KIPS Tr. Comp. and Comm. Sys.
Vol.6, No.6 pp.261~270 pISSN: 2287-5891

https://doi.org/10.3745/KTCCS.2017.6.6.261

TCP Session Recovery Technique for High Availability in Smart On-Devices

Seungtae Hong† · Beob-Kyun Kim† · Kwang-Yong Lee†† · Jeong-Si Kim†† · Chae-Deok Lim††

ABSTRACT

With the development of smart on-devices and communication technology, demand for non-stop services is increasing. Therefore, the high availability for continuously providing services in the event of system failure has been spotlighted. Meanwhile, because most internet-based services are provided by using TCP, an efficient TCP session recovery technique for providing non-stop services is required. However, the existing TCP session recovery techniques are inefficient because it has a high recovery cost or does not support failover operation, To solve these problems, in this paper, we propose a TCP session recovery technique for high availability in smart on-devices. For this, we first recover the TCP session without re-establish the TCP session by correcting a sequence number and a acknowledgment number. Second, we synchronize the TCP session recovery data between the master and the server, and then we operate the failover operation when master server fails. Finally, we provide the non-stop service to peer by using the virtual IP number and the transmission of GARP (Gratuitous ARP) packet.

Keywords : TCP Session Recovery, High Availability, Smart On-Device, Failover

스마트 온디바이스의 고가용성을 위한 TCP 세션 복구 기술

홍 승 태† · 김 법 균† · 이 광 용†† · 김 정 시†† · 임 채 덕††

요 약

최근 스마트 온디바이스 및 정보 통신 기술의 발전으로 인하여, 중단 없는 서비스에 대한 요구가 점차 증가하고 있다. 이에 따라 시스템의 장애 발생 시에도 지속적으로 서비스를 제공할 수 있는 고가용성이 주목받고 있다. 한편, 대부분의 인터넷 서비스는 TCP를 기반으로 제공되기 때문에, 스마트 온디바이스의 고가용성을 위해서는 효율적인 TCP 세션 복구 기술이 필수적이다. 그러나 기존 TCP 세션 복구 기술은 높은 세션 복구비용이 요구되거나 페일오버를 지원하지 않는 문제점이 존재한다. 이러한 문제점을 해결하기 위해, 본 논문에서는 스마트 온디바이스의 고가용성을 위한 TCP 세션 복구 기술을 제안한다. 이를 위해 첫째, 순서 번호와 확인 응답 번호의 보정을 통해 TCP 세션의 재연결 과정 없이 TCP 세션을 복원한다. 둘째, 마스터 서버와 백업 서버 간에 TCP 세션 복구 데이터를 동기화하고, 마스터 서버의 장애 발생 시 페일오버를 수행한다. 마지막으로, 가상 IP 주소와 GARP (Gratuitous ARP) 패킷의 전송을 통해 피어에게 무중단 서비스를 제공한다.

키워드 : TCP 세션 복구, 고가용성, 스마트 온디바이스, 페일오버

1. 서 론

최근 스마트 온디바이스 및 정보 통신 기술의 급속한 발전으로 인하여 인터넷 기반의 다양한 서비스들이 급격히 발전하고 있다[1, 2]. 특히, 인터넷 뱅킹, 증권 거래, 스트리밍 서비스 등과 같은 실시간 서비스가 날로 증가함에 따라, 콘텐츠 품질뿐만 아니라 중단 없는 서비스에 대한 요구도 점차 높아지고 있다[3, 4]. 이에 따라 시스템의 장애 발생 시에도 지속적으로 서비스를 제공할 수 있는 고가용성(high availability)이 서비스 품질 향상을 위한 핵심 요소로 부각되고 있다[5-7].

한편, 대부분의 인터넷 서비스는 TCP를 기반으로 제공되며 서버 또는 라우터(router)와 같은 서비스 시스템에 장애가 발생할 경우, 전송계층인 TCP는 프로토콜 특성 상 연결이 단절된다. 결과적으로 서비스 시스템에 장애가 발생할 경우, 서비스 시스템과 서비스 사용자인 피어(peer)와의 통

※ 이 논문은 2017년도 정부(미래창조과학부)의 재원으로 정보통신기술진흥센터의 지원(No.2017-0-00142, 스마트기기를 위한 온디바이스 지능형 정보처리 가속화 SW플랫폼 기술 개발)과 2016년도 정부(미래창조과학부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임(No.R0101-16-0081, 초소형 · 고신뢰(99.999%) OS와 고성능 멀티코어 OS를 동시 실행하는 듀얼 운영체제 원천 기술 개발)
† 정 회 원 : 한국전자통신연구원 선임연구원
†† 비 회 원 : 한국전자통신연구원 책임연구원
Manuscript Received : December 23, 2016
Accepted : February 17, 2017
* Corresponding Author : Seungtae Hong(sthong@etri.re.kr)

신 연결이 중단된다. 이러한 경우 장애가 복구되어 시스템이 재가동 되더라도, TCP 세션의 재연결 과정이 필요하게 되므로 많은 서비스 중단 시간이 야기된다. 따라서 스마트홈 게이트웨이 등 스마트 온디바이스의 고가용성을 위해서는 시스템 장애 발생 시 피어와의 TCP 세션을 유지할 수 있는 TCP 세션 복구 기술이 필수적이다. 이에 따라, 피어에게 서비스를 중단 없이 제공하기 위한 다양한 TCP 세션 복구 기술이 연구되었으며, 대표적인 기술로는 FT-TCP (Fault-Tolerant TCP) [8]와 TCPR (Transparent TCP Recovery) [9]이 존재한다.

그러나 기존 TCP 세션 복구 기술은 다음과 같은 문제점이 존재한다. 먼저, FT-TCP는 TCP 레이어에서 모든 TCP 세션 복구 과정을 담당하기 때문에, TCP 세션 복구 과정에서 높은 복구비용이 요구된다. 아울러 TCP 레이어의 위와 아래에 위치한 두 종류의 래퍼 함수(wrapper function)를 이용하여 TCP 세션 복구를 진행하기 때문에, 기존 커널의 수정이 필요한 한계점이 존재한다. 이에 비해, TCPR은 백업 서버를 통해 서비스를 재개하는 페일오버(failover) [10] 기술을 제공하지 않음에 따라, 시스템 자체적으로 장애 상황이 발생할 경우 TCP 세션 복구가 어려운 한계점이 존재한다.

