International Journal of Control and Automation
Vol. 10, No. 12 (2017), pp.131-138
http//dx.doi.org/10.14257/ijca.2017.10.12.12

Preventing DDoS Attack in Blockchain System Using Dynamic Transaction Limit Volume

Wonwoo Jung[1] and Sooyong Park[1]
[1]*Dept. of Computer Science and Engineering, Sogang University, Seoul, South Korea*
wonwoostar@gmail.com, sypark@sogang.ac.kr

Abstract

Blockchain is a trusting technology that allows nodes participating to share data reliably without a central management system. Blockchain uses the p2p system called 'flooding' as the data transfer method, where the data are defined as a transaction and a block. To prevent unnecessary propagation in such a transmission scheme, all nodes in the block chain check 'memory pool'. In current blockchain, there is no transaction creation rate limit. Since there is no transaction creation rate limit, DDoS attacker can easily attack blockchain using transaction creation. This DDoS attack called 'Overflood attack'. In this paper, to prevent 'Overflood attack', we propose a method to dynamically limit the maximum amount of creation transaction using Least Mean Square.

***Keyword**: Blockchain, p2p system, DDoS, flooding, Least Mean square, flow control*

1. Introduction

1.1. Blockchain Network

Blockchain is a p2p-based trusting technology that allows nodes participating in the p2p network to share data reliably without a central management system. Since all nodes in the blockchain system share the same data through the same process, the integrity of the shared data can be ensured. Bitcoin and Ethereum are electronic money trading platforms built on the blockchain technology, which makes the blockchain technology more popular. Thus, this technology has been used in a variety of fields requiring data trust and integrity, rather than being used only for electronic money transactions.

All nodes participating in a block-chain network share the same data. To share the same data, Blockchain uses the p2p system called 'flooding' as the data transfer method, where the data are defined as a transaction and a block. 'Flooding' is that the node that generated the data transmits to the directly connected node, not transmits data to all nodes of entire network, and the transmitted nodes transmit again to connected nodes. All nodes in blockchain do 'Data validation' and 'Memory pool management' to prevent unnecessary transmission rather than simply re-transmitting transactions and blocks. 'Data validation' is that transaction and block that are transmitted are correct.

1.2. Memory Pool Management

In blockchain, 'Memory pool management' is always performed by all nodes in order to maintain an entire blockchain network. The 'Memory pool' is a memory space that stores temporary transactions before creating a new block with the transactions. All transactions in memory pool are valid transactions through 'Data validation'. A node has its own memory pool. Memory pool management proceeds when the valid transactions and

Received (September 4, 2017), Review Result (December 1, 2017), Accepted (December 7, 2017)

blocks come in.

First, when a node receives a valid transaction, it is compared with transactions stored in memory pool and if received transaction is new transaction, it is stored in memory pool and then transmitted to connected nodes and ignored if the transaction is already received. Second, when a node receives a new block from other node, the node verifies the block, removes the transactions in its memory pool if the same transactions exist in the received block. Last, the transactions that cannot enter the block for certain period of time and remain in memory pool are automatically removed. In this way, all nodes' memory pool continues to store and remove transactions. As a block can only have transactions for a certain period, the memory pool always maintains transactions for a future block.

1.3. DDoS Attack in Blockchain

A Distributed Denial-of-Service (DDoS) attack can occur in this blockchain. New transactions that are created correctly are always stored in memory pool, and only some transactions of a given size can enter in one block. A DDoS attack in blockchain can be accomplished by an attacker who creates an infinite number of correct transactions for a short time and delivers them to the connected nodes. The node receiving the transactions becomes overwhelmed with its own memory pool getting full and cannot store any transaction in own memory pool. As a result, it takes more time to deliver to other connected nodes after memory pool management. Victim nodes can lose their ability as a blockchain node. Also, the memory pool of all nodes filled with transactions that created by the DDoS attack node, which increases the time for a transaction created by a normal node to enter a block. Seriously, transaction can be remain in memory pool for a long period time, then removed from memory pool. Hence, it eventually cannot use the blockchain system. We define this style of DDoS attack as an *overflood attack*. Through the overflood attack, an attacker may not only damage a specific node, but also gradually destroy the entire blockchain system itself.

