Extending Secure Data Storage in Decentralized Networks

by

Youngrak Ryu

Matriculation Number 380241

Technische Universität Berlin
School IV - Electrical Engineering and Computer Science
Department of Telecommunication Systems
Service-centric Networking

Master's Thesis Outline

May 18, 2020

supervisor:
Dirk Thatmann

Extending Secure Data Storage in Decentralized Networks

1 State of the Art

Most legacy network systems are composed as centralized architecture, in which the main system afford services to connected users. When users request any information, the front-end application takes over requests of the user and brings data from the connected server or back-end application. This manner takes advantage of development and management. However, as the growing scale of the system, the centralized network encounters difficulties, e.g., network connection instability, DDoS, storage cost with rising scale, and sensitive data management. In contrast to the centralized network system, the decentralized network system has become a big trend to avoid the disadvantages of the centralized network. Distributed Ledger Technology is one of the decentralized network technologies. It is composed of nodes that verify with the cryptographical identifier, then interconnect using the peer-to-peer network, and transmit data between nodes. The system to avoid data modification uses consensus algorithms for reliable nodes. The decentralized storage to share data uses the distributed file system, e.g., InterPlanetary File System (IPFS), [1][2], BitTorrent File System (BTFS)[3], Ethereum Swarm, and BigchainDB. The distributed file systems are characterized by sharing fragmented data as a chunk that is stored in nodes. It relies on distributed hash tables (DHT)[4], which is a lookup service using key and value or list paired. Using encrypted key DHT can get exact values. With distinguish DHT characters, more use cases are extended (e.g., e-Wallet, Digital Identification, Distributed File System, and Peer-to-Peer file sharing). The target scenario to be supported is a UDP based Kademlia system[5], thus as a DHT to extend the ability to store data packets larger than the MTU (1500bytes) distributed[6]. However, UDP has no transmission control, that lost packets cannot be re-requested, so packet fragmentation, i.e., splitting data across packets, becomes a problems[7][8]. Many different consensus algorithms, DHT, and databases have emerged to meet the challenge of distributed systems. In this thesis, a protocol and framework will be developed to help store large amounts of data in a UDP-based DHT. To enable secure decentralized data storage requires comparison and analyzation between other systems. Furthermore, I would like to develop a framework with research findings.

2 Problem

MTU has the limitation of the size of a single packet, 1500bytes. When the client transfers to another node data such as an image file over 1500bytes, the transmission use Packet Fragmentation, which splits packets into smaller pieces, and then fragmented packets are sending to avoid MTU. Fragmented packets are reassembled at the received host. However, the packet fragmentation can occur technical problems, which are packet filtering at the Firewall or NAT router, fragmented packets can be lost in the transmission. UDP cannot control the transmission, so the node cannot guarantee received data. Thus, to complete transmission, the node should keep sending data. Therefore the latency can increase with the retransmission of data. With the data loss problem, the interconnected nodes have trouble synchronizing data. The nodes cannot trust each other, and the reliability problem may arise from unsynchronized data. In Kademlia DHT, participated nodes are identified by number or node ID, which is composed as file hash or keywords. In a security aspect, file hash or keyword for the identification has a lack of functionality and security. It is necessary to find a way to identify participated nodes. The system is feasibly designed to interconnect between various platforms, e.g., PC, smartphone, and IoT. The framework will be developed into cross-compile based. Most IoTs are limited environment and offer minimized resource. Therefore the framework has to be compressed and not to be heavy.

3 Approach

The framework will be developed to solve issues problems. The overall approach in this thesis is to enable the extended DHT system to store data packets larger than MTU. The framework will apply UDP based Kademlia DHT. As improve the transmission ability of the lager data packet, the problem may solve using two different ways. First, transmission control will be developed using other ethernet frames such as the jumbo frame[9]. Jumbo frame can carry up to 9000 bytes of payload. However, enabling the jumbo frame may increase packet loss rates. Therefore checksum base on consensus algorithm should be applied to the framework. The other way is using packet fragmentation. UDP is a message-oriented protocol. Therefore it has no reordering or retransmitting mechanism. The framework using packet fragmentation also needs checksum functionality. Furthermore, the developed framework would like to make a trial of using IPFS and BTFS, and then with the findings, the prototype will be applied. Nodes make a connection via the P2P tunnel to enable data storage in decentralized networks. As establish P2P, between nodes have to verify credentials. Self-Sovereign Identity (SSI)[10] will be applied and uses the Decentralized identifier (DID)[11]. The framework will also implement SSI using DID. The prototype will be developed by ANSI C language for supporting extending various platforms such as IoT devices. For testbed, an environment will be configured using Docker-composed. Virtual nodes will be implemented using Docker-composed, and then the prototype will make tests on the environment.

