THE SYS4U DODUMENT

Peer-to-Peer Communication Across Network Address Translators

- Bryan Ford
- Pyda Srisuresh
- Dan Kegel

SYS4U
Beyond the e-Commerce

목차

I. Introduction

1. Introduction

1. Introduction

현재 인터넷 주소 체계의 방식은 global 주소 영역, private 주소 영역들이 네트워크 주소 변환 (NAT)를 통하여 연결되는 주소 체계이다.
서로 다른 private 네트워크에 존재하는 두 peer가 직접적으로 통신하게 하는 방법이 필요하다.

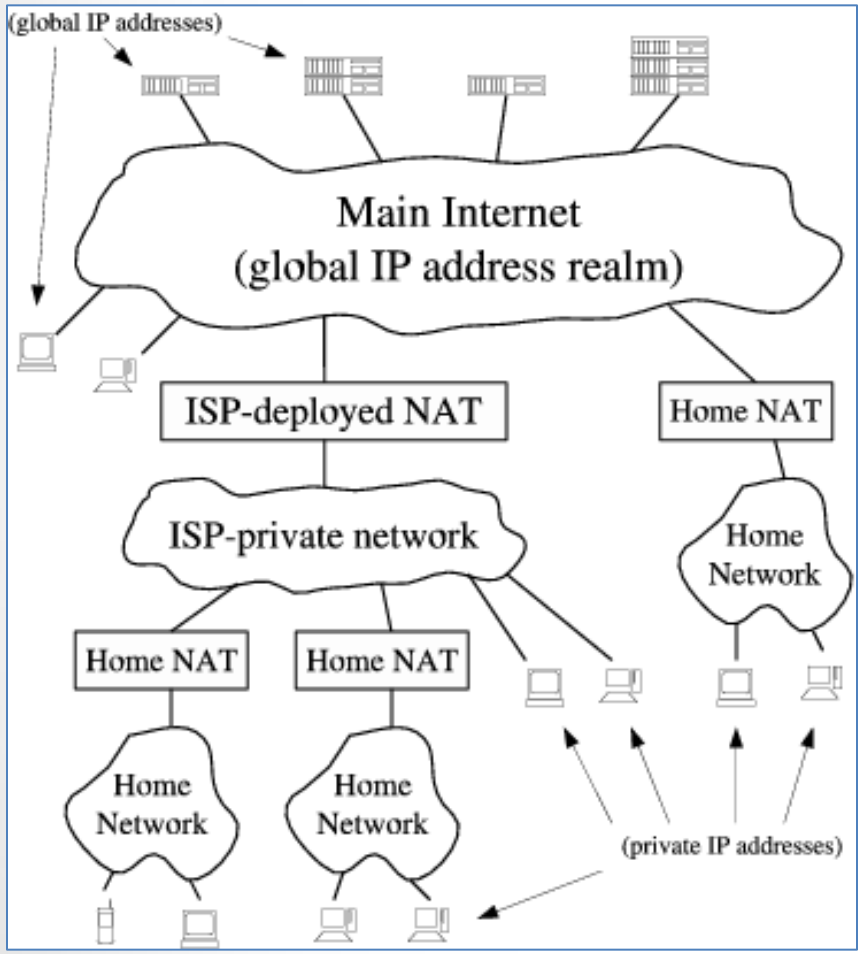


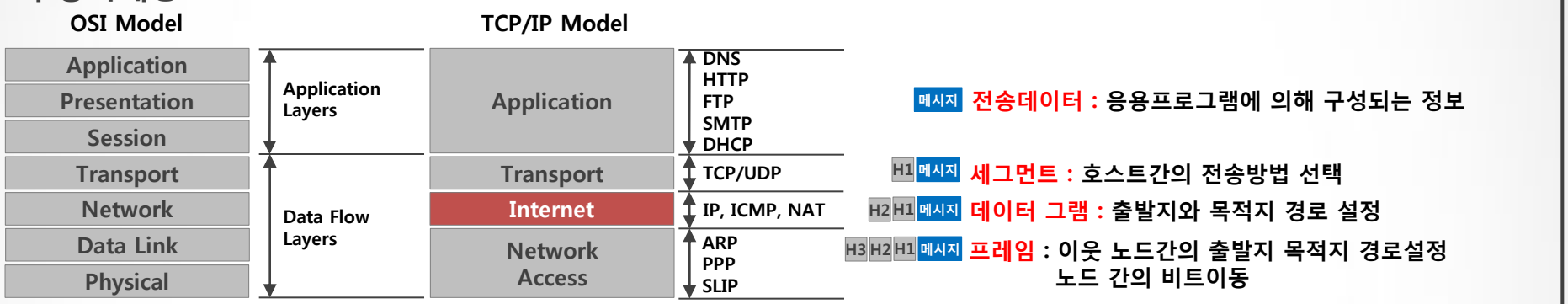
Figure 1: Public and private IP address domains

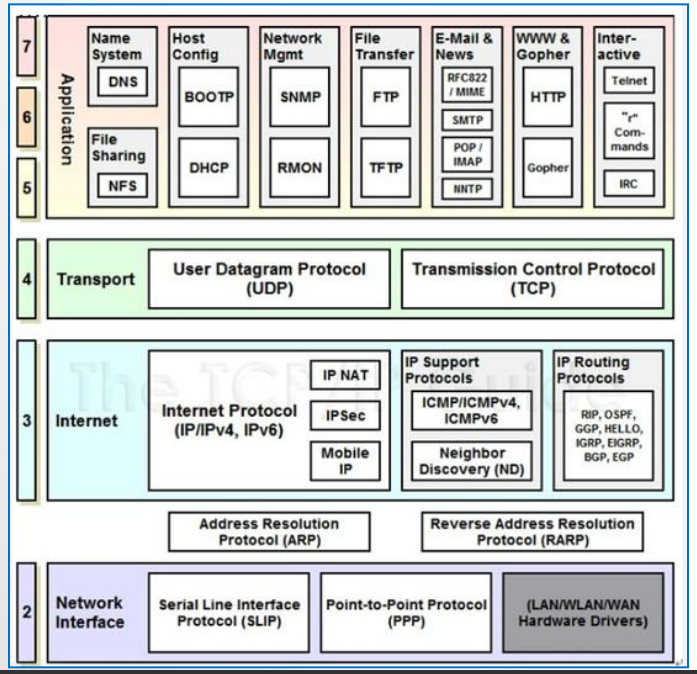
1. Hole Punching(홀 펀칭)