An overflood attack occurred with Bitcoin in 2015, which resulted in more than 10% reduction in the total number of Bitcoin nodes. Bitcoin proposed a countermeasure against the attack by raising the minimum transaction fee to 5 times the original fee when creating a new transaction. However, raising the minimum transaction fee is not an ultimate solution to an actual overflood attack, though attackers spend 5 times more to commit the attack. For the blockchain system that is not associated with electronic money, this countermeasure is not a preventive measure.

In 2017, the overflood attack also occurred in 'Ethereum Testnet', which is a blockchain network for testing the “Dapp” application and other mining programs before launching them into the real world. In 2017, Ethereum Testnet was taken over by a group of attackers with the overflood attack for a month, making it impossible for Ethereum application developers and mining program developers to experiment. Unlike Bitcoin, the transaction fee was not based on real currency, so the attackers could easily continue the overflood attack for a longer time. As Ethereum Testnet is a completely separate network with Ethereum, the Ethereum developers did not offer a way to prevent overflood attacks.

In current blockchain, when node wants to create a transaction, node can generate infinite transactions for a short time at any time and propagate it to the blockchain network. An attacker node that wants to perform an overflood attack can achieve the purpose of an attack even though there is a limit in generating a transaction for a long time because of a commission in the blockchain in which electronic money and real money are related. In various blockchains that are being studied for the purpose of providing various data integrity and reliability, transactions can be created infinitely without any limit. In this paper, we propose the use of a formula that dynamically sets the volume of transactions that one node can create during a certain period of time. Our formula is based on Least Mean Square formula. Our method is referred to as ‘Transaction limit volume’.

The rest of this paper is organized as follow. In Section 2, we present some references related to our work. In Section 3, we give a detailed description of our proposed approach for prevent overflood attack in blockchain. Section 4 explains the plan for experiments detail about our proposed method in Section 3. And we present the experimental results. Finally, a conclusion and future work are given in Section 5.

2. Related Work

There is no research of overflood attack in the blockchain as defined in this paper. However, various researches have been conducted to prevent DDoS attacks using the above-described attack types in network systems and P2P systems. In this paper, we describe briefly 'Attack Detection and Filtering', which is a method to detect the DDoS attacks and react on the network system, and 'Peer Reputation and Filtering', which is a method to take actions on the system through evaluation of P2P system nodes in a distributed P2P environment. Before explaining the two related studies, we describe ‘Least Mean Square’ which is one of the various formulas and algorithms used to detect DDoS attack in ‘Attack Detection and Filtering’ and basis of our proposed approach.

2.1. Least Mean Square (LMS)

LMS is widely used in 'Attack Detection and Filtering' to predict future values using actual measured values and present predicted values. The LMS passes from the nth to (n+1)th as time interval. The main purpose of the LMS is to minimize the error between the (n+1)th predicted value $\hat{x}_{n+1}$ and the measured value x_{n+1}. For this, the (n+1) th predicted value $\hat{x}_{n+1}$ uses a gradient descent method. The predicted value is calculated using Equation 1. In Equation 1, W_n is the nth weight vector and X_n is the measured value up to nth. The weight vector W_n is calculated using Equation 2. Value μ is the step size, which determines the process characteristics that minimize the error of the LMS's purpose. The larger value of μ, the higher probability that the result of Equation 1 will diverge. There are also studies that determine the optimal step size μ of LMS depending on what characteristics are used. The error between the measured value x_{n+1} and the predicted value $\hat{x}_{n+1}$ is minimized by iteratively calculating Equation 1 and Equation 2 over time.

Equation 1 $\hat{x}_{n+1} = W_n^T * X_n$

Equation 2 $W_{n+1} = W_n + \mu * (x_n - \hat{x}_n) * X_n$

2.2 Detection and Reputation

The 'Attack Detection and Filtering' monitors and analyzes the traffic flow that occurs in the entire network to determine DDoS attack and react it. Detection is determined based on the normal traffic flow pattern when the attack does not occur and determined that the attack is based on the predefined criteria through the traffic analysis. The Least Mean Square described above is one of the methods used to generate a normal traffic flow pattern. The 'Peer Reputation and Filtering' is a method to prevent attack node through the evaluation of the nodes participating in the system by blocking or disadvantage in system itself.