이러한 문제점을 해결하기 위해, 본 논문에서는 스마트 온디바이스의 고가용성을 위한 TCP 세션 복구 기술을 제안한다. 제안하는 TCP 세션 복구 기술은 마스터와 백업 서버로 구성된 이중화 시스템을 기반으로 구성되며, 마스터 서버에 장애 발생 시 백업 서버를 이용하여 TCP 세션을 복원한다. 제안하는 TCP 세션 복구 기술의 특징은 다음과 같다. 첫째, TCP 패킷의 순서 번호(sequence number)와 확인 응답 번호(acknowledgment number)의 보정을 통해 마스터 서버와 피어와의 TCP 세션을 유지한다. 둘째, 마스터 서버의 TCP 세션 정보를 백업 서버와 동기화하고, 마스터 서버의 장애 발생 시 백업 서버를 기반으로 페일오버를 수행함으로써 피어와 통신을 재개한다. 마지막으로, 피어를 위한 가상 IP 주소와 GARP (Gratuitous ARP) [11] 패킷의 전송을 통해, 피어의 페일오버 인지 과정 없이 TCP 세션을 복원한다.

본 논문의 구성은 다음과 같다. 먼저 2장에서는 기존 TCP 세션 복구 기술에 대해 살펴보고, 3장에서는 스마트 온디바이스의 고가용성을 위한 TCP 세션 복구 기술을 제안한다. 4장에서는 이중화 구조 기반의 채팅 서비스 시스템을 구축하고, 이를 통해 제안하는 기법에 대한 실험 및 성능분석을 수행한다. 마지막으로 5장에서는 결론을 제시한다.

2. 관련 연구

본 장에서는 대표적인 TCP 세션 복구 기술에 대해 살펴본다. 먼저, HydraNet-FT [12], CoRAL [13], HotSwap [14]은 전체 TCP 세션 정보에 대한 복사본을 백업 서버에 생성하고, 마스터 서버의 장애 발생 시 복사본을 이용하여 TCP 세션을 복원한다. 그러나 이러한 TCP 세션 복구 기법은 TCP 세션의 모든 정보를 복사본으로 저장하기 때문에, 높은 복구비용뿐만 아니라 복사본 유지비용이 요구되는 문제점이 존재한다.

이러한 문제점을 해결하기 위해 TCP 세션의 일부 정보를 이용하여 TCP 세션을 복구하기 위한 다양한 기법들이 연구되었다. 대표적인 연구로는 FT-TCP와 TCPR이 존재한다. 첫째, FT-TCP는 주 서버와 백업 서버로 구성된 이중화 시스템을 기반으로 시스템 장애 상황 발생 시 백업 서버를 이용하여 TCP 세션을 복구한다. Fig. 1은 FT-TCP의 시스템 구조를 나타낸다.

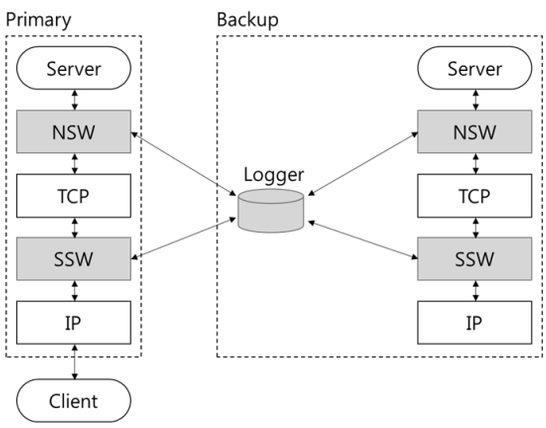

Fig. 1. System Architecture of FT-TCP

Fig. 1에서와 같이, FT-TCP는 TCP 레이어의 위와 아래를 각각 NSW (north side wrap)와 SSW (south side wrap)로 래핑(wrapping)한다. 이를 통해, FT-TCP는 TCP 레이어를 통과하는 모든 패킷에 대해 필터링을 수행하고, TCP 세션 복원을 위한 복구 데이터를 생성하여 백업 서버의 저장소(logger)에 저장한다. 만약, 주 서버에 장애가 발생한 경우, 백업 서버는 저장소에 저장된 세션 정보를 이용하여 클라이언트와의 TCP 세션을 재연결한다. 이와 같이, FT-TCP는 TCP 레이어에서 세션 복구를 모두 담당하기 때문에, 클라이언트에 대한 수정 없이 TCP 세션 복구 기능을 활용할 수 있는 장점이 존재한다. 그러나 FT-TCP는 TCP 레이어에 송/수신되는 패킷들을 필터링 하기 위하여 NSW와 SSW와 같은 래퍼 함수를 이용하였기 때문에, 커널 패치를 통해 기존 커널을 수정해야만 한다. 이에 따라, FT-TCP는 특정 커널에 의존전인 한계점이 존재한다. 또한 FT-TCP는 백업의 저장소에 저장된 데이터를 이용하여 피어와 세션을 재연결하기 때문에, 높은 세션 복구비용이 요구되는 문제점이 존재한다.

둘째, TCPR은 서버와 피어 사이에 위치한 네트워크 미들웨어를 이용하여 재연결 과정 없이 TCP 세션을 복구한다. Fig. 2는 TCPR의 시스템 구조를 나타낸다. TCPR의 네트워크 미들웨어는 서버와 피어 응용 사이에 송/수신되는

TCP 패킷의 헤더 정보를 이용하여 두 응용과 각각 지역적인 TCP 세션을 생성한다. 만약, 서버 응용이 중단된 경우, 네트워크 미들웨어는 서버 응용과의 TCP 세션을 복원하며, 피어 응용과의 TCP 세션을 지속적으로 연결을 유지한다. 이를 통해, Application-Driven TCP Recovery 기술은 세션의 재연결 과정 없이 TCP 세션을 복구하는 것이 가능하다. 그러나 TCPR은 시스템의 페일오버를 위한 TCP 세션 복구 데이터 동기화 및 세션 복구 기술을 제공하지 않음에 따라, 시스템 자체적으로 장애 상황이 발생할 경우 TCP 세션 복구가 어려운 한계점이 존재한다. 또한 TCPR은 응용 레벨에서 세션 복구를 수행하기 위해 기존 소켓 API와 함께 별도의 사용자 API를 함께 사용해야 하는 제약사항이 존재한다.