- 홀펀칭의 사전적 의미는 막힌 공간에 구멍을 뚫는 행위를 말하는 것
- 네트워크에서 구멍을 뚫는다는 의미는 NAT라는 네트워크 장비를 두고 있어서 직접적인 TCP/UDP 통신이 불가능한 Peer들간에 직접적인 통신이 가능할 수 있도록 (IP, PORT) 확보하는 것.
- 서로 다른 private 네트워크에 존재하는 호스트들 간의 P2P 연결을 구성하는 가장 효과적인 방법 중 하나.
- UDP기반의 어플리케이션들 사이에서 널리 이용되고 있으며 또한 본질적으로 TCP에도 적용
- 홀펀칭이 필요한 이유는 P2P (Peer-to-Peer) 기반의 게임이나 통신 방식에서는 클라이언트끼리 직접 연결을 맺고 통신을 진행하여야 하는데, 참여하는 클라이언트가 NAT 하위에 존재한다면 직접 연결이 불가능하기 때문에 유무선 공유기가 일반화가 된 환경에서는 P2P 게임이나 서비스를 제공하려는 경우에는 필수적인 기술

1. Introduction – 참고자료

NAT의 동작계층





◆**Application**

✓응용 프로세스 네트워크에 연결 할 수 있게 해서 자료를 송수신할 수 있는 인터페이스를 제공하는 계층(웹 브라우저등 응용프로그램)

◆**Presentation**

✓데이터 표현의 차이를 해결하기 위해 자료의 형식을 변환 해주거나 공통의 형식을 제공해주는 역할을 하는 계층 (인코딩/디코딩, 언어코드형식변환등)

◆**Session**

✓응용 계층 사이의 연결 설정 및 유지, 종료를 수행하는 계층

◆**Transport**

✓통신하는 컴퓨터 간에 자료를 전송하는 계층(자료단위 : 세그먼트)

◆**Network**

✓라우팅 프로토콜을 이용하여 최적의 전송 경로를 선택하고, 이 경로를 통해 자료를 전송하도록 해주는 계층(자료단위 : 패킷(TCP) 또는 데이터그램(UDP))

◆**Data Link**

✓물리적인 전송 링크를 통해 자료를 안전하게 전송하는 계층 (자료단위 : 프레임)

◆**Physical**

✓서로 연결하는 물리적인 링크의 활성화/비활성화, 링크 상태를 유지하기 위한 물리적 링크의 전기적, 기계적, 규약적, 명세를 정의하는 계층

II. General Concepts

1. NAT Terminology
2. Relaying & Connection Reversal

1. NAT(Network Address Translation) Terminology

다양한 NAT의 방식 중 가장 일반적인 형은 outbound NAT이다. Outbound NAT는 P2P 프로토콜들과 충돌을 일으키는데 이는 두 peer들이 서로 다른 두 NAT "뒤에" 존재하는 상황에서 서로 통신을 하려 할 때 , 누가 세션을 시작하려고 하든 상대방 peer의 NAT가 이를 거절하기 때문이다.
NAT traversal(선회)는 양쪽 NAT들에게 P2P 세션들이 "밖으로 향하는" 세션들로 보이도록 만든다.

1. 특징

- 주소변환 : NAT외부주소를 알아내어 IP주소를 패킷 내부에 포함시키기 위함.
- STUN(Session Traversal Utilities for NAT, RFC 5389) [1] - NAT 환경에서 동작하는 어플리케이션이, NAT의 존재유무와, NAT 공인 IP 주소, 그리고 어플리케이션의 UDP 포트(NAT이 현재 접속에 어떤 포트를 할당했는지) 정보를 알아낼 수 있도록 하는데 목적
- NAPT(Network Address Port Translation)를 포함하여 통상적으로 NAT라 함.

2. 장점

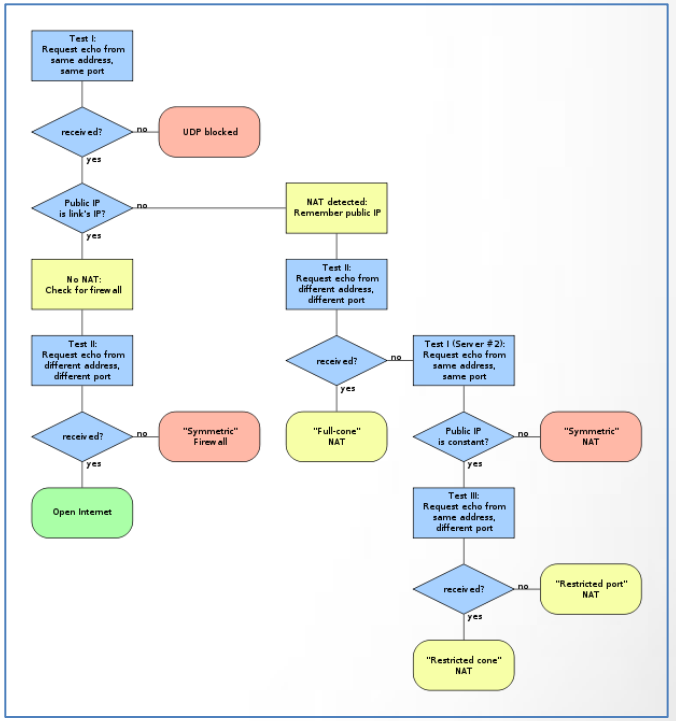
- private 네트워크 상에 있는 호스트들의 보안과 비공개의 장점
- IPv4 주소체계에서 부족현상을 해결하는 해결책(32비트 주소 체계에 기반)