There are limitations when the above-mentioned prevention methods are applied to prevent an ‘Overflood attack' occurring in the blockchain. First, large data sets are required to apply these prevention methods to the blockchain. The system can be heavy because method must always be on the blockchain to detect attack, and prevention methods can become useless depending on the attacker's capabilities. Third, DDoS attackers frequently use 'zombie node' and hiding behind them, these prevention methods will damage to ‘zombie node’. The biggest problem when applied to blockchain is that these methods can not completely decide attacks and non-attacks. There is always a probability of false negatives in these preventions.

3. Proposal

3.1. Attack Defines and Blockchain Model

The 'Overflood attack' described above is a DDoS attack using blockchain transactions. The characteristics of 'Overflood attack' are as follow.

- Attacker can create a transaction at any time. As a result, attacker can change the size of the transactions generate per second.
- Attack can be attempted through 'zombie node' which is the basic form of DDoS attack.

The blockchain to be covered in this study has several features, rather than public blockchain such as Bitcoin and Ethereum. Our blockchain characteristics are as follows.

- All nodes have the same role.
- Network connectivity between nodes is fixed.
- All nodes can be used for 'Overflood attack'.
- The size of all transactions is 1 Kb.

3.2. Transaction Limit Volume approach

We propose the use of a formula that dynamically sets the volume of transactions that one node can create during a certain period of time by modified Least Mean Square. Our prevention approach is considering transaction usability of blockchain node. Maximum size of transaction in certain period time called ‘Transaction Limit Volume’. The one node’s maximum transaction limit volume TLV_{n+1} is calculated by future predicted transaction creation size $\hat{x}_{n+1}$ which it is calculated by current measured transaction creation size x_n and current predicted transaction creation size $\hat{x}_n$. If the blockchain system limits the maximum transaction creation size TLV_n that all nodes can produce during a period T_L, attacker cannot succeed in an overflood attack. The maximum transaction creation limit volume that can be created in the next cycle is determined according to the cumulative transaction size of the previous cycle, rather than fixing the total transaction size that can be created in a certain period.

We describe the symbols and explanations of the variables included in the transaction limit volume formula through the LMS in this research. LMS is iterative formula. The value n is the nth period. The following table will describe the variables in this research proposal.

Table 1. Variable Symbol/Explanation

Name	Unit	Symbol	Explanation
Block period	second	T_B	The time it takes for one block to be generated and then the next block to be created
Limit period	second	T_L	The time it takes for pass nth cycle to (n+1)th cycle. $\frac{T_L}{T_B} = \mathrm{N}, \mathrm{N}$ is natural number
Block size	Kb	S_{B_tx}	The total size of transactions that can enter in one block.
Maximum transaction limit volume	Kb	$\mathrm{T}LV_{\mathrm{n}}$	Total transaction size that can be generated to the maximum during the nth cycle.

Predicted transaction generated size	Kb	$\hat{x}_n$	Total transaction size predicted to be generated during the nth cycle.
Measured transaction generated size	Kb	x_n	Total transaction size actually generated during the nth cycle
Maximum generation rate	%	C	Percentage of maximum transaction creation limit size allowed in predicted transaction generated size, 0-100
Increase/Decrease weight	Constant	W_n	A constant value that determines the predicted transaction generated size at (n + 1) cycle
Surrounding node importance	Constant	sur_I	The average number of nodes connected to other connected nodes
Importance weight	Constant	I	Weight constant value that determines how important the Surrounding node importance is.

3.3. Transaction Limit Volume Formulas

In this paper, nodes of a blockchain individually limit the maximum transaction creation to prevent attack. In order to consider the actual transaction size of the nodes generated, we modify the Least Mean Square formula to apply the blockchain domain. For the LMS, the next period predicted value $\hat{x}_{n+1}$ is calculated by the all measured value $x_1,\ x_2,\ \ldots\ x_n$ until the current period. However, we only use current measured value x_n to calculate predicted value $\hat{x}_{n+1}$. Accordingly, the weight vector W_n has only one weight value which is referred as Increase/Decrease weight.

