순수 P2P 네트웍 환경에서 프락시 서버 할당을 위한 동적 피어 선정 기법

김영진[0], 김문정, 엄영익
성균관대학교 정보통신공학부 분산시스템연구실
email : door21c@tjssm.co.kr, tops@ece.skku.ac.kr, yieom@ece.skku.ac.kr

Dynamic Peer Selection Scheme for Allocating Proxy-Server on Pure P2P Network Environments

Young-jin Kim[0], Moon Jeong Kim, Young Ik Eom
School of Information and Communication Engineering, Sungkyunkwan University

요 약

현재 네트웍 보안을 위한 방화벽과 유일한 공용 IP 주소 부족 문제를 해결하기 위한 NAT 사용이 일반화되고 있다. 그러나 이러한 환경 안에 있는 사용자들은 외부 피어의 연결을 받지 못하여 제한된 서비스만을 받고 있기 때문에, 순수 P2P 환경에서 피어들간의 연결을 방해하는 가장 큰 요소가 되고 있다. 본 논문에서는 이러한 문제점을 해결하기 위해 P2P 환경에서 방화벽/NAT가 아닌 피어들 중 네트웍 트래픽이 많지 않은 피어를 동적으로 선정하여 프락시 서버를 할당한 후 피어들의 통신 매개체로 사용하는 기법을 제안한다. 이 제안기법은 모든 운영체제에서 사용 가능하고, 양방향 통신을 가능하게끔 해주며, 네트웍 트래픽을 여러 피어들에게 분산키는 장점을 갖고 있다. 차후 파일 전송, 화상 채팅 등의 여러 가지 서비스를 사용하는데 유용한 기법이 될 것이다.

1. 서 론

최근 클라이언트/서버 모델의 문제점인 서버로의 네트웍 트래픽(Network Traffic) 증가와, 차후 서비스 제공자가 서버를 제공하지 않으면 사용자들는 더 이상의 서비스를 제공받을 수 없는 등의 이유로 각 단체에서는 보다 빠른 서비스를 제공할 수 있는 P2P (Peer-to-Peer)모델로 발전해가고 있는 추세이다[1].

그러나 각 피어간(Peer) 상호 연결하여 서비스를 받고자 할 때, 신뢰할 수 없는 네트웍으로부터 내부 네트웍을 보호하기 위해 네트웍 소켓 연결을 제한하는 방화벽(Firewall)에서는 특정 포트로 접속을 하는 외부에서의 연결이 실패하는 경우가 많이 발생한다. 또한 부족한 공용 IP 주소 문제를 해결하기 위해 임시적으로 사용하고 있는 NAT(Network Address Translator) 환경 안에 있는 피어들의 존재를 외부에서 알 수 없기 때문에 네트웍 소켓을 시도할 수 없게 된다. 이러한 이유 등으로 인해 P2P 환경에서의 피어들은 상호 연결을 못하게 되어 완벽한 서비스를 제공하거나 받지 못하고 있다[2, 3, 4]. 이를 해결하기 위해 여러 가지 기법들이 제안되고 있지만, HTTP 포트 연결 방법은 방화벽의 운영자가 웹 포트를 개방해야 하고 다른 응용 프로그램이 이 포트를 사용하지 않아야만 가능하다라는 제한점이 존재한다. 또한, 3way-handshake 방법은 현재 소스가 공개되어져 있는 리눅스에서만 가능하며, 양쪽 피어들이 동시에 연결을 시도해야 하기 때문에 이를 관리해주는 특정 서버가 존재해야 한다는 문제점이 있다. 그리고 프락시 서버(Proxy Server)를 이용한 방법은 특정 서버에 네트웍 트래픽의 증가가 발생하는 문제점들이 있다[5, 6, 7].

본 논문은 기존 P2P 환경의 여러 문제점들을 극복하기 위하여, 순수 P2P 기반에서 방화벽/NAT 환경이 아닌 피어들 중 네트웍 트래픽이 많지 않은 피어를 선정하여 프락시 서버를 할당하게 되며, 이를 이용하여 각 종단간의 원활한 통신을 제공할 수 있는 기법을 제안하고자 한다. 이러한 방법은 모든 운영체제에서 가능하고, 양방향 통신을 가능하게끔 해주며, 네트워크 트래픽을 여러 피어들에게 분산시켜 줌으로써 파일전송, 화상 채팅 등의 여러 가지 서비스를 가능하게끔 하여준다.

