SDN을활용한네트워크장애방지및탐지시스템기술동향에대한연구

강희도, 윤창훈, 신승원

한국과학기술원 정보보호대학원

{kangheedo, chyoon87,claude}@kaist.ac.kr

A Study on the trend of network error prevention and detection systemusingSDN

Heedo Kang, Changhoon Yoon, Seungwon Shin

Department of Graduate School of Information Security, KAIST

요 약

본 논문은 Software-defined-networking(SDN)을 활용한 네트워크 장애 방지 및 탐지시스템에 관한 국제 연구 동향 분석을 제시한다. 최근 SDN은 네트워크의 control-plane과 data-plane의 분리를 통해 programmable한 네트워크 환경을 제공함으로써 현재 네트워크 구조로는 구현하기 힘든 다양한 기능들을 네트워크 어플리케이션을 통해 지원한다. 그 중 하나가 네트워크 장애방지 및 탐지 시스템이다. 네트워크 장애 방지 및 탐지시스템은 크게 동적시스템과 정적시스템으로 나뉘는데, 본 논문에서는 네트워크 장애 방지 및 탐지 시스템에 대한 국제 연구들 중 가장 영향력 있는 동적분석 시스템과 정적분석 시스템을 하나씩 선택하여 자세한 분석을 하였으며, 결론으로 가상네트워크 환경(mininet)과 자체 오픈플로우 테스트베드에서의 실증을 통해 두 시스템을 비교분석하여 장단점과 그에 대한 타당성을 제시하였다.

Ⅰ. 서 론

현재 네트워크는 여러 프로토콜이 혼합되어있고 이 프로토콜들이 복잡하게 상호작용을 하며 네트워크를 구성하고 있다. 그에 따라 패킷전송이 복잡한 과정으로 이루어지고, 라우터나 스위치 같은 수천 개의 네트워크 장비들이 서로 다른 벤더들로부터 상호 작용하고 있다. 특히 복잡한 네트워크에서는 네트워크의 관리가 힘들뿐만 아니라 네트워크에서 발생하는 문제점들에 대해 예측하기가 힘든 상황이다. 따라서 이러한 네트워크의 correctness, security, fault tolerance를 보장하기 위해서는 많은 노력이 필요하다. 하지만 looping, black hole 같은 network state에서의 문제는 현재 네트워크에서 빈번하게 일어나고 있다. 현재 이러한 네트워크상에서 발생하는 문제를 분석하는 툴이 몇몇 존재하긴 하지만, 각 툴들은 프로토콜에 의존적이고 각 작업에 특화되어있다는 문제점을 가지고 있다. 최근 대두 되고 있는 SDN은 이러한 문제해결을 조금 더 쉽게 해준다. SDN은 네트워크의 control-plane, data-plane을 분리시켜 control-plane 기능을 하는 하나의 controller에서 data-plane기능을 하는 네트워크 장비들을 모두 관리할 수 있게 해준다.[1] 때문에 네트워크의 모든 정보들을 controller하나에서 수집이 가능하다. 또한 programmable 하기 때문에 네트워크 어플리케이션을 만들기 쉽다. 그에 따라 네트워크망의 장애 탐지를 쉽게 해줄 수 있으며, SDN을 활용한 네트워크 장애방지 및 탐지 시스템은 여러 연구가 진행되어왔다[2-6]. 연구되어진 장애탐지 시스템은 크게 동적 분석과 정적분석으로 나눠지는데, 현재로써 각각은 장단점을 가지고 있다.

본 논문에서는 SDN을 활용한 네트워크 장애방지 및 탐지 시스템에 대한 국제적 연구를 분석하고 이들 중 가장 영향력 있는 동적 분석 프레임워크 모델인 Veriflow와, 네트워크 정적 분석 프레임워크 모델인 Header Space Analysis(Hassel)을 분석하였으며, 결론으로 자체 오픈플로우 테스트베드와 가상네트워크 환경(mininet)을 이용한 실증을 통해 두 시스템의 실제 비교분석을 통해 장단점과 그에 대한 타당성을 제시한다.

Ⅱ. 본론

현재 SDN을 활용한 네트워크 장애방지 프레임워크 모델들은 크게 정적분석과 동적분석으로 나뉘어진다. 정적분석에서는 주로 data-plane을 verification하는 FlowChecker[2] Anteater[3],Header Space Analysis[4], 와 같은 모델이 연구되어지고 있으며 동적분석에서는 controller와 network device 사이에서 proxy형태로 controller message들을 가로채어 network state를 verification하는 Veriflow[5], Header Space Analysis[3]를 동적분석 시스템으로 발전시킨Netplumber[6]와 같은 연구가 진행되고 있다. 이와같은 SDN 환경에서의 네트워크 장애방지, 탐지 연구들은 현재 해외에서 활발하게 연구가 진행되어지고 있다. 본론에서는 이중에서, 정적분석 연구의 대표적인 Header Space Analysis[4]과 동적분석 연구의 대표적인 Veriflow[5]를 자세하게 분석하여 SDN을 활용한 네트워크 장애방지 프레임워크 모델의 현주소를 알아보고자 한다.