The maximum transaction limit volume $\mathrm{T}LV_{\mathrm{n+1}}$ is main idea of our research. It is calculated using Equation 3. In Equation 3, the maximum transaction limit volume $\mathrm{T}LV_{\mathrm{n+1}}$ is determined as the upper bound value according to the maximum generation rate C based on the predicted transaction generation size $\hat{x}_{n+1}$. The measured transaction generated size x_{n+1} is always less than or equal to maximum transaction limit volume $\mathrm{T}LV_{\mathrm{n+1}}$. The value C can be expressed as a percentage. As value C approaches 100%, the upper bound value $\mathrm{T}LV_{\mathrm{n+1}}$ becomes closer to predicted value $\hat{x}_{n+1}$.

$$\text{Equation 3}\ \ \mathrm{T}LV_{\mathrm{n+1}} = \frac{\hat{x}_{n+1}}{C}$$

The predicted transaction generation size $\hat{x}_{n+1}$ is calculated using Equation 4 iteratively. We modify Equation 1 to consider node's actual transaction usability. In LMS, only use weight vector W_n and measured value x_n to calculate the predicted value $\hat{x}_{n+1}$. Through the Equation 3, the upper bound of x_n is calculated as a maximum transaction limit volume. However, there is no lower bound of measured transaction generated size. If the size of transaction actually generated during a one period becomes 0, it means that x_n is 0, $\hat{x}_{n+1}$ becomes 0, and furthermore, the value $\mathrm{T}LV_{\mathrm{n+1}}$ also becomes 0. In order to prevent this problem, Equation 1 of LMS is modified and expressed by Equation 4. The key point in Equation 4 is to use $\max(\ )$. Thus, when the measured transaction generated size x_n is smaller than the predicted transaction generated size $\hat{x}_n$, the reduction rate the predicted value $\hat{x}_{n+1}$ of the next cycle becomes smaller than the current predicted value $\hat{x}_n$ can be reduced.

$$\text{Equation 4.}\ \ \hat{x}_{n+1} = \max(W_n * x_n\ , W_n * \hat{x}_n\ )\ ,\ 0 \le x_n \le \mathrm{T}LV_{\mathrm{n}}$$

We only use current measured value x_n to calculate predicted value $\hat{x}_{n+1}$. Accordingly, the weight vector W_n has only one weight value which is referred as

Increase/Decrease weight W_n. The Increase/Decrease weight W_n is calculated using Equation 5. As previously explained in Section 2, the step size μ determine how quickly reduce the error between the value x_n and $\hat{x}_n$.

$$\text{Equation 5. } W_{n+1} = W_n + \mu * (x_n - \hat{x}_n) * x_n$$

The step size μ that determines the degree of change the Increase/Decrease weight W_n is calculated through Equation 6. The variables in Equation 6 are those specialized in the blockchain. We use the value $\frac{1}{x_n * TVL_n}$ to normalize the error value $(x_n - \hat{x}_n) * x_n$ in Equation 5. Also in Equation 6, we use the block period T_B and the limit period T_l to increase degree of change increase/Decrease weight W_n through the form $\frac{T_l}{T_B}$. When a block is created, transactions entered into the block are deleted in the memory pool. Since the variable $\frac{T_l}{T_B}$ is N, the larger number N, the number of blocks generated during the limit period becomes large.

$$\text{Equation 6. } \mu = \frac{1}{x_n * TVL_n} * \frac{T_l}{T_B} * \frac{1}{I * sur_I}$$

Even if the nodes have the same measured transaction generated size x_n, they will have a different predicted transaction generated size $\hat{x}_{n+1}$ through different value of surrounding node importance sur_I. The larger the value of sur_I is, the more nodes connected to the user node are connected to many nodes. This means that the nodes that are connected to many nodes' transactions come directly from many nodes. We use the value $\frac{1}{I * sur_I}$ to protect import node from 'Overflood attack' more strongly. The importance weight I takes into the scalability of our research.

4. Experiment

In this paper, we use simulation to demonstrate that the proposed method is an effective way to prevent DDoS attack in a blockchain system. The blockchain used in the simulation is a self-developed one that goes through the memory pool management process proposed in this paper. We simulate a experiments in each scenario. For smooth experiments, we do not generate a transaction during block generated point. Because the memory pool is accessed by memory pool management to remove transaction at the point of block generate.

