SIP 메시지를 위한 Loose 라우팅 메커니즘 분석

박선옥° · 현욱 · 허미영 · 강신각

한국전자통신연구원

Analysis of Loose Routing Mechanism for SIP Message

Sun-Ok Park° · Hyun Wook · Mi-Young Huh · Shin-Gak Kang

Electronics and Telecommunications Research Institute

E-mail : {sunoko, whyun, myhuh, sgkang}@etri.re.kr

요 약

인터넷 사용자 수의 급격한 증가로 인터넷 서비스 보급이 대중화됨에 따라 인터넷 텔레포니 기술이 전기통신사업자 및 인터넷 서비스 사업자들의 주요 이슈가 되고 있다. 인터넷 텔레포니 서비스를 위한 호처리 프로토콜을 표준화하기 위해 1999년 9월 IETF산하 SIP WG이 구성된 이래, RFC2543 bis 문서들이 계속 소개되었으며, 2002년 7월초 bis-09문서를 기반으로 RFC3261 SIP 표준이 제정되었다. 본 논문에서는 SIP 메시지 전송을 위해 초창기에 사용하던 Strict 라우팅 방법과 RFC3261에서 사용하는 Loose 라우팅 방법을 비교하고, RFC3261 SIP 표준에서 제공하는 둘 간의 호환성 문제 해결 방안을 분석한다.

키워드

SIP, RFC3261, Strict Routing, Loose Routing

Ⅰ. 서 론

ITU-T의 H.323의 대안으로 IETF에서도 1999년 3월 IETF산하 MMUSIC(Multiparty MUltimedia Session Control) WG(Working Group)에서 인터넷상의 멀티미디어 세션을 제어하기 위해 SIP 표준(RFC2543)을 처음 개발한 이후, 1999년 9월 IP 텔레포니 서비스를 위한 호 처리 프로토콜을 표준화하기 위하여 SIP WG을 분리하였다. SIP WG에서 2000년 6월 rfc2543bis draft가 처음으로 제안되었으며, 2002년 2월말 bis-09버전이 제안된 이후, 2002년 7월 초, rfc2543bis-09 draft를 기반으로 하는 RFC3261를 제정하였다. 또한 bis문서에 포함되었다가 별도의 IETF draft로 제안되었던 기술들도 각각 RFC3262, RFC3263, RFC3264 표준문서로 제정되었다.

현재, 대부분의 국내외 SIP 기반 인터넷 폰은 bis-03 버전을 기준으로 개발된 상태이며, RFC3261가 제정된 이후 RFC3261로의 재개발을 추진하고 있는 중이다. 이전 버전에 비해 RFC3261에서는 Proxy 서버 동작이나, 트랜잭션 (transaction) 처리 등에 대해 명확하게 기술하고 있으며, Via 헤더의 branch 파라미터 뿐만 아니라 SIP 메시지의 라우팅과 관련된 Record-Route나 Route 헤더, Request-URI에 있어서도 많은 부분이 변화되었다.

본 논문에서는 SIP 메시지 전송을 위해 초창기에 사용하던 Strict 라우팅 방법과 RFC3261에서 사용하는 Loose 라우팅 방법을 비교하고, RFC3261 SIP 표준에서 제공하는 둘 간의 호환성 문제 해결 방안을 분석한다.

2장에서는 SIP 메시지 라우팅을 위해 사용되는 관련 헤더들에 대해 간단히 소개하고, 3장에서는 Strict 라우팅 방법에 대해 살펴보고, 4장에서는 호환성 문제 등을 고려한 Loose 라우팅 방법에 대해 살펴보며, 마지막으로 5장에서 결론을 내리기로 한다.

Ⅱ. SIP 메시지

SIP 메시지는 Start-Line을 통해 요청 메시지인지 응답 메시지인지 구분이 되며, 응답 메시지인 경우, Via 헤더에 기록된 요청 메시지 경로를 역으로 거쳐 라우팅이 되며, 요청 메시지인 경우에는 Start-Line의 Request-URI와 Route 헤더를 기반으로 이루어진다. 응답 메시지에 대한 라우팅처리는 비교적 간단하므로

본 논문에서는 요청 메시지에 대한 라우팅 부분만 다루기로 한다.