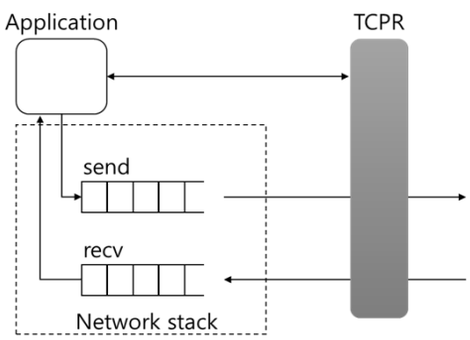

Fig. 2. System Architecture of TCPR

3. 스마트 온디바이스의 고가용성을 위한 TCP 세션 복구 기술

3.1 전체 구조

기존 TCP 세션 복구 기술의 문제점을 해결하기 위해, 본 논문에서는 이중화 구조에 기반을 둔 TCP 세션 복구 기술을 제안한다. 제안하는 TCP 세션 복구 기술은 마스터 및 백업 서버의 장애 감지 및 모드 전환을 위한 페일오버 관리자(Failover Manager) [5], 세션 복구 데이터 동기화 및 복구를 위한 세션 관리자(Session Manager), 그리고 응용 관리자(Application Manager) 구성된다. 한편, 제안하는 TCP 세션 복구 기술은 FT-TCP와 같이 이중화 구조에 기반을 두고 있으나, TCPR과 같이 응용 레벨에서 TCP 세션 복구를 지원하기 때문에 별도의 커널 패치 과정 없이 사용 가능하다. 또한 제안하는 TCP 세션 복구 기술은 응용 관리자를 통하여 기존 소켓 API와 동일한 입력 파라미터를 가지며, 기존 소켓 API에 일정한 접두어를 추가한 형태의 사용자 API를 제공함으로써 사용성을 극대화한다. 제안하는 TCP 세션 복구 기술의 시스템 구조는 Fig. 3과 같으며, 그림에서 점선은 페일오버 이후의 수행과정을 나타낸다.

첫째, 페일오버 관리자는 주기적인 하트비트(heartbeat)를 상대 서버에 전송함으로써 마스터 또는 백업 서버의 장애 상황을 감지한다. 만약, 마스터 서버에 장애가 발생할 경우 백업 서버로 자동적으로 페일오버를 수행한다. 또한 피어 BGP 응용에게 서버 접근을 위한 가상 IP(virtual IP) 주소를 제공하며, 가상 IP 주소는 현재 마스터로 활성화된 서버의 이더넷 인터페이스에 자동적으로 할당된다.

둘째, 세션 관리자는 마스터 서버와 피어 BGP 응용 간의 TCP 세션을 유지하고, 두 서버 간에 세션 복구 데이터를 동기화한다. 아울러 세션 관리자는 페일오버 관리자로부터 현재 모드 정보(마스터/백업)를 확인하며, 만약 현재 서버가 마스터로 활성화 된 경우 피어와 지역적인 TCP 세션을 생성한다. 페일오버 수행 시, 백업 서버는 저장된 세션 복구 데이터를 이용하여 순서 번호와 확인 응답 번호의 보정을 통해 세션 재연결 과정 없이 TCP를 복원한다.

마지막으로, 응용 관리자는 제안하는 TCP 세션 복원 기술을 사용하기 위한 사용자 API를 제공한다. 사용자 API는

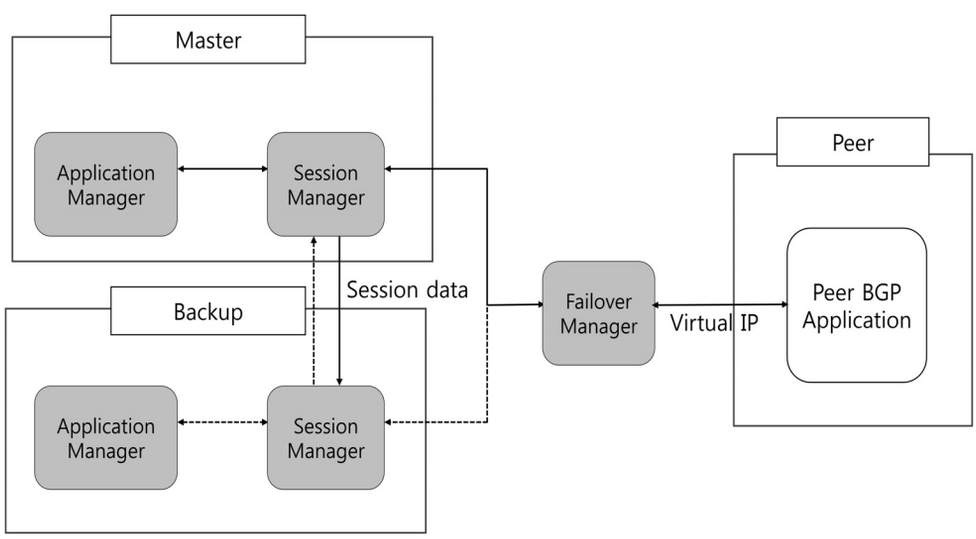

Fig. 3. System Architecture of Proposed TCP Session Recovery Technique

기존 TCP 응용의 수정을 최소화할 수 있도록 기존 소켓 API에 일정한 접두어를 추가한 형태로 사용자에게 제공된다. 사용자 API는 이중화 시스템에서 운영되는 TCP 응용에서만 호출되어 사용되며, 피어 BGP 응용은 기존의 소켓 API를 사용한다. 또한 응용관리자는 페일오버 이후 새롭게 복원된 TCP 세션을 기반으로 새롭게 활성화된 마스터 서버의 응용을 자동으로 재실행함으로써, 페일오버 이후에도 피어 BGP 응용에게 서비스가 지속될 수 있도록 한다.

한편, TCP 응용은 마스터와 백업 서버에서 각각 운용되며, 마스터 서버의 응용만이 피어 BGP 응용과 접속을 유지한다. 피어 BGP 응용은 각 서버의 물리적 이더넷 디바이스에 할당된 IP가 아닌 페일오버 관리자의 가상 IP 주소를 통해 이중화 시스템의 마스터 서버에 접근한다. 가상 IP 주소는 현재 활성화된 마스터 프로세서의 이더넷 인터페이스에 자동적으로 할당되며, 페일오버 후 세션 관리자는 피어 BGP 응용이 이를 인식할 수 있도록 GARP 패킷을 전송한다. GARP 패킷을 수신한 피어 BGP 응용은 자신의 ARP 테이블을 갱신하며, 피어 BGP 응용은 갱신된 MAC 주소를 통해 동일한 가상 IP 주소를 통해서 새롭게 활성화된 마스터 프로세서에 접근할 수 있다. 이를 통해 피어 BGP응용은 페일오버 수행 후에도 페일오버 과정을 인지하지 않은 상태로 서비스를 제공받을 수 있다.

3.2 TCP 세션 복구 데이터

TCP 세션 복구를 위해 세션 관리자는 마스터 서버와 피어 BGP 응용 간에 전송되는 모든 TCP 패킷들을 리눅스 커널에서 제공하는 netfilter와 iptables를 이용하여 사용자 영역의 큐(queue)에 저장한다. 이후, 큐에 저장된 패킷에서 TCP 헤더 정보를 추출하여 TCP 세션 복구 데이터를 생성한다. TCP 세션 복구 데이터는 마스터 서버의 로컬 파일 시스템에 저장되며, 세션 관리자를 통해 마스터에서 백업 서버로 전송된다. 한편, TCP 세션이 정상적으로 종료될 경우, 백업 서버의 세션 관리자는 저장된 TCP 세션 복구 데이터 파일을 삭제한다.

