4: Old balance 16 New balance (-3) 13
P1: Old balance 17 New balance (+6) 23
P2: Old balance 16 New balance (+6) 22
Pn: Old balance: 20 New balance: (+5) 25
Previous hash 0000fj2K94jFGaf5
Hash 0000ioFs78aFiAf5

Fig. 12: A simple blockchain visualised. A much more beautiful and comprehensive animated demonstration can be found at: https://anders.com/blockchain/.

The origins of the ideas that led to implementing blockchain technology can be traced back to the early 1990s[37]. However, it was successfully popularised for the first time by Satoshi Nakamoto's Bitcoin paper in 2009[38]. He cleverly combined the previously developed concepts of a Proof of Work (PoW) mechanism[39] with that of digital scarcity[40] and timestamped transactions into an ongoing chain. Bitcoin has seen a tremendous surge in popularity ever since. A very substantial evolution has taken place as blockchains became faster, more efficient, more secure, and a lot more open with respect to the shared data being stored. Bitcoin's blockchain is used to store unspent transaction balances, whereas

---

36 *"Cryptographic Hash Function", Wikipedia, https://en.wikipedia.org/wiki/Cryptographic_hash_function.*

37 *S. Haber, W.S. Stornetta, "How to Time-Stamp a Digital Document," In Journal of Cryptology, vol. 3, N. 2, p. 99-111, 1991.*

38 *S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System", https://bitcoin.org/bitcoin.pdf, 2008.*

39 *A. Back, "Hashcash - A Denial of Service Counter-Measure," http://www.hashcash.org/papers/hashcash.pdf, 1997.*

40 *W. Dai, "B-Money," http://www.weidai.com/bmoney.txt, 1998.*

Ethereum's one, introduced four years later[41], is now used to store computer programs written in a Turing-complete language influenced by C++, Python, and JavaScript called Solidity[42]. These programs are referred to as smart contracts[43] and can be reviewed and executed by all participating nodes in the Ethereum network, making them a perfect tool for accountability.

Since the rise of Ethereum, all sorts of applications involving hundreds of different use cases of blockchains have sprung up[44], ranging from sharing files to distributed DNS services[45] to liquidity-providing automated exchanges[46].

First applications built on blockchains

Given how many industries are plagued by middlemen, it is reasonable to predict that blockchains will do for transactions what the internet did for information. It is arguably the most important networking innovation since the rise of the commercial internet in the early 1990s and the most significant evolution in cross-party transaction recording since double-entry bookkeeping was defined by Luca Pacioli in Italy in the XV century[47].

It is important to note that despite the proliferation of ideas involving blockchain technologies, the vast majority of them that are actual products and are already operational are still focused on financial transactions. The late adoption from fields outside of finance is largely due to the different nature of transactions, characterised by:

- Their sheer volume - most blockchains have serious deficiencies in transactions volume capacity[48].
- Their computability - financial transactions are binary, either successful or not, whereas other transactions might be more subjective and require more complex governance or the presence of verifying oracles supplying external information[49].
- Their usability - international money transfers were an obvious application for blockchain transactions that has easily attracted enough interest to sustain development, whereas other fields willing to adopt blockchain solutions might need more dedicated users in order to cope with the initial complexity of understanding the dynamics of the market and the intricacies of this new technology.

It is still early days for non-financial applications of blockchain technology. Ever since it launched, AdHash has proven to be a great example of a non-financial blockchain solution that positively impacts the experience of millions of internet users.

---

41 *"Ethereum White Paper", Vitalik Buterin, Ethereum Foundation, https://github.com/ethereum/wiki/wiki/White-Paper, 2013.*

42 *"The Solidity Contract-Oriented Programming Language", GitHub, https://github.com/ethereum/solidity.*

43 *"Smart Contract", Wikipedia, https://en.wikipedia.org/wiki/Smart_contract.*

44 *"100 Cryptocurrencies Described in Four Words or Less", TechCrunch, https://techcrunch.com/2017/11/19/100-cryptocurrencies-described-in-4-words-or-less/, 2017.*

45 *"NameCoin", https://namecoin.org.*

46 *"Bancor Protocol", Eyal Hertzog et al., https://about.bancor.network/static/bancor_protocol_whitepaper_en.pdf, 2017.*

47 *"Summa de Arithmetica, Geometrica, Proportioni et Proportionalita", Luca Paccioli, 1494.*

48 *"Bitcoin Scalability Problem", Wikipedia, https://en.wikipedia.org/wiki/Bitcoin_scalability_problem.*

49 *"Oracles: Bringing Data to the Blockchain", Jules Dourlens, https://ethereumdev.io/oracles-getting-data-inside-blockchain/, 2017.*

Building an advertising ecosystem on a blockchain

As mentioned in Chapter II, blockchain's value for the advertising industry lies in providing an immutable, universally-available record of unique ad hashes. Having such a record would help marketers, publishers, and users alike to prevent bad actors from having access to advertising space. It would allow the ecosystem to self-regulate on commonly agreed-upon rules. If such a system had been in place, there would be no doubt about the source of the controversial ads in the 2016 US presidential elections that are believed to have had at least some effect on the end result[50],[51]. It would be technically impossible for an advertiser to run Javascript malware on the computers of millions of users through the websites of unsuspecting publishers[52],[53]. And there would be no disturbances to end users who are tired of seeing annoying ads without being able to block them, which unfortunately is the norm in today's ecosystem.

Below we will present our solution to the blockchain scalability problem within the context of RTB advertising. Our product is built on top of the Ethereum blockchain. While it would be easier to build a new blockchain than dealing with the limitations of an existing one, our team decided that the network effect, strong ecosystem, the various promising scaling roadmaps that Ethereum has presented[54], and the ERC20[55] token's easy integration with crypto wallets and exchanges far outweigh the negatives. This decision is reversible in the off-chance that Ethereum does not deliver on its technological promises, fails to move to a proof of stake protocol[56], or starts losing relevancy and support in the future.

While blockchain technology has revolutionised accountability, security, and trust, it is still relatively limited in its capacity for storing large volumes of transactions. The volume of RTB transactions in digital advertising is so large (5M+ transactions per second, each transaction could contain dozens of data points) that storing that information becomes computationally and financially impossible. An obvious first step in overcoming this limitation is to only store the most valuable information on the blockchain and find an alternative way to store the rest.

On-chain transactions

AdHash stores on the blockchain only singular bulk transactions for each participant in the ecosystem, the unique ad hashes, and the voting results that control the blocklists.
Funds are not exchanged every time an ad is clicked and the bidder only syncs the transactions once every day (we will call this the bidder period T, as it can be configured to different values). We have set T = 24h, which would mean that the blockchain will only need to record a total number of N transactions per day where:

[50] *"Google Uncovers Russian-Bought Ads on YouTube, Gmail and Other Platforms", The Washington Post, https://www.washingtonpost.com/news/the-switch/wp/2017/10/09/google-uncovers-russian-bought-ads-on-youtube-gmail-and-other-platforms/, 2017.*

[51] *"Facebook Criticised for 'Not Answering Questions' on Russia Ads", The Financial Times, https://www.ft.com/content/3bc945a2-e026-11e7-a8a4-0a1e63a52f9c, 2017.*

[52] *"The Chameleon Botnet", spider.io, http://www.spider.io/blog/2013/03/chameleon-botnet, 2013.*

[53] *"HummingBad: A Persistent Mobile Chain Attack", Check Point, https://blog.checkpoint.com/2016/02/04/hummingbad-a-persistent-mobile-chain-attack/, 2016.*

[54] *"Ethereum: Platform Review", r3, http://www.r3cev.com/blog/2016/6/2/ethereum-platform-review.*

[55] *"ERC20 Token Standard", The Ethereum Wiki, https://theethereum.wiki/w/index.php/ERC20_Token_Standard.*

[56] *"Proof of Stake FAQ", Ethereum GitHub, https://github.com/ethereum/wiki/wiki/Proof-of-Stake-FAQ.*

N = transacting advertisers + transacting publishers

This would reduce the number of required on-chain transactions from millions per second to just thousands or tens of thousands per day. This is a reduction in transaction volume of approximately seven orders of magnitude compared to recording all RTB transactions on a blockchain. An important additional benefit is the enhanced privacy of the data of all parties involved. Advertisers and publishers will save all private bidding data on their own servers and will be able to explore it rapidly and reliably through the Server-Side Platform described in the previous chapter.

Off-chain transactions

So far we have described how the blockchain achieves a generalised record of all payment transactions and the Server-Side Platform contains a very detailed dataset of all advertising measurements. What is missing is a third record connecting the two and containing the breakdown of individual payments for each verified click between a publisher and an advertiser. We call this type of records “off-chain transactions” because they are stored on the Server-Side Platform and not on the blockchain. However, they are regularly synced with the on-chain records for verification.

An important security notice: bulk transactions can be tracked by anyone on the blockchain. However, detailed information about those transactions, such as advertisers’ patterns, bids, numbers of clicks, and targeting criteria will remain securely stored in the Server-Side Platform of each client.

This approach in its essence is very similar to the micropayment channels proposed by the Lightning Network protocol[57], which aims to store just the bare minimum of information on the blockchain and create relationships between two parties to perpetually update balances in a secure and private manner.

Therefore, each time an ad is clicked, both the publisher and the advertiser record the event with a timestamp and the precise amount due. The bidder is also keeping this history as an independent third party in the event that dispute resolution is needed. Here is a visual representation of this principle:

[57] *“The Bitcoin Lightning Network: Scalable Off-Chain Instant Payments”, Joseph Poon and Thaddeus Dryja, https://lightning.network/lightning-network-paper.pdf, 2016.*

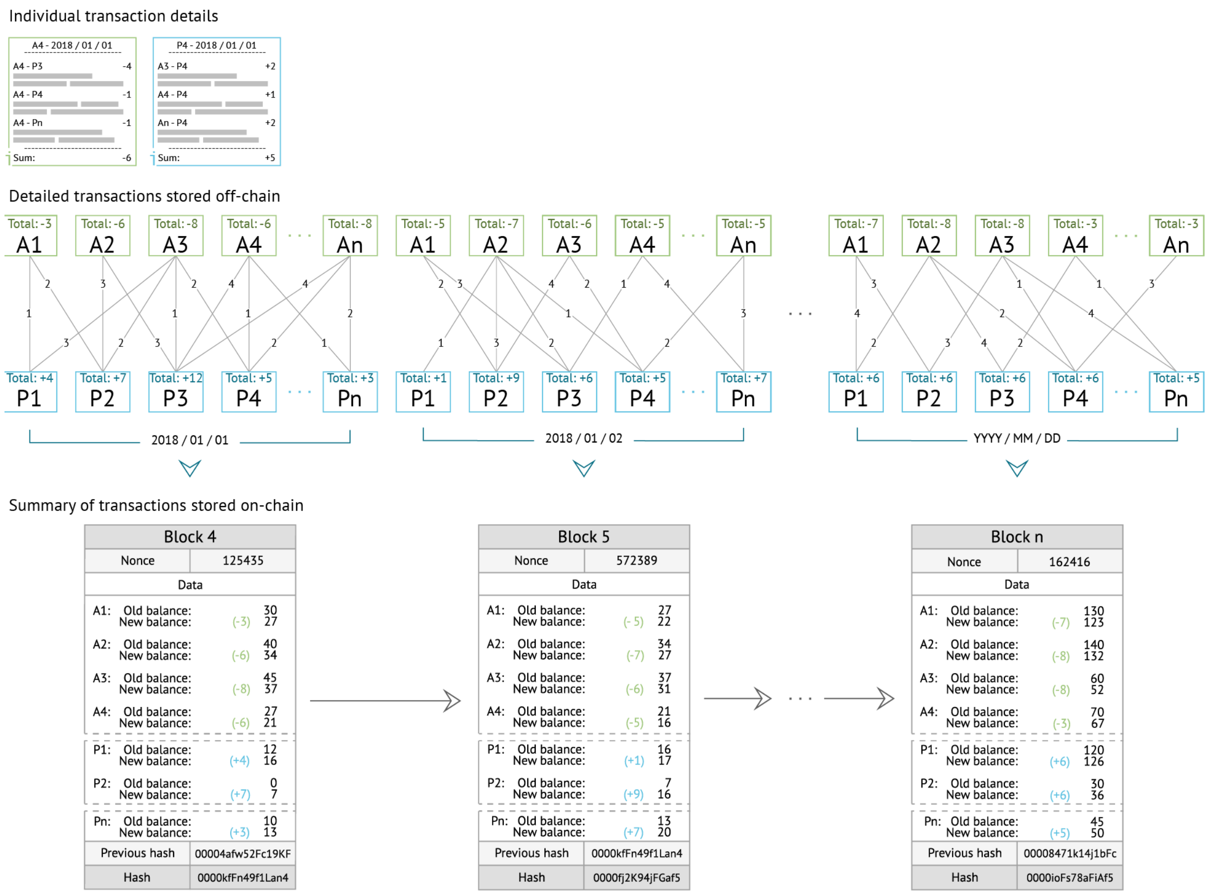

Individual transaction details
A4 - 2018 / 01 / 01
A4 - P3 -4
A4 - P4 -1
A4 - Pn -1
Sum: -6
P4 - 2018 / 01 / 01
A3 - P4 +2
A4 - P4 +1
An - P4 +2
Sum: +5
Detailed transactions stored off-chain
Total: -3 A1
Total: -6 A2
Total: -8 A3
Total: -6 A4
Total: -8 An
Total: +4 P1
Total: +7 P2
Total: +12 P3
Total: +5 P4
Total: +3 Pn
2018 / 01 / 01
Total: -5 A1
Total: -7 A2
Total: -6 A3
Total: -5 A4
Total: -5 An
Total: +1 P1
Total: +9 P2
Total: +6 P3
Total: +5 P4
Total: +7 Pn
2018 / 01 / 02
Total: -7 A1
Total: -8 A2
Total: -8 A3
Total: -3 A4
Total: -3 An
Total: +6 P1
Total: +6 P2
Total: +6 P3
Total: +6 P4
Total: +5 Pn
YYYY / MM / DD
Summary of transactions stored on-chain
Block 4
Nonce 125435
Data
A1: Old balance: 30 New balance: (-3) 27
A2: Old balance: 40 New balance: (-6) 34
A3: Old balance: 45 New balance: (-8) 37
A4: Old balance: 27 New balance: (-6) 21
P1: Old balance: 12 New balance: (+4) 16
P2: Old balance: 0 New balance: (+7) 7
Pn: Old balance: 10 New balance: (+3) 13
Previous hash 00004afw52Fc19KF
Hash 0000kfFn49f1Lan4
Block 5
Nonce 572389
Data
A1: Old balance: 27 New balance: (-5) 22
A2: Old balance: 34 New balance: (-7) 27
A3: Old balance: 37 New balance: (-6) 31
A4: Old balance: 21 New balance: (-5) 16
P1: Old balance: 16 New balance: (+1) 17
P2: Old balance: 7 New balance: (+9) 16
Pn: Old balance: 13 New balance: (+7) 20
Previous hash 0000kfFn49f1Lan4
Hash 0000fj2K94jFGaf5
Block n
Nonce 162416
Data
A1: Old balance: 130 New balance: (-7) 123
A2: Old balance: 140 New balance: (-8) 132
A3: Old balance: 60 New balance: (-8) 52
A4: Old balance: 70 New balance: (-3) 67
P1: Old balance: 120 New balance: (+6) 126
P2: Old balance: 30 New balance: (+6) 36
Pn: Old balance: 45 New balance: (+5) 50
Previous hash 00008471k14j1bFc
Hash 0000ioFs78aFiAf5

Fig. 13: On-chain and off-chain transactions operating in sync.

