링크 플러딩 공격 완화를 위한 소프트웨어 정의 네트워크 기반 허니넷

김진우[1] · 이승수[2] · 신승원[1*]

[1]KAIST 전기 및 전자공학부 · [2]KAIST 정보보호대학원

Software-Defined HoneyNet: Towards Mitigating Link Flooding Attacks

Jinwoo Kim[1] · Seungsoo Lee[2] · Seungwon Shin[1*]

[1]KAIST School of Electrical Engineering · [2]KAIST Graduate School of Information Security

E-mail : jinwoo.kim@kaist.ac.kr / lss365@kaist.ac.kr / claude@kaist.ac.kr

요 약

지난 몇 년간, 링크 플러딩 공격이라는 새로운 형태의 분산 서비스 공격 (DDoS) 이 제안되었다. 링크 플러딩 공격은 기존 DDoS 공격과는 다르게 선택적으로 라우터 간 코어 링크를 혼잡 시킴으로써 보다 넓은 범위에 지대한 영향을 끼친다는 점에서 큰 차이가 있다. 기존 네트워크 구조에서는 링크 플러딩 공격을 완화하는 것이 어려운데, 이는 공격자가 *traceroute*를 이용하여 취약한 링크를 사전에 파악하고 링크맵을 구축할 수 있는 원인에 기인한다. 기존에 링크 플러딩 공격을 감지하여 대응하기 위한 여러 연구가 제안되었으나 이들은 모두 목표 링크에 실제 공격이 발생한 직후에 이를 완화하는, 즉 사후 조치를 한다는 한계점이 존재한다. 본 논문에서는 링크 플러딩 공격 시나리오에서 공격자가 링크맵을 구축할 수 있다는 점에 주목하고 이를 사전에 방지하고자 하는 접근법을 제안한다. 소프트웨어 정의 네트워크의 장점을 활용하여 취약한 링크를 사전에 파악하고, 주변에 허니팟을 배치함으로써 중요한 링크를 공격자로부터 은닉하는 시스템인 SDHoneyNet을 보인다.

ABSTRACT

Over the past years, Link Flooding Attacks (LFAs) have been introduced as new network threats. LFAs are *indirect* DDoS attacks that selectively flood intermediate core links, while legacy DDoS attacks directly targets end points. Flooding bandwidth in the core links results in that a wide target area is affected by the attack. In the traditional network, mitigating LFAs is a challenge since an attacker can easily construct a link map that contains entire network topology via *traceroute*. Security researchers have proposed many solutions, however, they focused on *reactive* countermeasures that respond to LFAs when attacks occurred. We argue that this reactive approach is limited in that core links are already exposed to an attacker. In this paper, we present SDHoneyNet that prelocates vulnerable links by computing static and dynamic property on Software-defined Networks (SDN). SDHoneyNet deploys Honey Topology, which is obfuscated topology, on the nearby links. Using this approach, core links can be hidden from attacker's sight, which leads to effectively building proactive method for mitigating LFAs.

키워드

Distributed Denial of Service (DDoS), Link Flooding Attacks (LFAs), Software-defined Networks, Graph Theory

Ⅰ. 서 론

지난 20년간 인터넷이 발전함과 동시에 DDoS 공격 또한 다양한 공격 형태와 규모로 확대되어 왔다. 일반적으로 DDoS 공격은 단순한 방법을 통해 수행되지만, 본래 TCP/IP 네트워크의 특성 및 네트워크 카드 최대 전송 대역폭의 급격한 발전 등으로 인해 뚜렷한 해결책이 보이지 않는 상황이

* corresponding author

다. 하나의 예로써, 2018년 2월경 세계에서 가장 유명한 오픈소스 웹 호스팅 서비스인 GitHub에 최대 트래픽 대역폭이 1.35Tbps 에 이르는 대규모 DDoS 공격이 있었다[1]. 단순히 대규모 트래픽을 엔드 서버에 전송하는 공격이었지만, Akamai 같은 CDN 기반 DDoS 방어 시스템을 이용해서 트래픽을 우회시키는 방법 정도만을 사용할 수 있었다.

