PI Mechanism을 이용한 효율적인 Hop Count Filtering 구현

정용훈° 홍만표 예홍진
아주대학교 정보통신전문대학원
{tomaj°, mphong , hjyeh}@ajou.ac.kr

An Efficient Implementation of Hop Count Filtering using Path Identification Mechanism

Yonghoon Jeong° , Manpyo Hong , Hongjin Yeh
Graduate School of Information & Communication , Ajou University

요 약

서비스 거부 공격(Denial of Service)은 최근 가장 큰 관심의 대상이 되고 있는 공격 형태로 시스템이나 네트워크의 자원을 고갈시킴으로써 더 이상의 서비스를 제공하지 못하도록 하는 공격이다. 이러한 서비스 거부 공격은 IP 스푸핑(spoofing) 기법을 사용하기 때문에 공격지의 근원지를 파악하기 어렵고, 필터링 기법을 사용하여 공격을 차단하기도 어렵다. 이러한 공격을 탐지하기 위해서 Hop Count Filtering 기법이 제안되어 있지만, 피해자에게 유입되는 각각의 패킷을 분석하여 스푸핑을 탐지하기에는 많은 지연과 과부하(Overhead)의 문제점이 있다. 이 것을 해결하기 위해서 본 논문에서는 PI (Path Identification Mechanism) 기법을 적용시켜, 스푸핑 된 패킷을 이용한 서비스 공격을 차단할 수 있는 구현방안을 제안하고자 한다.

1. 서 론

서비스 거부 공격(Denial of Service)은 최근 가장 큰 관심의 대상이 되고 있는 공격 형태로 시스템이나 네트워크의 자원을 고갈시킴으로써 더 이상의 서비스를 제공하지 못하도록 하는 공격 방법이다.[1] 기술이 발전됨에 따라 공격의 효과를 극대화시키기 위해서 분산된 공격자로부터의 공격이 이루어지고 있으며 (Distributed Denial of Service), 최근에는 웜(worm)의 형태로 인프라에 피해를 주는 공격이 이루어지고 있다.

또한 대부분의 서비스 거부 공격은 TCP/IP 프로토콜의 근본적인 취약점을 이용한 IP 스푸핑(spoofing) 기법을 사용하여 패킷의 출발지 주소를 위조하기 때문에 공격을 당하는 피해 시스템에서는 공격의 근원지를 파악할 수 있는 방법이 없다. 또한 스푸핑된 IP를 필터링(filtering) 한다는 것은 결국 선의의 사용자에게도 서비스를 제공할 수 없기 때문에 공격 패킷을 차단하여 피해를 줄이는 방법에도 문제가 있다. 물론 다양한 시도와 연구가 계속 진행되어 오고 있으나 시스템과 네트워크에 가중되는 막대한 오버헤드(overhead)와 기존 인터넷 체계의 변경 등이 해결해야할 과제로 남아있다.

Hop Count Filtering 기법은 피해자(Victim)에게 유입되는 패킷의 근원지 호스트에서 피해자까지의 홉 수 (hop count) 정보를 수집하고, 수집된 홉 수를 기반으로 유입되는 패킷의 홉 수와 비교하여 동일하지 않으면, 스푸핑 되었다고 판단하여 필터링을 하는 기법이다. 하지만, 스푸핑 여부를 판단하는데 있어서 지연(delay)이 발생하고, 공격이 발생하는 중간에 경로가 바뀌었을 때에 정상적인 패킷을 스푸핑된 패킷으로 오인하고 필터링하는 경우가 발생하게 된다.

본 논문에서는 Hop Count Filtering 기법을 보완하기 위해 PI 기법을 이용하였다. 그렇게 함으로써 지연을 줄이고, 오탐지율을 줄일 수 있도록 구현 방안을 제안하고자 한다.

먼저 2장에서는 기존 연구들에 대해서 살펴본 후 3장에서는 본 논문이 제안하고자 하는 구현 방안에 대해서 제시하고자 한다. 마지막으로 4장에서 결론을 통하여 본 논문을 마무리하고자 한다.

2. 관련 연구

2.1 PI(Path Identification) Mechanism

PI 기법은 IP가 스푸핑(spoofing)된 패킷을 이용한 공격에 대해서 효과적으로 필터링 할 수 있는 기법이다.[2][3] 공격자(attacker)에 대한 정확한 경로를 추적하지는 않지만, 공격 패킷이 이동하는 - 공격자에서부터 피해자(victim)에 이르는 - 경로에 있는 라우터들이 패킷의 IP 헤더의 Identification field에 자신의 정보를 표시(marking)하고 패킷의 목적지인 피해자는 패킷에 - 경로를 지나오면서 - 표시된 PI와 공격 패킷의 PI를 비교하여 공격 패킷을 판단, 필터링하게 된다.