When, at period T, the transaction history is synced with the blockchain, each party's Server-Side Platform verifies whether the total amount transferred corresponds to the expected amount from their off-chain record. If it does, no action is taken. If it does not, a verification request is issued to all parties with whom transactions were made during that period.
We present a simplified code showing how that verification request works in practice:

```
/*
When payment is due, a response object is generated that contains the period
identifier.
Then a SHA1 checksum is generated for that object.
getCurrentPeriod() returns the current period identifier.
*/
var currentPeriod = getCurrentPeriod();
var response = {
    period: currentPeriod,
    checksums: {}
};

/*
The report data are measured by the Server-Side Platform and stored in its
database.
getReportData() returns the measured data as a table (example shown on next page).
*/
var reportData = getReportData(currentPeriod);

/*
For each row in the response, a SHA1 checksum of the JSON data is generated.
For example, a record of {"clicks":212,"price":116.48,"commission":3.66} would
return e72573df985ac28e41276e438d184f08e204cdcf.
*/
for (var key in reportData) {
    response.checksums[key] = SHA1(JSON.stringify(reportData[key]));
}

/*
In order to make the report verifiable by others, it has to be signed once more
with a checksum of all the combined transactions from the rows above. If any of
them are changed they will result in a different checksum.
*/
response.checksum = SHA1(JSON.stringify(response));
response.report = {};

/*
At this step, each row is encrypted with the public key of the counterparty it
belongs to.
Doing so ensures that only that counterparty will be able to decrypt it using its
private key.
encryptDataWithCertificate() returns a string encrypted with the public key of the
counterparty.
*/
for (var key in reportData) {
    response.report[key] = encryptDataWithCertificate(reportData[key], key);
}

/*
If the purpose of the response was to verify off-chain transactions then this was
the last step.
The response can be validated without the private keys by checking the checksums.
*/
console.log(response);

/*
If, however, the purpose of the response was to put a payment on the blockchain,
then an additional step is needed. The total amount due is the sum of all publisher
payments.
*/
var amount = 0;
for (var key in reportData) {
    amount += reportData[key].commission;
    amount += reportData[key].price;
}

/*
The amount calculated above is used to issue a payment to the bidder's wallet ID,
containing both the publishers' earnings and the bidder's commission.
makePayment() takes a wallet ID, an amount, and uses the combined checksum as a
reason for payment.
Once this checksum is announced to the blockchain, the report is permanent.
Any future change in the report will result in a SHA1 checksum that is different
from the one stored on the blockchain which can easily be verified by all
interested parties.
*/
var bidderWallet = getBidderWalletID();
makePayment(bidderWallet, amount, response.checksum);
```

getReportData() example output:

```
{
	"0x61f2d8c7003045D589D93E0e2Ab932E6121239a3": {
		"clicks": 102,
		"price": 52.91,
		"commission": 1.5873
	},
	"0x401715FB0728dcB8aD80a775F4446A32fd69a1eD": {
		"clicks": 212,
		"price": 116.48,
		"commission": 3.4944
	}
}
```

For the off-chain verification process to work, every advertiser and publisher provides a public node address when registering. It facilitates bidirectional communication between transacting parties, allowing advertisers and publishers to connect at any time and request each-other's metrics for verification. Each participant can update their public node address if necessary. If the link is unresponsive, the participant gets temporarily suspended from the network and risks his or her deposit.

Smart contracts and a native digital currency

Smart contracts allow us to create a whole new level of market dynamic by rewarding publishers for providing verified traffic to the ecosystem and voters for securing it. A rather limited analogy would be to compare them to Bitcoin miners. While they are rewarded for securing the ledger by spending computational resources, publishers in our system are rewarded for their traffic, and voters for their time and cognitive resources.

Building a rigid incentive system required a native digital currency. The AD token creates distinct benefits for our ecosystem in some important areas:

1. **Transactions** - AD tokens would enable micropayments between all members of the ecosystem. The AdHash Bidder is initially set up to issue payments every 24 hours, reducing the current reconciliation period by up to two orders of magnitude. Transaction costs would also be minimised compared to traditional bank transfers or credit card payments.
2. **Stakes** - as described in Chapter III of this paper, our ecosystem relies on self-governance and regulation from our voting community. To enable a functional voting system, AD token deposits are made and enforced through smart contracts.

Could the problems we have set out to fix be solved without a blockchain?

The goals of removing intermediaries by allowing self-hosted ads and removing inefficiencies through the introduction of unique ad hashes become so obvious when one looks at the messy advertising ecosystem of today that it is almost hard to believe that no one has built a solution yet. Current ad platforms have IDs assigned to their creatives that do not actually match with those of their partners and competitors, resulting in unavoidable double-bidding inefficiencies. Supply-side platforms actually benefit from this problem because they extract more money from unsuspecting advertisers. This is why it is impossible to rely on ad tech companies guarding their data to self-regulate against themselves. The only way forward is to make sure that they are all working with the same incorruptible IDs and to make them public. This is what the blockchain has been designed for.

The last two chapters have described in detail how all data are stored in AdHash's framework. It consists of three separate layers:

1. the server-side private databases containing all detailed advertising data;
2. the on-chain transactions containing bulk payments;
3. the off-chain transactions containing detailed financial transaction records.

However, merely introducing a clever combination of distributed databases could not by itself resolve all fraud and disputes. Advertisers could still refuse to issue payments and publishers could still attempt different ad fraud attack vectors. Our framework guarantees that all parties have incentives in the form of deposits to keep their records accurate and available upon request.

VI Voting System

Purpose

Currently, different advertising platforms have different and often completely random policies regarding creatives, formats, content, and various other aspects of advertising. We are firm believers in the fact that some rules are necessary to make sure that the experience of participating in the ecosystem matches everyone's expectations. What we do not agree with is the centralised manner in which those rules are set. Who decides whether an ad can have more or less text in it? Or contain specific words? Those decisions cannot remain in the hands of biased private policy makers who can discriminate against entire industries[58],[59].

AdHash proposes a solution in which all rules are subject to periodic evaluations by a voting community. This would enable decentralised self-regulation of the ecosystem. As the third and final necessary requirement to bring our product vision to life, this system plays a number of critical roles:

- Bolsters the security by allowing participants to proactively identify and block abusers.
- Eliminates the inefficiencies in the creative approval and ad quality control processes.
- Facilitates decentralised self-regulation, enabling the network to be self-sustaining.
- Provides a democratic framework for dispute resolution.
- Gives participants equal opportunity to contribute and earn rewards through AD token minting proportional to the value they add to the network.
- Democratises power distribution and token distribution.

Our product lays the architecture for a radically different advertising marketplace. The fundamental mechanisms have been carefully thought through and designed to address the inefficiencies of the current advertising space. This being said, it is by no means a complete or perfect solution. It is a solid and functional framework which will continuously mature and adjust to the needs of all participants.

Our long-term vision for the voting community is to find great synergy between the needs of the advertising market to have high-quality human curation at all times and the opportunity for voters to monetise their efforts. Advertising companies are presently wasting hundreds of millions of dollars a year employing tens of thousands of workers[60] in the US alone. We believe that by creating a more secure and reliable tech infrastructure, we can drastically reduce this number and redistribute those budgets in the hands of all interested users participating in our ecosystem as voters.

Voters

Advertisers, publishers, and users can all opt-in to become voters. The rewards they earn from participating in the voting system will be transferred to their wallets biweekly. The funds they accumulate during this period serve as their stakes in the rewards mechanism. Anyone in the ecosystem can call a vote in one of two ways:

---

[58] *"New Ads Policy: Improving Integrity and Security of Financial Product and Services Ads", Facebook Business, https://www.facebook.com/business/news/new-ads-policy-improving-integrity-and-security-of-financial-product-and-services-ads, 2018.*

[59] *"Google Will Ban All Cryptocurrency-Related Advertising", CNBC, https://www.cnbc.com/2018/03/13/google-bans-crypto-ads.html, 2018.*

[60] *"Inside Facebook's Fast-Growing Content-Moderation Effort", The Atlantic, https://www.theatlantic.com/technology/archive/2018/02/what-facebook-told-insiders-about-how-it-moderates-posts/552632/, 2018.*

1. By submitting a vote directly, in which case a deposit is required. This is done so that the time and effort of the voting community is not abused.
2. By flagging a creative or a participant. If enough flags are received within a time period, the flagged party or creative is sent to a vote. No deposit is required in this scenario.

As soon as a vote is called, elected voters receive an email request containing a link. The link leads to the AdHash Voting Platform where the voting process takes place. Everyone has thirty minutes to act before the link expires. This facilitates quick decision-making and conflict resolution. To ensure sufficient participation in the event of abstentions or absences from some of the elected voters, 75 emails are sent out but only the first 50 votes from each voting pool are counted towards the final decision (the link becomes inactive after 50 votes have been submitted). Those 50 voters are randomly selected based on their voting ranking (an important concept explained below). When the voting community matures and grows, additional filters (by time zone, languages, interests, etc.) will be introduced.
The decisions made by the voters take immediate effect and therefore have to demonstrate a sufficient level of certainty. This is why, for any proposal to be approved, a supermajority of two thirds (equivalent to 34 votes) needs to be in favour. The final decision is perpetually stored on the blockchain and everyone participating in the voting pool receives an email with the results of the vote.

There are many different proposals for distributed governance and voting. We have designed this one to be as simple as possible while maintaining strong resistance to most game-theory attacks that we could explore. It was therefore encouraging to subsequently discover that another team had developed successfully a very similar voting system, called the Tendermint Consensus Protocol[61]. It also expects a supermajority of voters to agree on a decision, tracks voting power (similar to our ranking), and performs voting in phases until the necessary number of validators is reached. Their number of validators is slightly higher than ours (ranging from 100 to 300), but ours can be adapted if necessary through scheduled ecosystem governance votes, as can be all other parameters in the system.

Ranking

We elect voters based on a resource that nobody can monopolise - their voting rank. The rank determines the likelihood of being selected to vote each time. A certain degree of randomisation is necessary to provide opportunities for everyone to participate, while also making the system increasingly difficult to manipulate as the network expands. The ranking system, on the other hand, allows participation based on merit, driving higher operational efficiency.

All votes in the AdHash Voting System are equal, regardless of the rank or stake of the voter. However, a higher rank would increase the likelihood of being elected to vote. Ranks are scaled by a point system whereby points are added or subtracted based on a number of factors. Each rank can fluctuate between 0 and 100. Each newly registered voter starts with an initial rank of 25 points. The maximum achievable rank is 100.

Min
Initial
Max
0
25
100

The voting process is binary as voters can select one answer from two possible: for or against. Therefore, a vote can either be right or wrong. A vote is considered correct when it

[61] *"Cosmos - A Network of Distributed Ledgers", Jae Kwon and Ethan Buchman, https://cosmos.network/about/whitepaper.*

represents the final verdict (agreeing with the supermajority), and incorrect when it does not. This is how a voter's rank is calculated, based on the accuracy of votes:

```
var correctAnswers = getCorrectAnswers();
var wrongAnswers = getWrongAnswers();
var currentVotePoints = getCurrentVotePoints();

if (correctAnswers == 0 && wrongAnswers == 0) {
    currentVotePoints--;
} else {
    currentVotePoints += correctAnswers - (wrongAnswers * 5);
}
```

The rule mandates that a penalty to the rank for an incorrect vote is five times larger than the reward for a correct vote. The ranking cap ensures that even the best ranked voters are only four times more likely to be selected than a newly registered voter. This diversifies participation and prevents excessive concentration of voting activity. Successful voters whose ranks reach 100 start to tokenise their ranking points into AD tokens. For every additional point beyond 100, the voter receives the equivalent amount of tokens a publisher mints per click (this 1:1 ratio can be adjusted during the governance votes). On the other hand, voters who allow their rank to fall to zero are perpetually disqualified from the voting system. Ranks for any account can be checked in real-time on adhash.com.