The experiment will test that our prevention method can prevent 'overflood attack'. To do this, we measure the time that one transaction is transferred to the entire blockchain system. This measured value is defined as 'Round Trip Time(RTT)'. Also we measure a memory pool size of blockchain node. In three scenarios we will measure 'RTT' and 'Memory pool size'. The three scenarios are:

(1) Non overflood attack: When the transaction generate slowly.

(2) Occurred overflood attack: When the transaction generates very fast.

(3) Occurred overflood attack after our proposal apply: When the transaction generates very fast.

In Scenario 1, the transaction generation rate is fixed at 2 per second. The rate of transaction generation in Scenario 2 and Scenario 3 is fixed at 100 per second. The block size S_{B_tx} is 100kb and block period T_B is 50sec for experiment.

The results of our experiments will be analyzed and presented graphically in this section. The 'full transfer time' and 'memory pool size' to be measured through experiments are shown based on the time axis. The number of blocks generated in accordance with the block cycle T_B is indicated on the horizontal axis. The number of blocks is often used as a time in the blockchain.

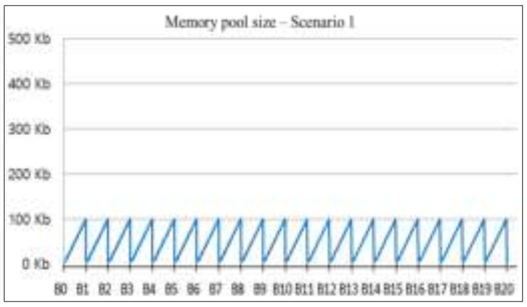

Figure 1. Result Scenario 1

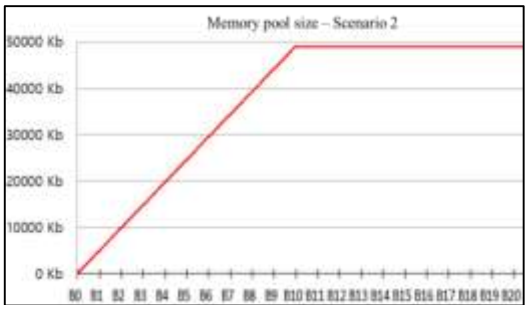

Figure 2. Result Scenario 2

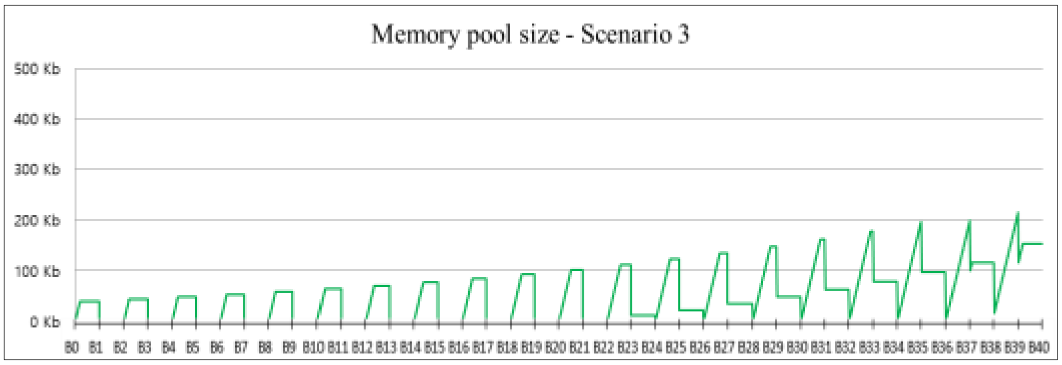

Figure 3. Result Scenario 3

The Figure 1 shows the size of the memory pool when two transactions per second are consistently generated when there is no attack in blockchain. If the total size of the transactions generated during the block period is smaller than one block size, it can be confirmed that the size of the memory pool becomes zero immediately after the block is generated.

The Figure 2 shows the size of the memory pool when an 'overflood attack' occurs in the blockchain. We generate 100 transactions per second for attack. In comparison with the Figure 1, Until a total of 20 blocks are created, the memory pool size of Scenario 1 shows a repetitive pattern between 0 and 100kb. However, the result of Scenario 2 shows that the size of the memory pool grows exponentially. This is an increase in memory pool size due to the 'overflood attack' mentioned in this paper.