3. 최소의 NAT 분류 (RFC3489)

- Full Cone
- Restricted Cone (Full cone + ip 검증)
- Port Restricted Cone (Full cone + ip/port 검증)
- Symmetric

4. NAT Session

- 출발지 [IP : Port] → Session Endpoint (세션종점)
- 목적지 [IP : Port] → Session Endpoint (세션종점)
- 세션의 4개 중요 요소 (로컬IP, 로컬포트, 리모트 IP, 리모트 포트)
  → 프로토콜을 포함하여 5요소로 구분하기도 함
- 세션의 방향은 세션을 처음 시작한 패킷의 흐름방향
  (TCP – 초기 SYN패킷, UDP – 처음 유저데이터그램)



STUN(Session Traversal Utilities for NAT)

1) NAT내외부의 패킷 교환을 조사하는 검증체계인 STUN(Simple traversal of UDP over NATs)은 RFC3489에서 명세하였지만, 현재 표준은 보완된 STUN(Session Traversal Utilities for NAT)으로 RFC 5389 으로 발표됨.

2. Relaying & Connection Reversal

1. Relaying

NAT를 넘나드는 가장 확실한 (그러나 가장 비효과적인) P2P 통신 방법은 중계(relaying)를 통해서 간단하게 표준 클라이언트/서버 모델 처럼 만드는 것 이다. TURN(Traversal Using Relays around NAT)기법이라 함.

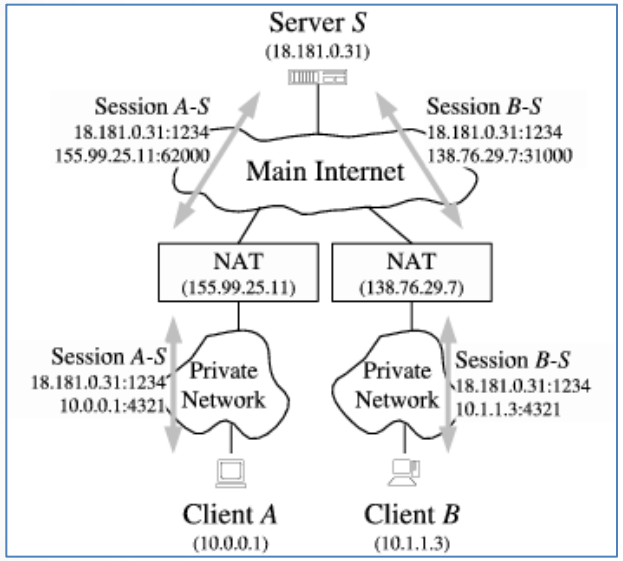


Figure 2: NAT Traversal by Relaying

1) Server S : 중계 서버
2) Client A → NAT (A) → Server S
3) Server S → NAT (B) → Client B

- **장점**
  - 두 client가 서버에 연결할 수 있는 한 Relay는 정상작동
- **단점**
  - 서버의 프로세싱 파워와 네트워크 대역폭 소모
  - 두 client간의 통신 지연발생 가능성

→ 일반적으로 STUN을 시도하고, 이것이 실패할 경우 사용함.

2. Connection Reversal

두 호스트가 모두 잘 알려진 랑데뷰 서버 S에 연결되어 있고 오직 하나의 피어(peer)만이 NAT 뒤에 있을 경우 통신을 가능케 한다.

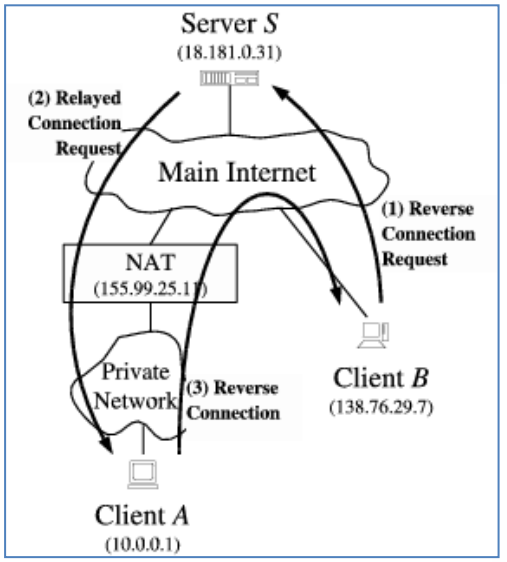


Figure 3: NAT Traversal by Connection Reversal

1) 연결 #1 : Client A → NAT → Client B (정상적인 작동)
2) 연결 #2 : Client B(연결요청) → Server S(연결중계)

→ Client A(연결시작) → NAT → Client B

(Connection Reversal)

랑데뷰서버(중계서버)를 이용해서 P2P 연결의 셋업을 돕는 핵심 아이디어는 홀펀칭 기법의 기본이 된다.

III. UDP Hole Punching

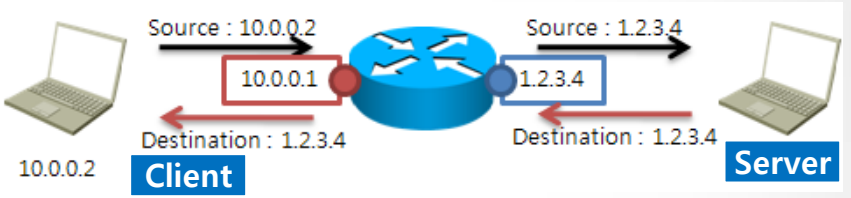
1. The Rendezvous Server & Establishing Peer–to–Peer Sessions
2. Peers Behind a Common NAT
3. Peers Behind Different NATs
4. Peers Behind Multiple Levels of NAT & UDP Idle Timeouts

1. The Rendezvous Server & Establishing Peer-to-Peer Sessions

UDP 홀 펀칭은 두 클라인언트가 모두 NAT 뒤에 있다고 하더라도 잘 알려진 랑데뷰 서버를 통해서 직접적인 P2P UDP 세션을 구성케 하는 기법이다. 또한 온라인 게임을 위한 다양한 고유의 프로토콜들도 역시 UDP 홀 펀칭을 사용한다.