라우터들이 패킷에 자신의 정보를 표시하는 과정은 [그림 1]과 같다. 공격자(A), 피해자(V) 그리고 중간의 라우터들(R) 로 구성되고, 각각의 라우터들이 n 비트(그림 1의 경우에는 n=1)를 표시한다고 가정했을 때, 먼저 라우터는 통과하는 패킷의 TTL 값을 가지고 PI를 표시을 할 위치($\left\lfloor \frac{16}{n} \right\rfloor$)를 찾는다. 표시할 위치에 라우터들은 자신의 IP를 이용하여 만들어낸 n 비트를 삽입하게 된다. 피해자는 이 PI을 비교하여 공격패킷을 차단하게 된다.

Pi 기법은 매우 단순하게 설계되어 있어서 중간 라우터들에게 큰 오버헤드(overhead)를 주지 않으며, 상위 라우터들의 도움 없이 피해자가 직접적이며 즉각적인 필터링이 가능하다는 장점이 있다.

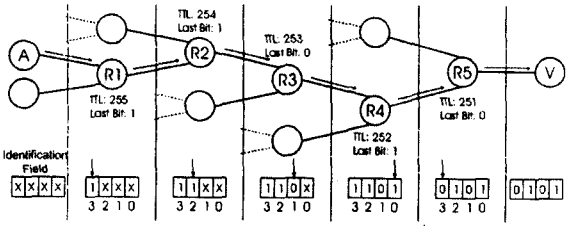

그림 1. PI 마킹 알고리즘

2.2 Hop-count Filtering

홉 수(Hop Count)를 이용한 필터링 기법은 IP 별로 피해자까지의 홉 수에 대한 정보를 테이블로 저장하고 있다가 피해자에게 유입되는 패킷들에 대하여 홉 수를 계산하여 기존의 테이블과 비교 후 차이가 생기면 스푸핑된 패킷이라고 판단하여 필터링하는 기법이다.[4] 홉 수를 계산하는 알고리즘은 아래와 같다.

```
모든 패킷에 대해서 :
  최종 TTL 값 Tf와 IP 주소 S를 추출 ;
  초기 TTL 값을 유추한 값 Ti ;
  패킷이 지나온 홉을 계산 Hc = Ti - Tf ;
  기존에 저장하고 있던 홉수 Hs ;
  if ( Hc ≠ Hs ) 패킷은 스푸핑 되었음 ;
  else 패킷은 정상 ;
```

이 알고리즘을 이용하여 피해자에게 들어오는 패킷에 대해서 스푸핑 여부를 판별하여 필터링하게 되는데, 이 과정은 경계 상태 (alert state) 와 실행 상태 (action state) 로 나눠지게 된다. 먼저 경계 상태는 공격의 징후를 판단하는 상태로 정의되여 스푸핑 되었다고 판단되는 패킷의 유입량이 소수일 때 공격 상태가 아니라고 판단하게 된다. 이 상태에서는 유입되는 패킷중에서 임의로 샘플을 추출하여 스푸핑 여부를 검사하게 된다. 만약 스푸핑이라고 판단되는 패킷의 유입량이 어느 임계치 이상을 넘어가면 실행 상태로 들어가게 된다. 실행 상태에서는 도든 패킷에 대해서 스푸핑 여부를 검사하여 스푸핑 된 패킷이이라고 판단된 모든 패킷을 폐기(drop)시킨다. 이 상태에서 스푸핑 되었다고 판단되는 패킷의 유입량이 떨어지면, 공격이 끝났다고 판단하여 다시 경계 상태로 변경된다.

홉 수를 이용한 필터링의 가장 큰 장점은 스푸핑 된 패킷의 탐지율이 높다는 것이다. 임의의 두 서버와 호스트 간의 홉 수는 자주 변하지 않고, 홉 수의 분포도 정규분포를 나타내고 있기 때문이다. 또한 다른 위치에 있는 임의의 사용자가 두 서버와 호스트 간의 홉 수를 예측하기 어렵기 때문에 회피하기 어렵다.[5]

하지만 이 기법도 몇 가지 문제점을 가지고 있다. 첫째로 패킷의 스푸핑 여부를 판단하는 과정에 있어서 지연(delay)이 생긴다는 것이다. 스푸핑 탐지 과정을 경계 상태와 실행 상태로 나눠지게 되는데, 그 이유는 탐지로 인한 지연 때문에 전체적인 성능의 저하를 불러일으킬 수 있기 때문이다. 모든 패킷의 홉 수를 계산하여 가지고 있는 테이블과 비교한다는 것은 패킷 처리에 많은 지연을 수반하기 때문이다.