In summary, the ranking system encourages and rewards participation based on merit. Those contributing in a meaningful way are given more opportunities to do so. Those who are not will gradually lose rank and eventually get disqualified. Voters are rewarded purely on the amount of work done, not on the amount of tokens they own.

Deposits

There are three types of deposits in the AdHash ecosystem and they are funded with AD tokens. Advertisers submit them to join the network and risk losing them if the voting community decides to ban them. The size of their deposits defines their maximum allowed daily ad spend which is equal to 20% of the deposited amount. Publishers, on the other hand, are not required to submit deposits. Instead, their earnings from minting (paid once every 14 days) and advertisers' payments (paid every day) are their collateral. Publishers and advertisers also stand to lose their deposits should they break the rules of keeping all off-chain transaction records on their Server-Side Platforms available for verification for at least 60 days, or if they violate any other terms.
The third kind of deposits applies when a participant wishes to engage the voting system to block a certain creative from the network. In this case, a collateral is required to prevent rampant and unnecessary block requests. If the block request is not approved by the voting community, it is considered unnecessary and the vote requester loses the deposit submitted for the vote. The confiscated funds enter the Rewards Pool.

Rewards Pool

The purpose of issuing rewards is to incentivise the voting community to perform the task of governing the quality of the ecosystem. Vote initiators and voting pools perform crucial functions within the network and should be rewarded for their contribution to the network's security and quality of the ads. Token ownership provides a financial incentive for them to keep the network healthy and foster its growth.

All voters act in a way that would maximise their own potential payoff. There is a short-term incentive to vote correctly to earn rewards, and a long-term incentive to vote in a way that increases the value of the ecosystem. If the protocol and incentives are designed well, then both would overlap and most voters would vote in the best interest of the network most of the time. This is a Nash equilibrium[62], a concept of game theory where the optimal outcome of a game is one where no player has an incentive to deviate from a chosen strategy after considering all opponents' strategies. In other words, no voter should be able to realise higher profit in the long term by deviating from the optimum strategy for the entire community.
Presumably, there will always be voters trying to game the system and this is precisely why voting rewards are held for two weeks before being issued, meaning that offending actors stand to lose significant earnings if they are caught cheating.

The Rewards Pool for voters minimises the risk of a concentration of token ownership in the hands of individuals or whole categories of participants (not only publishers will be able to mint new tokens, advertisers and users will also have that opportunity). A voter's reward is calculated by dividing the total amount of funds accumulated in the Rewards Pool by the total number of correct votes during a period of 14 days (from day i to day i+13) and then

$$\text{voting reward} = \frac{\sum_{d=i}^{i+13} \text{rewards pool}}{\sum_{d=i}^{i+13} \text{all correct votes}} * \text{correct votes}$$

multiplying it by the number of correct votes of the voter:
Voters are rewarded purely on the amount of honest work done, so the more correct votes they submit, the more they earn.

Those who initiate a vote and whose proposal is approved by the voting community are also rewarded for successfully identifying and eliminating problems. Their reward is equivalent to ten times the current amount earned by a publisher per click.

Fraud detection

Combatting the sophisticated and ever-evolving ad fraud techniques requires human intelligence and serious investigative work on top of algorithmic detection. That is why we intend to build a rewards mechanism based on proving ad fraud. Those votes would be more complex, require specialised technical voting pools and more time for assessment. But the end results would allow ad fraud investigators to submit reports of detected fraud and if confirmed, they would be rewarded not only by minting a significant reward, but also by receiving a percentage of the offender's confiscated deposit. In this regard, the logic behind our fraud detection mechanism is very similar to that of the *fishermen* in the Polkadot framework[63]. *Fishermen* also get their rewards through a timely proof that a party in the ecosystem acted illegally. And they are also required to leave deposits, preventing them from launching attacks wasting the time of community members.

62 *"Equilibrium Points in N-Person Games", Proceedings of the National Academy of Sciences of the United States of America, http://www.pnas.org/content/36/1/48, 1950.*

63 *"Polkadot", Dr Gavin Wood, https://github.com/w3f/polkadot-white-paper/raw/master/PolkaDotPaper.pdf, 2017.*

Creative approval

In today's advertising platforms, all submitted creatives are subject to independent approval by each individual ad exchange. Across the exchanges, the approval process could take from a few hours to over a week. It often leads to delays in campaign launches and is extremely inefficient. Thousands of employees are manually reviewing every single creative[64], yet many of the ads that pass their filters are still breaching their policies or are perceived as inappropriate, intrusive, or irritating by the users. The ever-increasing popularity of ad blockers can attest to that.

The AdHash Voting System eliminates this wasteful use of human effort. It is designed to allocate resources only where necessary, concentrating its human capital on identifying and removing the abusers, rather than wasting time on proactively checking all users. It can also offer more visibility into why certain creatives are banned and allow advertisers to get more opportunities for revision and improvement, which enables a faster learning curve.

The ad creative blocking mechanism is reactive in terms of involving the voting community, since any creative can be served over the network without going through human approval until it is flagged. However, that does not mean that there is no proactive mechanism in place. An evolving combination of vision APIs[65],[66],[67] is used to proactively check all creatives and warn against unacceptable images so that voters can review them.

AdHash Quality Control

The current AdChoices[68] solution has left people disenchanted with the lack of control they have over the ads they see. Not only does it provide insufficient information about why a specific ad has been targeted to a particular person, but it simply does not serve its intended function - that of removing undesirable ads from one's advertising experience.

The AdHash Voting System, thanks to the introduction of unique ad hashes, provides users with immediate and effective control over their ad exposure. The AdHash Quality Control icon in the upper right corner of each creative features an intuitive menu:

---

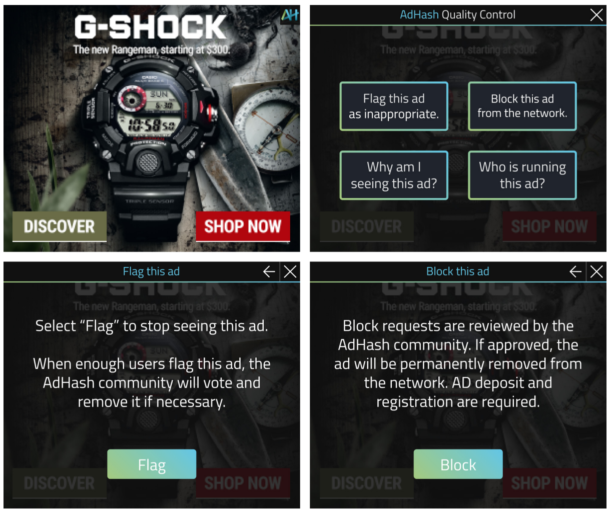
64 "Facebook to Add 1,000 Hires to Ad Review Process", Seeking Alpha, https://seekingalpha.com/news/3298739-facebook-add-1000-hires-ad-review-process?uprof=25#email_link, 2017.

65 Clarifai, https://www.clarifai.com.

66 Microsoft Azure Computer Vision API, https://azure.microsoft.com/en-us/services/cognitive-services/computer-vision/.

67 IBM Visual Recognition API, https://www.ibm.com/watson/services/visual-recognition/.

68 "YourAdChoices", the Digital Advertising Alliance, http://youradchoices.com.



Select "Flag" to stop seeing this ad.

When enough users flag this ad, the AdHash community will vote and remove it if necessary.

Block requests are reviewed by the AdHash community. If approved, the ad will be permanently removed from the network. AD deposit and registration are required.

Fig. 14: The AdHash Quality Control interface is accessible by clicking on the icon in the top right corner of any creative. This allows regular users to flag any ads they find inappropriate (no account needed), and users with wallet IDs to request for such ads to be immediately blocked (AD deposit and registration are required). The interface also allows users to find out which company is paying for an ad and what other ads it is running.

Flagging

Users can remove creatives they consider inappropriate from their personal advertising experience by flagging them. Flagging happens on a user level, not network-wide, and is equivalent to blacklisting since it does not impact the advertising experience of other users. The system relies on local storage and ad hashes to remember each user's flag requests. We could implement a more vigorous method of recording user preferences but it would be more invasive of user's privacy which goes against our philosophy regarding user data. Frequent flagging of the same creative may signal a potential problem on a network level. If an ad accumulates 10 flags within one hour, it automatically triggers a vote. The creative is temporarily suspended from serving while the voting pool determines if the ad needs to be blocked network-wide. Since the vote is initiated automatically and anonymously, no deposit is required.

The high bar that AdHash uses to choose its advertisers means that high rates of flagging rarely occur. In the first months after publicly launching, only about one in a million ad impressions resulted in a flag initiated by a user.

Blocking

Instead of relying on a sufficient number of flags to accumulate, participants can directly submit requests to block inappropriate creatives from the network. If the request is approved, the creative is removed with immediate and permanent effect. To submit a block request, one needs a wallet ID with sufficient amount of AD tokens to cover the deposit. Deposits are required to eliminate the risk of rampant block requests and predatory blocking of competitors.

Once a request is submitted, the creative is temporarily suspended until the voters make a decision. If two-thirds of the voters approve the request, the ad is permanently removed from the network and the advertiser's deposit is confiscated. The outcome of the vote is recorded on the blockchain to prevent the blocked creative from re-entering the system. Since all creatives have unique hashes, the same creative could never be served again by any advertiser.

If the request is rejected, the temporary suspension of the creative is lifted and the campaign continues to operate as planned. The deposit of the vote requester is confiscated and enters the Rewards Pool.

Reversibility

Banned accounts can submit a deposit to appeal the decision and request a revote. The case is reexamined and if the issues which triggered the ban in the first place are not fixed, in all likelihood the second vote would confirm the results of the first one and the appealing party would lose yet another deposit. This system is designed to dissuade banned participants from appealing before fully addressing the issues they were suspended for.

Banned creatives can also be appealed following the same procedure - advertisers can call for revotes by depositing a collateral.

Possible attack vectors

Since flagging is free and voters are rewarded for each correct vote, they may start flagging every ad and every account they see in an attempt to earn more rewards by creating more voting opportunities. Because the Rewards Pool is limited and is distributed among all correct voters, an explosion of voting requests would only reduce the size of the reward per correct vote. Abusers will have nothing to gain from such an attack as the size of their reward per vote will become smaller. Abusing the system and praying on the deposits of good actors would also diminish the trust of those good actors and would make them less likely to ask for a revote, effectively slicing the potential payout for the voters.

Since 10 flags over a period of one hour would automatically trigger a block request, advertisers may start flagging competitors' creatives to reduce their competition. However, because the voting pool is diversified and incentivised to vote in the best interest of the network, the approval of any unsubstantiated block requests is highly unlikely.

To minimise any disruptions from malicious and unfounded flagging, each time a creative block request triggered by flagging is rejected by the voters, the flag threshold for that particular creative is increased by 10. The next time the same creative would need to accumulate 20 flags in one hour before triggering a vote request, 30 the time after that, and so on. This makes it linearly more difficult for abusers to impact a good advertiser.

We believe that a perfect consensus mechanism in a state of equilibrium will rarely need to perform conflict resolution as there would be no conflicts to resolve. If the protocol and incentives are designed well, then most participants will follow the rules at all times and act in the best interest of the entire ecosystem. Such a mechanism can only be achieved by continuous refinement. Until then, the voting system will ensure that any potential abuse is not so rampant as to undermine the incentive to do honest work in support of the network and its digital currency. The proof of stake mechanism governing the ecosystem is designed to tolerate short-term abuses by incentivising the voting community to actively work on eliminating the offenders and thus preventing any long-term damages.

This concludes our description of the three main components of our ecosystem: the Server-Side Platform, the on-chain and off-chain transaction storage, and the voting community. The following chapter will describe in detail the AdHash Protocol allowing all of those components to work in perfect sync by complementing and verifying one-another at all times.

VII The AdHash Protocol

Overview

So far, we have described how different types of data are handled in our framework, including detailed advertising metrics, on-chain and off-chain transactions and how they are all vital for the operation of our ecosystem.

The protocol that enables the communication of these data between all parties and facilitates the decision-making can be divided into three separate temporal layers: a real-time bidding layer (responsible for ad trading), a synchronous layer (responsible for transactions and payments), and an asynchronous layer (mostly responsible for the voting system). In this chapter, we will describe how each one of them works and elaborate on the rules of storing information. We will also explain how user data are being analysed and communicated.

