블록체인 기반의 안전한 소프트웨어 정의 네트워킹

우승원* · 신승원**

한국과학기술원

Blockchain-based Secure Software-Defined Networking

Seungwon Woo* · Seungwon Shin**

KAIST

E-mail : seungwonwoo@kaist.ac.kr / claude@kaist.ac.kr

요 약

소프트웨어 정의 네트워킹(SDN, Software-Defined Networking) 기술은 기존의 네트워크 장비와는 다르게 중앙 집중화된 SDN 컨트롤러가 다수의 스위치를 관리하여 어떠한 네트워크 요구사항도 쉽게 적용할 수 있는 차세대 네트워크 기술이다. 하지만 최근 몇 년간 SDN에 대한 연구가 급격하게 진행되면서 이에 대한 보안 문제도 중요하게 여겨지고 있다. 따라서 본 논문에서는 SDN에서 가능한 주요 보안 문제들을 조사하고, 이를 해결할 수 있는 방안으로 블록체인(Blockchain) 기술을 SDN에 적용할 수 있는 방법론을 소개한다.

ABSTRACT

Software-Defined Networking, called SDN is a next-generation network technology that allows a centralized SDN controller to manage multiple switches and easily apply any network requirements, unlike traditional network devices. However, as the research on SDN has progressed rapidly in recent years, the security of SDN is also considered to be important. Therefore, in this paper, we investigate the major security problems in SDN and introduce a methodology to apply blockchain technology to SDN as a solution to solve them.

키워드

Software-Defined Networking, SDN, Blockchain, SDN Security

Ⅰ. 서 론

소프트웨어 정의 네트워킹(SDN, Software-Defined Networking) 기술은 기존의 스위치, 라우터와 같은 전통적인 네트워크 장비와는 다르게 데이터를 어떻게 전달할지 결정하는 제어평면(Control Plane)과 제어평면에서 결정된 규칙을 기반으로 데이터를 실제로 전달해주는 데이터평면(Data Plane)으로 분리되어 네트워크 서비스들을 제공하는 기술이다. 이렇게 분리된 제어평면과 데이터평면은 OpenFlow [1] 라는 프로토콜을 통해 통신하며 하나 또는 다수의 SDN 컨트롤러가 여러 스위치를 관리하는 중앙 집중화된 구조로 구성되어 있다. 기본적인 흐름은 다음과 같다. 먼저 데이터평면에 위치한 SDN 컨트롤러가 관리하는 스위치에 패킷이 들어오면 가장 먼저 스위치의 플로우 테이블을 확인한다. 만약 플로우 테이블에 들어온 패킷과 일치하는 플로우 규칙이 없다면 OpenFlow 프로토콜을 이용해 SDN 컨트롤러에게 해당 패킷을 전달한다. SDN 컨트롤러는 해당 패킷을 받아 어떻게 처리할지 결정한 뒤에 스위치에게 플로우 규칙을 전달하면 스위치는 해당 플로우 규칙을 플로우 테이블에 저장하고 그 이후의 패킷들은 SDN 컨트롤러를 거치지 않고 처리하게 된다.

SDN은 이러한 중앙 집중화된 구조로 수많은 스위치를 관리하기 때문에 다양한 네트워크 서비스들을 현재 네트워크 상태에 맞게 제공할 수 있으며 이러한 네트워크 기능들은 SDN 컨트롤러의 어플리케이션의 형태로 구현할 수 있다. 이러한 SDN

* speaker

** corresponding author

의 특징으로 기존에 비해 방화벽, IDS, IPS와 같은 서비스를 보다 유연하게 제공할 수 있기 때문에 많은 네트워크 사용자들에게 높은 보안성을 제공할 수 있다 [10, 11]. 하지만 SDN 환경에서는 이러한 중앙 집중화된 구조로 인해 SDN 컨트롤러에 문제가 생기면 전체 네트워크가 마비되는 치명적인 문제가 발생할 수 있다 [12, 13]. 따라서 다양한 오픈소스 SDN 컨트롤러들 [2, 3]은 분산 컨트롤러를 지원함으로써 이러한 문제를 해결하였으나 이로 인해 분산 컨트롤러들 간의 데이터 동기화, 불일치 등의 문제를 야기할 수 있다 [4, 8, 9]. 그 밖에도 SDN에서 발생할 수 있는 여러 공격 벡터들을 연구하거나 이를 대상으로 실제 공격들을 테스트 하는 등 SDN 보안에 관련된 여러 연구들이 활발하게 진행되고 있다 [14, 15, 16, 17]. 따라서 본 논문에서는 이러한 SDN에서 발생 가능한 보안 문제를 해결하고자 블록체인(Blockchain) 기술을 SDN에 도입하여 데이터를 저장 및 전달하는 방법을 제안한다.