TCP 세션 복구 데이터는 tcp_recovery_full 구조체와 tcp_recovery_sync 구조체로 구성되어 있다. tcp_recovery_full 구조체는 오픈 소스로 공개된 TCPR [15]의 복구 데이터 구조를 기반으로 하며, 이중화 구조에서 데이터 동기화가 가능하도록 확장하였다. 또한 tcp_recovery_full 구조체는 tcp_recovery_sync 구조체를 포함하며 총 57 바이트로 구성된다. tcp_recovery_full 구조체는 3번 악수 기법(3-way handshake)을 통해 TCP 세션이 처음으로 생성되거나 세션 종료 시에만 마스터에서 백업 서버로 전송된다. 3번 악수 기법 이후 데이터 전송이 수행될 때에는 tcp_recovery_sync 구조체만을 전송함으로써 데이터 동기화 비용을 최소화한다.

tcp_recovery_sync 구조체는 총 32바이트로 구성되며, 서버 응용의 포트 번호(internal_port), 세션 관리자의 포트 번호(external_port), 서버 응용의 확인 응답 번호(ack), 마지막으로 정상 전송된 확인 응답 번호(safe), 서버 응용의 수신 완료 여부(done_reading), 서버 응용의 송신 완료 여부(done_writing), 피어 응용의 주소(peer_addr), 피어 정보 구조체(struct peer), 그리고 최종 저장 패킷의 출발지(from_peer) 로 구성된다. tcp_recovery_sync 구조체는 세션 생성 후 데이터 전송 시에만 전송되며, 백업 서버의 세션 관리자는 수신된 정보를 이용하여 백업 서버에 저장된 TCP 세션 복구 데이터를 갱신한다.

3.3 TCP 세션 복구

TCP 응용은 3번 악수 기법을 통해 TCP 세션을 생성한 이후, 자신이 전송한 순서 번호와 회신된 확인 응답 번호의 비교를 통해 정상 수신 여부를 결정한다. 예를 들어 피어 BGP 응용에서 400으로 설정된 순서 번호를 보낼 경우, 서버 응용은 수신된 패킷의 수신 번호에 1을 더한 값을 확인 응답 번호로 설정하여 피어 BGP 응용에게 회신한다.

Fig. 4는 피어와 서버 간에 3번 악수 기법을 통해 TCP 세션이 생성되는 과정을 나타낸다. 피어와 서버 간 3번 악수 기법의 수행 과정은 다음과 같다. i) 피어는 서버에게 400으로 설정된 순서 번호를 가지는 패킷을 전송한다. ii) 서버는 순서 번호에 1을 더하여 401의 확인 응답 번호를 회신한다. iii) 피어는 회신된 패킷의 확인 응답 번호와 자신이 전송한 순서 번호를 확인하여 정확한 응답 번호가 전송되었는지 확인한다. iv) 피어는 서버로부터 수신한 패킷의 순서 번호에 1을 더하여 확인 응답 번호를 생성한 후, 서버에게 전송한다. 이때, 순서 번호는 수신한 패킷의 확인 응답 번호로 설정된다.

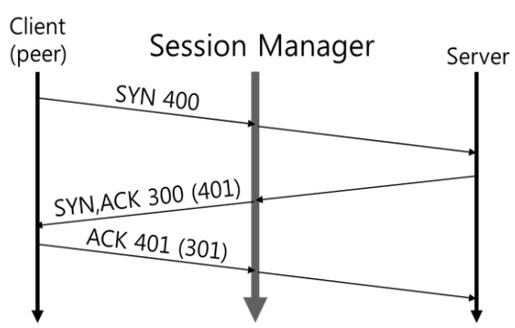

Fig. 4. Establish a TCP Session Through 3-way Handshake

이러한 TCP 응용의 특징을 이용하여 제안하는 기법은 순서 번호 및 확인 응답 번호의 보정을 통해 피어 BGP 응용과 서버 응용과의 TCP 세션을 복구한다. 또한 TCP 복구 시간을 최소화하기 위하여, 세션의 재연결 과정 없이 TCP 세션을 복구할 수 있도록 TCPR의 지역적 세션 생성 기법을 적용한다. 즉, 제안하는 TCP 세션 복구 기술은 서버 그리고

피어 BGP 응용과 각각 TCP 세션을 독립적으로 생성하고, 피어 BGP 응용과의 세션이 비정상적으로 종료되더라도 서버 응용과는 세션을 유지함으로써 두 응용의 TCP 연결을 지속적으로 유지한다.

TCP 세션 복구 데이터를 이용한 TCP 세션 복구 방법은 다음과 같다. 먼저, 피어로부터 전송된 패킷의 순서 번호와 세션 복구 데이터에 저장된 확인 응답 번호를 비교하여 TCP 세션의 장애 상황을 인식한다. 이후, 장애 상황이 인식될 경우, 피어의 확인 응답 번호에서 1을 뺀 값을 응답 패킷의 순서 번호로 설정한다. 이때 응답 패킷의 확인 응답 번호는 현재 패킷의 순서 번호를 이용하여 설정하며, TCP 세션 복구 데이터의 정보를 이용하여 보정된 응답 패킷의 TCP 헤더를 재구성한다. 마지막으로 세션 관리자는 재구성된 응답 패킷을 피어에게 전송하고, 피어는 보정된 확인 응답 번호를 받음으로써 서버와 끊임없이 세션을 유지한다.

Fig. 5는 피어로부터 잘못된 순서 번호가 전송될 경우, 저장된 TCP 세션 복구 데이터를 이용한 TCP 세션 복구 과정을 나타낸다. 이때, 이전 TCP 세션의 상태는 Fig. 4의 상태로 가정한다. i) 세션 관리자는 TCP 세션 복구 데이터에 저장된 피어의 확인 응답 번호에서 1을 뺀 값인 300을 피어에게 회신할 패킷의 순서 번호로 설정한다. 이때, 피어와 세션을 끊임없이 유지하기 위해, 확인 응답 패킷은 수신한 패킷의 순서 번호에 1을 더한 801로 설정한다. ii) 피어는 보정된 패킷을 수신함으로써 세션을 유지한다. 아울러 수신한 패킷에 대한 응답 패킷을 전송한다. iii) 세션 관리자는 피어로부터 전송된 응답 패킷의 확인 응답 번호를 저장된 세션 복구 데이터를 이용하여 보정한다. iv) 세션 관리자는 서버에게 보정된 패킷을 전송하며, 서버는 이를 통해 세션을 유지한다.

