Arachneum: Blockchain meets Distributed Web

Donghyeon Lee
E-mail: phdleedh@gmail.com

August 2016

Abstract

Appearance of Bitcoin raise up evolution in currency. Blockchain database also raise up possibilities to share important data between untrusted peers. In this paper, we propose Arachneum, a decentralized, distributed, and model-view-controller-based web service using blockchain and new privileges model. Therefore, we can enjoy the freedom against censorship of government or organizations, keep transparency of our web services, fork our web services, and create/read/update/delete(CRUD) all model-view-controller(MVC) components dynamically.

1 Introduction

World Wide Web (WWW) has grown significantly, therefore many users can enjoy the freedom. In some country and area, however, the advantages of the WWW is reduced because of censorship of government or private organizations [1]. This paper proposes Arachneum, decentralized and distributed web service that means no central web server exists, just many connected peers interact themselves. Therefore, the government or organizations cannot block central server, and we can enjoy the freedom continually. On the one hand, it can make less or zero cost of our web servers, therefore it can helps people has no special income, such as philanthropic organizations. One different point rather than other distributed web model is it logs all web transactions by blockchain, and the blockchain is publicly announced. Therefore, we can keep transparency of our web services, because all transactions included business logic are publicly announced. And forking our web services are also available freely. By introducing new privileges model, finally, we can CRUD all MVC components dynamically if you had granted proper rights from other administrator, therefore we can keep the quality of web services.

2 Overview of Bitcoin

In this section, we cover an overview of Bitcoin to propose Arachneum, distributed web service that be affected by Bitcoin. While summarizing the overview, we will propose more suitable design for Arachneum.

2.1 Transactions

In Bitcoin whitepaper, they define an electronic coin as a chain of digital signatures. And they propose that each owner transfer the coin to the next by digitally signing a hash of the previous transaction and the public key of the next owner and adding these to the end of the coin, therefore they can verify the signatures to verify the chain of ownership [2, Section 2].

We defines a web service as a chain of digital signatures, and an activity as a transaction. And we propose that each user of the web service transfer the activity to the next. A first transaction is always created by administrator, and its signature is not used.

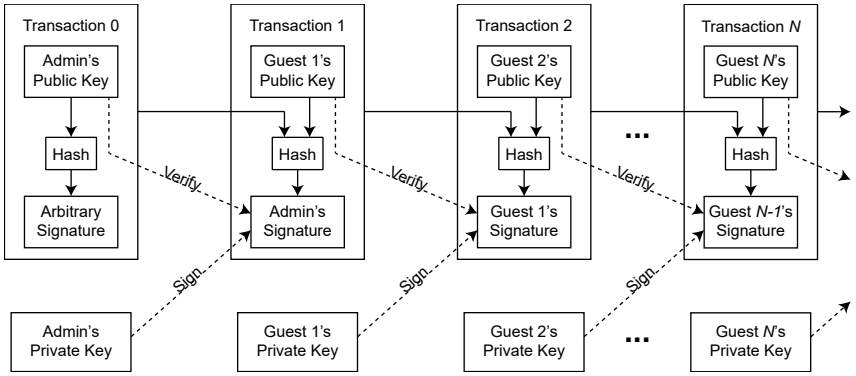

Figure 1: A chain of transactions

In Arachneum, double-spending problem is not occurred, because it's not coin, just web service. Therefore, we can think the solution that introduce timestamp server used in Bitcoin is not required. To log more details for convenience, however, we illustrate our timestamp server.

2.2 Proof-of-Work

To implement a distributed timestamp server, Bitcoin designer had to use a proof-of-work system similar to Adam Back's Hashcash [3]. Bitcoin's proof-of-work involves scanning for a value that when hashed, such as with SHA-256, the hash begins with a number of zero bits [2, Section 4].

Like Bitcoin, we implement the proof-of-work by incrementing a nonce in the block until a value is found that gives the block's hash the required zero bits, therefore it's irreversible. And it can also invalidate allocating many IPs to abuse, because it's one-CPU-one-vote, not one-IP-address-one-vote. Finally, difficulty of proof-of-work is adjustable based on mined blocks per hour [2, Section 4].

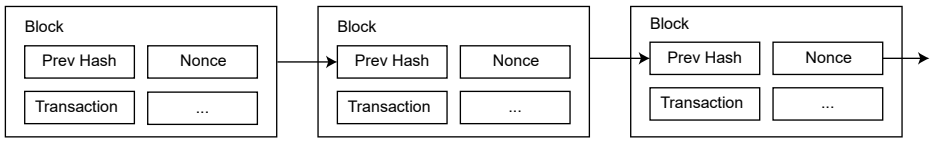

Figure 2: A chain of proved-of-work blocks

2.3 Network

In Arachneum, like Bitcoin, following steps are executed to run the network [2, Section 5]:

1. New transactions are broadcast to all nodes.
2. Each node collects new transactions into a block.
3. Each node works on finding a difficult proof-of-work for its block.
4. When a node finds a proof-of-work, it broadcasts the block to all nodes.
5. Nodes express their acceptance of the block by working on creating the next block in the chain, using the hash of the accepted block as the previous hash.