더욱이 최근, 링크 플러딩 공격 (Link Flooding Attacks, LFA) 이라는 새로운 형태의 DDoS 공격 방법이 제안되었다[2,3]. 이는 오늘날 인터넷의 라우팅 경로가 척도 없는 네트워크 (Scale-free network)를 따른 다는 점을 이용한 공격 방법이다. 현재 ISP가 사용자에게 서비스를 제공하는 인터넷 구조에서는, 종단 간 라우팅 경로는 특정 코어 라우터 및 링크를 반드시 거쳐갈 수 밖에 없는 형태로 이루어져 있다. 따라서 구조적으로 특정 링크에 라우팅 경로가 집중되는 이른바 병목 링크가 생겨날 수 밖에 없다.

LFA는 이러한 점을 이용한 간접적인 DDoS 공격 방법이다. 기존 DDoS 공격이 특정 서버나 라우터 등 종단 목표점에 트래픽을 직접적으로 전송하여 타격하는 방법이라면, LFA는 앞서 언급한 병목 링크에 대량의 봇넷을 활용하여 트래픽을 대량으로 전송하여 병목 링크의 대역폭을 소비하는 방법을 이용한다. 병목 링크를 혼잡시킴으로써 기존 DDoS 공격과는 달리 보다 넓은 범위에 효과적으로 공격을 할 수 있다는 큰 장점이 있다. 이 때, 개별 공격자가 사용하는 봇이 전송하는 트래픽은 각각 개별 플로우로 봤을 때는 일반적인 HTTP GET 트래픽 같이 적은 대역폭을 사용하기 때문에 구분하기 어렵다는 점도 특징이다.

LFA 시나리오에서 공격자는 병목 링크를 타격하기 위해 필수적으로 링크맵 (Linkmap)을 구축해야만 한다. 이는 실제 IP 네트워크에서 라우팅이 어떻게 이루어지는지에 관한 경로를 *traceroute*를 이용하여 경로 정보를 수집하여 기록한 것이다. 링크맵은 TopologyZoo* 나 Rocketfuel** 같은 정적 토폴로지 데이터가 아닌, 실제 레이어 3에서 동적으로 이루어지는 라우팅 경로에 관한 것이므로 실제 트래픽이 전송되는 신뢰성 있는 정보가 된다.

본 논문에서는 이러한 링크맵 구축 단계에 주목하여, 공격자가 *traceroute*를 이용하여 관찰하는 실제 네트워크 토폴로지를 숨기고, 거짓된 토폴로지를 제공하는 방법을 제안하려고 한다. LFA 시나리오 에서는 정확한 링크맵을 구축함으로써 병목 링크를 파악하는 것이 무엇보다 중요한데, 이를 사전에 파악하고 공격자로부터 은닉함으로써 공격 자체를 방지하고자 한다. 그러나 이와 동시에 LFA 공격을 탐지할 수도 있어야 하는데, 기존 허니팟 (Honeypot) 시스템에 영감을 얻어 공격자에게 병목 링크가 존재하는 것처럼 보이는 거짓된 토폴로지를 제공하는 *허니 토폴로지 (Honey Topology)*를 보이고자 한다. 만약 *traceroute*를 수행하는 호스트가 공격자라면, 허니 토폴로지를 대상으로 하여 DDoS 공격을 수행할 것이고, 이를 탐지함으로써 대상이 공격행위를 하는 것을 확신하고 대응을 할 수 있게 된다. 따라서 LFA의 공격 트래픽이 일반적인 트래픽과 대응하기 어렵다는 점도 해결될 수 있다.

