SybilGuard 에서의 부하 분석 및 부하균등 방법 제시

김형석*, 김기영*, 염헌영*
*서울대학교 컴퓨터공학부
e-mail : {hskim, kykim, yeom}@dcslab.snu.ac.kr

An Analysis of Random Routes in SybilGuard

Hyeong Seog Kim*, Ki Young Kim*, Heon Young Yeom*
*Dept. of Computer Science and Engineering, Seoul National University

요 약

P2P 및 Mobile Network, Reputations System 등의 분산 시스템은 sybil attack 에 노출되어 있다. sybil attack 은 한 명의 사용자가 다수의 식별자를 가진 것으로 위장하여 시스템 내에서 마치 실제 다수의 사용자인 양 시스템을 악용하는 공격방법이다. sybil attack 을 막기 위한 다양한 노력이 진행되었고, 최근에 SybilGuard 라는 social network 를 이용한 방어 방법이 제시되었다. SybilGuard 는 악의적인 사용자를 막기 위하여, Random Walk 의 변형이면서 결정적인 경로의 특징을 가지는 임의경로(Random Route)를 사용하여 악의적인 사용자의 sybil attack 을 방어한다. SybilGuard 는 sybil node 의 개수를 제한하고, 이들을 하나의 동일한 그룹으로 분류할 수 있도록 하여 시스템 내에서 가짜 식별자의 개수를 제한한다. 이를 위해 각 노드가 시스템에 돌어올 때 Verifier(V)노드가 이들 노드를 확인하게 되는데, 이를 위해 시스템 내의 선한 노드(Honest Node)를 사용하여 이들을 확인한다. 이 때, honest node 들은 verifier 의 요청에 따라 확인요청을 수행하게 되는데, social network 의 특성상 몇몇 노드들은 사회적인 명망으로 매우 큰 링크수를 가지게 될 것이며, 따라서 이들 노드들이 처리해야할 요청의 양이 매우 많아지게 될 것이다. 따라서 이들 honest node 들 간에 로드분포를 균등하게 하는 것이 요구되며, 이 논문에서는 부하 조절을 하기 위한 기법을 제시하고, 이들을 평가한다.

1. 서론

분산 시스템에서 사용자에 의한 Byzantine failure 와 같은 악의적인 행동들이 문제가 되고 있는데, 지금까지의 대부분의 시스템들에서는 각 노드들이 하나의 식별자를 가진다고 가정하여 연구를 진행해왔다. 이런 상황에서 악의적인 사용자가 다수의 식별자를 가진 것으로 위장하여 시스템에서는 마치 각각 다른 식별자를 가진 여러 사용자로 둔갑하게 되면, 노드의 개수에 따라서 시스템의 보안체계를 확립하는 byzantine-fault-tolerant 시스템에서는 큰 문제로 대두된다. 이런 sybil attack 을 완전히 차단하거나 혹은 sybil attack 을 줄이기 위해서 다양한 연구가 진행되었다.

믿을 수 있는 인증기관 (Trusted CA) 을 활용하면 sybil attack 을 차단할 수 있다. 하지만 식별자의 유일성을 보장하는 데 있어서 문제가 발생할 수 있다. 최근 많이 연구되는 방법은 자원검사 (Resource Testing) 을 통하여 sybil attack 을 줄이는 것이다. 이 방법은 각 노드가 현재 프로토콜상 가지고 있어야 할 자원보다 적은양의 자원을 가지고 있는지 체크하여 각 노드의 유효성을 검증하는 것이다. 여기서의 자원에는 계산능력, 저장능력, 네트워크 bandwidth, IP 주소 등 다양한 자원을 대상으로 연구가 진행되었다.