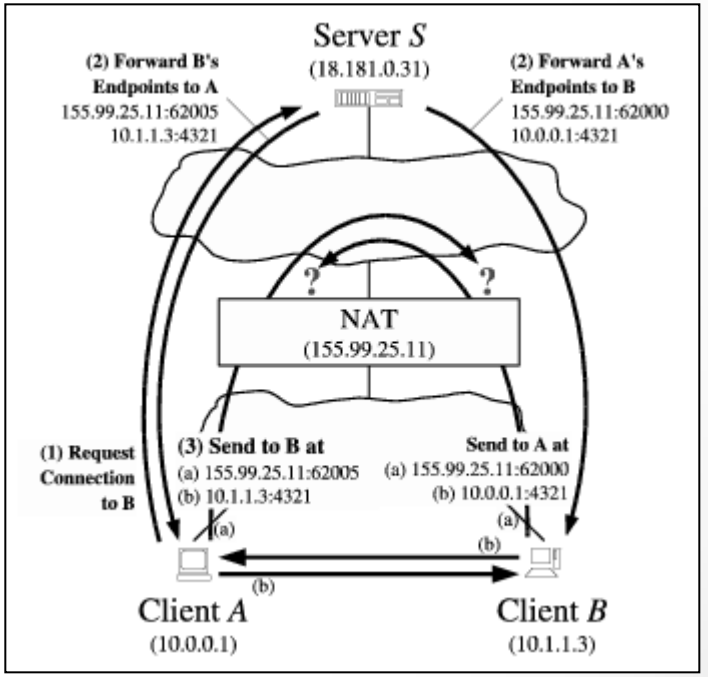
▪ The Rendezvous Server

▪ **랑데부 서버의 클라이언트 종점(클라이언트와 서버에 UDP세션이 있다고 가정)**
- Private 종점 : 클라이언트의 종점 (Private IP, UDP포트)
- Public 종점 : 서버에서 클라이언트를 바라보는 종점(Public IP, UDP포트)



▪ Establishing Peer-to-Peer Sessions

▪ **홀 펀칭 진행과정 (Client A →Client B방향으로 UDP 세션 구성)**
1) A → S (B와 UDP세션을 구성하기 위한 요청)
2) S → A (B의 public & private 종점), S → B (A의 public & private 종점)
3) A는 B의 두 종점에 UDP패킷을 전송
   → 두 종점 중 어떤 종점이든 B로부터 유효한 응답을 받은 종점과 연결.
4) 이와 유사하게, B도 A와 같은 방식으로 동작한다.



▪ 세 가지의 네트워크 토폴로지에 대한 UDP 홀 펀칭

▪ **두 클라이언트가 같은 NAT 밑에 있는 경우**
▪ **두 클라이언트가 다른 NAT 밑에 있는 경우**
▪ **두 클라이언트가 다중 레벨의 NAT들 밑에 있는 경우.**

→ STUN 과 같은 프로토콜로 다중 레벨의 NAT구성에 대한 정보는 신뢰하기 힘들지만, NAT가 대체적으로 RFC를 충족하도록 구현되어 있다면, 홀 펀칭은 대부분 잘 동작한다.

2. Peers Behind a Common NAT

첫 번째, 두 클라이언트가 같은 NAT 밑에 있는 경우
클라이언트 A가 서버 S를 중계서버로 이용, B와 UDP 세션을 구성한다고 가정한다.