The Figure 3 shows the size of the memory pool when 'overflood attack' occurred after applying our proposed method. For the experiment in Scenario 3, We generate 100 transactions per second for attack same as Scenario 2. We set the limit period T_L to two times the block period. Also maximum generation rate C is 75% for Scenario 3. The results show that the size of the memory pool grows as each limit period passes. . In comparison with the Figure 2, the size of the memory pool exceeded 40000kb during the time that 10 blocks were created. However in Scenario 3, it is impossible for the size of the memory pool to exceed 40000kb in short time. Also compare with Figure 1 which is the result of Scenario 1, the size of the memory pool is similar to Figure 1.

6. Conclusion and Future Work

If we apply the proposed method to blockchain to limit the total transaction size, the usability of the nodes is not good initially, but it gradually improves usability according to the transaction generation behavior of each node. The rate of increase or decrease in the total size limit of a transaction in our formula can vary depending on various blockchain systems. At the most extreme, we can fix the Maximum transaction limit volume TLV_{n+1}. The change ratio of Maximum transaction limit volume can be customized to satisfy the convenience part of the user. Our approach can prevent ‘Overflood attack’ in blockchain. As a result, attackers attempting an 'Overflood attack' cannot succeed due to the maximum transaction limit volume, and the following maximum transaction generation limit size is iteratively changed according to the transaction generation size during the previous cycle.In the future, when a blockchain is used for various data sharing, the formula which we proposed in this paper will be needed.

In this paper, we only used the measured transaction generated size x_n when calculation the predicted transaction generated size $\hat{x}_{n+1}$. In future work, we will be use average of all the measured values x_1, x_2, … x_n for calculating the predicted value $\hat{x}_{n+1}$. Then we will test whether it is appropriate when applied to a blockchain. Also, we will continuously research the Equation 6 for various type of blockchain. Through continuous research, the Equation 6 may be revised to more optimize. In order to test the scalability of our approach, we will test our approach by increasing the number of nodes more than hundreds. In the course of research on scalability, we will research the tracing absolute path among the nodes in network layer for dynamic network topology.

Acknowledgments

“This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program, (IITP-2017-0-01628) supervised by the IITP (Institute for Information and communications Technology Promotion)”.

References

[1] Y. Liu, X. Liu, C. Wang and L. Xiao, “Defending P2Ps from overlay flooding-based DDoS”, In Parallel Processing, 2007, ICPP 2007, International Conference on, IEEE. **(2007)**, pp. 28-28.

[2] L. Wang, “Attacks against peer-to-peer networks and countermeasures”, In T-110.5290 Seminar on Network Security, **(2006)**.

[3] J. Mirkovic and P. Reiher, “A taxonomy of DDoS attack and DDoS defense mechanisms”, ACM SIGCOMM Computer Communication Review, vol. 34, no. 2, **(2004)**, pp. 39-53.

[4] R. K. Chang, “Defending against flooding-based distributed denial-of-service attacks: a tutorial”, IEEE communications magazine, vol. 40, no. 10, **(2002)**, pp. 42-51.

[5] J. Yang, Y. Li, B. Huang and J. Ming, “Preventing DDoS attacks based on credit model for P2P streaming system”, Autonomic and Trusted Computing, **(2008)**, pp. 13-20.

[6] N. Naoumov and K. Ross, “Exploiting p2p systems for ddos attacks”, In Proceedings of the 1st international conference on Scalable information systems, **(2006)**, p. 47.

[7] O. Salem, A. Makke, J. Tajerand and A. Mehaoua, “Flooding attacks detection in traffic of backbone networks”, In Local Computer Networks (LCN), 2011 IEEE 36th Conference on, IEEE, **(2011)**, pp. 441-449.

[8] J. Yang, Y. Li, B. Huang and J. Ming, “Preventing DDoS attacks based on credit model for P2P streaming system”, Autonomic and Trusted Computing, **(2008)**, pp. 13-20.

[9] V. A. Siris and F. Papagalou, “Application of anomaly detection algorithms for detecting SYN flooding attacks”, In Global Telecommunications Conference, 2004, GLOBECOM'04. IEEE Vol. 4, **(2004)**, pp. 2050-2054.