SybilGuard 는 이 방식의 하나로서 제시되었는데, sybil attack 자체를 완벽히 차단하는 방법이 아니라, sybil attack 의 영향을 최소화하기 위한 분산된 프로토콜이다. SybilGuard 는 사회적 네트워크(social network)를 바탕으로 각 노드들간의 믿을 수 있는 관계를 기본적으로 가정한다. SybilGuard 에서 기본적으로 문제를 풀기 위한 방법은 전체 노드들을 선한 구역(Honest Region)과 악의적인 구역(Sybil Region)으로 나누는 것이 NP-Hard 문제이기 때문에 이를 우회하여 선한구역과 악의적인 구역을 나누는 중요한 에지인 attack edge 의 수를 최소화하여 sybil attack 의 영향을 최소화하는 것이다. 이를 위해 SybilGuard 에서는 각 노드들이 증인 테이블(Witness Table) 및, 등록테이블(Registry Table)을 가지고 있으며 이를 통해 각 노드가 시스템 내에 들어올 때 노드들의 테이블들을 비교하여 시스템에서 기대되는 정상적인 정보를 가지고 있으면 해당 노드를 선한 노드라고 분류하고, 만약 그렇지 않다면 이 노드를 sybil node 로 분류하는 것이다. 이 과정에서 증인 테이블과 등록 테이블을 확인하는 작업이 몇몇 선한 노드에서 일어나게 되는데, 이 과정에서 선한 노드는 츠가적인 계산을 하게 된다. 우리는 이 확인작업을 하는 노드가 해당 노드의 social network 에서의 링크수에 영향을 받는 것을 실험으로 확인해보았다. 실험 결과 링크수가 높은 유명한 노드는 확인 작업을 하는 확률이 높아졌으며, 이런 상황에서는 파일 공유 P2P 에서 유명 파일을 가지고 있는 노드가 다운로드 요청을 많이 받아 자신의 자원을 희생해야 하는 것처럼 유명 노드가 확인 작업을 해야

하는 경우가 많아져 부하의 불균형이 초래된다. 따라서 각 노드들 간의 공정성을 보장하기 위해서는 시스템내에서 링크수에 따른 verification request 를 적절히 분배하는 것이 필요하게 된다. 실제로 이를 적용하여 실험한 결과, 각 노드들의 부하 분포의 편차를 매우 줄일 수 있었다. 이 논문에서는 SybilGuard 시스템 내에서 공정성을 보장하기 위한 기법을 제시한다.

2. SybilGuard 의 Random Route 분석

각 노드는 자신의 모든 에지에 대해서 임의 경로(Random Route)를 생성한다. Random Route 는 각 노드의 경로 테이블(Routing Table)에 의해서 정해지는데 Routing Table 은 다음과 같이 구성된다. 각 노드는 자신의 에지인 $e_1 \ldots e_n$ 에 대해서 임의의 순서로 순열을 구성하여 $e_1' \ldots e_n'$의 시퀀스를 구성한다. 이렇게 하여 e_1 에서 들어오는 메세지는 e_1'으로 가고, e_n 에서 들어오는 메세지는 e_n'으로 전달되는 테이블을 구성한다. 모든 노드들이 Routing Table 을 구성하고 나면, 각 노드는 자신의 에지 $e_i(1 <= i <= n)$에서 시작하여 각 노드의 Routing Table 에 따라서 결정되는 w 의 길이를 가지는 Random Route 를 구성한다. SybilGuard 에서는 이 경로를 저장하고 관리하기 위해 각 노드가 증거 테이블(Witness Table)과 등록 테이블(Registry Table)을 유지하도록 한다. 증거테이블은 자신의 임의경로를 저장하는 테이블이고, 등록 테이블은 다른 노드의 임의경로가 자신을 거쳐갔을 경우, 이를 기록해두는 테이블이다. 임의경로를 구성한 노드는 시스템에 들어오기 위해 Verifier(V)라는 특정 노드에게 요청을 보내게 된다. 이를 받은 Verifier 는 확인 프로토콜(Verification Protocol)을 수행하여 각 노드의 적합성을 점검한다.

SybilGuard 에서 사용하는 Social Network 는 Kleinberg (1)의 모델을 따르고 있다. 모든 노드는 nxn 의 정방형 격자 안에 놓여있고, 이들 격자를 각각 lattice point 라고 정의한다. 두 노드 (i, j)와 (k, g)간의 lattice distance 는 다음과 같이 정의한다.

d((i, j),(k, g))=|k-i|+|g-j|

각 노드 u 는 lattice distance 가 최소인 p(>1)개의 local contact 를 가진다. 여기서 p 는 시스템 내에서 공유하는 상수이다. 또한 각 노드 u 는 q 개의 다른 노드와 링크를 맺는데 이 노드들을 long-range contact 라 한다. 이는 독립적인 임의시행(Random Trial)으로 정해지는데, 각 노드 u 는 v 와 $d(u,v)^{-r}$ 에 비례하는 확률로 링크를 가지게 된다. SybilGuard 에서는 백만 노드와 만개의 노드의 경우, p = q = 8, r = 1.98 을 적용하였고, 이 때 각 노드의 평균 링크수는 24 정도가 되고 분산은 2.54 정도 된다. 즉, 각 노드들의 링크수의 분포가 거의 균일하다는 것이다. 또한 SybilGuard 에서는 각 노드의 링크수의 최대값을 30 으로 두어 각 노드가 가질 수 있는 링크의 개수를 제한하였다. 하지만 실제 social network 에서는 이와 같은 가정보다는 각 노드들이 비슷한 환경에서 생활하거나, 같은 직장 혹은 학교의 노드들은 클러스터를 구성할 것이고, 또한 매우 유명한 노드는 평균 degree 보다 매우 큰 link 의 개수를 가지게 될 것이다. 따라서 이 같은 환경을 시뮬레이션하기 위하여 우리는 각 노드의 degree 가 Zipf 분포를 따르도록 위의 모델을 수정하였다. 수정한 모델에 따르면 각 노드는 위와 동일하게 p 개의 local contact 를 가진다. 달라지는 점은 각 노드의 long-range contact 의 개수이다. 각 노드의 링크수가 Zipf 분포를 따르도록 하기 위해 노드에 따른 q 를 아래와 같이 정의하였다.