위에서 언급한 아이디어를 구현하기 위해 본 논문에서는 소프트웨어 정의 네트워크 (Software-Defined Network, SDN) 기반으로 구축한 시스템인 SDHoneyNet을 제안한다. SDHoneyNet은 SDN의 장점 중 하나인 글로벌 네트워크 뷰 (Global Network View)를 활용하여 병목 링크를 사전에 파악하고 주변 지점에 허니 토폴로지를 배치한다. 이를 위해 병목 링크의 정적/동적 특성을 수집하고 분석한다. 허니 토폴로지는 공격자로부터 병목 링크를 가질 확률이 높고 충분히 복잡하게 보이게끔 척도 없는 네트워크 법칙을 따르는 토폴로지를 생성한다.

II. 관련 연구 및 문제 정의

LFA는 2009년 'The Coremelt Attack' 이라는 논문에서 처음으로 제안되었다. Coremelt 공격은 기존 DDoS 공격이 종단 지점에 직접 트래픽을 보내는 것과 달리, 코어 링크를 파악하고 해당 링크를 다수의 봇과 봇간 쌍을 이용하여 서로 정상적인 트래픽을 주고받으며 코어 링크를 혼잡시키는 방법을 사용하였다[2]. 이 후 2013년 'The Crossfire Attack' 이라는 논문에서 보인 Crossfire 공격에서는 봇넷이 전송하는 트래픽이 공개적으로 접근이 가능한 퍼블릭 서버 (Public Server)를 대상으로 하여, 단일 방향 트래픽 전송을 이용하여 병목 링크 공격이 가능하다는 것을 보임으로써, 특정 지역에 관계없이 좀 더 유연하게 공격을 할 수 있다는 점을 보였다[3].

이렇게 정교한 공격 방식의 LFA를 우회하기 위해 보안 학회에서는 여러 가지 다양한 해결책을 제시하였다. 기존에 제안된 연구 방법은 크게 두 가지로 분류될 수 있는데, 첫 번째로는 고전적인 DDoS 공격 방어법 같이 공격이 발생한 후에 대응하는 *사후 조치* 방법이다. 예를 들어, Spiffy [4]는 LFA 공격 상황에서 일반 사용자와 봇을 구분하기 위해 임시 대역폭 확장 (Temporary Bandwidth Expansion, TBE)을 이용하였다. 링크의 대역폭이 일시적으로 확장된다면 정상적인 호스트는 이를 인지하고 스루풋을 증가시키나, 봇은 사전에 병목 링크를 혼잡시키기 위해 개별 할당 대역폭만큼만 보내도록 되어있을 것이므로 스루풋을 증가시키지 않는다. 이러한 변화를 감지함으로써 봇과 일반 호스트를 구분할 수가 있다. Codef [5]는 AS 간 협업 재 라우팅 (Collaborative Rerouting) 및 협업 속도

* http://www.topology-zoo.org

** https://research.cs.washington.edu/networking/rocketfuel/interactive/

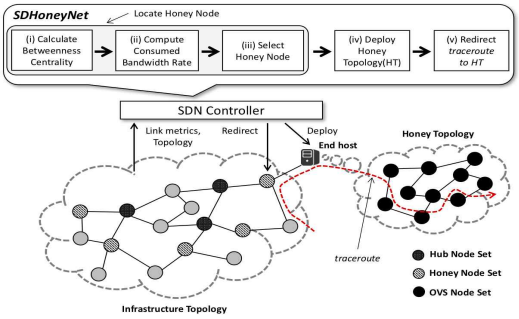

그림 1. SDHoneyNet 개요도

(Collaborative Rate Control) 테스트를 이용해 봇과 정상 호스트를 구분하는 방법을 사용하였다. 각 라우터가 LFA로 인해 혼잡될 때, 경로 식별자를 이용하여 출발지 AS를 파악하고 해당 라우터로 재라우팅 또는 속도 제어 요청을 보냄으로써, 공격을 완화하거나 이에 따르지 않는 봇을 찾아내는 방법이다. LinkScope [6]는 홉 바이 홉 측정을 통해 경로 별 성능 메트릭을 측정하고 이에 관련된 피쳐 벡터 (예: 패킷 손실률, RTT, 연결 실패율 등)를 추출해 내었다. 이를 바탕으로 정상 데이터를 학습하고, 공격을 당할 때의 경로 성능 측정 데이터에 이상 감지 (Anomaly Detection)을 수행하여 LFA를 감지하였다.