Let us start by diving head-first into the full complexity of our protocol by drawing the entire block diagram for all three layers it consists of:

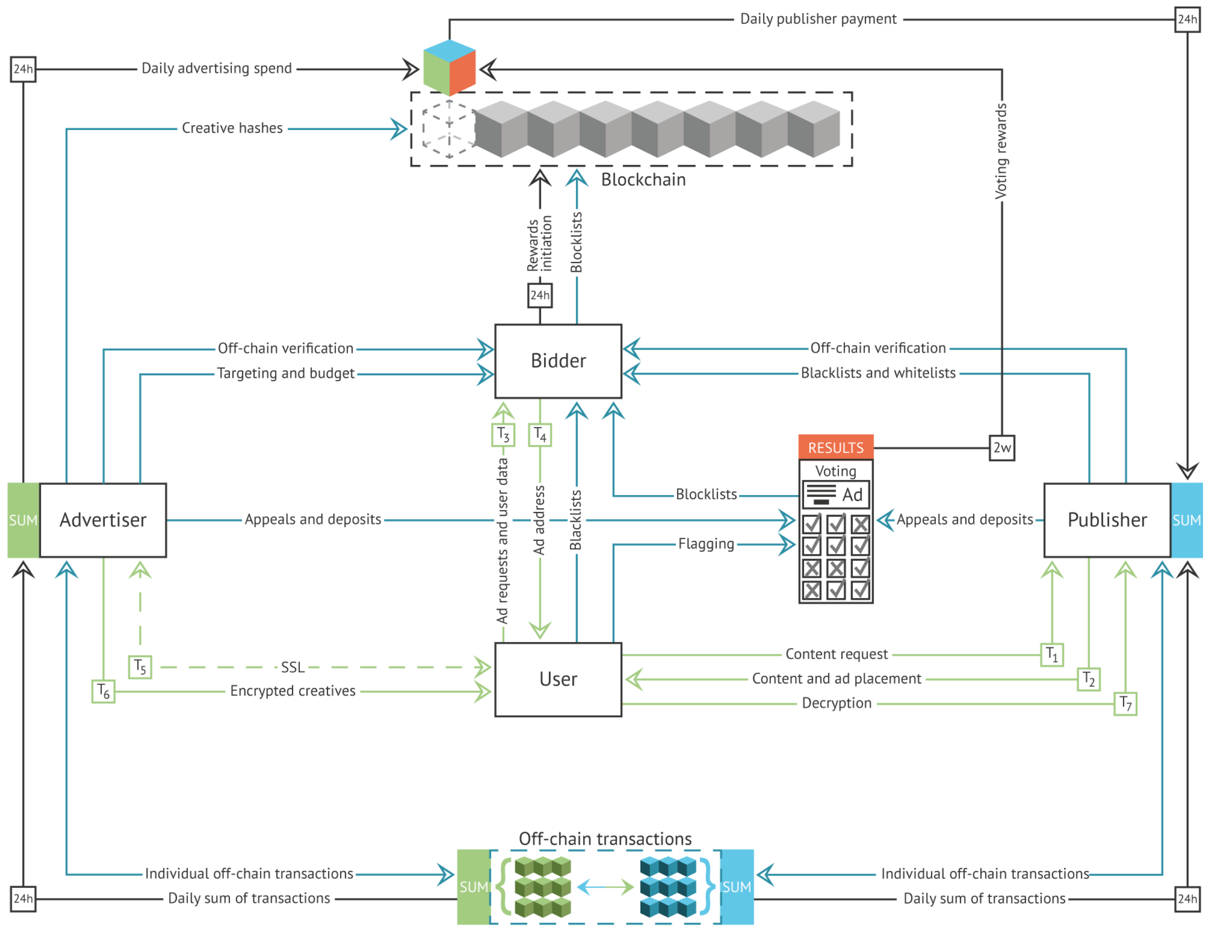


Fig. 15: This block diagram represents all aspects of the AdHash Protocol when just one bidder is present.

It may seem complex at first sight, but this includes all communications necessary to execute our vision. Dividing it into three separate layers should help the reader get a much clearer understanding of the way timing works in our protocol and would separate the three core components of our framework: the Server-Side Platform, on-chain and off-chain storage, and the voting community.

Let us start with the layer most familiar to anyone interested in digital advertising:

Real-Time Bidding layer

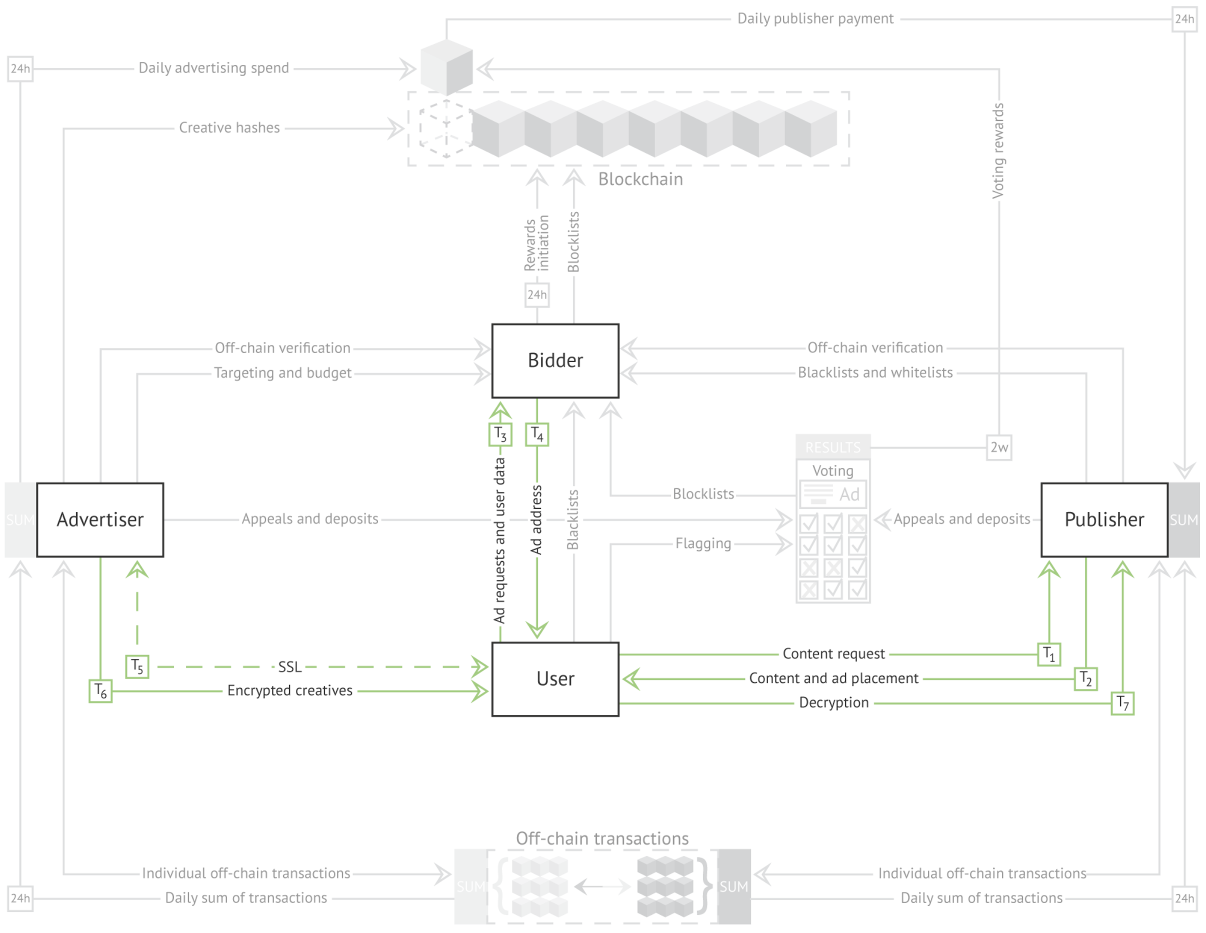


Fig. 16: The RTB layer of the AdHash Protocol includes all communications that are necessary to deliver an ad to a user.

We would like to remind the reader of Fig. 7 on page 12 and its stark contrast with the simplicity of our proposed solution. Instead of 30 distinct categories of participants, our RTB layer consists of just four categories: a user, a publisher, a bidder, and an advertiser (the final fifth one, the voting community, will be introduced on the next page).
The RTB layer of communication determines which ad will be shown to which user. It takes just over 100 milliseconds to execute and follows these seven steps:

T1. A user opens a webpage and requests content from the publisher.
T2. The content is served, along with an ad placement (still empty at this point).
T3. The user requests the ad address from the bidder.
T4. The bidder estimates the highest bid available from all advertisers and returns an address to the user.
T5. The user and advertiser establish a secure connection.
T6. The advertiser sends an encrypted ad. It can only be decrypted if the data provided by the user's browser match the advertiser's targeting criteria. If they do not, the ad will not be

clickable. Only after successfully passing through the advertiser's first-party data verification is the ad decrypted and a subsequent click can be considered verified.
T7. The publisher, the advertiser, and the bidder record in their Server-Side Platforms that an impression has been served and after successful decryption a click can be recorded too.

As already mentioned on page 25, the format of the ad requests exchanged between the Server-Side Platforms of publishers and advertisers and the bidder follow the OpenRTB 3.0 standard. This provides a low barrier to entry for integrations, as described in Chapter IX.

Synchronous layer

The synchronous layer of the AdHash Protocol is the one that governs all financial transactions in the system:

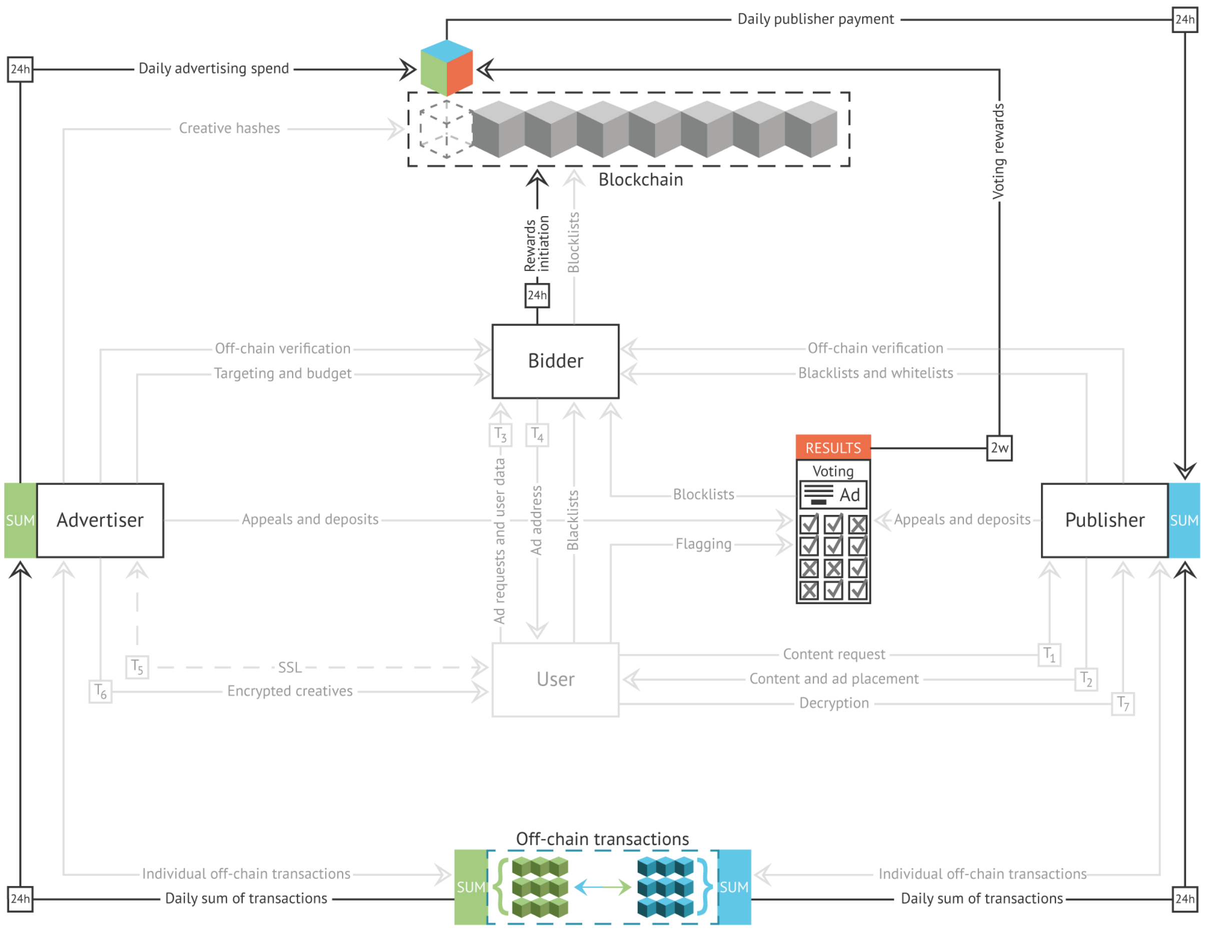


Fig. 17: The synchronous layer of the AdHash Protocol involves all communications that are scheduled at specific intervals, including payments and rewards relating to both ad trading and voting.

For a participant to pay or receive payment, that transaction has to be recorded on the blockchain. Those transactions happen at regular intervals T, hence the name synchronous. There are two types of them:

Daily:

- Advertisers pay their full daily spend through a single transaction.
- Publishers receive the full amount that they are due through a single transaction.
- If a publisher receives less than expected, off-chain transactions are verified.

Biweekly:
Voting and publisher rewards are held for two weeks to serve as a collateral. Once this period expires, they are recorded on the blockchain and therefore transferred to the voters and publishers.