$q_r = \frac{n}{r}$ (r = 노드의 순위, n=normalization parameter)

각 노드의 순위는 랜덤하게 결정하였고, normalization parameter 는 노드의 개수에 따라 다르게 결정하였다.

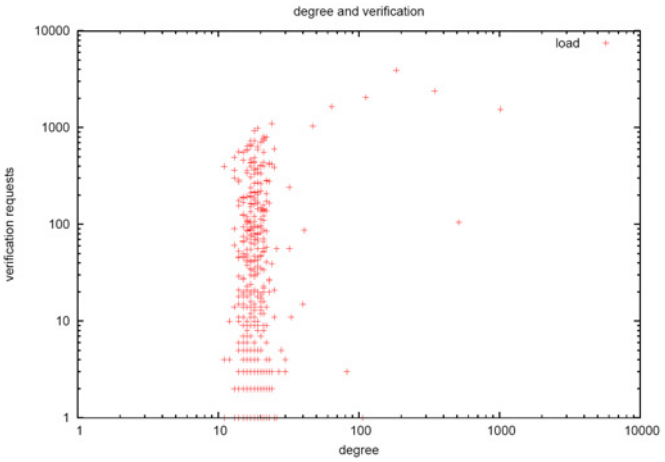

그림 1 Zipf 분포하에서의 SybilGuard 를 수행했을 때의 부하분포

그림 1 은 우리가 제시한 Zipf 분포를 따르는 Social Network 하에서 SybilGuard 를 수행했을 때 각 노드가 받는 부하를 각 노드의 링크수에 따라서 보여주는 그래프이다(x 축과 y 축 모두 log-scale 임을 유의). 이 그림을 보면 노드의 링크수가 커질수록 각 노드에서 수행되는 확인요청이 매우 커지는 것을 확인할 수 있다. 이를 확인하였기 때문에 우리는 각 노드의 링크수를 고려한 새로운 확인 프로토콜을 제안한다.

3. 제안하는 알고리즘

SybilGuard 에서의 부하 불균등의 가장 큰 원인은 노드의 링크수를 고려하지 않고, 임의 경로의 교차점 중 첫번째 노드에게 확인요청을 보내는 데에 있다. 이 방법은 가장 단순하면서, Verifier 의 부담을 덜어줄 수 있는 방법이다. 하지만 위에서 서술한 것처럼 부하 불균등의 문제가 나타나기 때문에 이를 다음과 같은 방법을 통해 해결하였다. 이 문제를 풀기 위한 우리가 제안하는 첫 번째 방법은 교차하는 Verifier 에서 임의경로의 원소를 선택할 때, 임의성을 주어 결정적인 행동을 막는 것이다. 다른 방법은, Verifier 가 Suspect 와의 모든 임의 경로의 교차점을 찾아, 해당 노드의 링크수에 반비례하는 확률에 임의선택하는 것이다. 표 1 에 우리가 제안한 두 번째 알고리즘을 기

술하였다.

표 1 제안하는 알고리즘

S 가 자신의 증거테이블과 public key 를 V 에게 보낸다 V 의 증거테이블의 모든 원소 T_V에 대해서 S 의 증거테이블과 T_V에 동시에 있는 원소 X 를 찾는다 V 는 X 와 통신하여 X 의 링크수를 얻어 링크수테이블에 저장한다 링크수테이블에 있는 원소들 중에서 각 링크수에 반비례하는 확률로 하나의 노드를 뽑는다 V 는 X 에게 통신하여 실제로 S 가 X 의 증거테이블 안에 존재하는지 확인하고, 존재한다면 S 를 받아들인다.