본 논문의 구성은 2장에서 P2P 문제점 및 방화벽/NAT에 대한 관련 연구를 소개한다. 3장에서는 P2P 환경에서 프락시 서버 할당을 위한 피어를 동적으로 선정하는 새로운 개념을 기술하며, 4장은 기존 방법과의 차이점을 기술하며, 마지막으로 5장에서는 이에 대한 결론 및 향후 연구로 결론을 맺는다.

2. 관련연구

본 절에서는 현재 P2P 상에서 데이터를 송수신하기 위한 연결에 있어서의 문제점과 이를 해결하기 위한 기존 연구들을 알아보도록 한다.

2.1 현재 P2P의 문제점

많은 단체에서 사용하는 방화벽은 내부 네트웍을 보호하기 위하여 허락되지 않은 포트로 접근하는 외부로부터의 연결을 막고 있다. 또한 NAT를 사용하고 있는 피어들은 각각 가상 IP를 가지고 외부와 통신하게 된다. 따라서, 외부의 피어가 NAT 환경 안에 있는 피어에게 접속을 하고 싶어도 NAT 시스템으로 접속을 하기 때문에 가상 IP를 사용하고 있는 실질적인 NAT 환경 안의 피어에게로는 연결이 불가능하다[2].

물론 부족한 IP 주소 문제를 해결하기 위해 IPv6가 사용된다 하여도 IPv4 주소를 사용하고 있는 사용자들이 IPv6 주소로 변경하기 위해 NAT를 사용할 것이며, NAT를 이용하여 방화벽의 역할을 구현하고자 하는 사용자들이 있을 수 있기 때문에 NAT는 완전히 없어지지는 않을 것이다.[8].

2.2 HTTP 포트 연결 기법

방화벽을 사용하는 대부분의 단체에서는 웹 포트를 이용하여 방화벽 안에 있는 컴퓨터에 연결할 수 있는 방법으로 사용하고 있다. 그러나 방화벽을 사용하는 모든 곳에서 웹 포트를 개방하여 두지 않고, 이미 다른 프로그램에서 사용하고 있는 경우가 있기 때문에 완벽한 해결책이라 할 수가 없다[9, 10].

2.3 3 way-handshake 기법

RFC793에 나와 있는 양방향 3 way-handshake의 사용으로 해결을 시도하는 방법으로, 소스가 공개되어 있는 리눅스 계열에서만 테스트가 성공되었을 뿐, 많은 사용자층을 가지고 있고 P2P 기반 프로그램이 동작할 윈도우즈 운영체제에서는 불가능한 상태이며, 이러한 방법은 양쪽 피어에서 연결을 동시에 시도해야 한다는 문제점이 있다[5, 7].

2.4 프락시 서버 기법

방화벽/NAT 환경에 있는 피어들에게 통신을 할 수 있는 프락시 서

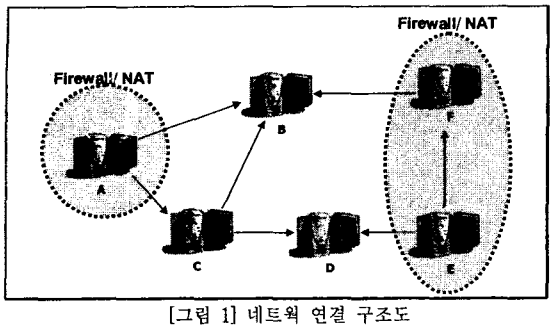
버를 서비스 제공업자가 제공하는 기법이다. 이 기법은 프락시 서버를 제공하는 서버에 네트웍 트래픽을 증가시켜 속도가 느려진다는 문제점이 있다.

3. 제안 기법

3.1 시스템 환경

네트웍 상태는 순수 P2P 상태를 전제로 하며 자기 자신이 방화벽/NAT안에 있는지를 이미 알고 있다고 가정한다. 순수 P2P에서는 자기 자신에 연결된 피어 이외에 어떠한 피어들이 존재하는지를 알 수가 없고, 알고 있다 하여도 그 즉시 IP와 서비스를 제공하기 위한 포트를 알 수가 없다.