Asynchronous layer

The asynchronous layer of the AdHash Protocol governs mainly the AdHash Voting System:

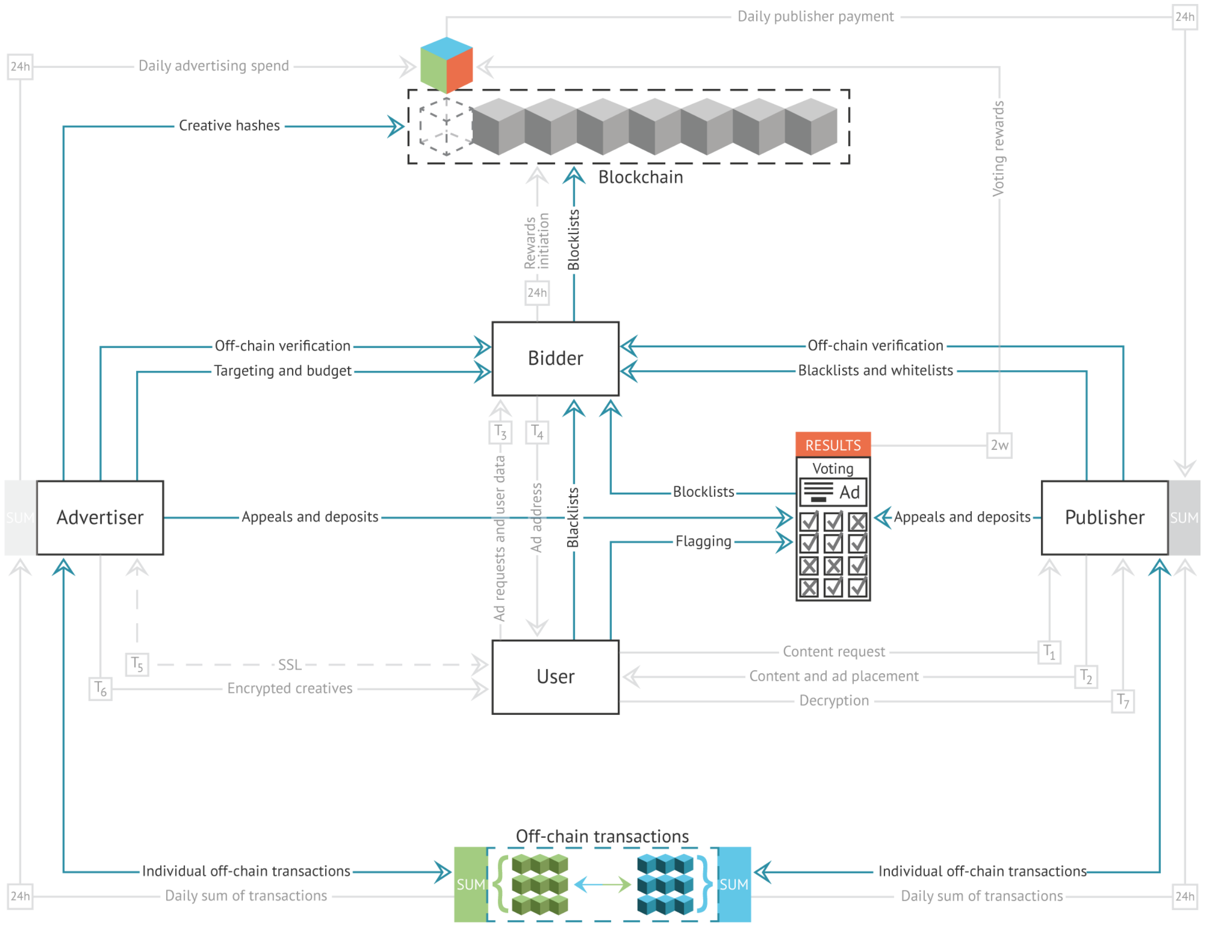


Fig. 18: The asynchronous layer of the AdHash Protocol consists of communications initiated by users or algorithms at non-deterministic moments. Some of those communications need to happen consecutively, others are completely independent.

It is initialised at random points in time (hence the name asynchronous) based on the users' interactions with ads, the decisions of the voting community, or specific events in the ecosystem. This layer includes all interactions that happen at arbitrary times and are not dependent on a schedule. There are seven types of such interactions in the ecosystem:

- All unique ad hashes are recorded on the blockchain.
- Advertisers upload their targeting criteria, budgets, bidding strategies, whitelists, and links to their creatives and landing pages to the bidder.
- Individual off-chain transactions are being recorded each time a user clicks on an ad.
- Off-chain transaction verification happens when disputes arise and the bidder is involved as an independent third party.
- Users, publishers, and advertisers can all create lists of counterparties that they do not wish to work with, as well as flag content that they do not approve of.

- Appeals and deposits are communicated between advertisers, publishers, and the voting system.
- Blocklists are updated when votes result in the removal of one or more parties from the ecosystem. The bidder records those decisions on the blockchain.

To summarise, we come back to the block diagram of the complete solution. This time third-party integrations or third-party bidders and trading desks are included as well:

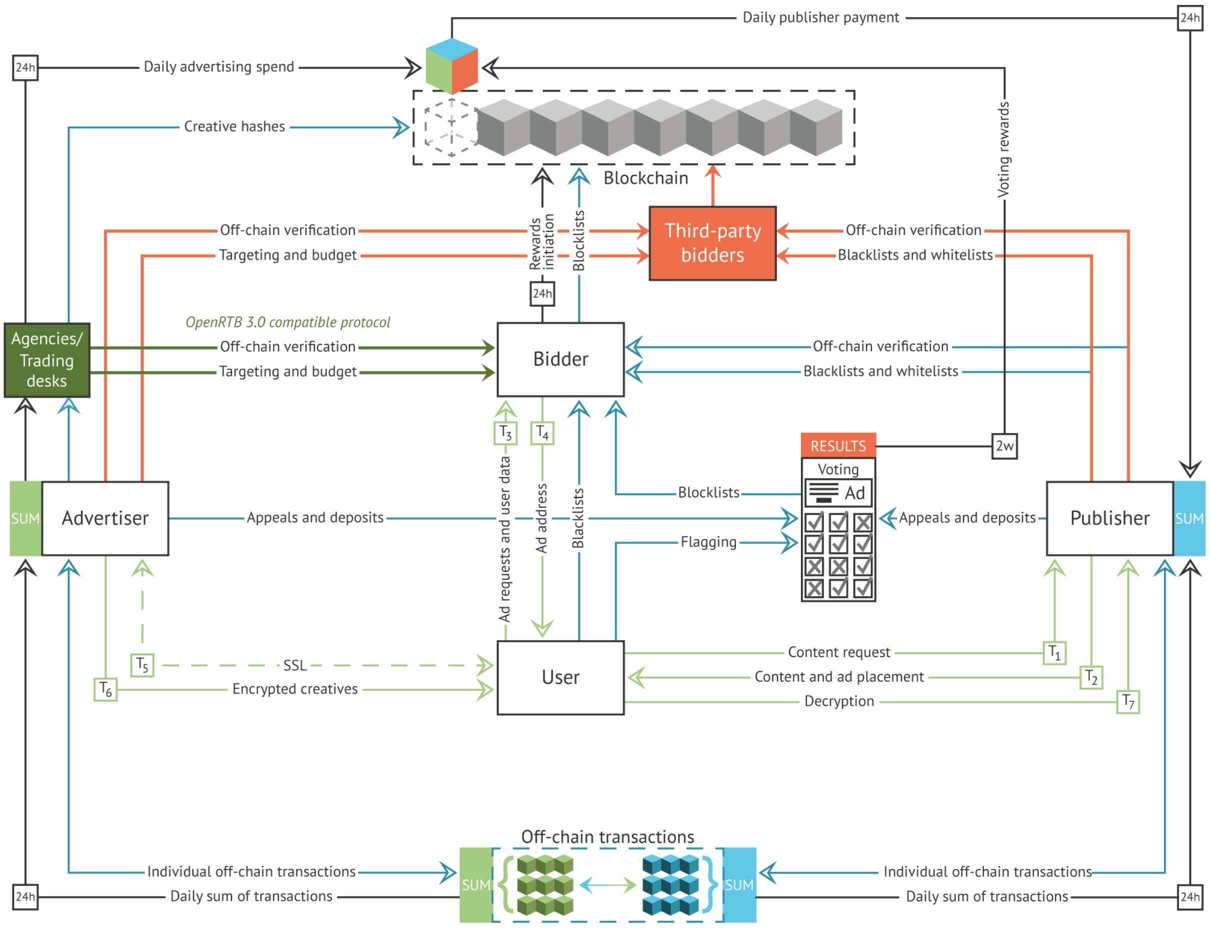


Fig. 19: Block diagram of the AdHash Protocol in its full intricacy, including the possible third-party bidder integrations which will be discussed in detail in Chapter IX.

Targeting users

After describing in detail how all communications between the participants in our framework operate, we need to elaborate on the way user data are communicated. This section will look at how user targeting is performed and how detailed it is. It is split into two subsections: the first refers to general web users and the second to mobile app users.

Web data exchange

At point T3 in Fig. 16, the user's browser sends the following information to the bidder:

- URL of the current webpage that can also be used for contextual targeting
- ID of the publisher
- Timezone
- Operating system name and version
- Browser version

- Device type and model (if available)
- Browser language
- Screen size
- Hashed IP addresses for geolocation (but targeting by IP is not allowed and IP storage can be disabled or IPs truncated)
- Keywords
- Content categories
- Contextual data about the content
- Real-world contextual data (financial, weather, sports, etc.)
- Local storage is used for ad preferences, flagging, frequency and recency capping

The advertiser validates whether the data provided by the user in the request header matches the targeting criteria sent to the bidder. This validation is secured by encrypting the landing page URLs of the creatives, which is executed in three steps:

- All data are written in JSON format and SHA1 hashes are generated from them.
- The landing pages' URLs are encrypted using AES-256. The SHA1 hashes from the step above are used as the encryption keys.
- Advertisers send their creatives with their encrypted URLs back to the users.

To produce a valid landing page URL, the user's machine has to transmit the exact same data the advertiser used to encrypt the creative. Should the ad land on a different URL or if something in the environment does not match, its landing page will not be decrypted and the ad itself will not be clickable. The advertiser will not be charged in such instances.

Mobile SDKs data exchange
The data accessible through our mobile SDKs include all of the information available in the web data exchange, as well as the following additional fields:

- Advertising IDs of the devices instead of local storage. Should the user decide to limit ad tracking, that request will be passed through and will be honoured.
- Name, ID, and version of the mobile app
- Internet connection type - WiFi, LTE, EDGE, HSDPA, etc.
- Internet service provider / carrier ID
- Screen orientation
- Hardware version
- Regional and language settings of the mobile device
- Exact make and model of the mobile device
- IP addresses are still used to determine geolocations, and GPS data can be used in certain cases with the necessary user permissions for more precise localisation.
- Information about finger gestures: swipe, pinch in and out, double-tap, pan, etc.
- Screenshot events allow advertisers to know when users take screenshots of creatives they liked. To preserve privacy, only the Unique Hash ID of the creative is stored.
- All ads are disabled if features for visually impaired people are enabled on the device.

Transaction history encryption and exchange

Lastly, we would like to describe the communication protocol between Server-Side Platforms and bidders for off-chain transactions encryption and decryption. It is a crucial part of how our solution guarantees security while maintaining privacy for the bidding activity of both publishers and advertisers. When publishers and advertisers sync their transactions with the blockchain once every 24 hours, they might find themselves surprised by the records submitted by their counterparties. For example, a publisher might be expecting a reward of X

AD for Y clicks from N advertisers and instead find the block of transactions containing just 0.7*X AD for 0.8*Y clicks from N-1 advertisers. Providing that the publisher is honest and had not experienced any disruptions of service during the block period, the publisher would expect his records of transactions to fully match those of all advertisers and the bidder.

So far, our off-chain transactions have been discussed as an innovation enabling the storage of transactions at very high frequencies. However, another key property of our off-chain communication system is the added security for all parties. A checksum line in each public record guarantees that the information pertaining to the individual transactions within each bulk transaction cannot be manipulated. Thus, once a record of the summary transaction is put on the blockchain, the full breakdown of transactions comprising it is secured. Any participant's public node must provide exactly the same response regardless of who is requesting it or else the node will be banned from the ecosystem and its deposit withheld. Here is a simplified code showing how this works:

```
/*
Sets the wallet ID and the period identifier.
*/
var walletID = getWalletID();
var currentPeriod = getCurrentPeriod();

/*
The publishers' Server-Side Platform has a database of all measured ad
interactions.
getMeasuredData() returns a table containing these data: (example table on p.63)
- Wallet ID - the wallet ID of an advertiser who purchased clicks
- Clicks - amount of clicks measured for that advertiser
- Price - payment due measured for that advertiser
- Commission - the bidder's commission for that payment
- Public node - the URL through which to communicate with the advertiser
*/
var expectedSum = 0;
var measuredData = getMeasuredData(currentPeriod);
for (var key in measuredData) {
  expectedSum += measuredData[key][2];
}

/*
getActualAmountFromBlockchain() returns the actual sum received by the publisher
for the period.
*/
var actualAmount = getActualAmountFromBlockchain(currentPeriod);

/*
This line checks if the received amount is equal to the expected sum.
If they are different, a verification is initiated to track the sources of the
discrepancies.
*/
if (expectedSum === actualAmount) {
  console.log('No discrepancies found');
} else {
  console.log('Expected ' + expectedSum + ', received ' + actualAmount);

/*
The public nodes of all counterparties are contacted (all advertisers who purchased
clicks during the period).
requestReportData() returns the JSON data from the public node request (more on
this below).
*/
  for (var key in measuredData) {
    var reportData = requestReportData(measuredData[key][4], currentPeriod);

/*
The data are encrypted with the public key of each counterparty.
Each party can only decrypt their own data as they do not have the private keys of
the others.
decryptData() returns the decrypted data in raw JSON format (more details below).
*/
    var myReportData = decryptData(reportData['report'][walletID]);

/*
A SHA1 checksum of the decrypted JSON data is performed to validate it against the
record in "checksums". If they are different, the offending counterparty is
flagged. A hash list of all individual clicks measured in the transaction period
between the two parties can also be requested and verified. This ensures that no
tampering with the data can occur.
*/
    var expectedSHA1 = SHA1(JSON.stringify(myReportData));
    if (expectedSHA1 !== reportData.checksums[walletID]) {
      console.log('Report was modified. Take action.');
      continue;
    }

/*
The final checksum is stored on the blockchain when the advertiser pays.
If the advertiser modifies any information in the report, the checksum validation
would fail.
If the advertiser modifies the data along with the checksums, the validation would
fail because it would not correspond to the checksum recorded on the blockchain.
SHA1Whole() returns a SHA1 hash of the whole report.
Before generating it, the checksum field is removed as well as the report object.
What remains is a checksum of the checksums object along with the period field.
*/
    var expectedWholeSHA1 = SHA1Whole(myReportData);
    if (expectedWholeSHA1 !== reportData['checksum']) {
      console.log('The report was modified. Take some action.');
      continue;
    }

/*
Note that minor discrepancies are to be expected.
*/
    if (myReportData['price'] !== measuredData[key][2]) {
      console.log(
        'Expected price ' + measuredData[key][2] +
        ', received ' + myReportData['price']
      );
    }
  }
}

JSON of the encrypted data supplied by an advertiser’s public node:
```

```
{
  "period": "12345",
  "report": {
    "p1": "__ENCRYPTED DATA WITH P1'S PUBLIC KEY__",
    "p2": "__ENCRYPTED DATA WITH P2'S PUBLIC KEY__",
    "p3": "__ENCRYPTED DATA WITH P3'S PUBLIC KEY__",
  },
  "checksums": {
    "p1": "__CHECKSUM OF THE DATA FOR P1__",
    "p2": "__CHECKSUM OF THE DATA FOR P2__",
    "p3": "__CHECKSUM OF THE DATA FOR P3__",
  },
  "checksum": "__CHECKSUM__"
}
```