II. 배경지식

SDN 환경에서의 여러 스위치들은 하나의 SDN 컨트롤러로 관리할 수 있지만 여러 SDN 컨트롤러들을 통해 관리할 수 있다. 분산 SDN 컨트롤러를 사용하는 가장 큰 이유는 장애 허용성(Fault-Tolerance)이다. SDN 환경에서의 가장 큰 단점은 중앙 집중화된 SDN 컨트롤러가 DDoS와 같은 네트워크 공격을 당하거나 컨트롤러 내부 로직의 문제로 오류가 발생되어 중지되는 경우, 해당 컨트롤러가 관리하고 있는 스위치들은 새로운 패킷에 대해 어떠한 처리도 할 수 없으며 이는 전체 네트워크에 심각한 영향을 줄 수 있다. 이러한 문제를 해결하고자 분산 SDN 컨트롤러를 도입해 하나의 컨트롤러가 문제가 생겨 종료되는 경우 다른 컨트롤러가 종료된 컨트롤러를 대신하여 스위치들을 관리할 수 있다. 이러한 분산 컨트롤러를 지원하기 위해서는 다수의 컨트롤러가 서로 통신하면서 동일한 데이터를 가지고 있어야 한다. 기존에는 동일한 데이터를 유지하기 위해 하나의 데이터베이스를 만들어 여러 컨트롤러가 이 데이터베이스에 접근해 플로우 규칙이나 스위치 정보 등을 관리하였다. 하지만 이 경우 하나의 데이터베이스에 다수의 컨트롤러가 접근하기 때문에 속도가 제한되며 기존과 동일하게 이 데이터베이스에 문제가 생기면 분산 컨트롤러가 제공하는 이점이 사라지게 되는 문제점이 있다. 다른 방법으로는 각각의 분산 컨트롤러마다 동일한 데이터를 유지하는 방법이 있다. 하지만 이 경우에는 특정 컨트롤러가 악의적으로 데이터를 수정하거나 삭제할 수 있는 문제점이 존재한다. 따라서 본 논문에서는 이러한 문제점을 해결하고자 다수의 분산 컨트롤러가 동일한 블록체인을 가지고 데이터를 저장하는 방법을 제안한다.

블록체인은 최근 비트코인 [5]과 이더리움 [6]과 같은 가상화폐가 가능하게 된 근본적인 기술이다. 블록체인은 명칭이 의미하는 바와 같이 하나의 블록이 이전 블록을 참조하면서 하나의 체인을 이루는 구조로 구성되어 있다. 블록체인에서의 하나의 블록 안에는 관리하고자 하는 데이터를 저장할 수 있으며 이러한 데이터의 추가, 삭제, 수정과 같은 행위를 하나의 트랜잭션이라는 단위로 저장하게 된다. 가상화폐의 경우로 예를 들면, 비트코인이나 이더리움에서 사용하는 코인이 하나의 데이터이며 이러한 코인(화폐)을 특정 소유자에게 전달하거나 새로 생성하는 등의 행위를 트랜잭션에 저장하게 된다. 일반적으로 하나의 블록 안에는 여러 트랜잭션의 정보와 타임스탬프, 그리고 이전 블록의 해쉬값으로 구성되어 있다. 처음 블록의 경우 이전 해쉬값은 존재하지 않으며 그 이후의 블록이 생성될 때마다 이전의 해쉬값을 포함해 블록을 구성한다. 이렇게 생성된 블록체인은 P2P 방식을 기반으로 네트워크에 속한 많은 사람들에게 전달된다. 이러한 블록체인의 특징으로는 탈중앙화된 구조로 네트워크에 속한 모든 노드들이 동일한 정보의 데이터를 소유하고 있기 때문에 중앙 집중형 데이터베이스를 관리할 필요가 없다. 또한 악의적인 사용자가 특정 데이터를 수정해도 해당 데이터에 대한 정보를 다수의 다른 노드들이 검증할 수 있으며 이전 블록에 쓰여진 데이터를 변경하면 해쉬값의 특성상 전체 해쉬값이 변경되므로 데이터의 변경에 대해 쉽게 알아차릴 수 있다. 따라서 블록체인의 특성상 이전 기록에 대한 정보의 변경이 불가능하다. 새로 생성되는 블록에 있어서는 네트워크에 속한 모든 노드들이 받아들일 수 있는 합의 알고리즘을 이용해 해당 알고리즘에 일치하는 경우에 특정 블록을 생성할 수 있으며 생성된 블록은 인터넷을 통해 전체 노드들에게 전달된다. 이와 같은 합의 알고리즘은 PBFT [7], PoW(Proof-of-Work), PoS(Proof-of-Stake) 등 다양한 합의 알고리즘이 있다. 이러한 블록체인 기술은 가상화폐 뿐만 아니라 여러 비즈니스 모델에도 사용되고 있으며 현재 많은 기업들이 블록체인 기술의 도입을 고려하고 있는 추세이다. 따라서 본 논문에서는 이러한 블록체인 기술을 SDN에 적용하는 방법을 제안한다.

