한국산학기술학회논문지
Vol. 10, No. 10, pp. 2733-2740, 2009

싱크홀 라우터 기반 IP 추적 시스템 설계 및 구현

이형우[1*]
[1]한신대학교 컴퓨터공학부

Design and Implementation of Sinkhole Router based IP Tracing System

Hyung-Woo Lee[1*]
[1]**School of Computer Engineering, Hanshin University, Korea**

요 약 최근 All-IP 네트워크 환경이 구축되면서 다양한 형태의 트래픽이 송수신되고 있으며, 이와 더불어 악의적 공격이 급증하고 있어 이에 대한 능동적 대응 방안이 제시되어야 한다. 기존 연구로는 SPIE 시스템을 통해 일방향 해쉬함수와 Bloom Filter 방식을 적용한 라우터 중심 패킷 경로 추적 기법이 제시되었으나, DDoS 공격이 발생할 경우 이를 능동적으로 차단하면서 공격 근원지를 효율적으로 추적하기에는 문제점이 있다. 따라서 본 연구에서는 기존 SPIE 및 Sinkhole 기반 라우터 기법의 장단점에 대한 분석을 통해 두 방식의 장점을 결합하여 All-IP 네트워크 환경에 적합한 IP 추적 방식을 설계하고 이를 구현하였다. 본 연구에서 제시한 기법은 기존의 Sinkhole 방식과 유사하게 공격 패킷에 대한 수집/모니터링 기능을 제공하면서도 추적 패킷 Manager 시스템을 기반으로 공격 패킷에 대한 판단 및 수집/제어기능을 제공하여 성능 향상과 함께 DDoS 공격에 대한 능동적 대응이 가능하였다.

Abstract An advanced and proactive response mechanism against diverse attacks on All-IP network should be proposed for enhance its security and reliability on open network. There are two main research works related to this study. First one is the SPIE system with hash function on Bloom filter and second one is the Sinkhole routing mechanism using BGP protocol for verifying its transmission path. In this study, we proposed an advanced IP Tracing mechanism based on Bloom filter and Sinkhole routing mechanism. Proposed mechanism has a Manager module for controlling the regional router with using packet monitoring and filtering mechanism to trace and find the attack packet's real transmission path. Additionally, proposed mechanism provides advanced packet aggregation and monitoring/control module based on existing Sinkhole routing method. Therefore, we can provide an optimized one in All-IP network by combining the strength on existing two mechanisms. And the Tracing performance also can be enhanced compared with previously suggested mechanism.

Key Words : Sinkhole Routing, IP Tracing, DDoS Attack, Security

1. 서론

현재 인터넷의 개별적인 유/무선 환경에서의 공격탐지 기술과 추적 기술이 존재하고 있다. 하지만 차세대 네트워크 환경의 All-IP에서의 문제는 그리 간단하지 않다. 각각의 망에서의 단순한 공격탐지의 문제가 아닌 통합망에서 유연성을 가진 통합 공격탐지 기술과 추적 기술에 대한 연구가 미흡한 실정이다. 따라서 기존의 탐지 기법들처럼 각 망에 대한 공격만을 탐지하는 것이 아니라 All-IP 환경에서의 능동적이 탐지가 필요한 실정이다.

기존의 IP 추적 방법으로 Bloom Filter[1] 기반 SPIE(Source Path Isolation Engine)[2] 시스템이 제시되었다. Bloom Filter는 Burton H. Bloom에 의해서 고안된 확률적 데이터 구조(Probabilistic Data Structure)로서 공간

본 논문은 한신대학교 2009년도 교내 일반연구비 지원으로 수행되었음.
*교신저자 : 이형우(hwlee@hs.ac.kr)
접수일 09년 09월 18일 수정일 09년 10월 13일 게재확정일 09년 10월 14일

의 효율성을 높이는 장점을 가지고 있다. Bloom Filter는 일방향성 해시 함수를 사용하여 데이터를 저장하기 때문에 사용되었던 키를 복원하는 것이 불가능하다. 즉, 해당 데이터가 데이터 셋에 속하는지에 대한 유무만을 판단할 뿐이고 해당 데이터를 원형대로 저장하지는 않다. 따라서 Bloom Filter는 검색하고자 하는 데이터가 해당 데이터 셋에 속하는 지에 대한 유무를 판단하는데 매우 효율적이다. 이러한 방식의 데이터 구조는 매우 작은 데이터 공간 (=메모리)을 사용하여 매우 많은 정보를 저장할 수 있고 운영하는 방식에 따라 검색에 소모되는 프로세스 부하, 소요 시간을 줄일 수 있어 효율적인 검색을 할 수 있다[3].

SPIE 시스템에서는 Bloom Filter를 사용함으로써 라우터를 지나가는 패킷의 정보를 저장하게 된다. Bloom Filter는 시스템의 저장 공간을 최소화 할 수 있다는 큰 장점이 있지만, False positive 라는 단점이 있다. SPIE 시스템에서 발생하는 False Positive는 DGA의 Bloom Filter에서 공격 패킷이 라우터를 지나가지 않았는데도 지나갔다고 판단하는 경우이다. 이 문제는 SPIE에서 사용하는 Bloom Filter에 중복되는 값을 표시하기 방법이 없기 때문에 발생한다[2,3].