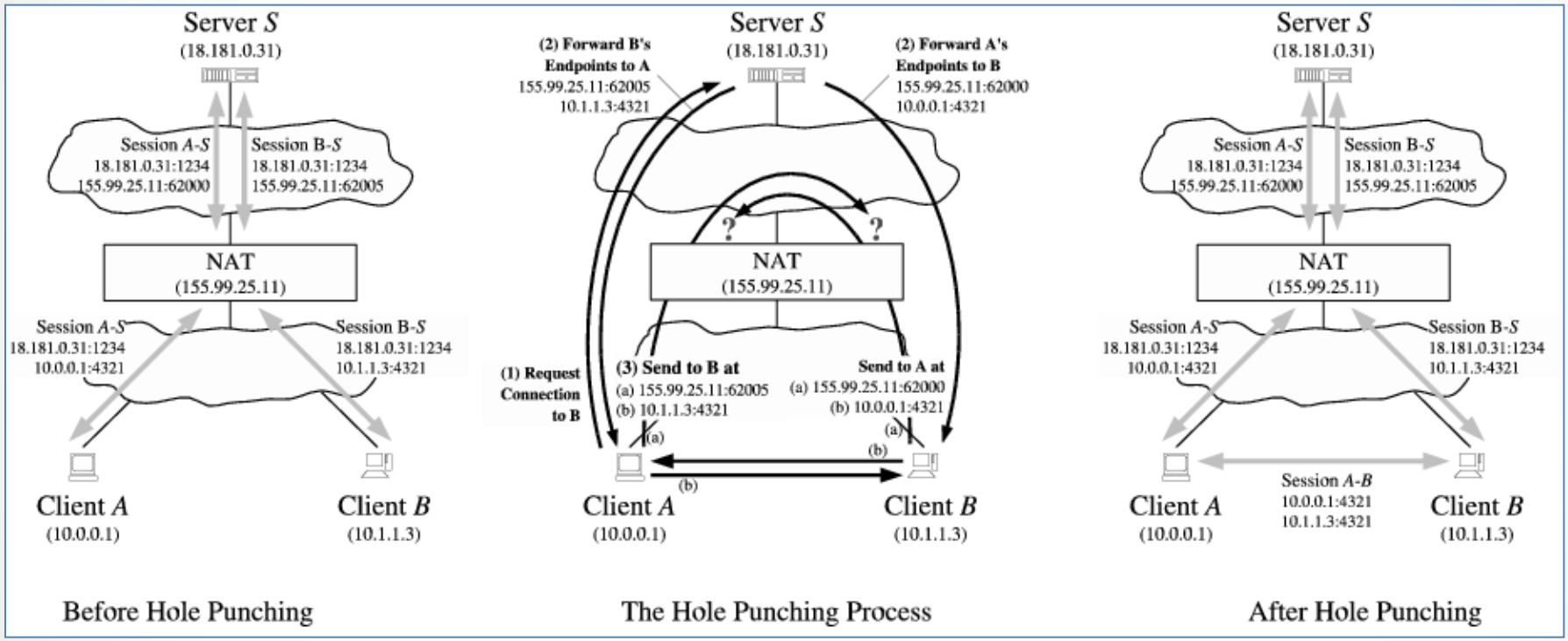


Figure 4: UDP Hole Punching, Peers Behind a Common NAT

1) Client A → Server S (Client B에 연결요청)
2) Server S → Client A (B의 Public, Private 종점정보 응답)
   Server S → Client B (A의 Public, Private 종점정보 전달)
3) **Client A → B의 Private 종점에 UDP 데이터그램 전송시도**
   Client A → B의 Public 종점에 UDP 데이터그램 전송시도(NAT hairpin 지원여부)
   Client B → A의 Private 종점에 UDP 데이터그램 전송시도
   Client B → A의 Public 종점에 UDP 데이터그램 전송시도(NAT hairpin 지원여부)

Private 네트워크를 통한 직접 route 가 NAT를 통하는 간접루트보다 빠를 것이므로, 클라이언트들은 이후의 연결에서는 대부분 private 종점들을 선택하게 됨.

3. Peers Behind Different NATs (most case)

두 번째, 두 클라이언트가 다른 NAT 밑에 있는 경우. 클라이언트 A와 B가 서로 다른 NAT들 뒤에 private 주소를 가진다고 가정.

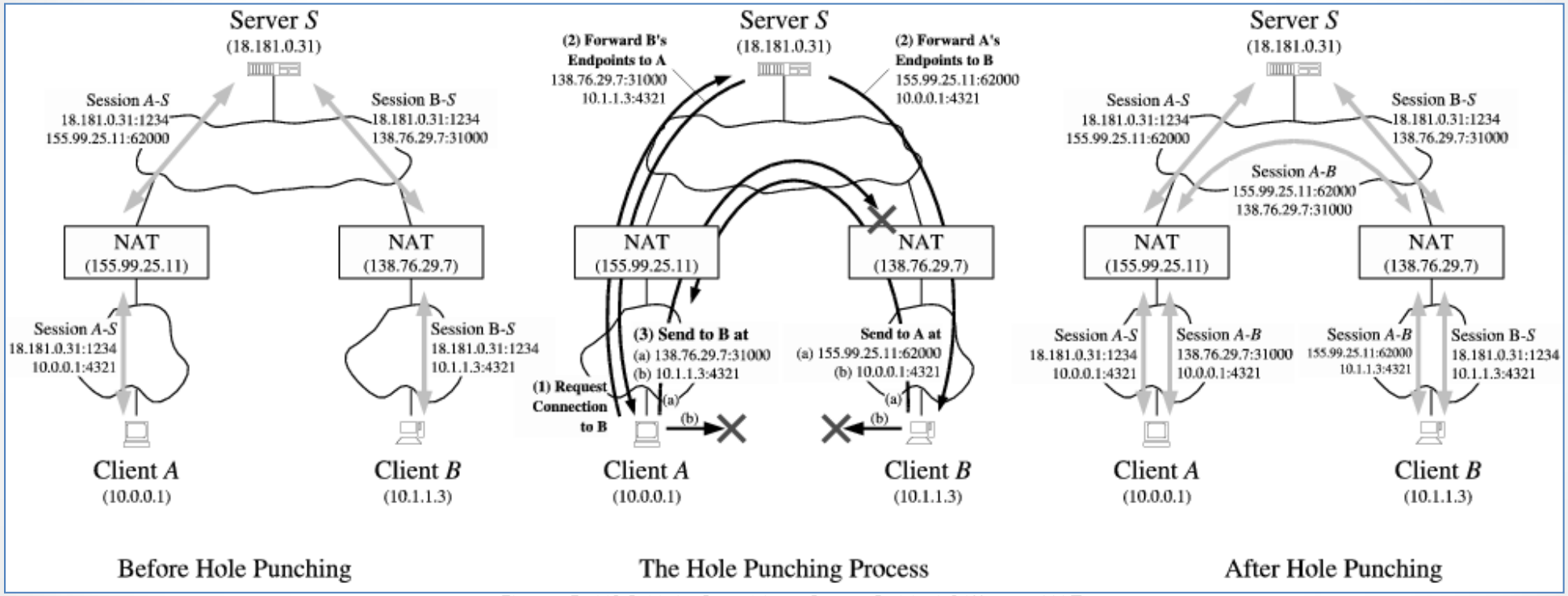


Figure 5: UDP Hole Punching, Peers Behind Different NATs

1) Client A → Server S (Client B에 연결요청)
2) Server S → Client A (B의 Public, Private 종점정보 응답)
   Server S → Client B (A의 Public, Private 종점정보 전달)
3) ~~Client A → B의 Private 종점에 UDP 데이터그램 전송시도~~(실패)
   Client A → B의 Public 종점에 UDP 데이터그램 전송시도
   ~~Client B → A의 Private 종점에 UDP 데이터그램 전송시도~~(실패)
   Client B → A의 Public 종점에 UDP 데이터그램 전송시도

A측) Client A → NAT(A) : B's Public종점(홀펀칭)
(10.0.0.1:4321 – A의 private 주소, 138.76.29.7:31000 – B의 public 주소)
→ 이 세션은 인터넷 상에서 두 종점들(155.99.25.11:62000 – A의 public 주소, 138.76.29.7:31000 – B의 public 주소)로 정의된다

B측) Client B → NAT(B) : A's Public종점(홀펀칭)
(10.1.1.3:4321 – B의 private 주소, 155.99.25.11:62000– A의 public 주소)
→ 이 세션은 인터넷 상에서는 (138.76.29.7:31000 – B의 public 주소, 155.99.25.11:62000 – A의 public 주소)

4. Peers Behind Multiple Levels of NAT & UDP Idle Timeouts

두 클라이언트가 다중 레벨의 NAT들 밑에 있는 경우. NAT-C는 소수의 ip 주소로 수많은 사용자들을 받아들이기 위한, ISP에 의해 설치된 거대한 산업용 NAT이라고 하고, NAT-A와 NAT-B는 NAT-C로부터 할당 받은 ip를 소규모 단위로, 홈 네트워크 구성 등의 이유로 쓰이는 NATs 라 가정.