본 논문에서 제안하는 기법에 적용될 네트웍 구성은 그림 1과 같다.

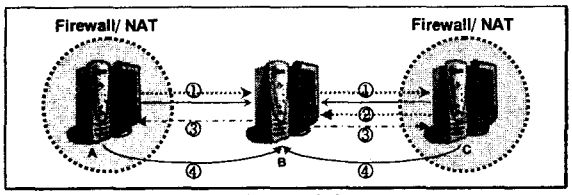


[그림 1] 네트웍 연결 구조도

그림 1은 각 피어간 네트웍 소켓으로 연결이 이루어져 있는 상황을 보여주고 있다. 일반 네트웍 사용자가 방화벽/NAT안으로 연결을 시도할 수 없고, 반대의 연결만 허락되기 때문에 위와 같은 화살표 방향으로 구성되었다.

3.2 기본 동작

본 논문에서의 제안 기법은 그림 2에서와 같이 크게 4단계로 구성된다.



[그림 2] 동작과정

그림 2에서 임의의 피어(A)가 목적지 피어(C)에게 IP/포트를 요청할 때 연결되어 있는 피어(B)에게 연결 요청 메시지(CQP)를 보내게 된다. 중간 피어(B)는 자신이 목적지가 아니기 때문에, 메시지에 자신의 노드를 추가하고 다른 피어(C)에게 메시지(CQP)를 보내게 된다. 연결 요청 메시지를 받은 피어(C)는 최종 목적지가 자기 자신이라는 사실을 알고, 바로 전 피어(B)에게 프락시 서버 요청 메시지(PQP)를 보내게 된다.

메시지를 받은 피어(B)는 프락시 서버를 할당하고 요청한 피어(A)와 연결을 수락할 피어(C) 각각에게 연결하라는 메시지(CPP)를 보낸다. 각각의 피어는 이 메시지를 받은 후 프락시 서버에 접속을 하게 되며 확인 메시지(CFP)를 보내어 인증 작업을 거친다.

3.3 프락시 서버 역할

프락시 서버는 방화벽/NAT 환경 안에 있는 피어들을 위해 할당되는 것으로써, 원활한 네트웍 통신이 되게끔 지원하는 역할을 한다.

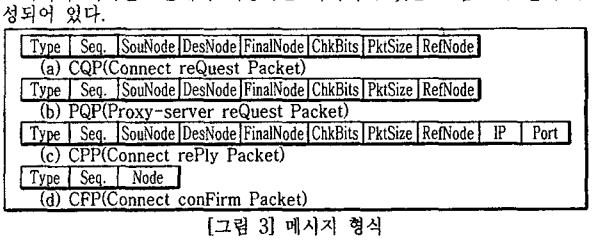
프락시 서버에는 2개의 피어가 연결하여, 한쪽에서 들어온 패킷을 다른 곳으로 패킷을 전달하는 역할을 한다. 물론 다른 피어들의 연결을 수락할 수 있어야 하기 때문에 방화벽/NAT 환경이 아닌 피어만이 생성할 수 있다.

차후 피어가 프락시 서버에 연결할 때 인덱스 값을 보내게 되며, 이 때 프락시 서버는 인덱스를 이용하여 서비스를 제공할 피어인지 아닌지 판별한다.

3.4 제안 기법 설계

(1) 메시지 형식

위에서 제시한 4단계에 해당하는 메시지 포맷은 그림 3과 같이 구성되어 있다.



[그림 3] 메시지 형식

각 필드의 정보는 표 1과 같다.

[표 1] 필드 정보

명 칭	설 명
Type	패킷 구별 필드
Seq	Message Sequence
SouNode	연결을 원하는 노드
DesNode	연결하고자 하는 목적지 노드
FinalNode	메시지 최종 목적지
ChkBits	RefNode Node의 방화벽/NAT 체크 비트
PktSize	RefNode 개수
RefNode	중간 노드들

(2) 알고리즘

메시지 구성은 다음과 같이 총 4가지로 구성되며, CQP, PQP, CRP는 그림 4의 a, b, c로, CFP는 그림 4의 d로 메시지를 구성된다.