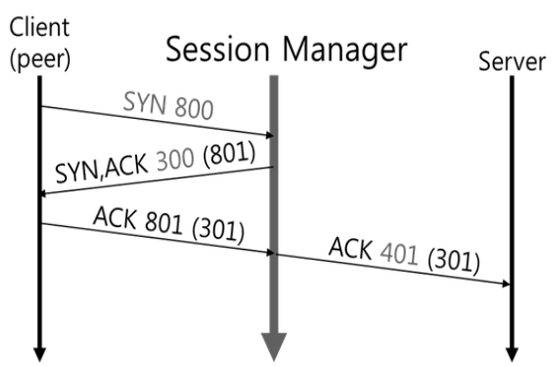

Fig. 5. Procedure of TCP Session Recovery in the Proposed Technique

3.4 TCP 세션 동기화 및 페일오버

시스템의 장애 상황 발생 시에도 피어 BGP 응용에게 끊임없이 서비스를 제공하기 위해, 제안하는 TCP 세션 복구 기술은 TCP 세션 동기화 및 페일오버 기능을 제공한다. 이를 위해, 마스터와 백업 서버에서 운영되는 페일오버 관리자는 지속적으로 상대 서버에게 하트비트를 전송한다. 이때, 하트비트가 일정 시간 내에 수신되지 않는다면 해당 서버에 장애가 발생한 것으로 간주한다. 만약, 백업 서버의 페일오버 관리자가 마스터 서버로부터 하트비트를 수신하지 못한 경우, 페일오버 관리자는 현재 서버의 모드 정보를 백업에서 마스터로 변경한다. 또한 페일오버 관리자는 피어 BGP 응용의 접속을 위한 가상 IP 주소를 생성하여, 현재 마스터로 활성화된 서버의 이더넷 디바이스에 할당한다. 이는 만약 피어 BGP 응용이 각 서버의 물리적 IP 주소를 이용하여 서버에 접근할 경우, 페일오버 이후 변경된 물리적 IP 주소로 사용자가 재접속해야하기 때문이다. 따라서 제안하는 TCP 세션 복구 기술은 가상 IP를 통해 피어 BGP 응용이 현재 마스터로 활성화된 서버에 접속할 수 있게 하며, 페일오버 후에 GARP 패킷을 세션 관리자를 통해 전송함으로써 사용자가 새롭게 활성화된 마스터 서버에 접속할 수 있게 한다.

한편, 제안하는 TCP 세션 복구 기술은 별도의 커널 패치 과정 없이 사용 가능하며, 이를 위해 기존 소켓 API와 내부 함수를 통합한 사용자 API를 제공한다. 제안 기술의 사용자 API는 기존 소켓 API와 동일한 입력 파라미터를 가지며, 기존 소켓 API의 함수명에 특정 접두어를 추가한 형태로써 사용자의 사용성을 극대화한다. 또한 페일오버 이후 서버 응용을 자동으로 재실행함으로써, 복원된 TCP 세션을 기반으로 피어 BGP 응용에게 서비스를 지속적으로 제공한다.

Fig. 6은 제안하는 TCP 세션 복구 기술의 페일오버 알고리즘을 나타낸다. 페일오버 알고리즘은 동기화 페이즈와 모드 변경 페이즈, 그리고 응용 재시작 페이즈로 구성되며, 각 페이즈는 스레드(thread)로 구성되어 병렬적으로 수행된다.

먼저 동기화 페이즈의 수행 내용은 다음과 같다. 응용 관리자는 페일오버 관리자를 통해 현재 서버의 모드 정보를 확인한다(line 1). 모드 정보를 이용하여 TCP 세션 복구 데이터의 동기화를 위한 동기화 서버를 생성한다(line 2). 현재 서버의 모드 정보가 마스터일 경우, netfilter를 초기화하고 패킷의 ip와 tcp 헤더 정보를 이용하여 tcp 세션의 정보를 얻어온다(line 3-6). 피어 BGP 응용으로부터 전송된 패킷일 경우 패킷의 헤더 정보 추출을 통해 TCP 세션 복구 데이터를 생성하고, 패킷을 그대로 서버 응용에게 전달한다(line 7-9). 서버 응용으로부터 전송된 패킷일 경우, 패킷의 헤더 정보 추출을 통해 TCP 세션 복구 데이터를 생성하고 기존 TCP 세션 복구 데이터에 저장된 세션 정보와 비교를 수행한다(line 10-11). 세션 정보 비교 결과에 따라서 패킷 전달(deliver), 패킷 드랍(drop), 패킷 초기화(reset), 패킷 복구(recover)의 동작을 수행한다(line 12-15). 만약 3번 악수 기법을 통해 처음으로 생성된 세션일 경우, tcp_recovery_full

```
Failover algorithm
Synchronize phase (session manager)
Input  ip : struct ip, tcp : struct tcphdr tcp
 1  curr_mode = get_mode()
 2  setup_sync_server(curr_mode)
 3  while synchronize_phase is running
 4      if curr_mode is master
 5          setup_netfilter()
 6          connection = get_connection(ip, tcp)
 7          if from_peer(ip)
 8              tcp_filter_peer(connection, tcp)
 9              deliver(ip, tcp)
10          else
11              switch tcp_filter(connection, tcp)
12                  case DELIVER:  deliver(ip, tcp)
13                  case DROP: drop(ip, tcp)
14                  case RESET: reset(connection)
15                  case RECOVER: recover(connection)
16              if chk_first_connection(connection)
17                  send_full_state(connection)
18              else if chk_tcp_connection(connection)
19                  send_state(connection)
20      else if curr_mode is backup
21          has_fin = recv_state()
22          if has_fin is true(1)
23              connection = find_recv_connection()
24              teardown_connection(connection)
25  close_sync_server()
26  teardown_netfilter()

Mode change phase (session manager)
Input  old_mode : mode information before failover operation
 1  while mode_change_phase is running
 2      new_mode = get_mode()
 3      if new_mode is master and old_mode is backup
 4          send_garp()
 5          restart_sync_server(new_mode)
 6      else if new_mode is backup and old_mode is master
 7          restart_sync_server(new_mode)

Application restart phase (application manager)
Input  config : configuration for service
 1  curr_mode = get_mode()
 2  conf = get_configuration(config)
 3  while application_restart_phase is running
 4      if curr_mode is master
 5          wait_session_manager()
 6          service_setup(conf)
 7          service_handle_event()
 8      else if curr_mode is backup
 9          stop_service()
10          do_waiting()
11  teardown_service()
```

Fig. 6. Failover Algorithm of Proposed TCP Session Recovery Technique

구조체를 백업 서버에게 전송한다(line 16-17). 정상적으로 데이터 전송이 이루어진 경우, TCP 세션 복원 데이터의 동기화를 위해 tcp_ recovery_sync 구조체를 백업 서버에게 전송한다(line 18-19). 현재 서버의 모드 정보가 백업일 경우, 동기화 서버를 통해 TCP 세션 복구 데이터를 수신하며, 로컬 파일 시스템에 이를 저장한다(line 20-21). 수신한 TCP 세션 복구 데이터를 통해 마스터 서버의 TCP 세션 종료 여부를 검사한다(line 22). 마스터 서버의 TCP 세션이 종료된 경우, 저장된 TCP 세션 정보 데이터를 통해 TCP 세션의 정보를 획득하여 해당 TCP 세션을 삭제한다(line 23-24). 동기화 페이즈가 종료될 경우, 동기화 서버를 종료하고 netfiler를 해제한다(line 25-26).