According to Bitcoin whitepaper, nodes always consider the longest chain to be the correct one and will keep working on extending it. In a situation that peers receive two next block simultaneously, we solve the problem by saving the other branch become longer, and pick a longest branch.

A differential thing rather than Bitcoin is it doesn't check whether the transaction spent, because it's not currency. However, incentive system is also existed in Arachneum, because it requires to encourage volunteer to keep the network. Then, section 4 covers how to encourage volunteer without introducing new currency. Also, we can implement the network features by WebRTC that be compatible with modern web browsers.

3 Model-View-Controller

MVC is suitable for separation of concerns on many applications [4]. In Arachneum, it is used to separate and execute activities. A signatures can involve an activity related to model, view, and controller. Therefore, a chain of transactions is a chain of all activities about a web service. To distinguish a type of an activity either MVC, we should specify the type in the signature.

3.1 Model

A model seems like a row in relational database systems. In fact, many MVC web frameworks implement its a model to be linked with the row. The model can be CRUD by users, through controller, or immediately by administrator. In this paper, we demonstrates a model is a key-value pair.

3.2 View

A view seems like a web page except that it's constructed as not only vanilla HTML. The view also can be CRUD by users, however it's rare by guest, therefore granting proper permissions to users are important.

3.3 Controller

A controller is related to business logic. User always orders to controller, and get outcomes from view. In Arachneum, controller transactions can involve stack-oriented programming language like Forth, object-oriented programing language like JavaScript, or Low Level Virtual Machine (LLVM) intermediate representations as business logic in its signature. Controller also can be CRUD by users, however all activities except read is not many used like View.

Component	Guest	Admin	Related to
Model	R, CUD†	CRUD	A key-value pair
View	R	CRUD	A web page
Controller	R	CRUD	A business logic

Table 1: Comparison of MVC components († only through a controller)

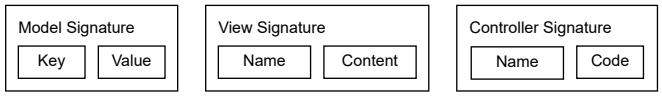

Figure 3: A signature of MVC components

4 Privileges and Incentive

Incentive system in Arachneum is replaced by privileges rather than currency. This means that, to do activity rather than just read, you must grant corresponding privileges by proof-of-work. Incentive system is able to construct by the network's designer. However, following cost equation is most widely used:

$$0 = M_R = V_R = C_R < M_C = M_U = M_D \leq V_C = V_U = V_D = C_C = C_U = C_D \tag{1}$$

In the equation 1, M_R means read activity of each model.

5 Processing of Controller

In the above, we covered about how to introduce blockchain to distributed web, and how to encourage to keep the network by introducing new privileges as incentive. However, there are one ambiguous thing that how to identify an activity as between immediate activity and through a controller. In this section, we'll modify the controller's signature to able to identify it.

By identifying an activity as above, we should modify the MVC signature that we designed. It can solve by append affecting controller in model and view, and append affected model and views in controller.

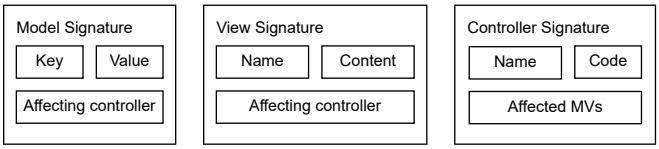

Figure 4: A signature of modified MVC components

6 Fork

Arachneum-based web services can be forked by everyone whoever is admin or guest. To fork, first, a forker is recently updated by all previous blocks and transactions what you want. If the web service be forked is hacked, the forker is required to update the previous transactions before be hacked.

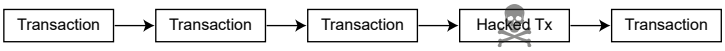

Figure 5: A chain of transactions involved be hacked

Now, the forker can fork the hacked web service to his network. But, all transactions after be hacked are not included. Therefore, we can keep the quality of our web services.

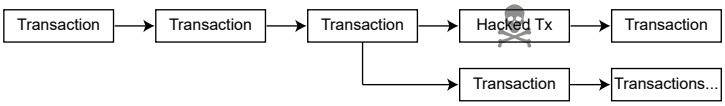

Figure 6: A forked chain of transactions

7 Conclusion

In this paper, we covered an overview of Bitcoin, and introduce it to distributed web. While introducing, we propose new privileges model and MVC components

as Arachneum's structure. If its implementation is appeared, it seems like to Ethereum, called Blockchain 2.0, because of existing of code of signatures [5]. This means that we can program our distributed and publicly announced web service as turing-complete programming language if implementation supports. This will be a light for all citizen that oppressed by the government or private organizations. Therefore, it will be useful to operate poor web service against censorship of government.

References

[1] E. S. Eric and C. Jared, "The future of internet freedom." `http://www.nytimes.com/2014/03/12/opinion/the-future-of-internet-freedom.html`, 2014.

[2] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[3] A. Back *et al.*, "Hashcash-a denial of service counter-measure," 2002.

[4] G. E. Krasner, S. T. Pope, *et al.*, "A description of the model-view-controller user interface paradigm in the smalltalk-80 system," *Journal of object oriented programming*, vol. 1, no. 3, pp. 26–49, 1988.

[5] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum Project Yellow Paper*, 2014.