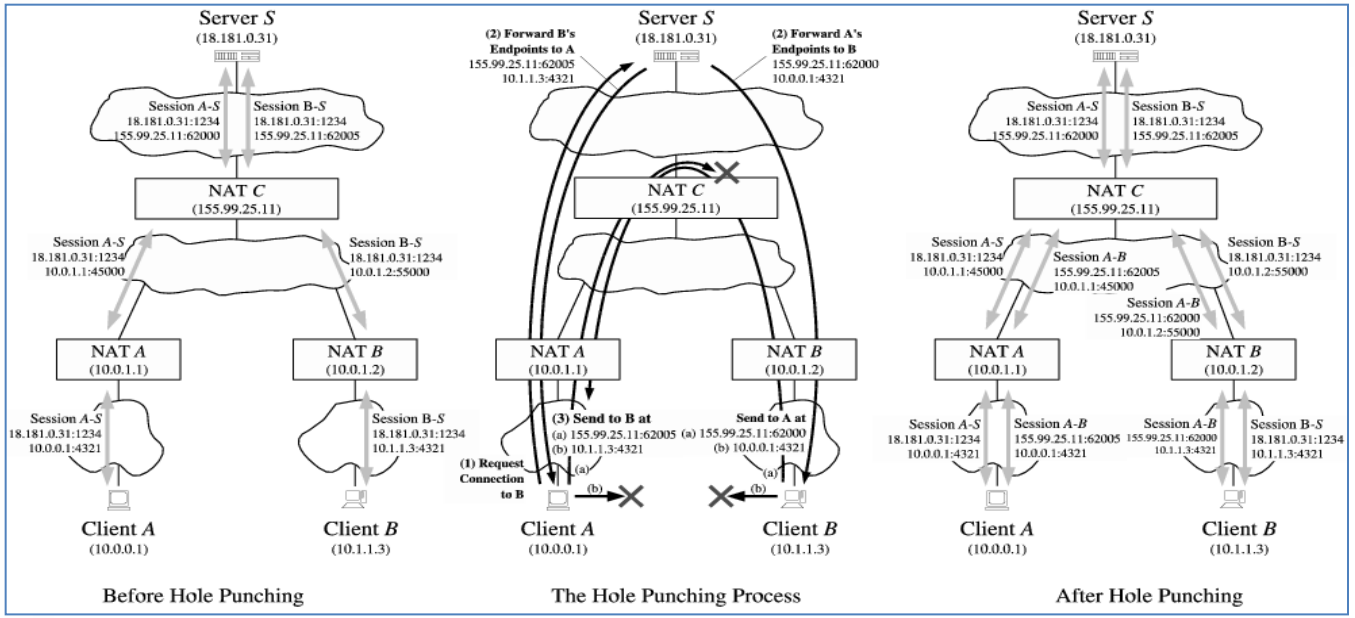


Figure 6: UDP Hole Punching, Peers Behind Multiple Levels of NAT

<NAT-C가 hairpin 지원할 때 가능한 시나리오>

1. A가 B의 public endpoint(155.99.25.11:62005)으로 UDP 패킷을 보낼 때, NAT-A는 먼저 패킷의 source endpoint를 10.0.0.1:4321 -> 10.0.1.1:45000으로 변환한다.
2. 이 패킷은 NAT-C에 도착하고 이 패킷의 destination이 NAT-C의 변환된 public endpoints 중에 하나라는 것을 알아차린다. (hairpin translation)
3. 만약 NAT-C가 잘 동작한다면, 패킷의 source endpoint와 destination을 모두 변환하고 이 패킷을 "돌려서" private 네트워크로 보낸다. (hairpin translation)
4. 이제 패킷의 source endpoint는 155.99.25.11:62000이고 destination은 10.0.1.2:55000 이다.
5. B의 private 네트워크로 이 패킷이 들어가면서 NAT-B가 이 패킷의 도착 주소를 변환, B에 도착하게 된다.

- **UDP Idle TimeOut**
- **NAT 하에서 어플리케이션과 관계없이 UDP 통신은 타임 아웃이 있다. 이 타임 아웃은 일정 기간 서로 통신이 없으면 hole을 닫아버리는 것을 의미한다.**
  - 불행하게도 이 타임 아웃에 대한 기준은 없으며, NAT 별로 다 다르다.
  - 서로 홀펀칭된 것을 유지하려면 열려 있는 세션들끼리의 최소한의 keep-alive는 보장이 되어야 한다.

IV. TCP Hole Punching

1. Sockets and TCP Port Reuse & Opening Peer–to–Peer TCP Streams
2. Behavior Observed by the Application & Simultaneous TCP Open & Sequential Hole Punching

1. Sockets and TCP Port Reuse & Opening Peer-to-Peer TCP Streams

NAT가 TCP 홀 펀칭을 지원 가능한 경우, TCP 홀 펀칭 역시 UDP 홀 펀칭 만큼 빠르고 신뢰할 만 하다. NAT들 사이를 지나는 P2P TCP 연결이 UDP 와 달리 TCP 프로토콜의 상태 기계가 NAT들에게 특정 TCP 세션의 수명을 정확하게 결정할 수 있는 표준 방법을 제공한다.