이와 같은 사후 조치 방법에 관련하여 효과적으로 LFA를 탐지 할 수 있는 여러 연구가 제안 되어 왔으나, 근본적인 한계점으로는 공격자가 이미 병목 링크에 공격 트래픽을 보내고 있다는 점을 들 수 있다. 따라서 네트워크 관리자는 공격 발생 후에 LFA에 대응해야 하므로 코어 링크에 어느 정도의 혼잡함을 감수해야만 한다. 또한, 공격자는 이미 링크맵을 구축하였기 때문에, 병목 링크에 이르는 다양한 경로를 알 수 있어서 우회 링크를 사용하여 재공격을 시도할 수 있다. 따라서, 네트워크 관리자는 공격자로부터 병목 링크를 보호하기 위한 다른 방법이 필요함을 알 수 있다.

III. 제안 방법

본 논문에서는 이러한 기존 연구의 한계점을 개선하고자 다른 방향으로 접근하여 LFA를 우회해보고자 한다. 앞서 설명한 사후 조치에 대비되는 *사전 조치*를 수행함으로써 공격자가 병목 링크를 찾는 것을 어렵게 만드는 것이다. 간단한 아이디어로써, 네트워크 관리자는 사전에 병목 링크가 될 만한 링크를 파악하고 주변 노드에 허니 토폴로지를 배치한다. 이 후, 공격자가 *traceroute*를 이용하여 링크맵을 구축할 때 *traceroute* 트래픽을 감지하고 이를 실제 병목 링크가 아닌 다른 목적지로 재라우팅 함으로써 실제와는 다른 링크맵을 구축하게끔 유도하는 것이다. 이를 통해 병목 링크 자체가 공격자에게 노출되지 않음으로써, 링크가 공격 트래픽으로부터 영향을 받지 않을 수 있다. 이 때 해결 해야되는 문제로써, i) 병목 링크를 어떻게 빠르고 정확하게 파악할 것인가? ii) 허니 토폴로지를 어떻게 빠르게 배치할 수 있을 것인가? iii) 어떻게 공격자에게 링크맵을 잘못 구축하게끔 할 것인가? 를 들 수 있다.

SDHoneyNet은 이러한 세 가지 문제를 해결하기 위해 첫 번째로, SDN 컨트롤러가 사전에 가지고 있는 토폴로지 정보를 바탕으로 병목 링크에 대한 *정적 메트릭*인 매개 중심성 (Betweenness Centrality)와 링크의 트래픽 상태에 따라 달라지는 *동적 메트릭*인 소비 대역폭 비율 (Consumed Bandwidth Rate)을 계산하여 후보 링크를 파악한다. 두 번째로는, 병목 링크 주변 호스트에 동적으로 소프트웨어 스위치를 이용한 가상 토폴로지를 배치하여 해결한다. 마지막으로, *traceroute* 트래픽이 배치된 노드 주변 스위치*로부터 감지되면, SDN 컨트롤러가 이를 허니 토폴로지에 재 라우팅 하도록 플로우 룰을 설치한다. 그림 1은 이러한 일련의 과정들을 나타낸 SDHoneyNet의 개요도이다.

SDHoneyNet은 허니 토폴로지를 배치할 스위치를 먼저 찾아야하는데 이러한 노드를 *허니 노드 (Honey Node)* V_{honey}라고 한다. 허니 노드를 찾기 위해서는 계산된 정적 메트릭과 동적 메트릭 모두 높은 값을 가지고 있는 노드를 먼저 찾아야하는데 이를 *허브 노드 (Hub Node)* V_{hub} 라고 정의한다. 허니 노드는 이러한 허브 노드의 인접 노드 (Neighbor)의 집합으로 정의한다.