모드 변경 페이지는 현재 서버의 모드 정보 갱신을 확인하며(line 2), 마스터 서버로 새롭게 활성화된 경우 피어 BGP 응용에게 GARP 패킷을 전송하고 동기화 서버를 재시작한다(line 3-5). 만약 기존 마스터 서버가 백업 서버로 갱신될 경우, 동기화 서버만을 재시작한다(line 6-7).

응용 재시작 페이지는 서버 응용에서 응용 관리자를 통해 수행되며, 페일오버 이후 응용을 자동으로 재실행한다. 이를 위해, 먼저 현재 서버의 모드 정보를 확인한 후 응용의 환경설정을 불러온다(line 1-2). 현재 서버가 마스터 서버인 경우, 세션 관리자가 실행될 때까지 대기한다(line 5). 세션 관리자가 정상적으로 실행된 경우, 응용 서비스를 시작한다(line 6-7). 만약 백업 서버일 경우, 실행 중인 기존의 응용 서비스가 존재할 경우 해당 서비스를 중지한다 (line 8-9). 페일오버 이후 신속히 서비스를 재개하기 위해 대기한다(line 10). 응용 재시작 페이즈가 종료될 경우, 서비스를 종료하고 모든 기능들을 해제한다(line 11).

4. 실험 및 성능 분석

본 장에서는 제안하는 TCP 세션 복구 기술을 검증하기 위한 실험 및 성능분석을 수행한다. 실험을 위해 Nvidia사의 TK1 보드 3대를 이용하여 Table 1과 같은 환경에서 이중화 구조 기반의 채팅 서비스 시스템 및 클라이언트를 구축하였다.

Table 1. Experimental Environment

H/W specification	Nvidia TK1
Kernel version	4.7.5
gcc version	4.8.4
Network bandwidth	1000 Mbps

실험 시나리오는 다음과 같다. 먼저, TK1 보드 2대를 이용하여 마스터와 백업 서버로 구성된 이중화 시스템을 구성한다. 마스터와 백업 서버에는 각각 페일오버 관리자, 세션 관

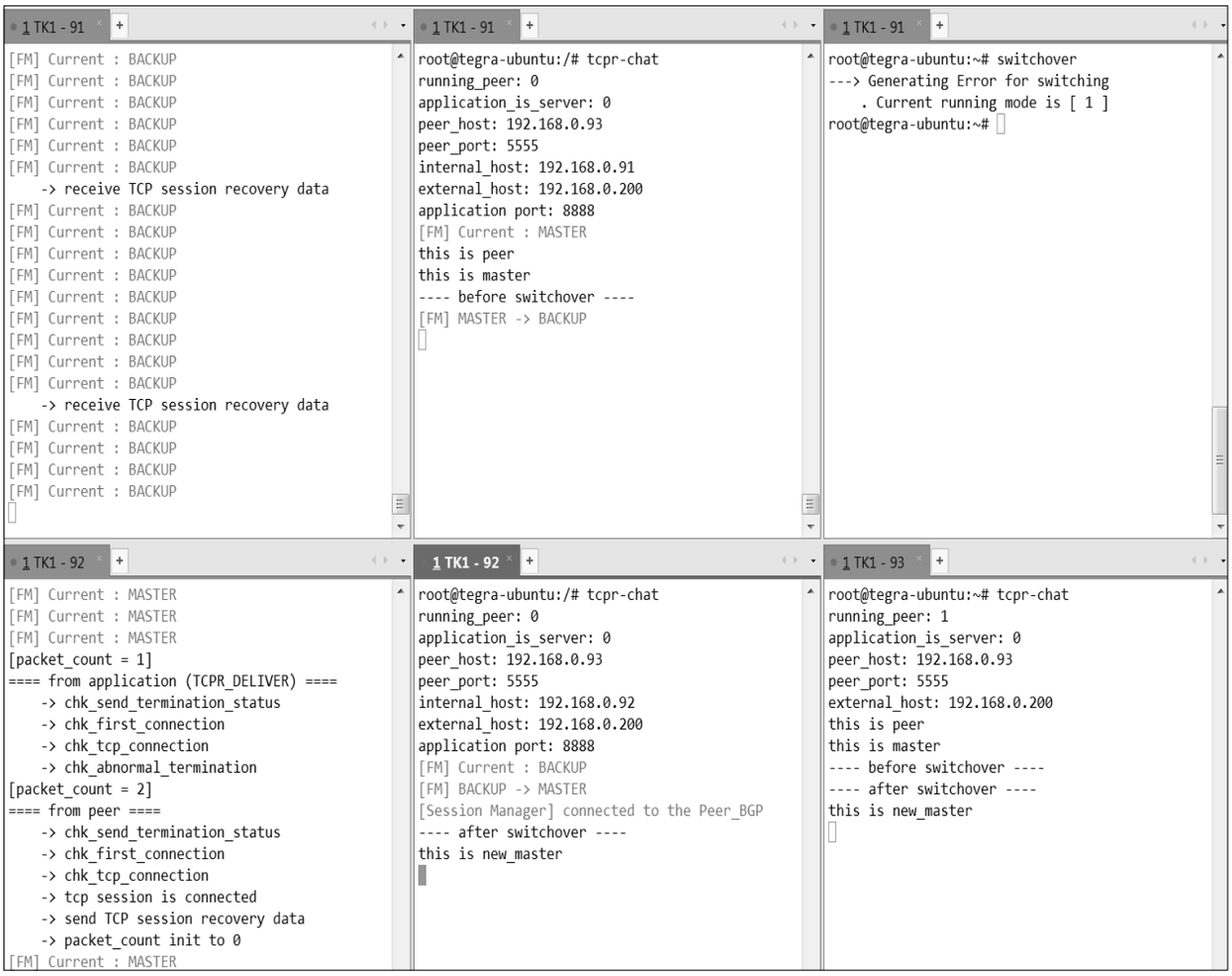

Fig. 7. Experimental Result of TCP Session Recovery by Using the Proposed Technique

리자가 실행되며, 응용 관리자를 포함한 채팅 서비스 응용이 운영된다. 이중화 시스템은 클라이언트에게 실시간 채팅 서비스를 지원하며, 이중화 시스템과 클라이언트는 1000 Mbps의 네트워크 환경을 통해 연결된다. 페일오버를 통한 서비스 재개 여부를 확인하기 위해, 마스터 서버에 스위치오버(switchover) 명령을 수행함으로써 인위적으로 장애를 발생한다. 스위치오버 이후, 클라이언트에서 재접속 없이 새롭게 활성화된 마스터 서버와 실시간 채팅이 가능한지를 확인한다.