Ⅱ-Ⅰ. Header Space Analysis

Header space analysis framework는 네트워크의 구성정보를 적절히 파싱한 후 테스트 패킷을 삽입하여, 패킷을 geometric space의 하나의 점으로 표시하고 각 점들이 라우터나 스위치 등의 네트워크 디바이스의 transfer function을 통해 변해가는 것을 연산하여 네트워크 문제를 찾아내는 Geometric model을 기반으로 하고 있다. 위에서 언급한 점들은, 패킷의 헤더 부분을 단순히 0과 1로 이루어진 길이L의 비트스트림으로 보고 이를 L 차원 space의 점으로 표현한 것이다. 0과 1로 이루어진 패킷의 헤더 하나가 점 하나로 표시되므로, 패킷의 flow는 하나의 region으로 표

현될 수 있으며, 본 논문에서는 헤더에 0,1 뿐만 아니라 wildcard인 x도 허용하기 때문에 wildcard expression은 L dimension space에서 hypercube 형태로 표현되게 된다. Transfer function은 일종의 rule로, 라우팅테이블의 엔트리와 유사하다. 조금 다른 점은 helper function도 지원하여 어떠한 포트로 해당 헤더정보가 들어오면, 그에 맞는 아웃풋포트로 원하는 헤더정보를 변환할 수 있다. 멀티캐스트를 위해 여러 가지 아웃풋 포트를 가질 수도 있는데, 이러한 transfer function을 라우터나 스위치 등 네트워크 디바이스들에 적용함으로써 패킷이 네트워크 장비들을 지날 때마다 헤더정보가 바뀌게 된다. 이러한 프로세스를 기반으로 reachability analysis, loop detection, network slice isolation을 수행한다.

Ⅱ-Ⅱ. Veriflow

Veriflow는 네트워크 컨트롤 어플리케이션, 장비, 관리자가 발생시키는 모든 네트워크 업데이트 이벤트들을 controller와 네트워크 장비사이에서 감시하여 network invariants를 검사한다. 업데이트가 발생했을 때 전체 네트워크를 검사하는 것은 비효율적이고 실시간을 제공하기 어려우며, 새로운 flow rule은 현재 존재하는 전체 rule의 작은 부분에만 영향을 미치기 때문에 Veriflow는 새로운 flow rule에 의해 영향을 받는 현재 존재하는 rule들만을 검증한다. 이를 위해 Veriflow는 네트워크를 equivalence class단위로 나누었다. equivalence class는 간단히 말해 rule들의 매칭필드 네트워크범위가 서로 겹치지 않도록 네트워크를 쪼개는 것이다. Veriflow는 이렇게 나뉘어진 equivalance class에 대한 rule들을 효율적으로 검사,저장,탐색하기 위하여 multidimensional prefix tree(trie)데이터 구조를 사용하였다. trie에서 각 노드는 rule이 매치될 수 있는 3가지 가능한 값들(0,1,*(wildcard))을 가진다. trie에서 각 level은 forwarding rule에서의 비트들을 의미하며 trie는 몇몇 sub-trie나 dimension이 계층적으로 이루어져있다. forwarding rule의 매칭rule은 여러 가지를 가질 수 있으므로(예를 들어 source ip, destination ip, port번호 등등) 이 sub-trie나 dimension 하나가 매칭 rule하나를 의미한다. trie구조의 leaf node는 device, forwarding rule pair로 이루어져있다. 만약 새로운 forwarding rule이 내려오면, Veriflow는 trie구조를 탐색하면서 새로운 forwarding rule에 대해 equivalence class단위로 나누고, 다시 한 번 탐색하면서 forwarding graph를 생성한다. 각 그래프는 equivalence class에 속한 패킷들이 네트워크를 통해 어떻게 전송되는지를 나타낸다. 그래프에서 노드는 특정 네트워크 디바이스에서의 equivalence class를 나타내며 엣지는 forwarding decision(equivalence class,device pair)을 나타낸다. 이 그래프는 network invariant들을 검사하는데 사용되며, 만약 새로운 forwarding rule이 내려 왔을 때 equivalence class에 변화가 생긴다면 변화된 equivalence class들에 대해 검사할 invariant들을 forwarding graph를 이용하여 차례대로 검사한다. Veriflow에는 검사해야할 network invariant들 (예를 들어 reachability, looping, consistency)이 저장되어있는데 API들로 invariant들을 더 추가할 수도 있다. 만약 invariant 확인 과정에서 문제가 있으면, Veriflow는 보안적으로 forwarding rule을 drop시켜 버리거나 rule은 설치되도록 하되 관리자에게 alarm을 통해 이를 알려준다.