SPIE 시스템에서 추적 기능을 제공하기 위해서 STM, SCAR, DGA의 3 가지 Component가 설치되어 있어야 한다. 이 중에서도 모든 라우터에 DGA가 설치가 되어 있어야만, 라우터에 공격 패킷이 지나간 것에 대한 여부를 알 수가 있다. DGA가 설치되어 있지 않은 라우터에 Request 메시지를 보낸다고 하더라도 이 Request 메시지를 처리할 수 없을 뿐만 아니라, Request 메시지를 받더라도 DGA의 Bloom Filter에 패킷의 Digest 정보를 가지고 있지 않기 때문에 Tracing을 할 수 가없다. 따라서 SPIE 시스템의 큰 취약점은 Tracing을 하기 위한 라우터에는 DGA가 모두 설치되어 있어야 한다.

따라서 본 연구에서는 기존의 SPIE 시스템에서 사용하는 Bloom Filter 기반 추적 기법 기법의 문제점을 개선하기 위해 Sinkhole 라우터 기반 트래픽 분석 기법을 적용한 IP 추적 기법을 제시하였다. 본 연구에서 제시한 기법은 기존 SPIE 시스템의 문제점을 해결하면서 Sinkhole 라우터와 연계한 추적 방법으로 최근 급증하고 있는 DDoS 공격에 능동적으로 대응할 수 있는 방법을 제시할 수 있었다.

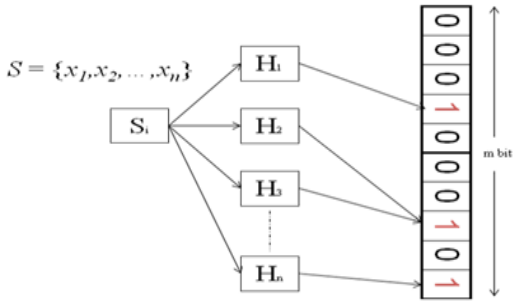
2. Bloom Filter 기반 SPIE 시스템

2.1 Bloom Filter 구조 및 동작방식

Bloom Filter는 일방향성 해시 함수와 비트열의 벡터로서 구성 된다. n개의 원소를 가지는 집합 S = {x1,x2,⋯,xn}가 있다. 집합 S의 각각의 원소들은 저장하고자 하는 데이터를 의미하며 m 비트열과 k 개 (h1,h2,⋯,hn)의 서로 독립적인 해시 함수 (Hash function)들로서 표현된다. 각 해시 함수들의 입력 값에 대한 출력 값의 범위는 {0, 1,⋯ , m-1}까지 되고 모든 출력 값이 동일한 확률로 나오게 되는 uniformly distributed 난수의 형태를 가진다. 따라서 집합 S의 각 원소가 평균적으로 사용하는 비트의 수는 m/n이 된다[1,3].

Bloom Filter는 초기화, 입력, 그리고 검색의 세 가지 과정으로 동작한다. 초기화 과정에서는 m 비트의 열을 모두 0으로 초기화를 한다. S 집합에 속하는 하나의 원소인 si에 대해서 k개의 해시 함수에 각각 si 를 입력 값으로 넣는다. 그리고 m 비트열 중 각각의 해시 함수의 출력 값에 해당하는 위치의 비트를 1로 설정한다. 만약 해시 값이 같을 경우에도 1로 설정할 수 있으며 다른 제약은 없다.

그림 1은 데이터에 대한 Bloom Filter의 동작과정을 보여주고 있다. 일반적으로 해시 함수를 이용한 해시 테이블(Hash table)은 k가 1인 특별한 경우의 Bloom Filter라 할 수 있다.



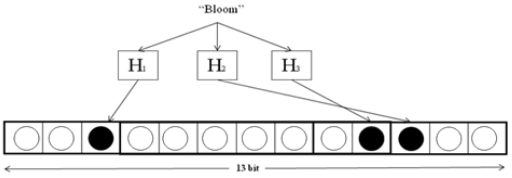
[그림 1] Bloom Filter 구조

앞서 본 입력 과정 이후에 각 데이터가 집합 S에 속하는지에 대한 질의 과정은 쉽게 수행된다. 주어진 하나의 원소 x 가 집합 S 에 포함되는지 확인하기 위해서 k개의 해시 함수에 x를 입력한다. 각 해시 함수의 출력 값에 해당하는 비트가 모두 1이면 해당 원소는 집합 S에 포함되고 하나라도 1이 아니면 포함되지 않는 원소임을 판단 할 수 있다.

3개의 해시 함수와 14비트의 비트열을 가지는 Bloom Filter가 있을 경우 비트열에서 속이 빈 원은 0을 의미하고 속이 찬 원은 1을 의미한다. 다음 그림 2와 같이 모든 비트를 초기화 한다.