SDN 환경에서는 기본적으로 SDN 컨트롤러가 오픈플로우 채널로 연결된 데이터 평면의 라우터에 대한 토폴로지 정보를 가지고 있다. 본 논문에서는 모든 데이터 평면 라우터가 SDN 컨트롤러와 연결되어있다고 가정한다. 토폴로지 정보를 이용하여 병목 링크의 정적 메트릭을 계산하는데 SDHoneyNet은 매개 중심성[8]를 이용한다. 매개 중심성은 소셜 네트워크 분야에서 노드의 상대적 중요성을 나타낼 때 사용 되는 지표로, 해당 노드가 다른 노드 간 최단 경로에 포함될 확률을 나타낸다. 따라서 데이터 평면에서 매개 중심성이 높은 노드는 병목 링크와 인접할 가능성이 높은 노드가 된다. 이 때, AS간 네트워크의 경로는 주로 관리자의 정책에 의해 결정되는 경우가 많으므로, 매개 중심성을 구할 때는 최단 경로가 아닌 정책 기반 라우팅 (Policy-based Routing)에 의해 결정되는 경로를 사용하여야 한다[6].

SDN 환경에서는 SDN 애플리케이션이 관리자 정책에 따라 경로를 결정하므로, SDHoneyNet은 SDN 컨트롤러로부터 우선순위가 가장 높은 애플리케이션으로부터 결정된 라우팅 경로를 이용하여 매개 중심성을 계산한다. 따라서 임의의 노드 v에 대한 매개 중심성의 수식은 아래와 같다. 수식에서

* 본 논문에서는 SDN 환경에서의 데이터 평면의 오픈플로우 스위치를 지칭하기 위해 스위치 또는 라우터 혼용하여 사용한다.

$path_{st}$는 SDN 애플리케이션으로부터 결정된 노드와 *s*와 *t*간 종단 라우팅 경로가 되며, $path_{st(v)}$ 는 노드 *s*와 *t*의 라우팅 경로 중 노드 *v*가 포함된 경로를 의미한다.

$$C_{B(v)} = \sum_{s \neq t \neq v \in V} \frac{path_{st(v)}}{path_{st}} \quad [6]$$

각 동적 메트릭인 소비 대역폭 비율을 구하기 위해서는 주기적으로 링크 별로 사용되고 있는 대역폭을 알아야하는데, 이를 위해 SDN 컨트롤러가 주기적으로 데이터 평면으로부터 통계정보를 수집한다는 점을 이용한다. SDN 컨트롤러는 오픈플로우의 *Port Statistics* 메시지를 주기적으로 데이터 평면에 전송하여 스위치별 포트의 사용량을 모니터링 할 수 있다. SDHoneyNet은 SDN 컨트롤러가 수집하는 포트 통계 정보로부터 각 링크 *e(s,t)*에 대한 소비 대역폭 비율인 *CBR(e(s,t))*을 아래와 같이 계산한다.

$$CBR(e(s,t)) = \frac{\frac{CurrTXBytes(s) - \mathrm{Pr}evTXBytes(s)}{meantime} \times 1.25 \times 10^9}{MaxBandwidth(e(s,t))}$$

*Port Statistics*에는 포트별로 전송한 누적 바이트 수를 기록한 *tx_bytes* 필드가 있는데, 이를 일정 시간 *meantime* 마다 수집한다. 노드 *s*에 대해 현재 시간의 *tx_byte*인 *CurrTXBytes(s)*에서 이전 *tx_byte*인 *PrevTxByte(s)*를 빼고 이를 bps 단위로 변환한다. 그리고 이를 링크별 최대 대역폭인 *MaxBandwidth(e(s,t))*로 나누어주면 각 링크별 실시간 소비 대역폭 비율을 알 수 있다.

이렇게 구한 모든 링크에 대한 소비 대역폭 비율 집합 *CBR(e(s,t))*과 매개 중심성 집합 $C_{B(v)}$을 목적지 노드 *t*에 대해 그룹핑하여 값이 높은 순에서 낮은 순으로 내림차순 정렬한다. 이 때 관리자로부터 배치할 허니 토폴로지 개수 *k* 에 따라 두 집합에서 상위 랭크 k개의 공통 노드를 허브 노드로 선택하며, 해당 노드가 공격을 당할 가능성이 가장 높은 허브 노드 집합 V_{hub}가 된다. 이러한 허브 노드 집합과 인접 노드이면서 정적/동적 메트릭이 낮은 노드를 구하는데 이들이 허니 노드 집합 V_{honey}가 된다.