4 Challenge

The most important challenge is the efficient data transmission between nodes, which are the usability of larger data packet transmission and transmission control. As develop larger data transmission, the framework should control data transmission, e.g., compression, propagation speed, message latency, fragmentation. If as possible, the prototype can combine an extending Kademlia DHT using TCP and UDP-lite[12]. This solution may help the limitation problem of packet size. The decentralized network can extend transmittable feasible data types, e.g., image, the large amount of the dataset, and music, over the existing limitation. Furthermore, nodes of different types should be feasible, connecting each other. The growth of IoT would be the main node platform in the decentralized network. Numerous IoT device nodes will deploy on the network, thus developing the cross-compiling framework is a significant assignment. For reaching the requirement framework will be Ansi C implementation and composed scale for using limited IoT resources as well.

5 Evaluation

The completely developed prototype will be evaluated by demanded procedures. The prototype will make an interoperability test. To evaluate the prototype progresses a simulation of transmission mass data distribution. In the simulation, the between nodes will read or write data through the findings that can be analyzed as pros and cons. The evaluation should analyze the comparison of the implemented prototype between other systems such as IPFS, BTFS, and standard Kademlia by measurement of transmission rate, error rate, latency, propagation speed, and reliability. Through the user test, the developed system can verify more issues. Discovered new challenges or developable problems can be further work.

6 Schedule

• 1/2 month	Research of decentralized network systems and related work
• 1/2 month	Design the prototype
• 1/2 month	Implement the network and other systems
• 1 month	Develop and implement the prototype
• 1 month	Improve the developed prototype
• 1/2 month	Simulation and evaluation
• 2 month	Write thesis

References

[1] (2020) Ipfs documentation - what is ipfs? [Online]. Available: https://docs.ipfs.io

[2] J. Benet, “Ipfs-content addressed, versioned, p2p file system,” *arXiv preprint arXiv:1407.3561*, 2014.

[3] B. Inc. (2020) Btfs documentation - what is btfs? [Online]. Available: https://docs.btfs.io

[4] S. Sivaraja, M. Thiyagarajah, T. Piranavam, C.-H. Lung, and S. Majumdar, “Efficient multiple-keyword search in dht-based decentralized systems,” in *2008 International Symposium on Performance Evaluation of Computer and Telecommunication Systems.* IEEE, 2008, pp. 406–412.

[5] P. Maymounkov and D. Mazieres, “Kademlia: A peer-to-peer information system based on the xor metric,” in *International Workshop on Peer-to-Peer Systems.* Springer, 2002, pp. 53–65.

[6] R. Bonica, F. Baker, G. Huston, B. Hinden, O. Trøan, and F. Gont, “IP Fragmentation Considered Fragile,” Internet Engineering Task Force, Internet-Draft draft-ietf-intarea-frag-fragile-17, Sep. 2019, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-ietf-intarea-frag-fragile-17

[7] “Internet Protocol,” RFC 791, Sep. 1981. [Online]. Available: https://rfc-editor.org/rfc/rfc791.txt

[8] “IP datagram reassembly algorithms,” RFC 815, Jul. 1982. [Online]. Available: https://rfc-editor.org/rfc/rfc815.txt

[9] E. Alliance and B. Kohl, “Ethernet jumbo frames,” 2009.

[10] A. Tobin and D. Reed, “The inevitable rise of self-sovereign identity,” *The Sovrin Foundation*, vol. 29, no. 2016, 2016.

[11] M. Sabadello, K. Den Hartog, C. Lundkvist, C. Franz, A. Elias, A. Hughes, J. Jordan, and D. Zagidulin, “Introduction to did auth,” *Rebooting the Web of Trust VI*, 2018.

[12] L.-E. Jonsson, L. Åke Larzon, G. Fairhurst, S. Pink, and M. Degermark, “The Lightweight User Datagram Protocol (UDP-Lite),” RFC 3828, Jul. 2004. [Online]. Available: https://rfc-editor.org/rfc/rfc3828.txt