둘째로 공격이 발생하는 중에 피해자까지의 경로가 바뀌게 된다면, 무조건 폐기되는 문제점이 있다. 경계상태에서는 홉 수가 다른 패킷이 유입되더라도, 즉각적인 필터링을 하지 않아 오탐지율을(false positive) 줄일 수 있으나 실행 상태에서는 즉각적으로 폐기된다. 공격 중에라도 홉 수가 변경된 선의의 호스트로부터의 패킷은 폐기 되지 말아야 된다.

3. PI기법을 이용한 효율적인 Hop Count Filtering 구현

본 논문에서는 앞서 보았던 홉 수를 이용한 필터링의 몇 가지 문제점을 보완하고 효과적인 필터링을 위해 PI 기법을 적용하여 Hop Count Filtering방안을 제안하고자 한다.

3.1 가정

먼저 본 논문에서는 몇 가지 가정을 하고자 한다.

· 공격자에서부터 피해자까지의 모든 라우터들은 PI 표시(marking)를 지원한다고 가정한다.

· 공격은 단일 공격자에 의해 이루어 질수도 있고, 다수의 공격자에 의해 이루어 질수도 있다.

· 공격자는 공격 패킷을 생성함에 있어서 패킷의 모든 부분을 변경(modify)할 수 있다.

3.2 탐지 및 필터링 과정

본 논문에서 제안하는 모델은 [그림 2]과 같다. 이 과정은 크게 두 부분으로 나누어진다. 첫 번째 부분은 시그니쳐로 만들어진 PI를 이용하여 필터링하는 부분이고, 다른 한 부분은 유입되는 패킷의 스푸핑 여부를 판별하여 의심이 되는 근원지에 대한 PI 시그니쳐를 만드는 부분이다.

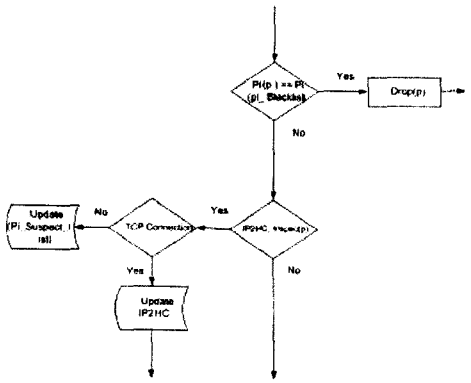

그림 2. 스푸핑 탐지 및 필터링 과정

[그림 2]의 구성 요소들은 아래와 같다.

· *P* : 피해자에게 들어오는 패킷

· *PI_Blacklist* : 공격자의 PI들을 모아놓은 리스트(list)로서 필터링의 시그니쳐로 사용된다.

· *IP2HC* : IP 별로 피해자까지의 홉 수를 저장해 놓은 테이블로서 <IP, Hop Count>의 집합이다.

· *IP2HC_Inspect(p)* : 패킷의 초기 TTL값을 유추하여 공격자에서 피해자까지의 홉을 계산, 기존의 정상적인 홉과 비교하여

스푸핑 되었는지 판단하게 된다.

· *PI_Suspect_list* : 스푸핑 되었다고 판단되는 패킷의 PI들을 모아놓은 리스트로서 <PI,Count>의 집합이다. " Count > 임계치 " 이 만족되면, PI는 PI_blacklist에 추가된다.

3.2.1 공격 PI 시그니쳐 생성 부분

먼저 공격 PI 시그니쳐 생성 부분은 [그림 3]과 같이 필터링 부분을 거친 패킷들에 대해서 스푸핑 여부를 판별하는 부분이다. 먼저 필터링 부분을 거친 패킷은 비록 공격이 아니라고 판단 되었지만, 잠재적인 공격 가능성은 아직도 가지고 있게 된다. 그렇기에 이 패킷들을 임의로 선택하여 스푸핑 가능성 여부를 판단하게 된다.

먼저 임의로 선택된 패킷을 IP2HC_Inspect를 통해 스푸핑 가능성 여부를 확인한다. 만약 스푸핑 되었을 가능성이 있다고 판단되면, 이후 TCP 연결이 이루어지는가를 확인한다. 만약 이루어진다면, 즉 연결이 된다면, 서비스 거부 공격 패킷이 아니라고 판단될 수 있으며, 홉 수가 상이한 것도 스푸핑된 것이 아닌 자연 상태에서 경로의 변화로 인한 홉의 변화로 판단 될 수 있으므로, 이후의 같은 IP를 가진 패킷에 대해서 변화된 홉으로 판단하기 위해 IP2HC를 업데이트 시킨다.

