Rethinking Service Systems: A Path Towards Secure and Equitable Resource Markets

Ghada Almashaqbeh
University of Connecticut, CT, USA

Abstract

Escalating demand for digital services has motivated the community to revisit the old ideas of peer-assisted models for resource trading. Cryptocurrencies have strengthened this trend by providing a fully distributed mechanism to compensate service provision. We present a generic framework for building decentralized service markets by utilizing these new technologies. We discuss the security and efficiency challenges associated with the open-access work model of these markets along with some potential solutions.

1. Introduction

Escalating demand for digital services has motivated the community to revisit the old ideas of peer-assisted models for resource trading. Cryptocurrencies have strengthened this trend by providing a fully distributed mechanism to compensate service provision. To promote practical deployment, in this article I present a generic framework for building decentralized service markets by utilizing these new technologies. I also discuss the security and efficiency challenges associated with the open-access work model of these markets along with some potential solutions.

When obtaining digital services, usually we deal with traditional systems that are centrally managed. Most of the time, we resort to third party providers, or commercial companies, to obtain services like file storage, content distribution, computation outsourcing, and many others. Despite being effective and widely deployed, this centrally-managed paradigm introduces several trust, cost, and transparency issues. Companies may impose complex business relationships in which customers usually overprovision their needs in order to handle future peak demands. These companies also constrain customers with the service specifications that can be offered, such as geographic coverage and speed. Not to mention infrastructure dependencies where a system implementation is tied to the API provided by the underlying resource vendors.

Table 1: Examples of centrally-managed digital services and their counterparts of P2P-based ones.

Service Type	**Traditional Solution**	**P2P-based Solution**
Payments	Banks	Bitcoin
File storage	Dropbox	Filecoin
Video Transcoding	Amazon Elastic Transcoder	Livepeer
Key management	Azure Key Vault	NuCypher

These issues raised many questions regarding the future of the Internet and its digital services; can we build flexible and equitable systems that allow anyone, from anywhere, to obtain or provide a service? Can we even deploy them in the form of decentralized resource markets in which computing services are exchanged for payments without intermediaries? And can we optimize these systems to be highly performant, and hence practical, without compromising security?

The pursuit of answering these questions motivated the community to revisit the old ideas of peer-to-peer (P2P) based models in which anyone is allowed to join the system and serve others. In order to encourage collaborative work and compliance with the protocol, payments are provided in return, which creates a market for trading resources. This paradigm builds flexible systems, scales more easily with demand, and extends the network coverage since peers from anywhere can join. Furthermore, P2P-based models implement transparent and equitable ecosystems in which participants can negotiate service terms and price directly instead of having a few entities monopolizing the market.

However, most of existing solutions for monetizing P2P-based services introduce some form of centralization or trust. They either use centralized payment services, place trust in specific parties to handle these payments and resolve disputes, or even rely on some centralized entities to manage participants and hold

them accountable. Such design choices bring us back to the central management model and the trust issues of traditional solutions.

The evolution of cryptocurrencies and blockchain technology has provided new templates for reshaping this service paradigm. Cryptocurrencies implement a decentralized virtual currency exchange medium that permits participants to be rewarded without any pre-authentication or identification requirements. And their underlying blockchains and consensus protocols support public verifiability, auditing, and decentralized governance without needing to place trust in any entity. These features can be exploited in P2P-based schemes to manage and pay for the correct service without driving the system toward centralization (see Table 1 for examples of traditional service solutions and their P2P/cryptocurrency-based counterparts).

2. Decentralized Resource Markets Design

The open access environment of P2P networks (i.e., allowing anyone to join and dealing with untrusted participants) introduces several security and performance challenges that need to be addressed before having any practical deployment. In addition, having monetary incentives motivates attackers to attack the system in novel ways to maximize their financial profits. Thus, traditional practices of secure systems design need to be modified and expanded to account for such factors.

To address these issues, I present a generic framework for designing secure, scalable, and equitable resource markets to provide services in a fully distributed way. The proposed framework consists of systematized design steps distilled from experiences in building blockchain-based services and large-scale distributed systems (a high level diagram is captured in Figure 1). The framework accounts for the security, performance, and economic aspects of monetary-incentivized decentralized systems. The framework also highlights how such an emerging work model requires more sophisticated techniques (for risk management, threat mitigation, service-payment exchange, service pricing, etc.) than these employed by traditional, infrastructure-based services. These challenges, along with some potential solutions, and the framework steps are discussed below.

Viability Assessment. Before looking into building a distributed resource market, one has to assess its viability. This includes studying the demand side (who is interested in the service) and the supply side (who can provide it) to answer several questions; are there tangible advantages to encourage replacing traditional solutions with fully distributed ones? Can the system match the reliability and performance offered by these traditional solutions? Does providing the service require large amounts of resources that exceed the capabilities of average end-users? Such a viability assessment is an essential step to assess the potential for practical adoption before investing time and effort into building the system.