위 알고리즘으로부터 구한 허니 노드 집합 V_{honey}에 허니 토폴로지를 배치하는데, 이때 병목 링크가 포함될 만큼 충분히 복잡한 토폴로지를 생성하기 위해 *Barabási-Albert (BA)* 모델을 사용한다. BA 모델은 확률 모델에 기반하여 무작위한 척도 없는 네트워크를 생성하는 알고리즘이다. 네트워크 토폴로지가 척도 없는 네트워크의 형태일 경우 허브 노드를 포함할 확률이 높아지므로 병목 링크가 생길 가능성이 높아진다. 따라서 이러한 모델에 기반하여 허니 토폴로지를 생성한다면, 해당 토폴로지에 있는 허브 노드가 병목 링크를 가질 것이라 믿을 확률이 높아진다고 할 수 있다.

이 후, 허니 노드에 TTL 값이 1인 패킷이 올 경우 이를 컨트롤러에 오픈플로우 Packet-In으로 알리는 플로우 룰을 설치한다. 이러한 패킷이 올 경우, SDHoneyNet은 공격자의 traceroute로 간주하고, 이후 플로우를 허니 토폴로지에 라우팅하는 임시 플로우 룰을 설치한다. 공격자는 traceroute의 결과만을 가지고 링크맵을 구축하므로, 허니 토폴로지에서 오는 응답을 보고 해당 지점에 공격 트래픽을 보낼 시 이를 보고 공격자를 구분 할 수 있다.

Ⅳ. 결론 및 향후 연구

본 논문에서는 SDN의 이점과 LFA가 링크맵에 기반한다는 점에 기반하여 공격자에게 거짓 토폴로지를 보일 수 있는 시스템인 SDHoneyNet을 제안하였다. 이를 위한 방법론으로 SDN 컨트롤러가 수집 및 추출할 수 있는 정적/동적 메트릭을 제안하고, 이에 기반하여 병목 링크를 찾아 주변 노드에 거짓 토폴로지인 허니 토폴로지를 배치하였다. 향후 연구에서는 확장성 있는 시스템의 구현 및 병목 링크를 찾기 위해 사용할 수 있는 다른 메트릭에 대해 더 고찰해보고자 한다.

Acknowledgement

이 논문은 2017년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.2018-0-00254, SDN 보안 기술개발)

References

[1] GitHub Engineering. February 28th DDoS Incident Report [Internet]. Available : https://githubengineering.com/ddos-incident-report/

[2] Studer, Ahren, and Adrian Perrig. "The coremelt attack." European Symposium on Research in Computer Security. Springer, Berlin, Heidelberg, 2009.

[3] Kang, Min Suk, Soo Bum Lee, and Virgil D. Gligor. "The crossfire attack." Security and Privacy (SP), 2013 IEEE Symposium on. IEEE, 2013.

[4] Kang, Min Suk, Virgil D. Gligor, and Vyas Sekar. "SPIFFY: Inducing Cost-Detectability Tradeoffs for Persistent Link-Flooding Attacks." NDSS. 2016.

[5] Lee, Soo Bum, Min Suk Kang, and Virgil D. Gligor. "CoDef: collaborative defense against large-scale link-flooding attacks." Proceedings of the ninth ACM conference on Emerging networking experiments and technologies. ACM, 2013.

[6] Xue, Lei, et al. "Towards Detecting Target Link Flooding Attack." LISA. 2014.

[7] Schuchard, Max, et al. "Losing control of the internet: using the data plane to attack the control plane." Proceedings of the 17th ACM conference on Computer and communications security. ACM, 2010.

[8] Freeman, Linton C. "A set of measures of centrality based on betweenness." *Sociometry* (1977): 35-41.