Fig. 7은 제안하는 기법을 이용한 TCP 세션 복구 실험 결과를 나타낸다. 클라이언트(TK1_93)는 페일오버 관리자에서 제공하는 가상 IP 번호(192.168.0.200)를 이용하여 마스터 서버(TK_91)에 접속하였으며, 마스터 서버는 백업 서버(TK1_92)에게 TCP 세션 복구 데이터를 전송한다. 이후 마스터 서버에 장애가 발생함을 가정하고, 스위치오버 명령을 통해 기존 백업 서버를 마스터 서버로 활성화한다. 새롭게 활성화된 마스터 서버는 동기화된 TCP 세션 복구 데이터를 이용하여 클라이언트와의 TCP 세션을 복원하고, 서비스를 자동으로 실행한다. 이를 통해 클라이언트는 새롭게 활성화된 마스터 서버에 재접속 하지 않고도 지속적으로 서비스를 제공받을 수 있다.

Fig. 8은 제안하는 기법의 벤치마크 결과를 나타낸다. 클라이언트(TK1_91)와 서버(TK1_93) 간에 256개의 TCP 세션을 생성한 후, 이에 대한 동기화와 TCP 세션 복구를 테스트하였다. 벤치마크 결과, 다중 TCP 세션에 대해서도 동기화 및 복구가 가능함과 함께, 평균 TCP 세션 복구 시간이 약 0.0007초임을 확인하였다. 이는 장애 발생 시 500 msec 이내에 서비스를 재개해야 하는 5-nines(99.999%)의 고가용성을 만족한다[6, 7].

Fig. 9는 제안하는 기법의 세션 수에 따른 평균 TCP 세션 복구 시간을 나타낸다. 성능 평가를 위해 세션 수를 64개에서 512개로 증가하면서 각 세션의 평균 복구 시간을 측정하였다. 성능 평가 결과, 세션 수의 변화에 따라 각각 0.000512초, 0.000628초, 0.000705초, 0.000758초가 소요된다. 이를 통해, 제안하는 기술은 세션의 수가 증가하더라도 세션 복구 시간이 크게 증가하지 않으며, 이를 통해 실제 서비스에도 활용 가능함을 확인하였다.