For example, in online content distribution, studies argued that up to $\sim$88% of the traffic can be offloaded to peers during peak demand hours [6], and several large commercial companies have utilized this option to supplement their infrastructure, such as Akamai NetSession, Swarmify, and Velocix. For file storage, a Harvard-led study [11] found that almost 50% of all hyperlinks cited in US Supreme Court opinions are broken since the content is no longer available at the cited locations. This advocates for building a decentralized content-addressed (instead of location-addressed) file storage network to promote the robustness and resilient of the web, e.g. [1].

Threat Modeling. Despite the many advantages they offer—decentralization, transparency, and lowered service costs— there is still a big gap between the promise of P2P-based systems and their performance in practice. Adding monetary incentives, by using another P2P-based payment service, widens this gap. This is due to the perception that these systems are not secure, where the recent large number of security breaches give credence to these doubts [2].

The best practice for designing a secure system requires a threat modeling step to investigate potential security risks. Such a model can guide designers in deploying the proper countermeasures, and evaluating the security level of the system in the after design stage. For resource markets, building a threat model requires a framework that can handle large-scale distributed systems, explicitly account for the financial motivations of the attackers, and help in spotting any potential collusion between these attackers.

Thus, existing threat modeling frameworks, which either target secure software development or small-scale systems, need to be adapted to address these issues. For example, the ABC framework [3] was designed to achieve these goals by accounting for both the underlying cryptocurrency medium and the service provided on top of it. ABC enables building comprehensive threat models by holistically analyzing the threat space while managing its complexity, and distilling the impactful cases that need to be neutralized. ABC also allows for classifying threats based on their mitigation techniques, i.e., threats that can

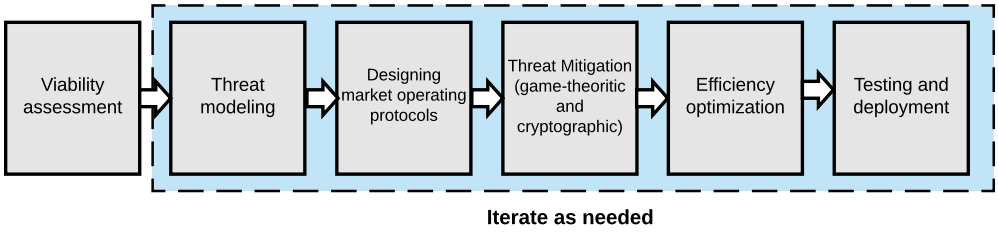

Figure 1: Design process of distributed resource markets.

be addressed cryptographically or algorithmically, and those that require game theoretic means, thus providing insights about the proper measures to deploy.

It should be noted that the threat modeling step need to be revisited each time the system design is altered. Furthermore, it should be performed as the last step before shipping the system to argue formally about its security.

Unique Aspects of Operating a Decentralized Market. Operating in a flexible open access ecosystem comes at a cost. Dealing with untrusted parties means that fair exchange is impossible [8], which raises the question of when to pay servers — before or after providing the service? If paid first, a malicious server may not serve the customer, and if served first, a malicious customer may not pay afterwards.

Moreover, accounting attacks, in which participants collude with each other, pretending that the service has been delivered, could be a hammer that destroys the market. This is a particular problem in systems that require sponsoring service requests. For example, in content distribution, a publisher (e.g., Netflix) can hire caches (or servers) to distribute content to its clients, and hence, it pays for the service. In this case, caches (or servers in general) and clients may collude so that clients pretend to be served, allowing servers to collect payments from the publisher (the sponsor) for free.

These security issues (and many others depending on the service type) require a careful design of a decentralized service-payment exchange protocol that can reduce the risks of dealing with untrusted, possibly colluding parties. Such a protocol represents the backbone of the resource market; if it fails the whole market fails. Servers will not be willing to participate if they are not being paid. The same holds true for customers; they will not be willing to use the system if they pay for a service that they do not receive. Operating the market also requires devising mechanisms for service pricing, term negotiation for server recruiting, and matching protocols to match these servers with interested customers.

Financial and Cryptographic Security Measures. Usually, security threats are mitigated by using cryptographic means (e.g., encryption and digital signatures), algorithmic approaches (e.g., ordering the actions in a way that enforces secure behavior), or a variety of network/systems security techniques (e.g., firewalls and access control mechanisms). Monetary-incentivized systems introduce new types of attacks that cannot be addressed using conventional approaches. In particular, having financially-motivated attackers may require addressing certain threats using financial techniques. These fall into three categories. Detect-and-punish mechanisms, where parties are required to lock penalty deposits that are forfeited if they are caught cheating. Designing algorithms that, if performed maliciously, require larger amounts of resources than when performed honestly. Or devising service pricing and payments mechanisms that make it more profitable on the long run to act honestly in every service request than cheating or ignoring the request (even if cheating or ignoring are not detectable). Such techniques make cheating unprofitable so that rational parties will choose to adhere to the protocol.