[그림 2] Bloom Filter 비트 초기화

만일 'Bloom' 이라는 문자열을 3개의 해시 함수에 각각 입력할 경우 입력 된 문자열에 대해서 각각의 해시 함수의 출력 값이 각각 다음 그림 3과 같고 이에 해당하는 비트를 1로 설정한다.



[그림 3] Bloom Filter 입력 과정

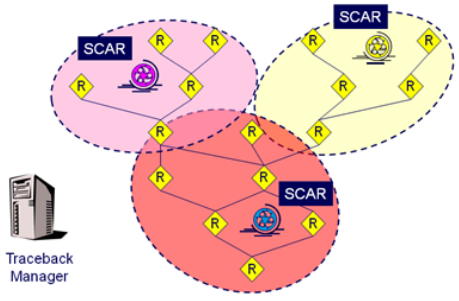
2.2 Bloom Filter의 특징 및 적용분야

Bloom Filter는 다양한 컴퓨터 응용 분야에서 사용되고 있다. Bloom Filter는 분산 캐시 시스템, DB 및 Peer-to-Peer 응용 프로그램 등에 활용될 수 있으며, 최근 IP Tracing 구조에 활용되고 있다[3]. 네트워크상에 한 패킷에 대한 라우팅 기반 추적 과정을 수행하고자 할 경우 모든 라우터들이 자신이 처리한 모든 패킷들에 대한 정보를 기록하는 방법을 사용할 수 있다. 그리고 추적하고자 하는 패킷에 대한 정보를 각각의 라우터에게 질의하고 라우터 내에 일치하는 정보가 있다면 질의가 시작된 처음 시스템으로 응답을 함으로서 패킷이 거친 경로를 추적할 수 있다. 하지만 이 방법은 라우터의 리소스 낭비와 프로세스 오버헤드가 발생 하는 문제가 있다.

이러한 문제를 해결하기 위해서 Bloom Filter를 사용할 경우 각각의 라우터에서 기록해야 할 패킷 정보를 Bloom Filter로서 저장하고 추적하고자 하는 패킷에 대한 질의가 왔을 때 단순히 Bloom Filter를 검색하여 존재 여부를 판단하기 때문에 IP 추적 모듈에 적용시 라우터의 리소스 사용량과 프로세스 오버헤드를 줄일 수 있는 이점이 있다.

2.3 기존 시스템[2,4]의 문제점

SPIE 시스템은 IP 추적 기능을 제공하기 위해 Bloom Filter 기반 해쉬 기술을 이용한다. SPIE 시스템에서는 라우터에서 최근에 포워딩한 트래픽에 대한 패킷 Digest 정보를 DGA(Data Generation Agents) 시스템 내부에 Bloom Filter 형태로 저장 관리한다. 만약에 IDS에 의해서 어떤 패킷이 위험성이 있고 공격적인 패킷이라는 것이 판단이 되면, Alerts를 발생시킨다. Alerts이 발생하면 아래 그림 4와 같이 SPIE 시스템의 STM(SIPE Tracing Manager)은 어떤 경로를 통하여 패킷이 지나갔는지 확인하기 위해서, 공격 패킷에 대한 Request 메시지를 만들어서 SCAR(SPIE Collection and Reduction Agents)에게 보낸다. 이 메시지를 시작하는데, 이 때 Bloom Filter의 Digest 값과 비교하게 된다.



[그림 4] SPIE 시스템 구조

기존의 SPIE 시스템에서는 라우터에 설치된 DGA에서 라우터를 지나가는 패킷의 정보를 Bloom Filter에 저장한다. 이 Bloom Filter는 메모리에 저장이 되는데, Paging 기법에 의해서 10초에 한 번씩 저장 된다. 따라서 최대 100개 까지 저장이 되는데 이것을 시간으로 환산하면 약 16분이 지나면 패킷의 정보가 저장된 Bloom Filter 정보가 사라지게 된다.

결국 16분 안에는 메모리에 라우터를 지나간 패킷의 Digest값이 Bloom Filter에 저장이 되어 있지만, SPIE 시스템에서는 이 정보를 가지고 다시 Tracing 하는 기능이 포함이 되어 있지 않다. 즉, 일정 시간이 지난 후에 다시 추적 과정을 수행할 수 있는 기능은 포함되어 있지 않기 때문에 16분이 지나면 패킷에 대한 정보를 저장하고 있는 Bloom Filter가 메모리에서 삭제가 되어 Tracing을 할 수가 없다.

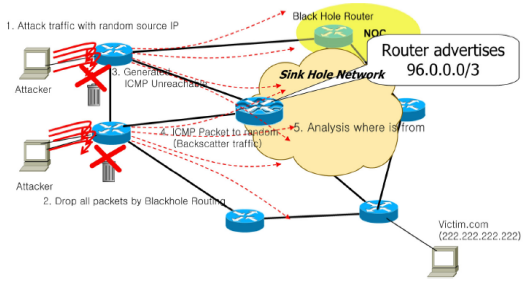
기존에 다양한 형태의 추적 기법[4,5]이 제시되고 있으며 DDoS 대응을 위한 추적[7,8,10] 등이 제시되었다. 하지만 아직까지 Sinkhole 라우터 방식을 적용한 추적 방식은 제시되고 있지 않다. 따라서 본 연구에서는 기존의 SPIE 시스템에서 제시한 Bloom Filter 기반 IP 추적 방법의 단점을 보완하면서 DDoS 공격 대응에 능동적인 기능을 제공하는 Sinkhole 라우터 기반 방식을 결합하여 보다 개선된 IP 추적 기법을 제시하고자 한다.