SIP는 크게 단말을 의미하는 user agent와 네트워크 서버로 나뉜다. 이들은 다시 요청 메시지를 생성하는 UAC(User Agent Client)와 요청 메시지에 대한 응답 메시지를 생성하는 UAS(User Agent Server)로 구성되며, 네트워크 서버로는 Proxy 서버, Redirect 서버, Registrar등이 있다.

UAC와 UAS에서는 다이얼로그가 설정되면, 해당 다이얼로그와 관련된 상태정보들을 [테이블 1]과 같이 저장하게 되며, 이후 이를 기반으로 SIP 메시지를 생성하게 된다.

테이블 1. 다이얼로그 상태 정보

Dialog state		UAC	UAS
Dialog ID	Call-ID	Request의 Call-ID (생성)	Request의 Call-ID
	Local tag	Request의 From tag (생성)	Response의 To tag(생성)
	Remote tag	Response의 To tag	Request의 From tag
local sequence number		Request의 CSeq	Empty
Remote sequence number		Empty	Request의 CSeq
local URI		From 헤더의 URI	To 헤더의 URI
remote URI		To 헤더의 URI	From 헤더의 URI
remote target URI		Response의 Contact 헤더 URI	Request의 Contact 헤더 URI
secure flag (boolean flag)		TLS를 통해 Request 송신 시 True로 설정	TLS를 통해 Request 수신 시 True로 설정
route set		Response의 Record-Route 헤더의 URI 리스트 (파라미터를 포함하며 역순으로)	Request의 Record-Route 헤더의 URI 리스트 (파라미터를 포함하며 순서대로)

UAC와 UAS는 요청 메시지 수신시, Record-Route 헤더의 URI 리스트(파라미터들을 포함)를 다이얼로그 상태정보 중 route set에 저장해 둔다. UAS는 INVITE 요청 메시지의 Record-Route 헤더에 있는 URI 리스트를 순서대로 route set에 넣으며, UAC는 INVITE 요청 메시지에 대한 응답 메시지의 Record-Route 헤더의 URI 리스트를 역순으로 route set에 기록한다. Record-Route 헤더가 없으면, pre-existing route set이 있다 하더라도 route set은 반드시 비워 두어야 한다. UAC는 다이얼로그 설정 이후에는 route set을 기반으로 요청 메시지의 Route 헤더를 생성하며, 다이얼로그 설정 이전에는 pre-existing route set을 기반으로 Route 헤더를 생성한다.

III. SIP Strict 라우팅

RFC2543에서부터 rfc2543bis-05버전까지 사용하던 SIP 메시지 라우팅 방법이다. 이 장에서는 Strict 라우팅을 기반으로 UAC가 요청 메시지를 어떻게 생성하고, Callee의 UAS로 해당 요청 메시지를 어떻게 전송하는지 알아본다.

3.1. user agent에서의 메시지 생성

UAC와 UAS에 저장될 route set 상태정보 생성시, [테이블 1]에서 설명한 방법과 다소 차이가 있다. remote target URI라는 상태정보를 별도로 관리하지 않고 route set 상태정보에 상대 단말의 Contact 주소를 함께 저장하고 유지한다. 따라서, route set 상태정보에는 Record-Route 헤더의 URI 리스트 뿐아니라, remote target URI가 리스트의 마지막에 추가된 후 route set에 저장된다.

UAC에서는 route set의 첫번째 URI는 Request-URI에, 나머지 URI 리스트는 Route헤더에 각각 넣어 요청 메시지를 생성한다. 이렇게 생성된 요청 메시지는 Request-URI에 적힌 URI로 전송되어져야 하나, 만약 outbound proxy가 설정되어 있다면 Request-URI로 전송하지 않고 outbound proxy로 먼저 전송한다.

3.2. Proxy에서의 라우팅

Proxy 서버에서의 요청 메시지 라우팅은 비교적 간단한 편이다. Route 헤더의 첫번째 URI를 Request-URI에 넣고, 해당 URI를 Route 헤더에서 제거시킨다. 그리고 나서, Request-URI로 해당 메시지를 전송한다

IV. SIP Loose 라우팅

일반적으로 Strict 라우팅이라는 용어는 명시한 특정 경로만을 따라 메시지 전송이 이루어질 때 사용된다. 그러나 outbound proxy를 사용하는 SIP Strict라우팅은 엄밀히 따지자면 Strict 라우팅이라 할 수 없다.