For example, to reduce the risks of the impossibility of fair exchange, micropayments can be employed. That is, instead of paying a large chunk of money for the full service, the payment is divided into small values, each of which is exchanged for a small service amount. For instance, one can pay for retrieving a file in small data chunks instead of paying for the full retrieval all at once. Hence, a server loses a small payment if a client does not pay after receiving a chunk. Similarly, a client loses a small payment if it pays in advance and the server does not send a data chunk in return.

On the other hand, to thwart accounting attacks, system designers need to incorporate suitable techniques to prove or confirm resource expenditure, and consequently, confirm that payments are well deserved. For example, for online content delivery, the CAPnet puzzle [5] can be used to ensure that caches have delivered the requested content. And for

file storage, proof-of-replication [9] can be used to prove that a server is still storing the clients' files with the agreed-upon number of replicas. As noted, there is *no one solution fits all*; the service type and its operational costs impact the type pf countermeasures needed to secure its design.

Optimize for Efficiency. Efficiency is an important driving factor of practical adoption and deployment. System designers need to exploit any opportunity to optimize performance. This also involves choosing the right trade-off between security and efficiency in the sense of risk management. That is to say, threats that have high impact need to be prioritized over low impact ones. In addition, looking into alternative cryptographic primitives that are lightweight (or optimizing their implementation) while maintaining the required security guarantees is another effective venue to utilize.

Another important aspect is system scalability—whether it is in terms of interactivity, amount of data exchanged between the participant, or the amount of data logged on the blockchain. A prime example is employing (distributed) probabilistic micropyament schemes [10], [7], [4] to aggregate small transactions into few larger ones before processing. In such scheme, payments take the form of lottery tickets, and only winning tickets are processed in the system with values that compensate properly for the tickets exchanged so far. Another example is batching client requests and replies together to reduce interaction, or even batching work confirmations to reduce the amount of data logged on the blockchain.

Testing and Deployment. To examine the viability of the system, conventional practices of prototyping, benchmarking, and controlled deployment can be used to evaluate both efficiency and resistance to attacks. These provide a starting point to attract early adopters and test the system at a large scale. Nowadays, it is popular in blockchain-based systems to have what is called a testnet, that runs with fake coins, to serve this purpose. This testing stage may inspire designers to revisit specific parts of the system for further optimization based on the results of the conducted experiments, or feedback from the community even after the official launch (or production stage).

3. Conclusion

Decentralized resource markets represent a monetized version of P2P-based systems that evolved to build flexible, equitable, and transparent ecosystems. In order to encourage practical deployment, in this article I put forward a generic framework for designing efficient and secure resource markets. The framework is distilled from experiences in building blockchain-based large-scale systems. In particular, cryptocurrencies and their blockchains have drawn a huge interest recently as a powerful economic tool. This resulted in a rush into implementing novel ideas without a thorough design process, leaving the produced systems vulnerable to sever security risks. The presented framework is believed to provide a useful guiding maps to avoid such pitfalls.

References

[1] Filecoin. https://filecoin.io/.

[2] *The Largest Cryptocurrency Hacks So Far*. https://www.investopedia.com/news/largest-cryptocurrency-hacks-so-far-year/.

[3] Ghada Almashaqbeh, Allison Bishop, and Justin Cappos. Abc: A cryptocurrency-focused threat modeling framework. In *INFOCOM 2019-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2019.

[4] Ghada Almashaqbeh, Allison Bishop, and Justin Cappos. Microcash: Practical concurrent processing of micropayments. In *Financial Cryptography and Data Security*, 2020.

[5] Ghada Almashaqbeh, Kevin Kelley, Allison Bishop, and Justin Cappos. Capnet: A defense against cache accounting attacks on content distribution networks. In *IEEE CNS*, 2019.

[6] Nasreen Anjum, Dmytro Karamshuk, Mohammad Shikh-Bahaei, and Nishanth Sastry. Survey on peer-assisted content delivery networks. *Computer Networks*, 116:79–95, 2017.

[7] Alessandro Chiesa, Matthew Green, Jingcheng Liu, Peihan Miao, Ian Miers, and Pratyush Mishra. Decentralized anonymous micropayments. In *EUROCRYPT*, 2017.

[8] Shimon Even and Yacov Yacobi. Relations among public key signature systems. Technical report, Computer Science Department, Technion, 1980.

[9] Ben Fisch. Tight proofs of space and replication. In *EUROCRYPT*, 2019.

[10] Rafael Pass and Abhi Shelat. Micropayments for decentralized currencies. In *CCS*, 2015.

[11] Jonathan Zittrain, Kendra Albert, and Lawrence Lessig. Perma: Scoping and addressing the problem of link and reference rot in legal citations. *LIM*, 14:88, 2014.