3. 기존의 Sinkhole 기반 공격 대응

3.1 Sinkhole 기반 공격경로 추적

대부분의 서비스거부공격에서는 근원지 추적을 피하기 위해 Source IP주소를 Spoofing하는 것이 일반적이어서 이를 추적하는 것은 쉽지 않다. 기존의 추적 기술 중 Cisco사의 Netflow를 이용하여 추적이 가능하지만, 공격 경로상의 라우터들이 Cisco사의 제품이어야 하고, 라우터에 접근할 수 있는 권한을 가져야 하고, 공격이 진행중인 동안에만 분석할 수 있다는 단점을 가지고 있다. 따라서, 공격 관련된 패킷들을 특정 네트워크로 끌어들여서 분석하는 기술이 필요하다.

다음 그림 5는 목적지 기반 원격구동 블랙홀 라우터에 의해 drop된 패킷들에 대한 ICMP Unreachable 패킷을 수집하여 공격 진입지점을 추적하는 시나리오를 보인다.



[그림 5] Sinkhole 라우터 기반 공격경로 추적

3.2 Sinkhole 설정 방식 및 문제점

선택된 주소공간을 관할하는 Sinkhole 라우터에 다음과 같이 ACL을 설정한다.

```
access-list 150 permit icmp any any unreachables log
access-list 150 permit ip any any
```

이 설정은 ICMP unreachable 메시지를 기록하도록 하는 명령으로, ACL이 설정된 라우터의 터미널 모니터링을 통해 unreachable 패킷이 발송되는 근원지를 찾을 수 있다. unreachable 패킷이 발송된 인터페이스가 공격 트래픽이 유입되고 있는 지점이며, 이 인터페이스에 연결되어져 있는 네트워크에서 공격 트래픽이 생성되는 근원지라는 것을 확인할 수 있다.

아래 내용은 공격으로 인해 Sinkhole 라우터에 ICMP unreachables 메시지가 기록된 예이다.

```
SLOT 6:6w6d: %SEC-6-IPACCESSLOGDP: list 150 permitted icmp
65.208.80.242 -> 99.141.160.46 (3/1), 1 packet
SLOT 6:6w6d: %SEC-6-IPACCESSLOGDP: list 150 permitted icmp
65.208.80.242 -> 97.147.124.71 (3/1), 1 packet
```

위의 로그에 나타난 ICMP 패킷의 목적지 주소(99.141.160.46와 97.147.124.71)는 공격시 처를 Spoofing하기 위해 임의로 생성된 공격자의 Source 주소로써, 패킷이 수집된 것을 확인할 수 있다.

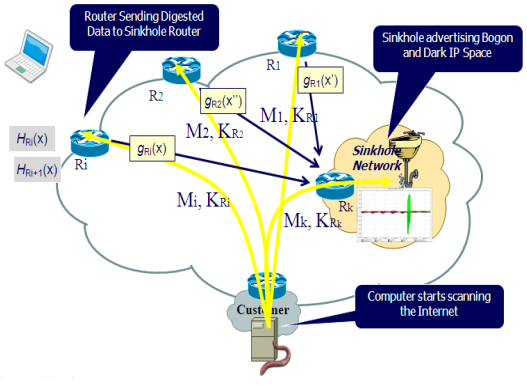
위 로그에서 중요한 것은 ICMP 패킷의 출발지 주소이다. 이 로그를 통해 공격 패킷이 유입되고 있는 네트워크 인터페이스가 65.208.80.242임을 쉽게 알 수 있다. 이 주소는 Null0 라우팅 되도록 미리 설정한 edge 라우터들 중의 하나로써, 해당 edge 라우터에 연결된 sub 네트워크로부터 공격이 들어오고 있음을 알 수 있다. 이처럼 Sinkhole 기반 공격 추적 기법은 아주 짧은 시간에 효과적으로 공격 근원지가 속한 네트워크까지 찾아낼 수 있다.

하지만 기존의 Sinkhole 라우팅 기반 대응 방식은 추적 근원지에 대한 정보까지는 제공하지 못하며 네트워크 전체에 대한 관리 및 공격이 발생하였을 경우 공격 패킷에 대한 근거자료 등을 제공하지 못한다는 문제점이 있다. 따라서 SPIE 시스템에서 제공하는 Bloom Filter를 적용하여 보다 효율적이면서도 성능이 개선된 추적 기법을 제공할 필요가 있다.