JSON of an individual auction:

```
{
  "clicks": 2,
  "price": 2.25,
  "commission": 0.0675
}
```

Table of advertisers' balances:

Wallet ID	Clicks	Price	Commission	Public node
Hash of A1	1	1.05	0.03	https://example1.com/advertiser/protocol.php
Hash of A2	2	2.25	0.07	https://example2.com/advertiser/protocol.php
Hash of A3	10	11.25	0.34	https://example3.com/advertiser/protocol.php
Hash of A4	15	16.05	0.48	https://example4.com/advertiser/protocol.php

While clearly providing transparency, this solution could also lead to unwanted security breaches if it had not been designed correctly. For example, by having every node on the network provide a list of all past transactions publicly, one could very easily reconstruct the monetisation strategy of any publisher or the bidding strategy of any advertiser. This is a common pitfall of many proposed blockchain advertising ideas. AdHash's solution to this problem comes in the form of public-key cryptography[69]. Each separate data exchange is encrypted by the public key of the corresponding counterparty. Thus, anyone can download the full list of transactions and verify their authenticity by comparing the checksums to the blockchain records. However, only the transaction for which the private key is available can be decrypted. Therefore, no details are revealed about any of the other transactions, except whether they are all valid or not. Here is what that would look like when an advertiser having interacted with nine publishers is requested to share off-chain transaction records with Publisher 6 who can only decrypt his or her own data:

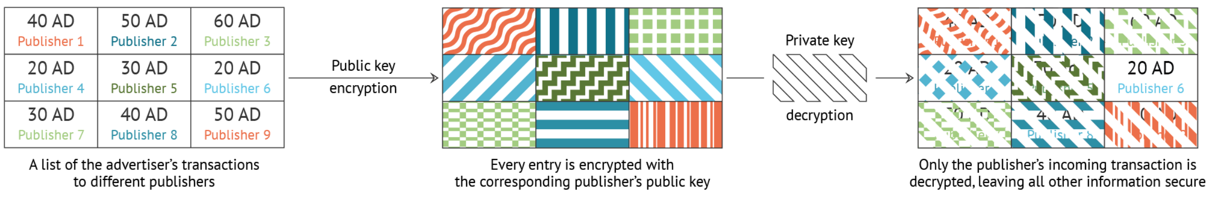

40 AD
Publisher 1
50 AD
Publisher 2
60 AD
Publisher 3
20 AD
Publisher 4
30 AD
Publisher 5
20 AD
Publisher 6
30 AD
Publisher 7
40 AD
Publisher 8
50 AD
Publisher 9
A list of the advertiser's transactions to different publishers
Public key encryption
Every entry is encrypted with the corresponding publisher's public key
Private key
decryption
20 AD
Publisher 6
Only the publisher's incoming transaction is decrypted, leaving all other information secure

Fig. 20: A simple illustration of the code snippet above, showing how public and private keys are matched to decrypt data in the AdHash Protocol for transaction verification purposes.

To put this in the context of our previously given example, our publisher knows that at least one advertiser who was supposed to pay has failed to do so. But the publisher also needs to verify whether all other advertisers have paid in full. For this to happen, when an inconsistency is detected between off-chain transactions and blockchain records, the Server-Side Platform automatically sends sync requests to all counterparties and compares the results. Any inconsistencies are displayed as notifications to the publisher or advertiser concerned (note that small discrepancies are to be expected due to the nature of the advertising business and should be tolerated). Any gross inconsistencies (in the example above, that would be the advertiser who did not pay at all) result in the offending party being

[69] *"Public Key Cryptography", Wikipedia, https://en.wikipedia.org/wiki/Public-key_cryptography.*

automatically removed from the whitelist of the affected counterparty. As described in Chapter VI, this also results in a flag for the offending party.

If a publisher gets paid a little bit less than expected, the Server-Side Platform can automatically block the non-paying advertisers. If the advertisers kept doing the same to other publishers, they would soon run out of publishing partners to show their ads on. If publishers decide to act in bad faith by inflating their numbers, the Server-Side Platforms on the advertisers' servers would automatically stop bidding on their URLs. That would leave the publishers with no clients to sell their inventory to. So, from a game theory perspective, it is clearly in everyone's best interest to not tamper with their records.

The bidder facilitating all RTB auctions acts as an independent third party in conflict resolutions and syncs its own records of data when disputes arise. If neither the publisher nor the advertiser are trying to tamper with their data, all three parties should have closely matching records. In cases that require more details, conflict resolution does not need to rely only on the sums of all clicks between two parties. Our protocol enables the communication of hash lists of all individual clicks measured in the transaction period between any two parties.

VIII Economic Model

Transaction types

Our ecosystem relies on an economic model governing the transactions between users, publishers, advertisers, voters, and bidders. There are five types of economic transactions:

1. Advertisers pay publishers for the verified clicks their landing pages have received. This is done through the bidder which deduces its cut. It is set at 3% for direct deals and 6% for RTB for the AdHash Bidder and can be different for third-party bidders.
2. Users pay publishers for not seeing ads by bidding on equal ground against advertisers. Exactly as above, this is executed through the bidder but the users do not need a Server-Side Platform to submit their bids, just a wallet ID.
3. Deposits confiscated from non-compliant publishers and advertisers fund the Rewards Pool which is distributed among voters biweekly.
4. The blockchain issues tokens to voters for each correct vote. Those transactions are executed biweekly.
5. The blockchain may issue rewards to publishers for every verified click they deliver. Those transactions are also executed biweekly.

Digital currency

All transactions and smart contracts in our ecosystem are enabled and executed by our native digital currency. As already explained on page 16, a unit of this currency is called an AD token. At launch, there will be a total of one hundred million AD tokens and they will be distributed gradually. This is the total supply of tokens that will exist, no further tokens will be issued and there is no inflation predefined in the smart contracts. AdHash's goal is to provide sufficient decentralisation by distributing as many of the tokens as possible to interested publishers, advertisers, users, and ad tech developers.

Rewards

As mentioned at the beginning of this chapter, there are two types of blockchain-generated minting rewards in our economy: click rewards and voting rewards. Publishers are rewarded for contributing valuable content and quality traffic, and voters are compensated for securing the network.

While the relative ratio of the two types of rewards can vary based on the terms agreed upon by our community, the total Rewards Pool at any given moment is determined by the confiscated deposits of offending parties and AdHash's contribution to the pool which will be carefully adapted as we scale.

Resistance to volatility

There is another very important note that needs to be made regarding the economy of AD tokens and it is its link to the real-world economy. Almost all existing digital currencies suffer from tremendous volatility[70]. That is great news for speculators while prices are rising and bad news for anyone who wants to use the tokens for their intended purpose. When prices are falling, it is bad news for everyone involved.

70 *"Total Cryptocurrency Market Capitalisation", CoinMarketCap, https://coinmarketcap.com/charts/.*

AD is a digital currency native to the AdHash Protocol created with a sole purpose: to realise our vision of a secure and fair advertising ecosystem that benefits from the transparency of a blockchain. This means that dramatic price fluctuations would not benefit our clients as they would have to readjust their bidding strategies accordingly. In this regard, AD's link to the real-world economy gives it a significant benefit compared to other purely financial digital currencies: since the tokens are directly linked to a very large and liquid market in the real economy - that of digital advertising, if the price of AD tokens suddenly halves, that would mean that publishers would be earning just half of what they used to. However, advertisers would be able to purchase twice the amount of advertising for the same price. So they would immediately start taking advantage of the situation which would naturally drive the price of AD tokens up again.
In the contrary scenario, if the value of AD tokens doubles overnight, advertisers would be reluctant to buy much inventory as it would be twice the price originally planned by their marketing departments. However, such a price jump would generate twice as much revenue for publishers, attracting new ones to the system. More of them joining the ecosystem would introduce more supply for the current demand, driving the average bid price down towards its original value and allowing advertisers to resume their normal levels of spending. Publishers and advertisers would be able to adjust their bids to adapt to the changing conditions.

The described market logic means that the effect of any volatility experienced by other digital currencies should be greatly dampened in the case of AD tokens. They should therefore remain usable for the intended purpose - that of trading ads.

New business models for digital advertising

The economic model we presented will govern the AdHash ecosystem and its future development. We have constructed it in a manner that would incentivise positive behaviour and create two entirely new business models around digital advertising:

1. It would allow users to reward publishers for their content without relying on advertisers as an intermediary by bidding against them.
2. It would allow users, publishers, and advertisers alike to contribute to the ecosystem by becoming voters and receiving real financial rewards for their time. Our projections show that saving even a tiny fraction of the hundreds of millions of dollars spent annually on ad reviews and moderation would allow a group of a few hundred committed voters to earn a decent income while providing value to everyone else.

IX Open-Source and Integrations

Open-source philosophy

Since we started working in the advertising space over six years ago, our team has been on a mission to make digital advertising more transparent and beneficial to everyone involved and reduce the number of unnecessary middlemen plaguing the RTB ecosystem. In that time, we have seen the broader advertising ecosystem collapse into an even more hopeless state of ever decreasing transparency and mounting costs. Our experience in this industry makes us realistic enough to know that we are not going to solve every problem that every advertiser, publisher, or user has single-handedly. Instead, we have tried to create a solid base upon which the entire industry can build. The goal of our framework is to be the technological layer on which all advertising bidding for the open internet is performed. For that we need a vibrant supporting community to push the technology forward.
Other developers can use our GitHub repository[71] to fork the latest version of the AdHash Bidder and Server-Side Platforms and build on top of them under the GNU GPLv3 license[72]. This could mean creating bidders with faster sync rates, lower fees, different rewards structure, or even a completely different targeting and bidding methodology. For example, one could introduce behavioural targeting, CPM or CPA bidding, data trading, rich-media creatives, or any other differentiation.

Governance

In the past year, we have seen tremendous excitement among blockchain enthusiasts to build self-governing ecosystems[73]. We hope to channel this excitement and utilise the talent and wisdom of this community towards building a better advertising framework.

We believe that imposing centralised and authoritative decisions[74] on a global advertising ecosystem makes little sense and hurts users, publishers, and advertisers alike. This is why we decided to open AdHash's policy making to its community. Hopefully this will also give more power to independent regulators and industry-governing bodies.
There will be regularly scheduled governance votes ruling on desired changes and improvements. Those could include anything from minor alterations to the smart contracts and algorithms to accepting or removing certain groups of publishers and advertisers or ad formats and bidding strategies, to adjusting the different classes of minting rewards.

There will be three platforms facilitating the self-governing process:

- **A proposal platform** for submitting and accepting proposals.
- **A voting platform** executing and recording all votes on the blockchain.
- **A database platform** serving as a database for all creatives ever hashed and recorded on the blockchain. This will allow anyone to check the hash of any ad and find its source and any other related ads. We hope this will bring significant transparency and control, alleviating some of the unease users feel about the anonymity and lack of governance in online advertising.

---

71 "AdHash", GitHub, https://github.com/adhashprotocol.

72 "GNU General Public License", GNU Operating System, https://www.gnu.org/licenses/gpl-3.0.en.html, 2007.

73 "Tezos", The Tezos Foundation, https://www.tezos.com.

74 "ProBeat: Facebook's Bitcoin Ban is Baffling", VentureBeat, https://venturebeat.com/2018/02/02/probeat-facebooks-bitcoin-ban-is-baffling/, 2018.