각각의 메시지들에는 시퀀스(Sequence) 값이 들어있어, 이미 받은 메시지는 다시 송신하지 않는다. 또한 최종 목적지에 온 동일한 시퀀스 메시지(CQP) 중 가장 처음에 오는 메시지의 경로가 가장 빠른 것으로 처리한다.

1) CQP(Connect reQuest Packet)

다른 피어로의 연결에 필요한 상대방의 IP와 포트를 얻어오기 위해 생성하는 메시지이다. 연결하기 원하는 피어에서는 CQP 메시지를 보냄으로써 상대방의 IP와 포트를 수신할 수 있다.

알고리즘 1은 연결 요청 메시지를 만드는 피어에 해당한다.

[알고리즘 1] CQP 메시지 생성 알고리즘

```
input : other peer node
output: none
{
    static int seq=0;
    message = type(CQP)+Seq(seq)
        +SouNode(myNode)+DesNode(other peer node)
        +FinalNode(other peer node);
    send message to all peer that is connected;
}
```

알고리즘 2는 CQP 메시지를 받았을 때 중간 노드인지 최종 목적지 노드인지에 따라 처리하는 알고리즘이다.

[알고리즘 2] CQP 메시지 처리 알고리즘

```
input : Seq, SouNode, DesNode, FinalNode, RefNode
output: none
{
  if ( FinalNode == myNode ) {
    if ( this.CheckBit == FALSE ) { // (1)
      create service port;
      make CPP message and send message to previous peer;
      return;
    }
    if ( SouNode.CheckBit == TRUE ) { // (2)
      while(_peer=(get last peer of RefNode)) {
        if ( _peer.CheckBit == FALSE ) {
          make PQP message and send message to previous peer;
    /* continued next page */
```

```
            remove last peer of RefNode;
            _peer = get last peer of RefNode;
            if ( peer.CheckBit == FALSE || peer.IP != this.myPublicIP )
               connect _peer for P2P;
            return;
         } // end if
         remove last peer of RefNode;
      } // end while
   } // end if
   make PQP message and send message to previous peer;   // (3)
   return;
} // end if
if ( history.CQP.SouNode.Seq == Seq ) return;
history.add(CQP, Seq, SouNode);
message.RefNode += this.Node;
send message to all peer except previous peer;
}
```

CQP 메시지는 방화벽/NAT 환경에 목적지가 해당이 안될 경우(1), 요청지와 목적지 모두 해당될 경우(2), 그리고 요청지만 해당될 경우(3)로 크게 구분된다.

2) PQP(Proxy-server reQuest Packet)

CQP 요청 피어나 CQP 수락 피어 모두 보낼 수 있는 메시지이며, 프락시 서버를 할당하여 양쪽 노드에게 IP와 포트를 전송하여 연결하라는 메시지를 전달한다. 단, PQP메시지의 목적지 노드가 연결을 요청했던 노드와 일치할 경우, 상대방이 접속할 수 있도록 IP와 포트 정보를 CPP에 담아 상대방 피어에게 전달한다.

[알고리즘 3] PQP 메시지 처리 알고리즘

```
input : Seq, SouNode, DesNode, FinalNode, RefNode
output : none
{
   if ( FinalNode == myNode ) {
      if ( SouNode == FinalNode ) {
         create service port;
         make CPP message and send message to previous peer;
         return;
      }
      create PS(Proxy-Server);
      make CPP message for SouNode;      // message1
      make CPP message for DesNode;      // message2
      send message1 to SouNode;
      send message2 to DesNode;
      return;
   }
   if ( history.PQP.SouNode.Seq == Seq ) return;
   history.add(PQP, Seq, SouNode);
   message += this.Node;
   send message to all peer except previous peer;
}
```

3) CPP(Connection rePly Packet)

알고리즘 4에서는 IP와 포트정보를 바탕으로 프락시 서버, 또는 상대방의 피어에게 연결을 시도하게 된다.

[알고리즘 4] CPP 메시지 처리 알고리즘

```
input : Seq, SouNode, DesNode, FinalNode, RefNode, IP, Port
output : socket handle
{
   if ( FinalNode == myNode ) {
      connect IP, Port;
      return socket handle;
   }
   if ( history.CPP.SouNode.Seq == Seq ) return;
   history.add(CPP, Seq, SouNode);
   message += this.Node;
   send message to all peer except previous peer;
}
```

4) CFP(Connection conFirm Packet)

프락시 서버에 연결된 상태에서, 인증 과정을 거치는 메시지이다.