하지만 TCP 연결이 설정되지 않는다면, 스푸핑 되었다고 판단, PI_Suspectlist에 해당 PI를 추가 시키거나, 이미 존재한다면, 카운트(count)를 증가 시킨다. 만약 PI의 카운트가 임계값을 넘으면, 해당 PI를 PI_Blacklist에 update 시키게 된다.

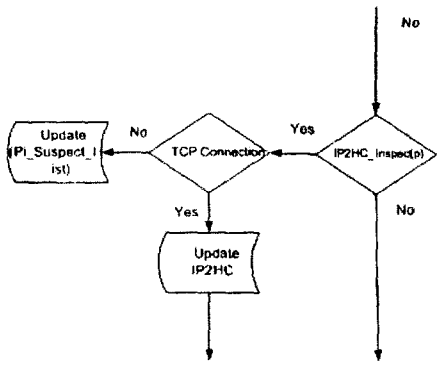

그림 3. Pi 시그니쳐 생성 과정

이 부분을 통하여 공격을 받고 있는 상태에서도 경로가 변경된 정상 패킷에 대한 폐기를 방지할 수 있게 된다. 즉, 피해자가 공격 상태에 있다고 하더라도 정상적인 호스트의 홉 수의 변화에 대해서도 수용 가능하게 된다.

또한 막고자 하는 공격 정도에 따라서 임계값을 조정함으로써 방어하고자 하는 공격에 맞춰 효율적인 방어가 가능하게 된다. 예를 들어 DDoS 공격의 방어를 위한다면 임계값을 높이고, 해킹에 대한 방어라면, 임계값을 줄여서 섬세한 방어가 가능하게 된다.

3.2.2 필터링 과정

[그림 4]와 같이 피해자에게 유입되는 패킷에 대해 필터링이 이루어지는 부분이다. PI 시그니쳐 생성부분을 통해서 만들어진 PI_Blacklist의 값들과 피해자에게 유입되는 패킷의 PI를 비교해서 필터링이 이루어지게 된다.

PI를 시그니쳐로 필터링함으로써 모든 패킷에 대한 스푸핑 여부를 판별하기 때문에 생기는 지연을 없앨 수 있으며, 동일한 공격자가 여러 스푸핑된 IP로 공격하는 것에 대해서도 필터링이 가능하게 되었다.

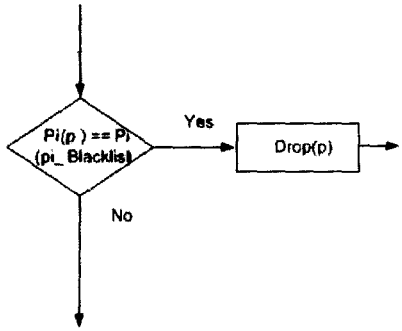

그림 4. 필터링 과정

4. 결 론

본 논문은 Hop Count Filtering 기법을 보완하기 위해 PI 기법을 이용하였다. 그렇게 함으로 지연을 줄이고, 오탐지율을 줄일 수 있는 구현 방안을 제안하였다. 홉 수를 이용한 필터링을 통해 스푸핑을 강력하게 탐지 할 수 있으며, PI를 이용하여 지연을 줄이고, 오탐지율을 줄일 수 있었다.

실제 환경에 있어서 효율성에 대한 검증과 스푸핑 된 패킷을 이용한 공격에 초점을 맞추었기 때문에 그 이외의 공격에 대한 효율성에 대한 연구를 향후 과제로 남겨 놓는다.

5. Reference

[1] A. Yaar, A. Perrig, D. Song. Pi: A Path Identification Mechanism to Defend against DDoS Attacks. *In Proceedings. 2003 Symposium on Security and Privacy, May 11-14, 2003*

[2] A. Yaar, A. Perrig, D. Song. Pi:A Path Identification Mechanism to Defend against DDoS Attacks. *In Proceedings. 2003 Symposium on Security and Privacy,* May 11-14, 2003

[3]A. Yaar, A. Perrig, D. Song. StackPi: A New Defense Mechanism against IP Spoofing and DDoS Attacks. Feb 2003

[4] C.Jin, H.Wang, K.G.Shin. Hop-Count Filtering : An Effecitve Defense Against Spoofed DDoS Traffic. *In proceeding of the 10th ACM Conference on Computer and Communications Security*, washington, DC,Oct 2003

[5] V. Paxson. End-to-End Routing Behavior in the Internet . *IEEE/ACM Transaction on Networking (5),* pp.601-615

[6] S.J. Tmpletton, K.E. Levitt, Detecting Spoofed Packets. *In proseedings of The Third DARPA Information Survivability Conference and Exposition (DISCEX III)'2003* Washington, D.C., April 2003