따라서, rfc2543bis-06 버전에서 Loose 라우팅이라는 방법이 처음 제안되었으며, bis-07이후의 bis 문서 및 RFC3261에서도 이러한 Loose 라우팅 방법을 사용하고 있다. Loose 라우팅에서는 Strict 라우팅과의 호환성 문제를 해결하기 위해 lr 파라미터를 하용하고 있으며, rfc2543bis-07 버전에서 처음으로 사용되었다.

4.1. user agent 동작

UAC와 UAS에서의 다이얼로그 상태정보들은 2장에서 살펴본 [테이블 1]과 동일한 방법으로 생성된다.

4.1.1. 요청 메시지 생성

-다이얼로그 설정 이전

일반적으로 UAC는 요청 메시지 생성시 Request-URI에는 저장되어 있는 다이얼로그 상태 정보 중 remote target URI를, Route 헤더에는 route set의

URI 리스트를 이용하여 값을 설정하나, 다이얼로그 설정이 이루어지기 이전이므로 remote target URI나 route set을 이용할 수 없다. 따라서, 초기 Request-URI는 To 헤더 필드와 마찬가지로 Callee의 SIP URI로 설정하고 Route 헤더는 비워두어야 한다. 하지만 pre-existing route set이 설정되어 있을 때에는 다이얼로그가 설정되기 전이라 하더라도 이를 기반으로 Route 헤더를 생성해야 한다. user agent에는 outbound proxy를 pre-existing route set에 설정할 수도 있다.

-다이얼로그 설정 이후

다이얼로그가 설정된 이후, UAC에서 생성되는 요청 메시지는 저장되어 있는 상태정보들을 기반으로 헤더 값이 결정된다. [그림 1]은 요청 메시지에서 라우팅과 관련된 Request-URI와 Route 헤더 값을 어떻게 결정하는지를 보여주는 순서도이다.

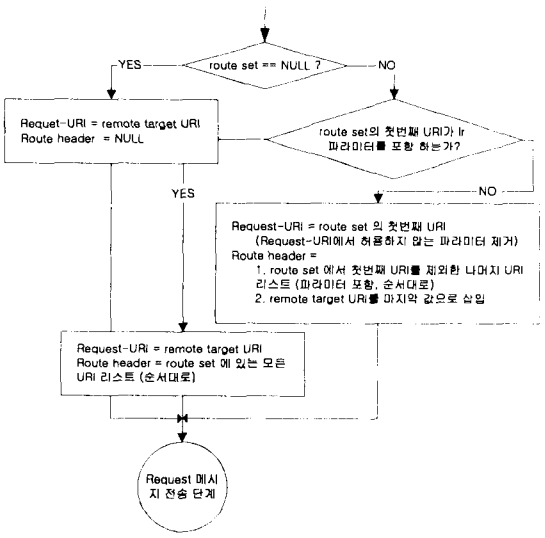

그림 1. 요청 메시지의 헤더 생성 방법

4.1.2. 요청 메시지 전송

필요한 모든 헤더 생성이 끝나면, 요청 메시지를 어디로 전송할지 Next-Hop 주소, 포트, 전송 프로토콜을 결정한다. 특정 Next-Hop을 명시하는 별도의 local policy가 있는지 확인하고, 만약 없다면 DNS Lookup을 통해 하나 이상의 Next-Hop을 결정한다. DNS Lookup시에 사용할 URI는 route set을 기반으로 결정되며 경우에 따라 Request-URI가 될 수도 있고 route set의 첫번째 URI가 될 수도 있다.

-route set의 첫번째 URI가 lr 파라미터를 포함하지 않으면 : Request-URI 이용

-route set의 첫번째 URI가 lr 파라미터를 포함하면 : route set의 첫번째 URI 이용

-route set이 비어 있으면 : Request-URI 이용

Outbound proxy를 명시하는 방법에는 두 가지가 있을 수 있다. Outbound proxy로 사용할 하나의 proxy 주소를 pre-existing route set에 명시하는 방법과 local policy로 특정 Next-Hop을 설정해두는 방법이 있다. 그러나 bis-09버전에서는 첫번째 방법을 사용하여 Outbound proxy를 설정하도록 권고하고 있다.