III. 본 론

본 논문에서 제안하는 블록체인 기반 SDN은 기존의 분산 SDN 환경에서 발생 가능한 3가지 보안 문제들을 해결하고자 한다.

첫 번째는 장애 허용성(Fault-Tolerance)을 유연하게 제공할 수 있다. SDN 컨트롤러는 DDoS와 같은 네트워크 공격이나 컨트롤러 또는 내부 어플리케이션의 잘못된 구현으로 인해 컨트롤러가 동작하지 않는 등의 내부 로직의 문제로 적절한 네트워크 서비스들을 제공하지 못할 수 있다. 따라서 이러한 문제가 발생했을 때 다른 SDN 컨트롤러가

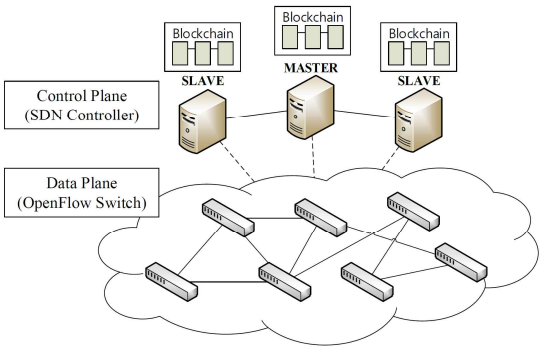

그림 1. 블록체인 기반 분산 SDN 구조

효율적으로 대체할 수 있다.

두 번째는 악의적인 컨트롤러 및 어플리케이션으로 인한 악성 행위를 방지하기 위해 검증(Verification) 과정을 제공할 수 있다. 기존의 분산 SDN 환경에서는 공격자가 악의적인 컨트롤러나 어플리케이션을 설치하여 네트워크 관리자가 원하지 않은 플로우를 설치하거나 특정 IP주소를 차단하는 등 악성 행위를 할 수 있다. 따라서 이를 위해 다른 분산 SDN 컨트롤러가 각 행위에 대해 검증함으로써 이러한 문제를 방지하는 메커니즘을 제공할 수 있다.

마지막으로 네트워크 서비스의 문제가 생겼을 때의 디버깅(Debugging)을 효율적으로 제공할 수 있다. 본 논문에서는 제안하는 블록체인의 트랜잭션으로 추가/변경 되는 네트워크 리소스에 대해 타임스탬프와 함께 기록할 수 있다. 뿐만 아니라 블록체인에 데이터가 저장되면 이를 수정할 수 없기 때문에 악의적인 컨트롤러가 과거 데이터를 수정할 수 없다. 따라서 이러한 수정 불가능한 과거 이력을 통해 효율적으로 디버깅 할 수 있다.

블록체인 기반의 분산 SDN 구조는 크게 그림 1과 같은 구조를 가진다. 본 논문에서는 그림 1의 구조를 크게 분산 SDN 컨트롤러, 블록체인, 트랜잭션으로 나누어 설명한다.