▪ 제약사항

▪ **표준 버클리 소켓 API : TCP 소켓은 호스트 상에서 TCP 포트와 일대일 관계.**
- 첫번째 소켓에서 로컬 TCP 포트를 사용하면 두 번째 소켓에서 같은 포트 할당 불가

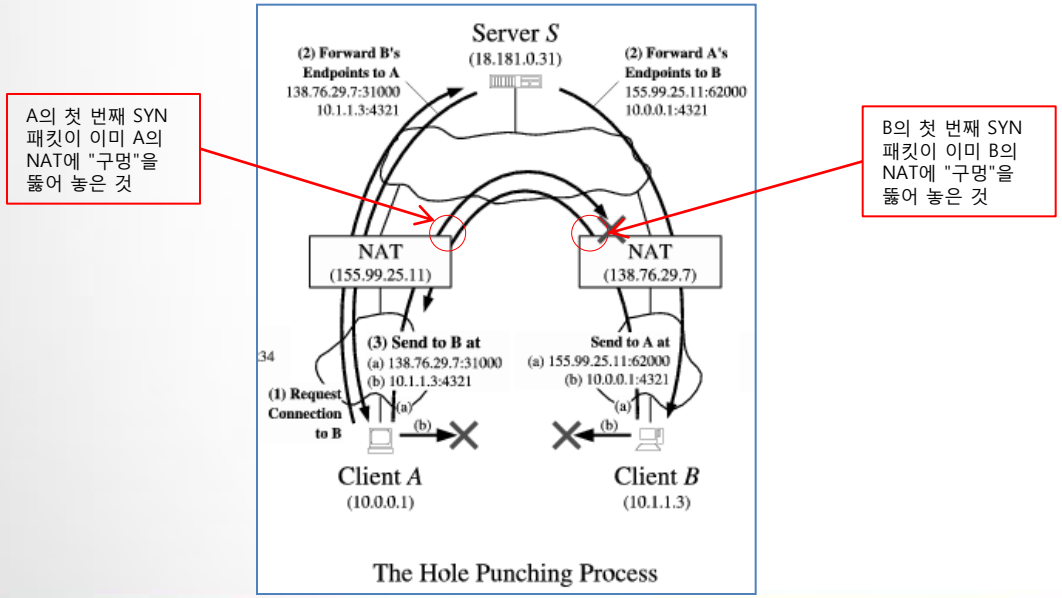
→ TCP 홀 펀칭 사용 불가 : 하나의 포트로 Listen 과 Connection 수행해야 함.

▪ 해결방안

▪ **SO_REUSEADDR(모든 메이저 운영체제 지원), SO_REUSEPORT(BSD 시스템지원)**
- 주소 재사용과 별개로 포트를 재사용을 제어

▪ Opening Peer-to-Peer TCP Streams

▪ **각각의 TCP 클라이언트 어플리케이션은 하나의 로컬 TCP 포트에 바운드된 여러 개의 소켓을 사용해야 한다.**



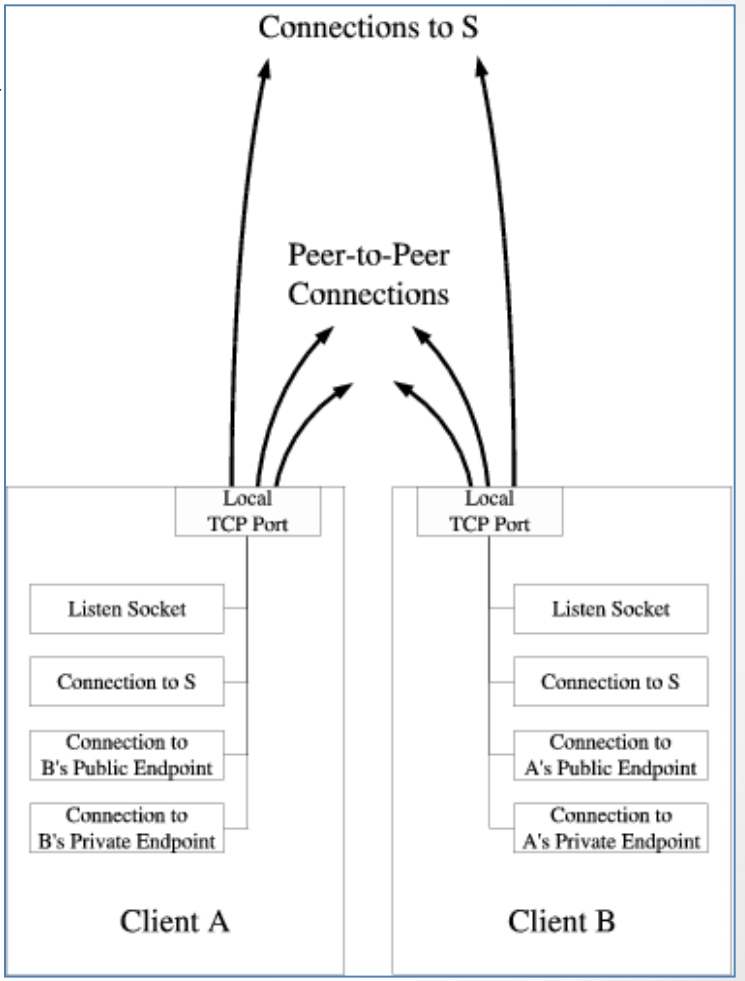
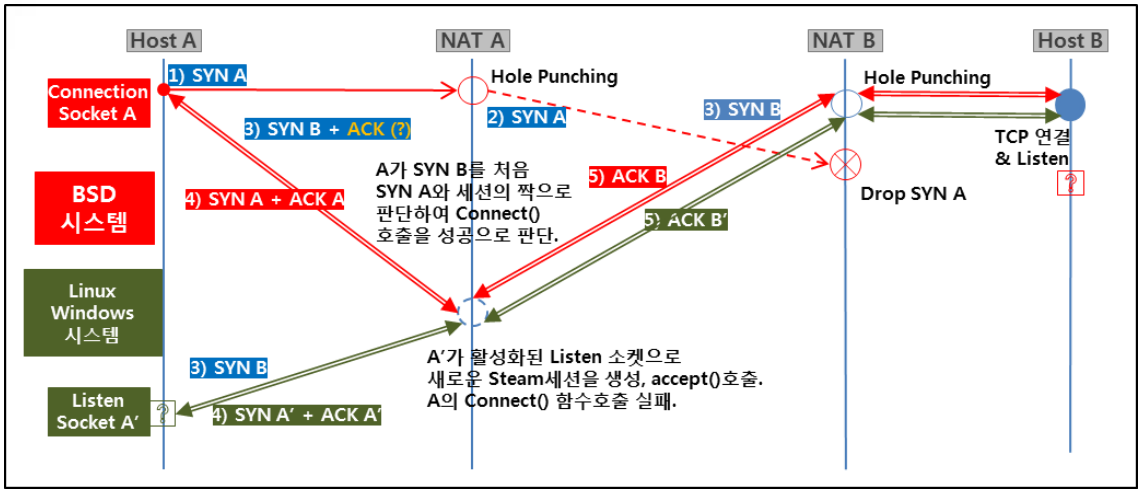