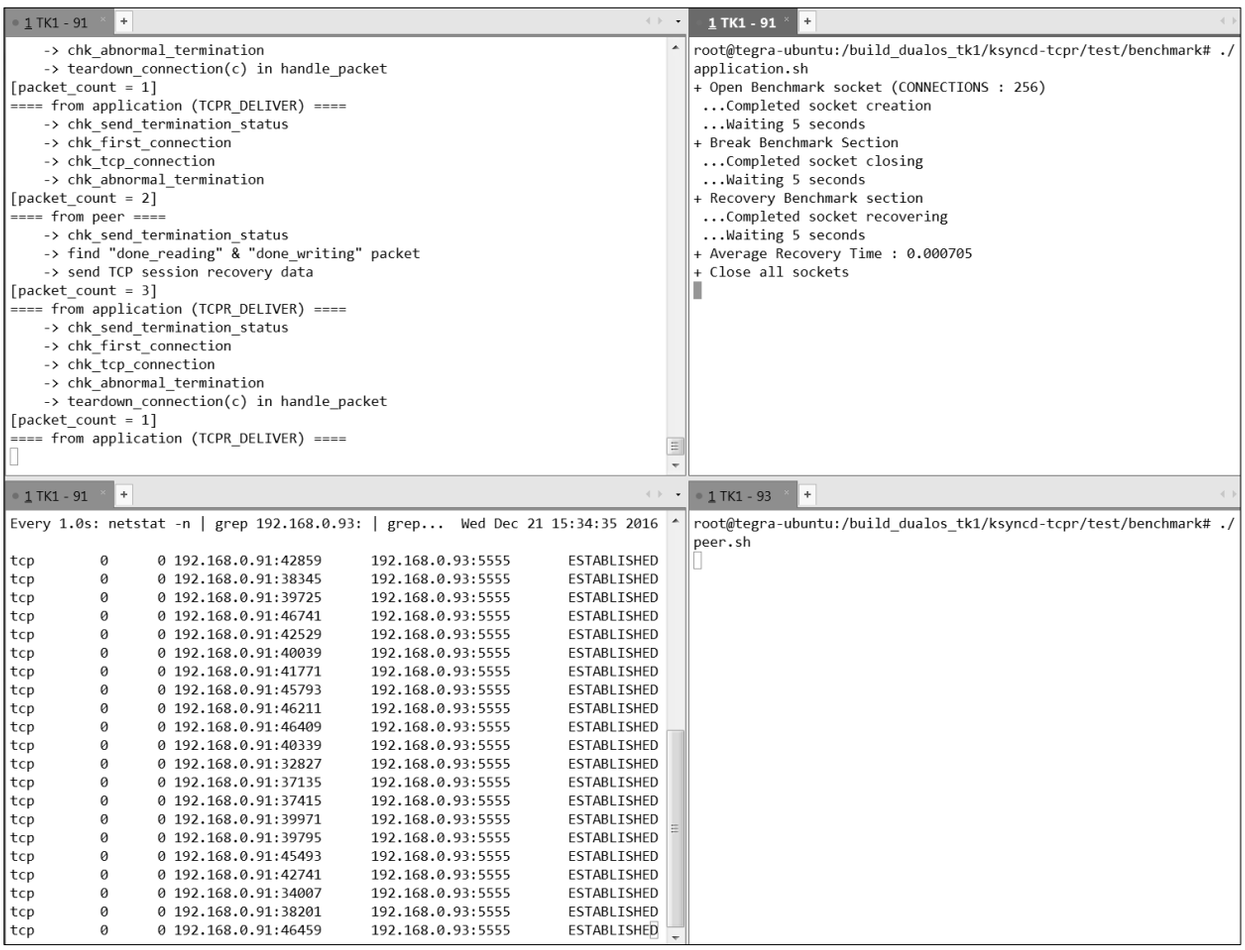

Fig. 8. Benchmark Result of Proposed TCP Session Recovery Technique

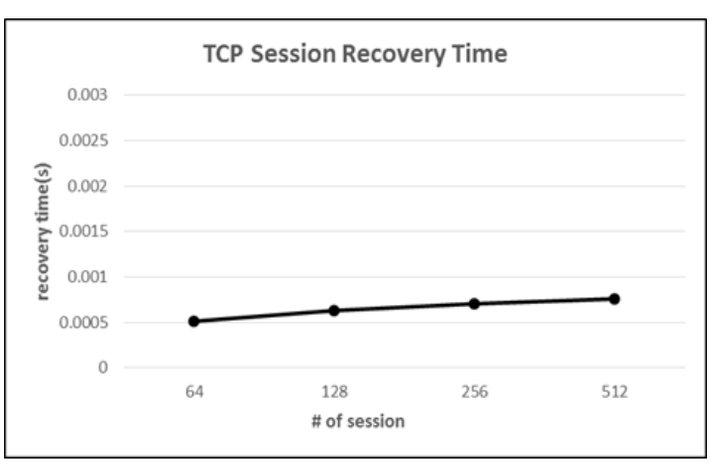

Fig. 9. TCP Session Recovery Time of Proposed Technique

5. 결 론

본 논문에서는 스마트 온디바이스의 고가용성을 위한 TCP 세션 복구 기술을 제안하였다. 제안하는 TCP 세션 복구 기술을 TCP 헤더의 정보를 이용하여 세션 복구 데이터를 생성하며, TCP 헤더의 순서 번호와 확인 응답 번호의 보정을 통해 TCP 세션의 재연결 과정 없이 TCP 세션을 복구한다. 아울러 마스터와 백업으로 구성된 이중화 시스템을 기반으로 세션 복구 데이터를 동기화함으로써, 마스터 시스템에 장애 상황 발생 시에도 페일오버를 통해 사용자에게

서비스를 지속적으로 제공할 수 있다. 또한 사용자는 페일오버 발생 시에도 서버에 재접속이 불필요함에 따라, 무중단 서비스를 제공 받을 수 있다.

아울러 실험 결과를 통해 제안하는 TCP 세션 복구 기술은 다중 TCP 세션 환경에서 5-nines 이상의 고가용성을 지원 가능함을 확인하였다.

향후 연구로는 TCP 세션뿐만 아니라 응용 서비스 데이터의 동기화 및 복구를 지원하도록 제안 기술을 확장하는 것이다.

References

[1] K. S. Lee and S. S. Choi, "TFRC Congestion Control for Mobile Streaming Services Based on Guaranteed Minimum Transmission Rate," *KIPS Transactions on Computer and Communication Systems*, Vol.2, No.3, pp.117-124, 2013.

[2] S. K. Kim and K. S. Chung, "A Performance Improvement of Linux TCP Networking by Data Structure Reuse," *KIPS Transactions on Computer and Communication Systems*, Vol.3, No.8, pp.261-270, 2014.

[3] S. I. Kim and H. S. Kim, "BGP Session Takeover Method Based on Docker," *The Journal of The Korean Institute of Communication Sciences*, Vol.41, No.2, pp.238-240, 2016.

[4] D. H. Kim, J. C. Shim, H. Y. Ryu, J. H. Park, and Y. T. Lee, "Networking Service Availability Analysis of 2N Redundancy Model with Non-stop Forwarding," *Information Science and Applications*, Vol.339, pp.1063-1069, 2015.

[5] B. K. Kim, S. T. Hong, K. G. Lee, J. S. Kim, Y. J. Jung, and C.D. Lim, "Failover Manager on PTN device for 5-nines," in *Proceedings of the IEMEK Symposium on Embedded Technology*, pp.142-143, 2016.

[6] J. H. Lee and K. Y. Lee, "SYNCEYE: An Availability Measurement Tool for Embedded Systems," in *Proceedings of the Asia-Pacific Software Engineering Conference*, pp.15-18, 2014.

[7] High availability from Wikipedia [Internet], https://en.wikipedia.org/wiki/High_availability

[8] L. Alvisi, T. Bressoud, A. El-Khashab, K. Marzullo, and D. Zagorodnov, "Wrapping server-side TCP to mask connection failures," in *Proceedings of the IEEE INFOCOM*, pp.329-337, 2001.

[9] R. Surton, K. Birman, and R.V. Renesse, "Application-driven TCP recovery and non-stop BGP," in *Proceedings of the IEEE/IFIP International Conference on Dependable Systems and Networks*, pp.1-12, 2013.

[10] K. Jayaswal, "Administering Data Centers: Servers, Storage, And Voice Over Ip," Wiley-India, 2005.

[11] GNU General Public License, Gratuitous_ARP [Internet], https://wiki.wireshark.org/Gratuitous_ARP

[12] G. Shenoy, S. K. Satapati, and R. Bettati, "HydraNet-FT: Network support for dependable services," in *Proceedings of the International Conference on Distributed Computing Systems*, pp.699-706, 2000.

[13] N. Aghdaie and Y. Tamir, "Client-transparent fault-tolerant web service," in *Proceedings of the IEEE International Performance, Computing, and Communications Conference*, 2001.

[14] N. Burton-Krahn, "HotSwap - transparent server failover for Linux," in *Proceedings of the Systems Administration Conference*, 2002.

[15] R. Surton, Transparent TCP recovery [Internet], https://github.com/rahpaere/tcpr/

홍 승 태

e-mail : sthong@etri.re.kr
2008년 전북대학교 전자정보공학부(학사)
2010년 전북대학교 전자정보공학부(석사)
2015년 전북대학교 전자정보공학부(박사)
2015년～현 재 한국전자통신연구원 선임연구원
관심분야 : Embedded System, System Software, Big Data, Network Programming, Machine learning

김 법 균

e-mail : kyun@etri.re.kr
2005년 전북대학교 컴퓨터공학과(박사)
2005년～2007년 전북대학교 공학연구원
2007년～2012년 한국과학기술정보연구원 선임연구원
2012년～현 재 한국전자통신연구원 선임연구원
관심분야 : Embedded System, Distributed Systems, Clouds, Grids

이 광 용

e-mail : kylee@etri.re.kr
1991년 숭실대학교 전자계산학과(학사)
1993년 숭실대학교 전자계산학과(석사)
1997년 숭실대학교 전자계산학과(박사)
1997년～1998년 한국전자통신연구원 박사후연수연구원
1999년～현 재 한국전자통신연구원 책임연구원
관심분야 : Embedded System, Software Engineering, Artificial Intelligence, Virtualization System

김 정 시

e-mail : sikim00@etri.re.kr
1992년 경상대학교 전산학과(학사)
1994년 경상대학교 전산학과(석사)
1999년 경상대학교 전산학과(박사)
2000년～현 재 한국전자통신연구원 책임연구원
관심분야 : Embedded System, Software Development Tool, GPGPU computing, On-device Machine learning

임 채 덕

e-mail : cdlim@etri.re.kr
2005년 충남대학교 전산학과(박사)
2006년～2007년 미국 U.C. Irvine 방문연구원
2012년～2016년 한국전자통신연구원 임베디드SW연구부 부장
1989년～현 재 한국전자통신연구원 책임연구원
관심분야 : Embedded System, System Software, Dual Operating System