Integrations

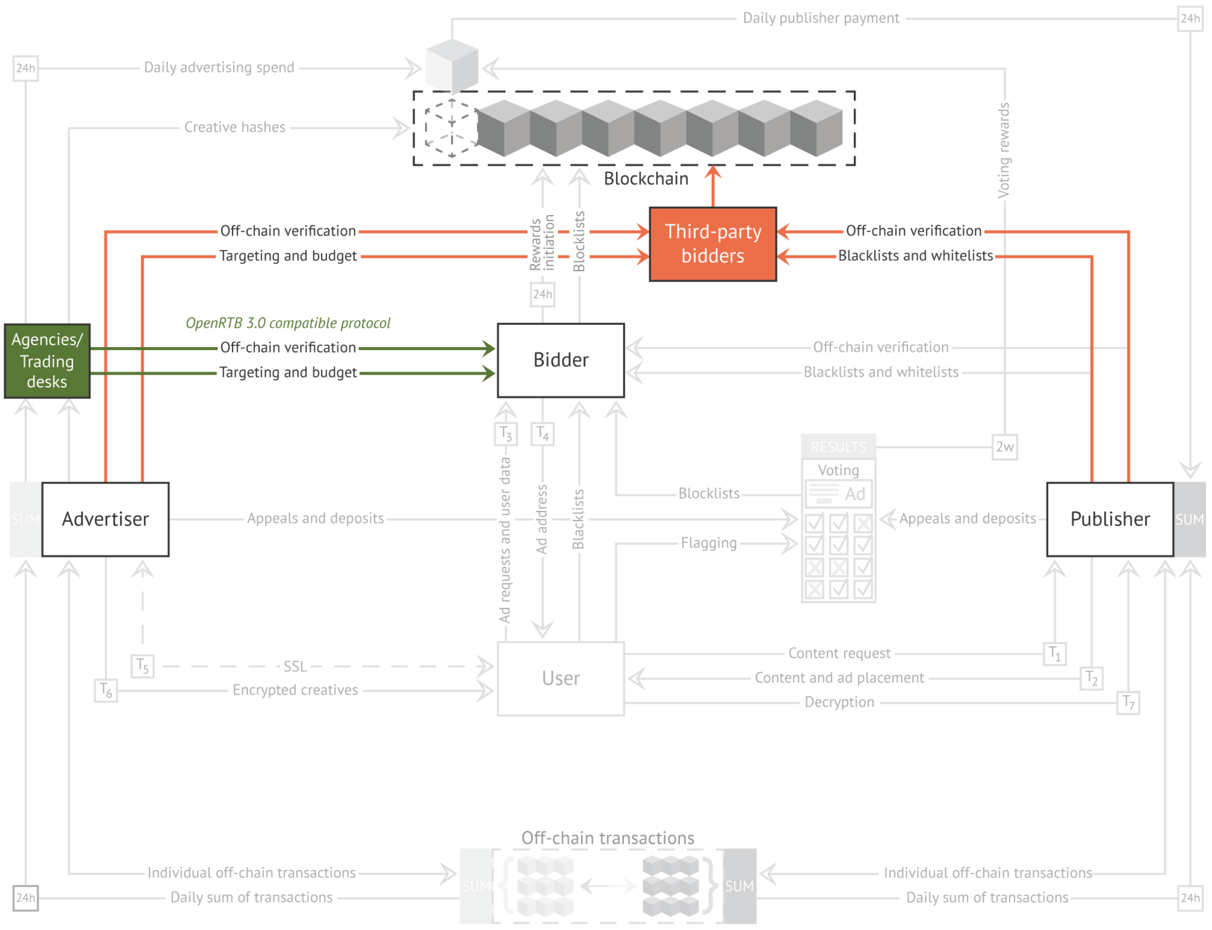


Fig. 21: A block diagram of the AdHash Protocol depicting the necessary communication channels between advertisers, publishers, the blockchain, and any third-party integrations. The orange lines concern parties who would like to introduce their own bidders to the ecosystem. The green lines show the integrations necessary for third parties who intend to plug directly into the AdHash Bidder.

Shown above is the block diagram depicting the AdHash Protocol discussed in detail in Chapter VII. The difference in this version is that the focus is on third-party bidders. In order to be truly open to the world, AdHash had to embrace competitors and make integrations as easy as possible.

What that means is that not only are third parties free to set up their own bidders in our framework, choosing their own rules and governance, but that we will also be opening the inputs and outputs of the AdHash Bidder to any third parties who wish to integrate with it. This is why we have chosen the communication between publishers, advertisers, and bidders to be based on the OpenRTB 3.0 protocol.

We expect ad networks and platforms willing to embrace the technology and bring their client bases to our system to choose to build bidders of their own. Ad agencies and their trading desks, on the other hand, would most likely prefer to directly plug into the AdHash Bidder to avoid having to educate their workforce on using yet another platform. The documentation for such integrations should make the adoption of our technology an almost trivial task.

We suspect that by now the reader has a strong-enough understanding of AdHash's technology to recognise that the universal ad hashes would allow for the introduction of multiple bidders and integrations without jeopardising the integrity of the bidding process. In the next paragraph, we will discuss how the bidding process can continue to operate productively in a multi-bidder environment.

Multi-bidder environment

In the presence of several bidders, the final decision-making for each bid is performed by the publishers' Server-Side Platforms. Advertisers would still submit their bids as they would in a single-bidder scenario, except they would push their requests to multiple bidders at once. Each bidder would calculate the winning bid and send the advertiser's targeting criteria and price to the publisher's Server-Side Platform. The platform would then compare all bids coming from the different bidders and the corresponding advertisers' unique ad hashes. If more than one bid is submitted by the same ID, the platform would only consider the highest bid. This process is similar in theory to server-side header bidding[75], but without header bidding's biggest pitfall - the latency created by bids being considered dozens of times for each ad placement. This architecture would only reward bidders that bring new inventory or innovative products. This should greatly reduce the commoditisation that is characteristic of the existing ad exchange business where the vast majority of competitors are essentially reselling the exact same inventory.

We hope that this section has given the reader a basic understanding of our plans regarding future integrations, collaborations, and developments. More information can be found in our open-source and integration documentations. They are available for download through the AdHash website[76] and our GitHub repository[77].

75 *"Envisioning the Future in a Server-Side Header Bidding World", AdExchanger, https://adexchanger.com/the-sell-sider/envisioning-future-server-side-bidding-world/, 2017.*

76 *"AdHash -A Fundamental Rethinking of RTB Advertising", https://adhash.com.*

77 *"AdHash", GitHub, https://github.com/adhashprotocol.*

X Differentiation

In this chapter we will give a brief overview of how AdHash is different from other products in the digital advertising space. We will look into various solutions along the spectre of the ad tech evolution including a few proposals based on blockchains.

Closed ad networks

Closed ad networks are the simplest form of advertising companies that represent just one intermediary in the ad serving process. Their isolated structure and lack of interoperability severely restrict the advertising reach and granularity of data, impacting both the return on investment for advertisers and the monetisation potential for publishers.
However, it is precisely this simplicity and lack of middlemen that could allow specific ad networks to stand apart, charging lower fees, allowing higher-quality participants, and ad formats that resonate better with advertisers and audiences. Unfortunately, without providing full transparency into their operations and fees, the reliability of those networks is only as good as the trust in them. And as already discussed in Chapter II, trust in the ad tech industry is a rare commodity. Without the transparency of blockchain, participants in ad networks start adding unnecessary verification middlemen, ad serving technologies, data brokers, and others, clogging up the ad delivery path and pushing fees to unreasonable heights.
Our open-source proposition is likely to attract many of those ad networks which would like to introduce trust into their operation and enhance the value they offer to their clients, while maintaining their independence from intermediaries. Stepping onto our platform would enable them to do so without having to expend time and resources.

Traditional SSPs, ad exchanges, and DSPs

Supply-Side Platforms (SSPs) working with publishers and Demand-Side Platforms (DSPs) working with advertisers are more evolved and complex variants of the ad network model which have gradually developed since the invention of RTB advertising and ad exchanges. They rely on many more middlemen and despite achieving a wider reach and more extensive data collection, they suffer from much higher costs, lowered efficiency, and a proliferation of ad fraud. We have already extensively covered some of the most catastrophic pitfalls of the current model in Chapters II and III. SSPs and DSPs are also notorious for exploiting user data and lacking transparency and consistency in their fees. For example, established ad tech companies structure and call their fees differently: AppNexus charges a “Seller Auction Service Charge”, Google has “tech fees” with advertisers and haggles with publishers over revenue share. Some of them are even inventing new types of fees, like the buy-side fee[78], the discovery of which caused a lot of controversy in 2017.
AdHash solves the lack of transparency and inconsistency in the current intermediary fees by introducing a single, transparent, and fully traceable commission which is an order of magnitude lower than what the combination of any DSPs, SSPs, or ad exchanges are currently charging. Also, by introducing unique ad hashes for all creatives and participants in the ecosystem, we eliminate the necessity of running ads through several platforms simultaneously in the hope of finding a better deal. Thus, double-bidding inefficiencies disappear. The off-chain and on-chain records of all transactions provide full transparency into the fees and also help alleviate the majority of ad fraud attack vectors known today. Most importantly, users will get true control over the advertisements that reach them and will

[78] *“Rubicon Got Rid Of its Buy-Side Fees – But Who Else Is Charging Them?”, AdExchanger, https://adexchanger.com/platforms/rubicon-got-rid-buy-side-fees-else-charging/, 2017.*

be able to choose an ad-free environment without hurting the publishers they like. Their data are protected by eliminating tracking scripts and personalised targeting.

Ads.txt and ads.cert

Introduced by the Interactive Advertising Bureau in 2017[79] to combat traffic arbitrage and domain spoofing, ads.txt is essentially a text file the webmaster posts on the publisher's domain. The file contains a list of authorised sellers (exchanges or SSPs) the publisher deals with. Buyers can crawl the web for those lists, and create filters to assure they transact with the intermediaries authorised to sell the publisher’s inventory. The seller’s ID listed in the bid request should match the authorised seller in the ads.txt file for the bid to be considered. Ads.cert complements ads.txt by using public-key cryptography to sign and verify bid requests. This ensures that inventory is not only being bought through authorised resellers, but it is also coming from verified sources.

However, it is important to note that ads.txt is nothing more that a simple text file; that makes it very easy to tamper with and also very vulnerable to social engineering attacks[80]. Even its proposed evolved version, adstxt.plus[81] written on the Ethereum blockchain would do nothing to prevent those social engineering attacks.
It is also relatively easy for botnets to change their spoofing scheme and generate traffic from other domains without ads.txt. This means that even if a small number of publishers on any given platform are non-compliant or get compromised, the advertisers will be scammed through them. Thus, the efficiency of ads.txt and ads.cert depends entirely on adoption rates and is not an all-encompassing solution on the protocol level the way AdHash is.
Possibly the weakest link of this solution is the fact that publishers’ private keys for ads.cert are stored by third parties (e.g. SSPs). This requires absolute trust between any publisher and all of the third-party platforms he is working with, which is unsustainable.

AdHash’s solution to ad fraud is considerably more complex and effective than ads.txt and ads.cert and covers a much broader field of attacks. Here we will only outline how it is superior to the two attacks discussed above - arbitraging and domain spoofing.
Arbitraging is easy to remove - it simply does not exist in an ecosystem where there are no middlemen, where all parties are disclosed and their identities verifiable on the blockchain. There is nothing to gain from merely reselling traffic without adding any value since the cut taken will be plain to see in the blockchain records of the transactions.
In cases such as the Methbot[82] and Hyphbot[83] operations, it can be impossible to confirm that a domain is being spoofed. AdHash’s technology does not prevent an ad from being shown on a spoofed domain in that scenario. Instead, it makes sure that payment is issued to the wallet ID of the original domain, not the one pretending to be it. This means that if a Methbot-like operation attacks our ecosystem with low-quality websites pretending to be Forbes.com, for example, it would result in Forbes.com earning money from those pretend-clicks. And since the AdHash Bidder executes payments every 24 hours, the attackers would realise very quickly that their operation is unprofitable and would therefore cease running it.

---

79 *“Ads.txt - Authorized Digital Sellers”, IAB Tech Lab, https://iabtechlab.com/ads-txt/, 2017.*

80 *“How Publishers are Getting Fooled by ads.txt Fraud”, Digiday, https://digiday.com/media/publishers-getting-fooled-ads-txt-fraud/, 2017.*

81 *“Ads.txt Plus”, MetaXchain Inc., https://adstxt.plus.*

82 *“The Methbot Operation” , Whiteops, http://go.whiteops.com/rs/179-SQE-823/images/WO_Methbot_Operation_WP.pdf, 2016.*

83 *“Hyphbot”, Adform, https://site.adform.com/knowledgecenter/whitepapers/how-adform-discovered-hyphbot, 2017.*

AdHash is also fully compatible with a mobile in-app environment, whereas ads.txt and ads.cert presently are not fully supported due to the lack of a reliable method to retrieve the public key since there is no known domain.

Other blockchain advertising ideas

The stellar rise in popularity of blockchain projects in the past few years has given birth to a tremendous variety of different projects, ranging from decentralised autonomous venture-capital funds[84] to literally useless tokens[85]. Not surprisingly, amongst thousands of proposed ideas, there were a few that tried to come up with solutions to the well-established problems of digital advertising[86],[87],[88],[89],[90],[91],[92],[93],[94],[95],[96],[97],[98],[99],[100],[101]. While discussing their merits and deficiencies at length is beyond the scope of this paper, and while revering the enthusiasm and urgency with which teams jump to offer solutions to the problems of the industry, as a general observation it is saddening to see that almost none of those ideas were released with even a minimum viable product. This is why we are referring to them as ideas, rather than projects. Most of them are based on empty promises, false assumptions about the advertising business, or broken economic models. What differentiates AdHash is the fact that over the course of over four years we have built a profitable ad tech company and we have spent the entire period of time researching and developing different ways to build a more transparent and efficient way to trade advertising. Or, to put it simply - the others have ideas, we have a product.