SybilGuard 에서는 T_V 와 S 의 임의경로가 교차하는 첫번째 노드 X 를 찾고난 후에 이 노드에게 확인요청을 보내 S 의 적합성을 검사한다. 우리는 이를 수정하여 S 와 V 의 모든 교차노드를 찾아내어 이 노드의 IP 와 링크수를 저장하는 step 을 추가하였다. V 의 임의경로와 T_V 와 S 의 임의경로의 모든 교차점을 찾은 후 이들에게 링크수를 요청하여 링크수테이블에 저장한다. 이들 중에서 하나의 노드 X 를 뽑는데 이를 링크수와 반비례하는 확률을 가지고 랜덤하게 선택하게 된다. 우리가 제안한 첫 번째 방식은 노드선택에 있어 임의성을 주면서 수행시간에 변화를 주지 않는 방법이고, 두 번째 방식은 모든 교차노드를 찾는 추가적인 시간이 발생하지만, 링크수를 고려하여 확인요청을 보낼 대상을 찾기 때문에 링크수의 분포가 확연히 다른 네트워크에서 좋은 성능을 발휘할 것이다.

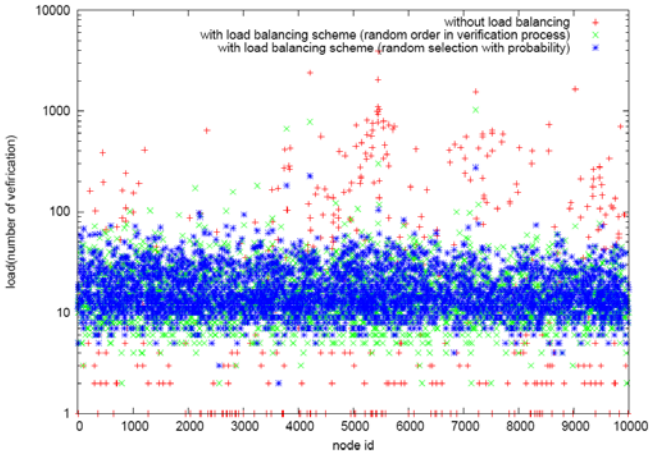

그림 2 Zipf 분포하에서의 부하 분포

그림 2 는 위에서 제안한 두 가지 방식의 성능을 보여주는 그래프이다. 이 그래프는 10000 개의 노드에서 p=8, normalization paratemer=1000 을 주어 만든 social network 에서 SybilGuard 와 제안한 두 가지 방식의 결과를 보여주는 그래프이다. x 축은 각 노드의 ID 를 나타내고, y 축은 해당 노드에서 수행한 확인요청회수를 나타낸다. 빨간점은 기본 SybilGuard 이고 녹색점은 첫 번째 방법, 파란점은 두 번째 방법이다. 그림을 보면 SybilGuard 는 로드의 불균형이 두드러지게 나타나고 있다.

이를 정량화하기 위해 각 방법의 Fairness Index (2)를 다음과 같이 정의하고 이를 계산하였다. Fairness Index 는 다음과 같이 정의된다.

$$I(D) = \frac{(\sum_{i=1}^{m} D_i)^2}{m \sum_{i=1}^{m} D_i^2}$$

여기서 D_j 는 각 노드에서 수행한 확인작업의 횟수에 해당된다. 이는 각 노드에서의 로드가 불균등하면 0 에 가까워지고, 균등해지면 1 에 가까워진다. 계산결과 I 값은 각각 .0077, .0949, and .2366 으로 나왔다. 즉, SybilGuard 에서의 불균등한 부하가 제시한 방법을 썼을 때에 약 열배에서 칠팔십배 이상으로 좋아지는 것을 확인할 수 있다.

4. 결론

SybilGuard 는 Sybil attack 의 영향을 최소화하기 위하여 social network 의 특성을 활용하여 sybil 노드의 개수를 제한하는 새로운 프로토콜이다. SybilGuard 의 임의경로를 활용한 프로토콜은 social network 에서 상대적으로 많은 링크를 가진 노드에게 많은 부하를 줄 수 있다. 우리는 임의성과 링크수를 고려한 확인 프로토콜을 제안하였고 이를 평가하여 매우 균등한 부하분포를 얻어낼 수 있었다.

참고 문헌

1. “The Small-World Phenomenon: An Algorithmic Perspective.” **KleinbergJon.** 출처 미상: ACM, 2000. Annual ACM Symposium on Theory of Computing.

2. **JainRaj.** “The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling.” 출처 미상: Wiley-Interscience, 1991.

3. **LevineNevilBrian, ClayShields, MargolinBoris.** “A Survey of Solutions to the Sybil Attack.” 출처 미상: University of Massachusetts Amherst, 2006.

4. “SybilGuard: Defending Against Sybil Attacks via Social Networks.” **YuHaifeng, 외.** 출처 미상: ACM, 2006. SIGCOMM.