분산 SDN 컨트롤러: 분산 SDN 컨트롤러의 경우에는 MASTER-SLAVE의 구조를 갖는다. 여기서

Key	Value
App	appId, appName, ...
Device	deviceId, type, version, ...
Flow	flowId, priority, deviceId, timeout, match, actions, ...
Group	groupId, type, actions, ...
Meter	meterId, type, band, ...
Host	hostId, mac, ipAddr, ...

표 1. Switch 타입의 트랜잭션 Key/Value 목록

MASTER 컨트롤러는 반드시 단 하나의 SDN 컨트롤러만을 지칭하고 SLAVE 컨트롤러는 MASTER 컨트롤러가 아닌 모든 SDN 컨트롤러들이며

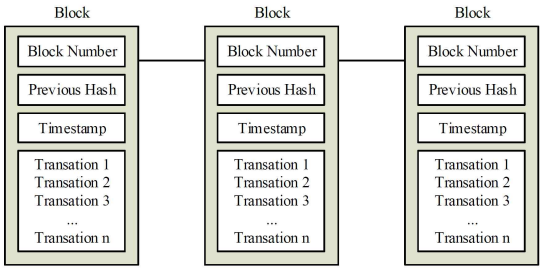

그림 2. 블록체인 기반 분산 SDN에서 사용하는 블록체인 구조

MASTER와 동일한 네트워크 기능들을 제공할 수 있는 Replica이다. 따라서 MASTER에서 동작하고 있는 SDN 어플리케이션이 그대로 SLAVE에서 동작하고 있어야 한다.

블록체인: 모든 SDN 컨트롤러는 동일한 블록체인을 가진다. 전체 블록체인의 구조는 그림 2와 같다. 먼저, 하나의 블록에는 이전 블록의 해쉬 값, 블록넘버, 타임스탬프, 그리고 트랜잭션들로 구성되어 있으며 블록은 오직 MASTER 컨트롤러만 생성해 블록체인에 추가할 수 있다. 블록 생성 시간의 경우에는 네트워크 관리자가 설정할 수 있으며 생성 시간 내에 아무런 트랜잭션이 발생하지 않았다면 블록을 생성하지 않고 다시 설정한 시간만큼 기다린다.

트랜잭션: 하나의 블록 안에는 여러 개의 트랜잭션을 가지고 있으며 하나의 트랜잭션은 Type, Key/Value, State, 그리고 Timestamp를 가진다. 가장 먼저 Type은 Controller, Switch 두 가지 유형을 가진다. Controller의 경우, 어떤 컨트롤러가 MASTER인지 SLAVE인지 정의할 수 있는 트랜잭션이다. 이때의 Value는 각 컨트롤러의 IP주소이며 MASTER는 하나의 Value만을 가지고 SLAVE는 여러 Value를 가질 수 있다. Switch의 경우, Key로 여섯 가지(App, Device, Flow, Group, Meter, Host)를 가질 수 있으며 각 해당하는 리소스에 맞는 데이터를 Value로 가진다. Switch 타입의 각 Key와 이에 해당하는 Value는 표 1과 같다. 마지막으로 State는 ADD 또는 REMOVE 두 가지 상태를 가지며 각 리소스가 추가되고 삭제될 때 마다 하나의 상태로 블록체인에 저장된다.

정리하면 MASTER 컨트롤러는 MASTER 컨트롤러를 포함한 모든 SDN 컨트롤러의 정보나 스위치 자체의 정보를 블록체인에 업데이트 하거나 OpenFlow 메시지를 통해 스위치 내부의 플로우 테이블, 그룹 테이블 등을 업데이트 하고 트랜잭션을 만들어 블록을 구성해 블록체인에 업데이트 할 수 있다. 또한 나머지 SLAVE 컨트롤러들은 스위치와 주고받는 OpenFlow 메시지를 통해 내부적으로 어떠한 리소스가 업데이트 되는지 확인하고 새로 생성된 블록 안의 트랜잭션들과 일치하는지 확인하는 검증 과정을 거친다. 이때 총 SLAVE 컨트롤러의 절반 이상이 동일한 결과를 가져야 하며 만약

MASTER나 SLAVE가 잘못된 행위를 하는 경우에는 해당 컨트롤러를 악성 컨트롤러로 판단하여 제외시킨다. 이를 통해 장애 허용성(Fault-Tolerant)과 검증(Verification)을 보다 효율적으로 제공할 수 있으며 추가적으로 이러한 블록체인은 특정 리소스에 대해 히스토리 뿐만 아니라 악의적인 사용자가 위조할 수 없는 무결성 또한 제공하기 때문에 신뢰할 수 있는 디버깅(Debugging)을 가능하게 만든다.