4.2. Proxy 서버에서의 동작

Proxy 서버에서 Request 메시지를 수신하면, 일단 Request-URI, Route 헤더 등 경로배정과 관련된 정보를 전처리 한 다음, Request 메시지를 어디로 보내야 할지 target set을 결정한다. target이 결정되면 필요에 따라 헤더들을 수정하고, 메시지를 내보내기 전에 Request-URI나 Route 헤더를 다시 한번 후처리 한 후, 결정된 Next-Hop으로 메시지를 전송한다. 이러한 일련의 과정들을 통해 Proxy 서버에서의 메시지 라우팅이 이루어진다.

4.2.1. 라우트(route) 정보에 대한 전처리

RFC3261에서는 Proxy 서버가 Record-Route 헤더에 자신의 주소를 삽입할 때, 반드시 lr 파라미터를 추가하도록 명시하고 있다. 하지만 Loose 라우팅을 지원하지 못하는 Proxy 서버와의 호환성 문제 때문에 Request-URI와 Route 헤더에 대한 전처리 과정이 필요하다. 수신한 요청 메시지의 Request-URI가 이전에 자신이 넣어 두었던 URI이면, Route 헤더의 마지막 URI를 Request-URI로 넣고 Route 헤더에서 해당 URI를 삭제한다.

이러한 경우는 요청 메시지를 전송해 준 바로 이전의 Proxy 서버가 lr 파라미터를 처리하지 못하는 Proxy서버일 경우에 발생한다.

라우트 정보에 대한 전처리는 Request-URI가 maddr 파라미터를 포함하고 있을 경우에도 필요하다. Request-URI가 maddr 파라미터를 포함하고 있다면, maddr 파라미터 값이 Proxy 서버에서 관리하는 도메인이거나 주소인지를 확인해 보아야 한다. 만약, Request-URI의 maddr 파라미터 값이 Proxy 서버가 관리하는 값이고, Request-URI에서 가리키는 포트와 전송 프로토콜 (디폴트 값이거나 명시된 값)로 해당 요청 메시지가 수신되었다면, 요청 메시지의 Request-URI에서 maddr 파라미터와 디폴트가 아닌 포트와 전송 프로토콜 파라미터를 각각 제거한다.

마지막으로, Route 헤더의 첫번째 URI가 Proxy 서버 자신을 가리키면, Route 헤더에서 첫번째 URI를 삭제한다.

Proxy 서버는 Request-URI와 Route 헤더 그리고

Request-URI의 maddr 파라미터 등에 대한 전처리 과정을 통해 Request 메시지를 수정한 후, 재구성된 요청 메시지를 수신 한 것처럼 다음 과정들을 수행한다.

4.2.2. 목적지(target) 결정 방법

Proxy 서버는 전처리 과정을 통해 재구성된 요청 메시지를 어디로 보낼지 결정해야 한다. Request-URI를 기반으로 목적지가 결정되며 결정된 목적지는 target set에 유지된다. [그림 2]는 Proxy 서버에서 target set을 어떻게 만드는지 보여주는 순서도이다.

target set에는 하나 이상의 목적지 리스트가 존재해야 하며, Request-URI를 이용하여 목적지를 결정할 수 없을 시에는 485(Ambiguous) 응답 메시지를 전송해야 한다.

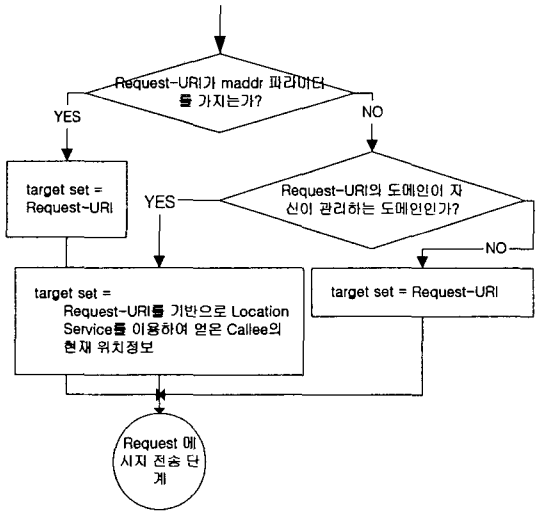

그림 2. 요청 메시지의 target set 결정 방법

4.2.3. 요청 메시지의 헤더 처리 방법

target set이 결정되면, 요청 메시지를 target set으로 전송해야 한다. target set으로의 전송을 위해 요청 메시지의 Request-URI를 수정한다. target set에 있는 URI로 Request-URI 값을 대체하며, Request-URI에서 허용하지 않는 파라미터들은 제거한다. 또한 Max-Forward 헤더 값을 1 감소하고, 필요에 따라 Record-Route 헤더에 값을 추가하거나 기타 다른 헤더들을 추가한다.