The most concerning fact about our blockchain competitors' viability is that nearly all of them rely on storing RTB transactions on a blockchain. First, that is a terrible idea for both publishers and advertisers, all of whom like to keep their specific bidding strategies a closely-guarded secret. And second, RTB activity accounts for over five million transactions

[84] *"The DAO Organisation", Wikipedia, https://en.wikipedia.org/wiki/The_DAO_(organization), 2016.*

[85] *"Useless Ethereum Token", https://uetoken.com, 2017.*

[86] *Basic Attention Token, https://madhive.com/solutions-advertisers/.*

[87] *NYIAX, https://www.nyiax.com.*

[88] *Adshares, https://adshares.net/assets/docs/Adshares.Network.whitepaper.pdf.*

[89] *Lucidity, https://lucidity.tech/yellowpaper/.*

[90] *AdChain, https://docsend.com/view/qnce6gy.*

[91] *AdEx, https://www.adex.network/adex/AdEx-Whitepaper-v.7.pdf.*

[92] *Qchain, https://qchain.co/files/Qchain_deck_web.pdf.*

[93] *EnvisionX, http://envisionx.co.*

[94] *ClearCoin, https://clearcoin.co/whitepaper/.*

[95] *MadHive, https://madhive.com/solutions-advertisers/.*

[96] *Ternio, https://ternio.docsend.com/view/4jji4v4.*

[97] *Ubex, https://www.ubex.com/wp/Ubex-Whitepaper-en.pdf.*

[98] *Thrive, https://ico.thrivelabs.io/documents/thrive_whitepaper.pdf.*

[99] *Kind Ads, https://kindads.io/Content/kind-ads-whitepaper.pdf.*

[100] *tribeOS, https://docsend.com/view/qh9g5ui.*

[101] *Zinc Protocol, https://docsend.com/view/hr3erk2.*

per second. That is six orders of magnitude larger than the current transaction capacity of the Ethereum blockchain on which almost all of those projects are based. That is an incredibly large gap that is certainly not going to be overcome in the foreseeable future, meaning that those teams would either have to completely change their stated roadmaps and plans for technological development, or worse - abandon the development of their ideas altogether. They cannot succeed if their product visions rely so heavily on pure conjectures about the future performance of blockchain. And even if those limitations disappeared overnight, storing millions of transactions per second would make no financial sense since most of those ideas are based on the Ethereum blockchain which has relatively high transaction costs.

Some of the mentioned ideas also propose storing user data on the blockchain and trading it with interested advertisers which is a far worse attack on user privacy than ever created by the advertising industry of today and a clear violation of the EU's General Data Protection Regulations (GDPR)[102] and the California Consumer Privacy Act[103].

Yet other examples are merely trying to rebuild today's cancerous ad industry with a blockchain twist, by introducing almost all of the present intermediaries in a new context. Unsurprisingly, such ideas are coming from companies that are acting as middlemen in the current advertising environment.

The lowest quality ideas tried to awkwardly retrofit the innovative blockchain technology to a transaction method that even predates RTB advertising. Those expect publishers and advertisers to manually find one another and reach deals which shows they have no experience in the advertising industry of the past decade.

AdHash, by contrast, is only storing information on the blockchain that is already public and the transparency of which actually increases the value of advertising for all parties involved. Executing micro-transactions off-chain and recording only bulk transactions on-chain significantly reduces the dependence on the blockchain by storing the vast amounts of data in a perfectly scalable way on the servers of all participating publishers and advertisers. It also ensures transparency while retaining a level of privacy.
Advertisers target users only by data readily available in their browsers and without exposing these data to any third parties. Users can participate in the ecosystem and can be paid for it, but only after delivering meaningful contributions. Most importantly, AdHash is launching a product that is fully functional, open-sourced, and ready for integrations.

Ad blockers

Some ad blocking companies[104],[105], while greatly annoying their customers[106], have decided to allow certain ads to slip through their filters as long as they meet some subjective criteria about being acceptable (or presumably if they offer enough money). Since the core

102 "The EU General Data Protection Regulation (GDPR)", GDPR Portal, https://www.eugdpr.org, 2018.

103 "California Consumer Privacy Act (CCPA)", California Legislative Information, https://leginfo.legislature.ca.gov/faces/billTextClient.xhtml?bill_id=201720180AB375, 2020.

104 "Allowing Acceptable Ads in Adblock Plus", https://adblockplus.org/acceptable-ads, 2016.

105 "Brave Ad Replacement", Brave Browser, https://www.brave.com/about-ad-replacement/.

106 "What Adblock Plus Forgot Before Its Acceptable Ads Platform Release", Forbes, https://www.forbes.com/sites/johnhall/2016/09/18/what-adblock-plus-forgot-before-its-acceptable-ads-platform-release/, 20176 users of ad blockers install them precisely because they do not want to see any ads, acceptable or not, this predictably has led to a very significant backlash.

But by that point, ad blocking companies had put themselves in a difficult position - they had created tools that were hurting publishers without generating any revenues in return. This made their net effect on the ecosystem negative and they had to quickly discover a business model that would not drive all of their clients away. Some of them found such a model in allowing clients to voluntarily donate a specific amount per month[107]. This amount would be distributed amongst the websites of participating publishers whose properties were visited by those users. Ad blockers would keep a commission and not infuriate their users by slipping acceptable ads through their filters.

This is an elegant concept that could bring additional funds to the publishing business. The problem with it is that it does not require publishers to give permission for their content to be monetised in this way. Thus, unless publishers know about the program and create accounts with the ad blocking company, the funds collected on their behalf would be taken from users and never delivered to them.

Another problem with this solution concerns its scalability - by using fiat currencies and regular bank transactions, it would never be possible to achieve the flexibility allowed by micropayments, as evidenced by the minimum payment threshold imposed by those platforms.

The most egregious problem with this solution, however, is the fact that there is zero transparency in the collected payments. There is no verifiable way of detecting how much of the users' contributions are actually paid to the publishers and how much are being kept by the ad blocking companies. It is also impossible to gauge whether those payments are comparable to the earnings the publishers would otherwise have made from advertising.

In light of these limitations, the decision to collect donations from users and distribute them amongst publishers seems like a desperate attempt from ad blockers to extract some revenue from a product that was never designed to have a working business model in the first place. The lack of transparency and constraints in scalability mean that it could never become the mainstream solution that would finally marry publisher monetisation to customer satisfaction.

By revolutionising ad serving and allowing advertisers to directly host their ads in a secure environment, AdHash could make the operation of ad blockers impossible. They would have to block access to the advertiser's server since that is where the ads would be delivered from and that, in turn, would mean blocking access to the entire website.
However, fighting ad blockers is not AdHash's goal. We believe that we could render ad blockers unnecessary for all but the most dedicated ad blocker users. In an ecosystem where ads offer meaningful content and uphold certain ethical and aesthetic standards, there is little incentive for users to install ad blockers in the first place. Those who insist on an ad-free environment can outbid the advertisers, thereby removing all ads without hurting the publishers and without having to pay individual monthly subscriptions for all websites and apps they use.

107 *"AdBlock Plus Teams Up With Flattr to Help Readers Pay Publishers", TechCrunch, https://techcrunch.com/2016/05/03/adblock-plus-teams-up-with-flattr-to-help-readers-pay-publishers/, 2016.*

Summary

In summary, a common point of failure among all the listed types of competitors is that they are all merely trying to patch things up: closed ad networks are trying to offer better quality inventory at the cost of reach; SSPs and DSPs are trying to deliver reach at the cost of opening up the gates to countless intermediaries and fraudsters; blockchain ad companies are trying to eliminate those intermediaries by reversing all of the technological progress that has happened in the advertising space in the past 25 years; and ad blockers are simply abusing the ecosystem to gain adoption, only to then betray the trust of all of their users by trying to discover a business model for themselves.

During our research and development we realised that the problems we were trying to solve could not be fixed by patching things up. We took a step backwards, looked at the entire ecosystem carefully, and concluded that retrofitting trivial fixes was not going to eliminate those problems at their roots. We realised that the solution we needed was not evolutionary, it was revolutionary.

XI Risks

AdHash's team has collectively invested over 40,000 hours of work re-engineering the framework on which real-time bidding advertising is executed. That is a considerable undertaking and in this chapter we would like to acknowledge some of the risks that lie ahead of us. They entail circumstances under which our technology and business operations could suffer.

The long-term followers of our project might notice that this list has shrunk significantly over the years. The explanation is simple - as we launched AdHash and opened it to clients, we confirmed that many of the risks listed before would in fact not affect us. The quality of our product met our clients' expectations.

Related to AdHash's technology and the advertising space

- **Advertisers may object to the fact that we are not tracking users as aggressively as they have come to expect from other companies**. But AdHash strives for ultimate quality and reliability. Only data that are freely visible and measurable by the advertisers' and publishers' servers when a user visits one of their properties are allowed to be considered by the AdHash Bidder. No assumed or extrapolated information will be collected by it. If certain advertisers find this targeting method insufficient, a market niche would open for building bidders on top of the AdHash Protocol to allow that sort of information trading.
- **A malicious ad on a high-profile publisher's site or app** - AdHash's solution allows advertisers to upload and start serving ads as soon as their hashes are recorded on the blockchain. Although each creative goes through an artificial-intelligence visual API, as soon as that happens, there is no formal creative approval process which means that there is a risk of unacceptable ads being uploaded. (Unacceptable only in a visual context, malware cannot be served due to the restriction to only serve static images and no scripts.)
  If this occurs on a high-profile website or an app with a lot of users, the ad would quickly trigger 10 flags from concerned users and would be therefore disabled and submitted for a vote. Almost immediately after that it will be permanently rejected and the advertiser that issued it would lose the entire deposit. This would make such experiments highly unprofitable. Additionally, the malicious advertiser would be banned from the ecosystem and would only be able to reapply by going through a vote.
  The more cautious publishers can also apply whitelists as a secondary safety measure.
- **The open-source nature of the AdHash solution may lead to undesirable forks made by third-party developers** - the operation of the AdHash ecosystem is not susceptible to malicious or substandard developments from third parties. Their work will only affect their particular products and sets of clients. All components necessary for successfully trading advertising in our ecosystem are built and tested by AdHash.
- **A better solution to digital advertising may be invented** - this is not merely a risk, but a statistical certainty. No technology has ever stayed dominant for too long before being disrupted by a more advanced solution. While our product is currently the only working solution for digital advertising that can combine the efficiency of RTB with the transparency of a blockchain, we embrace innovation and welcome competition. All parties need to collectively rebuild trust back into the advertising system and that is why we hope that our open-source approach will encourage more competitors to invest their time and resources to build on top of AdHash's technology, rather than reinventing it from scratch. AdHash will also continuously update its offering by exploring new areas of research and development, such as bringing non-intrusive and efficient video advertising into the blockchain era.

Related to blockchain technology and digital assets

- **Ethereum may fail to deliver** - in the unlikely scenario that we have to migrate from the Ethereum blockchain, we could always switch to another underlying blockchain or build our own, exchanging each Ethereum-based ERC20 token for a newly issued one. Should such necessity ever arise, we will do our best to consult and coordinate all important steps with our community in advance.
- **The regulatory regime governing nascent blockchain technologies and digital currencies is uncertain** - worldwide regulation changes might affect AdHash's operation or the utility of AD tokens. We intend to follow all policies and adapt to any changes in the landscape. Our top priority is the long-term viability of this project and that involves making sure that our technology is available worldwide and is in compliance with all regulations.
- **Unanticipated risks** - we have created an entirely new area of research and development in advertising and are at the beginning stage of a great experiment. There is no doubt that there will be unexpected hurdles along the way but the AdHash team is committed to meet all challenges head on.

XII Team

AdHash has three founders:

Martin Stoev, CEO
BSc and MA in Natural Sciences from the University of Cambridge, 2011.
An inventor, scientist, and entrepreneur with a decade-long experience in gathering, managing, and growing successful teams that deliver outstanding products.
stoev@adhash.com

Damyan Stanchev, CTO
BSc in Software Engineering from Sofia University, 2013.
A senior software developer with more than 10 years of experience in JavaScript, jQuery, PHP, and SQL who loves to coach people and bring ideas to life.
damyan@adhash.com

Adriana Taseva, COO
BA and MA in Chinese Studies from the University of Cambridge, 2012.
A sinologist by degree, strategic thinker by day, and artist by night with experience in developing, communicating, and sustaining business growth initiatives.
adriana@adhash.com

They are empowered by a world-class team of developers, designers, data scientists, and ad tech experts.

XIII Conclusion

This paper explored how today's advertising industry became a product of constant and desperate retrofitting. We concluded that yet another quick fix solution would not suffice to mend the leaks of a system that is broken at its core. Therefore, we presented an entirely new approach to securing the real-time bidding advertising environment. The AdHash Protocol makes advertising:

- **Simple** - hosting the creatives directly on advertisers' servers removes the vast majority of intermediaries, achieves a significant reduction in cost and complexity, and solves the issue of mistrust.
- **Decentralised** - delegating the governance of the ecosystem to a community of incentivised voters creates a more efficient and democratic policy-making process.
- **Transparent** - storing an immutable record of unique ad hashes on a blockchain eliminates ad fraud, solves the inefficiencies in the real-time bidding process, and introduces accountability and standardisation among all parties.
- User-centric - no personal user information is collected or resold by the AdHash Bidder. There is complete transparency into all ads, advertisers, and publishers. We do not use any kind of behavioural or demographic targeting; AdHash is focused entirely on optimising contextual targeting.

This paper has discussed in detail the building blocks of our transparent advertising protocol. After years of development, we finally have the fundamental technology to truly disrupt the existing advertising industry. But if there is one lesson to be learned from the network revolution that occurred over the past six decades, it is that success comes through numbers. And this is why we know that our success would depend on the combined efforts of the AdHash team and the vibrant community of incentivised users, passionate publishers and marketers, and like-minded developers willing to contribute. By open-sourcing our technology, we hope to supercharge innovation and harness the extraordinary value that a broad community can create. This is why we invite those compelled by the idea of revolutionising the digital advertising industry to join our effort.

AdHash

A Fundamental Rethinking of RTB Advertising.