4. 제안 시스템 설계 및 구현

4.1 Digested Sinkhole 기반 IP 추적 구조

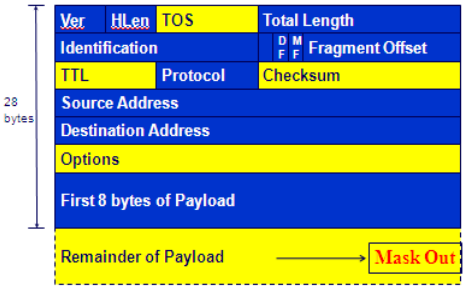
본 연구에서 제시하는 방식은 기존 SPIE 시스템과 Sinkhole 라우터 방식의 장단점을 비교 분석하여 최적의 해결 방안을 제시하였다. 그림 6과 같이 라우터를 중심으로 패킷에 대한 공격 발생시 Sinkhole 라우터로 해당 패킷에 대한 정보를 전송하고 공격 추적 정보를 생성/관리하는 구조를 설계하였다.



[그림 6] 제안하는 공격 패킷 추적 구조

위 그림에서 주요 작동 방식은 Victim 사이트로 공격 발생시 해당 네트워크에 소속된 라우터로 통보하여 Sinkhole 라우터에 대한 정보를 전송하고, 이후에 개별 라우터에 수신되는 패킷에 대해서는 Sinkhole 라우터로 전송하는 방식이다. 따라서 이를 위해서는 각 라우터에 기존의 SPIE 시스템과 유사한 형태로 패킷에 대한 요약(Digest) 정보를 저장하는 구조가 추가로 구현되어야 한다. 아래 그림과 같이 각각의 라우터에서는 개별적으로 캐쉬 구조를 갖고 있으면서 라우터를 거치는 패킷에 대해 Bloom Filter 방식을 적용해서 패킷에 대한 요약 정보를 구축하게 된다.

각각의 라우터에서 구축하게 되는 Bloom Filter 모듈에서는 아래 그림 7과 같이 IP 패킷의 헤더를 포함하여 처음 24바이트 정보를 취하여 이에 대해 해쉬 함수를 적용하게 된다.



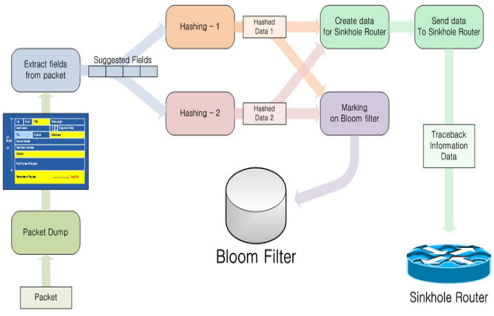
[그림 7] 패킷내 Bloom Filter 적용 24바이트

4.2 개선된 Bloom Filter 적용

아래 그림 8과 같이 각각의 라우터에서는 수신된 패킷의 처음 24바이트 정보에 대해 두가지형태의 해쉬 함수를 이용하여 해쉬 값을 생성하고 이를 Bloom Filter에 저장하고 라우터를 통해 다음 홉으로 패킷을 전송하게 된다.

만일 공격 형태에 해당하는 패킷일 경우 공격 리스트 파일에서 검색하고 만일 공격에 해당한다면 이를 Sinkhole 라우터로 전송하게 된다.

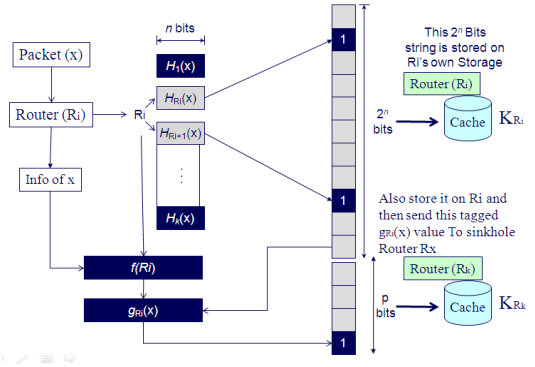
각각의 라우터에서 적용하는 해쉬 함수의 작동 과정을 좀더 자세히 살펴보면 다음과 같다. 암호학적으로 안전한 해쉬 함수는 SHA-256 함수를 사용할 수 있다. 이 경우 임의의 비트에 해당하는 패킷이 입력으로 들어오면, 항상 256비트의 출력값을 갖는 경우이다.



[그림 8] 제안하는 패킷 처리 구조

Bloom Filter인 경우 전체적인 크기는 해쉬 함수의 출력 비트 크기에 영향을 받기 때문에, 일반적으로 저장용량 및 Bloom Filter의 크기를 생각한다면 일반적으로 32비트 정도가 적절하다. 따라서 본 연구에서는 기존에 암호학적으로 안전하다고 알려진 SHA-256 함수를 이용하면서 Folding 방식을 이용하여 최종적으로 32비트 값의 해쉬 값을 계산하도록 하였다.

그림 9와 같이 각각의 라우터에서는 개별적으로 캐쉬 구조를 갖고 있으면서 라우터를 거치는 패킷에 대해 Bloom Filter 방식을 적용해서 패킷에 대한 요약 정보를 구축하게 된다.