[알고리즘 5] CFP 메시지 처리 알고리즘

```
input : Seq, Node
output : none
{
   if ( Seq != this.Seq ) {
      disconnect peer;
      return;
   }
   if ( Node != this.SouNode || Node != this.DesNode )    disconnect peer;
}
```

4. 비교 분석

기존에 제시된 기법들과 본 논문에서 제안하는 기법을 표 2에서 비교하였다.

[표 2] 방화벽/NAT 해결방법들의 비교 분석

	모든 OS	(Redhat)Linux	모든 OS	모든 OS
	양방향 연결 (설정 따라 단방향)	양방향 연결	양방향 연결	양방향 연결
	낮음	낮음	높음	중간
	방화벽 환경인 피어에 연결 가능	직접 연결이 가능	서비스 제공 용이	모든 OS에서 가능 순수 P2P에서 가능 네트웍 트래픽 분산
	포트 충돌 NAT 불가능 방화벽 환경설정에 종속적	동시 연결하기 어려움	프락시 서버의 네트웍 트래픽이 큼	직접적으로 참여하지 않는 피어에 네트웍 트래픽 부여

(상대적 비교 결과값임)

이 기법을 이용한 시스템은 다른 피어에게 네트웍 트래픽을 가중시키는 단점이 있지만, 각각의 피어는 다른 피어에게 서비스를 제공한다는 입장에서 보면 프락시 서버로서의 역할과 자료를 제공하는 입장이 동일하다고 본다. 이 기법을 사용함으로써 P2P 서비스를 받지 못하였던 방화벽/NAT 환경에 있는 사용자들에게 파일송수신, 멀티미디어 서비스 등을 제공함으로써, 서비스의 질을 높일 수 있게 된다.

5. 결론 및 향후 연구과제

본 논문에서는 순수 P2P환경에서 방화벽/NAT를 해결하기 위한 방법으로 프락시 서버 할당을 위한 동적 피어 선정 기법을 제안하였다.

기존에 지시된 기법들과 비교를 하였을 때 이 기법은 모든 운영체제에서 동작하고, 방화벽/NAT 환경 유무에 상관없이 통신을 할 수 있으며, 어느 한 피어에게 네트웍 트래픽을 가중시키지 않는다는 장점을 가지고 있다.

차후 이동 에이전트의 자원 검색 알고리즘에 적용을 시켜 전체 노드들에게 부여되는 네트웍 트래픽을 감소시킬 수 있는 방법으로 연구를 진행하고, 그 결과 순수 P2P 환경의 이동 에이전트에 적용을 시킴으로서 현실적으로 연결할 수 없는 방화벽/NAT 사용자들에게 보다 나은 서비스를 제공할 수 있을 것으로 기대된다.

참고문헌

[1] A. Oram, "Peer-To-Peer," O'Reilly, Mar. 2001

[2] P. Srisuresh, M. Holdrege, "IP Network Address Translator (NAT) Terminology and Considerations," IETF, RFC2663, Aug. 1999

[3] B. Carpenter, "Internet Transparency," IETF, RFC2775, Feb. 2000

[4] N. Freed, "Behavior of and Requirements for Internet Firewalls," IETF, RFC2979, Oct. 2000

[5] P2PWG, "Bidirectional Peer-to-Peer Communication with Interposing Firewalls and NATs," NAT/Firewall Traserval White paper, P2PWG, Aug. 2001

[6] D. C. Hyde, "How New Peer to Peer Developments May Effect Collaborative Systems," Department of Computer Science Bucknell University, Jan. 2002

[7] J. Postel, "Transmission Control Protocol," IETF, RFC793, Sep. 1981

[8] R. Gilligan, E. Nordmark, "Transition Mechanisms for IPv6 Hosts and Routers," IETF, RFC1933, Apr. 1996

[9] A. V. Anderson and S. H. Robinson, "Peer-to-Peer Network Communications in a World of NATs and Firewalls," An Intel Corporation White Paper, 2001

[10] JXTA, "Project JXTA Virtual Network," Sun Microsystems, Inc., Feb. 2002