4.2.4. 라우팅 정보에 대한 후처리 사항

위와 같은 일련의 과정을 통해 요청 메시지가 재구성되면, Next-Hop으로 메시지를 전송해야 한다. 그러나 Strict 라우팅을 사용하는 시스템과의 호환성을 위해 라우팅 정보에 대한 후처리를 수행한 이후에 Next-Hop으로 메시지를 전송한다.

요청 메시지에 대해 하나 이상의 특정 Proxy 서버를 반드시 지나도록 Local policy가 설정되어 있을 경우, Route 헤더에 명시된 Proxy 서버 리스트를 추가하는 후처리 과정이 필요하다. 요청 메시지에 Route 헤더가 있을 경우에는 헤더의 최상위 값으로 리스트를 추가하고, 없을 경우에는 Route 헤더를 생성하여 추가한다. 이 경우, 설정되어 있는 모든 Proxy 서버들은 lr 파라미터를 반드시 가져야 한다.

이러한 과정이 모두 끝나게 되면 lr 파라미터에 대한 후처리 과정이 필요하다. 요청 메시지에서 Route 헤더의 첫번째 URI를 보고 lr 파라미터를 포함하지 않으면, 요청 메시지의 Request-URI를 Route 헤더의 마지막 값으로 삽입하고, Request-URI에는 대신 Route 헤더의 첫번째 URI를 넣는다. 그리고 Route 헤더의 첫번째 URI는 헤더에서 제거한다.

이러한 후처리 과정이 끝나면, Nest-Hop 주소, 포트, 전송 프로토콜을 결정한다.

4.2.5. Next-Hop 결정방법

요청 메시지의 Request-URI나 Route 헤더와는 독립적으로 특정 IP 주소, 포트, 전송 프로토콜로 메시지를 전송하도록 Local policy가 설정되어 있을 수도 있으나 Route 헤더를 이용하여 사용하도록 권고하고 있다.

별도의 Local policy가 없다면, Request 메시지의 Request-URI나 Route 헤더의 URI를 이용한 DNS Lookup을 통해 Next-Hop IP 주소, 포트, 전송 프로토콜을 결정한다.

앞에서 설명한 후처리 과정에서 Route 헤더의 첫번째 URI가 lr 파라미터를 포함하지 않음으로 인해 Request-URI, Route 헤더가 재구성되었다면, 재구성된 메시지의 Request-URI를 이용하여 Next-Hop을 결정한다. 반면, 요청 메시지가 재구성되지 않았다면 Route 헤더의 첫번째 URI를 이용하며, Route 헤더가 없는 경우에는 Request-URI를 이용하여 Next-Hop을 결정한다.

DNS Lookup을 통해 얻은 모든 Next-Hop에 대해 순차적으로 메시지 전송을 시도하였으나 실패하였을 경우 전송하려던 Request 메시지에 대해 408(Request Timeout) Response 메시지를 수신한 것처럼 동작한다.

V. 결 론

rfc2543bis-05버전까지 사용되던 Strict 라우팅 방식과 bis-06버전에서 처음 제안된 이후, RFC3261 SIP 표준에서까지 사용되는 Loose 라우팅 방식에 대해 각각 살펴보았다. Loose 라우팅 방식에서는 Strict 라우팅을 사용하는 Proxy 서버와 Loose 라우팅을 사용하는 Proxy 서버와의 호환성 문제를 해결하기 위해 Request-URI와 Route 헤더에 대한 전처리 과정과 후

처리 과정이 존재한다. 이후 RFC3261 SIP 표준을 기반으로 하는 제품 구현시 개발자들에게 많은 도움이 되리라 기대한다.

참고문헌

[1] J. Rosenberg, H. Schulzrinne, et al., "SIP: Session Initiation Protocol," rfc2543bis-03.
[2] J. Rosenberg, H. Schulzrinne, et al., "SIP: Session Initiation Protocol," rfc2543bis-09.
[3] J. Rosenberg, H. Schulzrinne, et al., "SIP: Session Initiation Protocol," RFC3261.