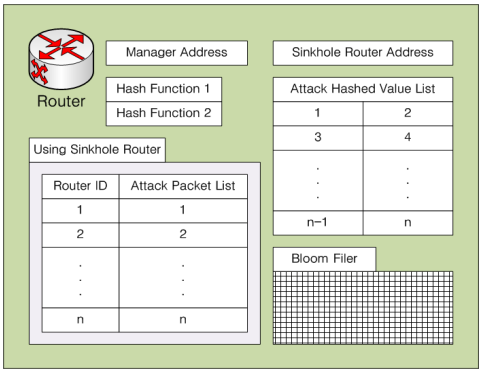
[그림 9] 패킷에 대한 Bloom Filter 적용

개별 라우터에서는 다음 그림 10과 같은 데이터 구조를 갖는다. 라우터 내부에는 각각의 라우터에서 설정한 두 개의 해쉬 함수에 대한 정보를 저장하고 있으며, Sinkhole 라우터에 대한 정보 등을 관리하면서, 공격 패킷에 해당하는 정보 등을 해쉬 함수에 대한 정보와 함께 리스트를 관리하고 있다. 물론 각 라우터에는 Bloom Filter 정보도 저장/관리하고 있게 된다.

각 라우터가 속한 네트워크 내 설정된 Sinkhole 라우터에 대한 정보를 가지고 있으면서 공격 패킷에 대해 전

송하는 기능을 수행하며 Bloom Filter 값을 저장하기 위한 구조도 설정되어 있다.

또한, 위에서 제시한 Bloom Filter 생성 구조를 이용하여 각각의 라우터에서는 개별 패킷에 대해 해쉬 함수를 적용하여 자신의 필터 구조 내에 기록하게 되며, 부가적인 정보에 대해서는 Sinkhole 라우터로 전송하여 공격 경로 등을 판별하게 하는 구조를 제안하였다.

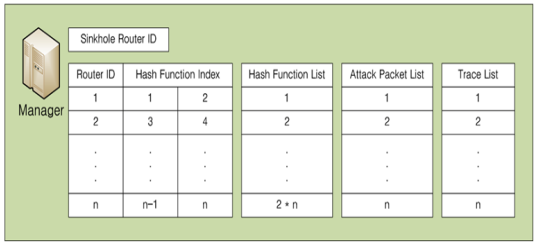


[그림 10] 라우터 내부 구조

4.3 패킷 추적을 위한 Manager 시스템

본 연구에서 제안한 Manager 시스템은 기본적으로 네트워크 그룹에서의 라우터 관리 기능을 한다. 자신이 관리하는 라우터들 안에서 Sinkhole 라우터를 설정하고 모든 라우터들의 정보를 유지, 갱신, 관리를 한다. Manager 시스템은 라우터 관리를 이용하여 설정된 Sinkhole 라우터를 통해 공격 패킷에 대한 정보를 획득하며, 획득된 공격 패킷을 이용하여 Bloom Filter를 사용하는 라우터들을 이용하여 공격 트래픽을 확인하고 추적을 수행한다.

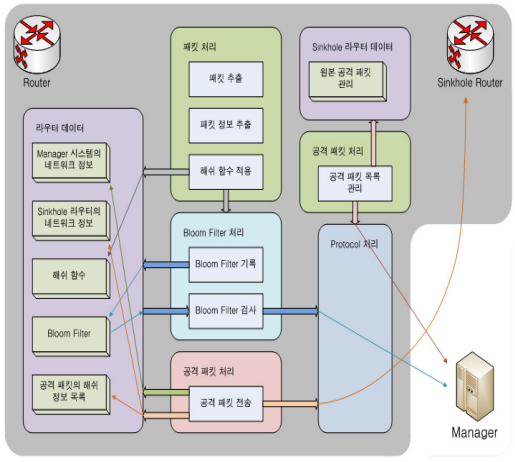
Manager 시스템에서 관리하는 데이터 구조는 다음 그림 11과 같이 해당 네트워크 내에 있는 각 라우터에 대한 정보와 각각의 라우터에서 사용한 해쉬 함수에 대한 정보를 기록하고 있으며, 공격으로 판단된 패킷에 대한 리스트와 IP 추적에 관한 정보를 관리하게 된다.



[그림 11] Manager 시스템 내부 구조

Manager 시스템은 기본적으로 네트워크 그룹에서의 라우터 관리 기능을 한다. 자신이 관리하는 라우터들 안에서 Sinkhole 라우터를 설정하고 모든 라우터들의 정보를 유지, 갱신, 관리를 한다. Manager 시스템은 라우터 관리를 이용하여 설정된 Sinkhole 라우터를 통해 공격 패킷에 대한 정보를 획득하며, 획득된 공격 패킷을 이용하여 Bloom Filter를 사용하는 라우터들을 이용하여 공격 트래픽을 확인하고 IP 추적 기능을 제공한다.