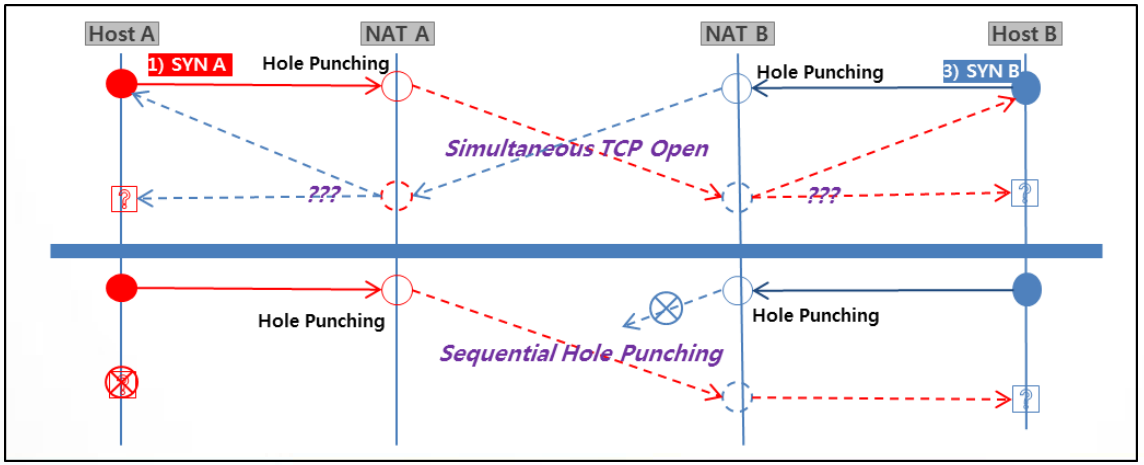
Figure 7: Sockets versus Ports for TCP Hole Punching

2. Behavior Observed by the Application & Simultaneous TCP Open & Sequential Hole Punching

▪ Behavior Observed by the Application



▪ Simultaneous TCP Open & Sequential Hole Punching



V. Properties of P2P–Friendly NATs & Conclusion

1. Consistent Endpoint Translation & Handling Unsolicited TCP Connections & Leaving Payloads Alone & Hairpin Translation
2. Conclusion

1. Properties of P2P – Friendly NATs & Conclusion

▪ Consistent Endpoint Translation

▪ NAT가 private 네트워크 상에 주어진 TCP혹은 UDP의 소스의 종점을 NAT에 의해 조정되는 "하나"의 public 주소로 일관되게 연관을 시킬 경우에만 자동적으로 작동

→ Symmetric NAT 는 일반적으로 P2P 지원이 어려움.

▪ Handling Unsolicited TCP Connections

▪ NAT가 자신의 public 쪽에서 SYN 패킷을 받았는데 이것이 요청되지 않은 내부로 향하는 연결 시도라면, 그냥 조용히 이 SYN packet을 버리는 것이 중요

→ 특정 NAT는 TCP RST (역주 : connection reset) 혹은 ICMP 에러 보고까지 되돌려 보내면서 연결을 적극적으로 거부하는 방식은 TCP 홀 펀칭 과정을 오랜 시간동안 방해.

▪ Leaving Payloads Alone

▪ 현재 소수의 NAT들은 IP주소 처럼 보이는 4byte 값을 패킷으로 부터 "맹목적으로" 스캔하고 이를 패킷 헤더에 있는 IP 주소마냥 해석.

→ 이런 방식은 일반적이지 않고 어플리케이션들이 자신이 보내는 메세지들에 있는 IP 주소들을 알아보지 못하게 만드는 것으로 방어가 가능

▪ Hairpin Translation

▪ 어떤 다중 레벨 NAT의 경우에 TCP 혹은 UDP 홀 펀칭이 작동하기 위해서는 hairpin 변환 지원을 요구

→ IPv4 주소 공간 고갈이 계속됨에 따라 다중 레벨 NAT가 일반적이 되고있다. 따라서 미래의 NAT 구현에서 hairpin 변환 지원은 중요

▪ Conclusion

▪ 홀 펀칭은 NAT가 존재할 때 P2P 연결을 구성하는 다목적의 기술이다.
▪ 연관된 NAT가 요구되는 몇 개의 방식만 따라준다면, 홀 펀칭은 TCP와 UDP 모두의 경우에 일관적이고 튼실하게 작동한다.
▪ 그리고 어떤 특별한 권한이나 특수한 네트워크 망과 상관없이 일반적인 어플리케션들에 의해 구현될 수 있다.
▪ 홀 펀칭은 완전한 투명성을 유지하는데 이는 가장 중요한 NAT의 특징과 장점들 중에 하나다.
▪ 또한 다중 레벨의 NAT에서도 hairpin 변환을 요구하지만 홀 펀칭은 작동한다.

감사합니다.