Ⅲ. 결론

Veriflow는 controller와 네트워크 장비 사이에서 모든 rule insertion,deletetion,modification등을 관찰하며 network invariants를 검사하는 동적 분석인 반면, header space analysis는 네트워크 장비 설정파일들을 통한 정적 분석이다. 동적 분석은 정적 분석보다 실시간으로 탐지가 가능하다는 장점이 있다. 또한 시시각각 변하는 네트워크 환경에서 유연하게 대처할 수 있다는 장점이 있다. 기능면에서의 장점만 비교하자면 동적 분석이 좋지만, 현재로써 가장 큰 문제는 성능이다. 동적분석은 실시간으로 탐지해야하기 때문에 네트워크에서의 오버헤드가 매우 클 수 밖에 없다. Veriflow논문에서의 실험에서도 자세히 보면 나타나있지만, 오버헤드로 인해 실시간지원이 힘들다. 실제로 자체 오픈플로우 테스트베드와 가상네트워크(mininet)를 활용하여 작은 네트워크와, 큰 네트워크를 구성해 성능을 실증한 결과 Veriflow는 네트워크 크기가 커지면 equivalence class가 커지기 때문에 성능이 좋지 않게 나타났다. 오버헤드로 인해 응답속도가 느려지는 것은 네트워크의 성능을 현저히 저하시킬 수 있으므로 이는 곧 큰 네트워크에서는 사용하기 힘들다는 의미이다. 또한 Veriflow는 처음 실행시킬 때 네트워크 토폴로지 정보가 담긴 텍스트파일을 인자값으로 주고 실행시켜야하기 때문에 controller를 껐다 켜기 전에는 네트워크에서 장비가 추가되었을 때의 정보를 탐지해낼 수 없는 단점이 있다. Header space analysis(Hassel)은 네트워크에서 문제가 생긴 이후에야 탐지해낼 수 있다는 단점이 존재하지만, 큰 네트워크에서의 성능 측면에서 Veriflow보다 효율적이며 네트워크의 성능저하 없이 network invariants를 탐지해낼 수 있다는 장점이 있다. 하지만 네트워크장비 설정파일들을 수집하여 파싱하는 작업은 수작업이기 때문에 네트워크 관리자가 번거롭게 느껴질 수 있는 문제가 있다. Veriflow와 Header space analysis(Hassel)은 각각의 장단점이 있다. 네트워크는 다양하기 때문에, 둘 중 어느 하나가 분명하게 좋다고 말하기는 어렵다. 따라서 네트워크 관리자의 판단에 따라 선택되어야 한다. 예를 들어 큰 네트워크 환경일 경우 성능 저하의 문제로 인해 Header space analysis(Hassel)을 선택해야만 하며, 작은 네트워크 환경일 때 에는 Veriflow를 선택하는 것이 좋을 것이다.

ACKNOWLEDGMENT

본 연구는 한국과학기술정보연구원의 지원으로 수행되었음.(과제번호 P14014)

참 고 문 헌

[1] 강세훈, 김영화, 양선희, “SDN 핵심 기술 및 진화 전망 분석” 한국통신학회지 (정보와통신) 제30권 제3호, 2013.2, 3-8

[2] AL-SHAER, E., AND AL-HAJ, S. “FlowChecker: Configuration analysis and verification of federated OpenFlow infrastructures.” In SafeConfig (2010).

[3] MAI, H., KHURSHID, A., AGARWAL, R., CAESAR,M., GODFREY, P. B., AND KING, S. T. “Debugging the data plane with Anteater.” In SIGCOMM(2011).

[4] KAZEMIAN, P., VARGHESE, G., AND MCKEOWN,N. “Header space analysis: Static checking for networks.“ In NSDI (2012).

[5] A. Khurshid, X. Zou,W. Zhou, M. Caesar, and P. B. Godfrey. “Veriflow: verifying network-wide invariants in real time.“ In NSDI(2013).

[6] Peyman Kazemian, Michael Chang, Hongyi Zeng, George Varghese, Nick McKeown, Scott Whyte “Real Time Network Policy Checking using Header Space Analysis.” In NSDI(2013).