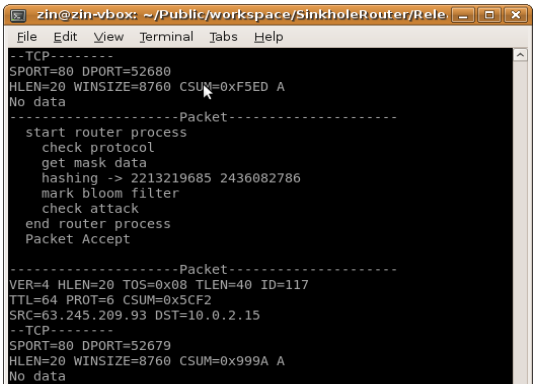
아래 그림 12와 같이 라우터 시스템은 크게 두 가지로 분류할 수 있다. 하나는 일반적인 라우터 기능을 수행하며 Bloom Filter를 적용하는 방법을 제공하는 라우터와 다른 하나는 그 기능과 더불어 Sinkhole 기능을 수행하는 Sinkhole 라우터이다. 두 라우터는 모두 자신을 지나가는 패킷에 대해 해쉬 함수를 적용하여 두 개의 해쉬 값을 추출한다. 추출된 해쉬 값은 Bloom Filter에 적용하여 패킷이 지나갔는지의 유무를 판단할 수 있다.



[그림 12] Manager 시스템의 모듈 구성도/작동방식

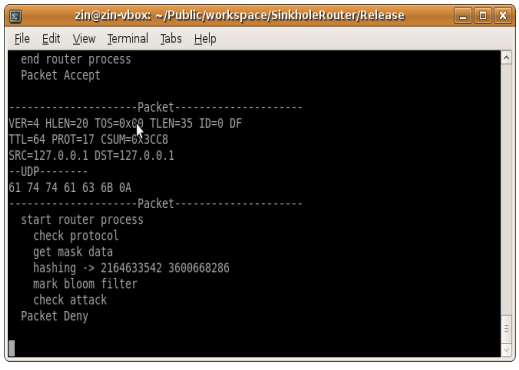
4.4 구현결과

본 연구를 통해 개발된 시스템은 리눅스 환경에서 'IP Queue'를 이용하여 패킷을 추출하며, 추출된 패킷에서 필요한 바이트를 'SHA-256'알고리즘을 이용한 해쉬 함수를 적용하여 해쉬 값을 생성한다. 파일시스템에 생성된 Bloom Filter에 생성된 해쉬 값을 기록하고 패킷의 처리를 진행한다. 그림 13과 같이 라우터에 들어오는 패킷은 모두 'IP Queue'에 쌓이게 된다. 개발 시스템은 'IP Queue'에 쌓인 패킷을 하나씩 읽어 처리한다.



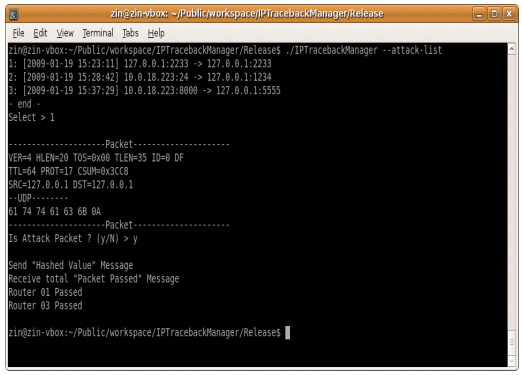
[그림 13] 라우터 수행 과정

또한, ‘SHA-256’ 알고리즘을 이용하여 256비트의 해쉬값을 생성하여 Bloom Fliter에 기록된다. 그 후에 아래 그림 14와 같이 라우터가 소유하고 있는 공격 해쉬 목록에서 해쉬값을 검사하고, 존재할 경우 패킷을 거부하는 방식으로 공격을 탐지하고, Sinkhole 라우터로 전송하며 존재하지 않을 경우 패킷을 승인시켜 일반적인 라우팅이 진행되도록 한다.



[그림 14] 공격 패킷의 판단 화면

아래 그림 15와 같이 공격 패킷 리스트에 있는 3개의 패킷 중 실험 대상이 되는 ‘패킷 1’에 대해 ‘Attack Packet’으로 설정한다. 설정과 함께 Manager 시스템은 ‘Hashed Value’메시지를 모든 라우터에게 전송하고, 그 메시지를 받은 라우터는 그 결과를 ‘Packet Passed’메시지로 응답한다. Manager 시스템은 모든 라우터에게 ‘Packet Passed’메시지를 받은 후, Passed 값이 true인 라우터의 ID를 보여줌으로써 어느 라우터를 통해 전송되었는지 확인이 가능했다.



[그림 15] 공격 경로 추적 실험 결과

5. 결론

최근 All-IP 네트워크 환경이 구축되면서 다양한 형태의 트래픽이 송수신되고 있으며, 이와 더불어 다양한 형태의 공격이 급증하고 있어 이에 대한 능동적 대응 방안이 제시되어야 한다. 이에 본 연구에서는 All-IP 네트워크 환경을 중심으로 DDoS 공격 등이 발생하였을 경우 공격 근원지 정보를 추적할 수 있는 기술에 대해 연구하였다.