Ⅳ. 결 론

본 논문에서는 블록체인을 이용해 SDN 환경에서의 분산 컨트롤러를 제공하기 위한 새로운 방법론을 제안하였다. 기존 연구와의 가장 큰 차이점은 현재까지 나와 있는 다양한 분산 SDN 컨트롤러의 경우 분산 데이터베이스를 사용하는 것이 일반적인 반면 본 논문에서는 처음으로 블록체인을 이용해 분산 컨트롤러 간 데이터를 공유하는 방법을 소개하였다. 따라서 본 논문에서의 블록체인 기반의 SDN은 장애 허용성(Fault-Tolerant), 검증(Verification) 그리고 디버깅(Debugging)을 제공함으로써 기존에 발생하는 보안 문제점들을 효율적이고 유연하게 처리할 수 있다.

Acknowledgment

본 논문은 2018년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No. B0190-16-2012, 글로벌 SDN/NFV 공개소프트웨어 핵심 모듈/기능 개발)

References

[1] McKeown, Nick, et al. "OpenFlow: enabling innovation in campus networks." ACM SIGCOMM Computer Communication Review 38.2, 69-74, 2008.

[2] Berde, Pankaj, et al. "ONOS: towards an open, distributed SDN OS." Proceedings of the third workshop on Hot topics in software defined networking. ACM, 2014.

[3] Medved, Jan, et al. "Opendaylight: Towards a model-driven sdn controller architecture." 2014 IEEE 15th International Symposium on. IEEE, 2014.

[4] Koponen, Teemu, et al. "Onix: A distributed control platform for large-scale production networks." OSDI. Vol. 10. 2010.

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

[6] Wood, Gavin. "Ethereum: A secure decentralised generalised transaction ledger." Ethereum project yellow paper 151, 1-32, 2014.

[7] Castro, Miguel, and Barbara Liskov. "Practical Byzantine fault tolerance." OSDI. Vol. 99. 1999.

[8] Katta, Naga, et al. "Ravana: Controller fault-tolerance in software-defined networking." Proceedings of the 1st ACM SIGCOMM symposium on software defined networking research. ACM, 2015.

[9] Panda, Aurojit, et al. "SCL: Simplifying Distributed SDN Control Planes." NSDI. 2017.

[10] Yoon, Changhoon, et al. "Enabling security functions with SDN: A feasibility study." Computer Networks 85, 19-35, 2015.

[11] Shin, Seungwon, and Guofei Gu. "CloudWatcher: Network security monitoring using OpenFlow in dynamic cloud networks (or: How to provide security monitoring as a service in clouds?)." Network Protocols (ICNP), 2012 20th IEEE International Conference on. IEEE, 2012.

[12] Shin, Seungwon, et al. "Rosemary: A robust, secure, and high-performance network operating system." Proceedings of the 2014 ACM SIGSAC conference on computer and communications security. ACM, 2014.

[13] Shin, Seungwon, et al. "Avant-guard: Scalable and vigilant switch flow management in software-defined networks." Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security. ACM, 2013.

[14] Kreutz, Diego, Fernando Ramos, and Paulo Verissimo. "Towards secure and dependable software-defined networks." Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking. ACM, 2013.

[15] Yoon, Changhoon, et al. "Flow wars: Systemizing the attack surface and defenses in software-defined networks." IEEE/ACM Transactions on Networking 6, 3514-3530, 2017.

[16] Lee, Seungsoo, et al. "DELTA: A Security Assessment Framework for Software-Defined Networks." NDSS. 2017.

[17] Jero, Samuel, et al. "BEADS: Automated Attack Discovery in OpenFlow-Based SDN Systems." International Symposium on Research in Attacks, Intrusions, and Defenses. Springer, Cham, 2017.