기존의 기법은 All-IP 네트워크 환경에 적용할 경우 최적의 성능을 제공하지 못하기 때문에 이에 적합한 공격 추적 방법을 제시할 필요가 있었다.

따라서 본 연구에서는 기존 SPIE 및 Sinkhole 기반 라우터 기법의 장단점에 대한 분석을 통해 두 방식을 결합하여 All-IP 네트워크 환경에 적합한 방식으로 개선하였으며, 이를 통해 공격 추적 기능을 개선하고자 하였다. 제안한 기법에서는 특정 네트워크를 관리하는 Manager 시스템을 두어 패킷에 대한 관리 및 모니터링 기능을 제공하며 공격 발생시 Sinkhole 라우터로 해당 패킷을 전달하는 기능을 제공한다.

본 연구에서 제시한 개선된 추적 기법인 경우 기존 SPIE 시스템보다 간편하고 최소화된 메모리 구조를 이용하여 Bloom Filter 정보를 저장할 수 있으면서도 효율적인 공격자 추적 기능을 제공한다. 또한 본 연구에서 제시한 기법인 경우 기존의 Sinkhole 방식과 유사하게 공격 패킷에 대한 수집/모니터링 기능을 제공하면서도 Manager 시스템을 기반으로 공격 패킷에 대한 판단 및 수집/제어 기능을 제공한다.

따라서 본 연구에서 제시한 기법은 기존 기법의 장단점에 대한 분석을 통해 두 방식을 결합하여 All-IP 네트워크 환경에 적합한 방식으로 개선하였으며, 이를 통해 공격 추적 기능을 개선할 수 있었다.

참고문헌

[1] B. Bloom, "Space/Time Tradeoffs in Hash Coding with Allowable Errors," Cmmunications of the ACM 13(7) pp.422─426, 1970.

[2] A. C. Snoeren, C. Partridge, L. A. Sanchez, C. E. Jones, F. Tchakountio, S. T. Kent, and W. T. Strayer, “Hash-Based IP Traceback,” ACM SIGCOMM Computer Communication Review (Proceedings of the 2001 SIGCOMM Conference) 31(4), pp.3-14, 2001.

[3] Andrei Broder and Michael Mitzenmacher, "Network Applications of Bloom Filters: A Survey," Internet Mathematics Vol. 1, No. 4: 485-509, 2003.

[4] S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Network support for IP traceback," ACM/IEEE Transactions on Networking, 9(3), pp.226-239, 2001.

[5] Andrey Belenky, Nirwan Ansari, "On IP Traceback," IEEE Communication Magazine, pp.142-153, July, 2003.

[6] Victor Oppleman, "Network Defense Applications using IP Sinkholes," http://vostrom.com/get/netdef_en.pdf

[7] Chen Kai, Hu Xiaoxin, Hao Ruibing, “DDoS Scouter : A Simple IP Traceback Scheme,” http://blrc.edu.cn/blrcweb/publication/kc1.pdf

[8] Udaya Kiran Tupakula12 and Vijay Varadharajan, "Tracing DDoS Floods: An Automated Approach," Journal of Network and Systems Management, Vol. 12, No. 1, pp.111-135, March 2004

[9] A. Yaar, A. Perrig, and D. X. Song, "FIT: Fast internet traceback," In Proceedings of IEEE INFOCOM, Miami, FL, Mar. 2005.

[10] Hong-bin Yim and Jae-il Jung, "IP Traceback Algorithm for DoS/DDoS Attack," APNOMS 2006, LNCS 4238, pp. 558-61, 2006.

[11] Byungryong Kim, "Efficient Technique for Fast IP Traceback," CDVE 2006, LNCS 4101, pp. 211 -218, 2006.

[12] 김태욱, 성경상, 오해석, “효율적 키 관리 방식 적용을 통한 전자문서 암호화에 관한 연구,” 한국산학기술학회논문지, Vol.10, No.5, pp.1000-1008, 2009. 5.

[13] 정민경; 신승수; 한군희; 오상영, “스마트카드를 이용한 원격 시스템 사용자 인증 프로토콜,” 한국산학기술학회논문지, Vol.10, No.3, pp.572-578, 2009. 3.

[14] 이영민; 노영섭; 조용갑; 오삼권; 황희융 , “SIP 프록시 서버의 부하 최소화를 위한 분산 처리,” 한국산학기술학회논문지, Vol.9, No.4, pp.929-935, 2008. 8.

이 형 우(Hyung-Woo Lee) **[정회원]**

- 1994년 2월 : 고려대학교 전산과학과 (전산학 학사)
- 1996년 2월 : 고려대학교 일반대학원 전산과학과 (전산학석사)
- 1999년 2월 : 고려대학교 일반대학원 전산과학과 (전산학박사)
- 1999년 3월 ~ 2003년 2월 : 백석대학교 정보통신학부 조교수
- 2003년 3월 ~ 현재 : 한신대학교 컴퓨터공학부 교수

<관심분야>
네트워크 보안, 정보보호, 무선네트워크