출처: http://crowback.tistory.com/category/Programming/Protocol%20-%20RUDP

RUDP 의 기초 : draft-ietf-sigtran-reliable-udp-00.txt [1/4]

Programming/Protocol - RUDP
RUDP : Reliable User Datagram Protocol (Reliable UDP)

IETF 상에서 아직(2007-04 현재) 정식으로 등록되지 못한 MEMO 등급의 문서이다.
http://www.ietf.org/proceedings/99mar/I-D/draft-ietf-sigtran-reliable-udp-00.txt
DRAFT 로 등록되어 있으며 초기 문서인 00 버전이 ietf 사이트에 등록되어있다.
현재까지 지속적으로 업데이트 되고 있으므로, 이를 구현하였을 경우 벤더가 다르면
서로 호환되지 못할 수도 있으므로, 만약 RUDP 를 구현하여 어떤 장비나 기나 endpoint 와
통신할 경우는 이 점에 대하여 인지하고 대처하여야 한다.

본 문서는 RFC908(v1), RFC1151(v2) RDP(Reliable Datagram Protocol)에서 보강된 프로토콜이다.

아래의 내용은 드레프트를 대충, 그냥.. 간단하게 생략 번역한 것이다.
이해가 모자란 부분은 원문을 참조하기 바란다.

본 드레프트 문서는 다음과 같은 목차를 가지고 있다.

1. 소개

본 드래프트 문서는 데이터를 전송하는 간단한 시퀀스 베이스 프로콜(즉, UDP)을 이용하는 어플리케이션에 신뢰성을 부여하는 것에 대하여 논한다. RUDP 는 RFC908 및 RFC1151 - Reliable Data Protocol 을 기초로 작성되었다. 또한 RUDP 는 UDP/IP 레이어 위에 설계되었다.

1.1 시스템 오버뷰

목차에는 있던데, 원문에는 없다.. -_-;;;

1.2 백그라운드

IP 네트워크상에서 텔레포니 시그널을 전송하기 위해서는 신뢰성있는 전송 프로토콜이 필요하다. 이 신뢰성있는 전송 프로토콜은 다양한 어플리케이션의 아키텍처를 지원할 수 있어야 한다. 현존하는 전송 프로토콜을 면밀히 조사해본 결과 텔레커뮤니케이션 시그날링을 위한 새로운 신뢰성있는 메카니즘이 필요하게 되었다. 이 새로운 프로토콜은 다음과 같은 기준을 반드시 포함해야 한다.

* 전송계층은 최대크기의 데이터 의 신뢰성 있는 전송을 지원해야만 한다.
* 전송계층은 순차전송을 보장해야 한다.
* 전송계층은 메시지 베이스여야 한다.
* 전송계층은 플로우 컨트롤 메카니즘을 지원해야한다.
* 전송계층은 적은 오버헤드에, 최고의 퍼포먼스를 내야한다.
* 개별적인 가상 연결을 핸들링 할 수 있어야 한다.
* 킵 얼라이브 메카니즘을 지원해야 한다.
* 에러 디텍션을 지원해야 한다.
* 암호화된 전송을 지원해야 한다.

RUDP 는 이러한 것들을 지원하기 위하여 디자인 되었다.

1.3 데이터 구조

1.3.1 데이터 전송을 위한 최소 헤더 six octet

UDP 처럼 전송되는 RUDP 는 앞단에 여섯개의 옥텟헤더로 시작해야한다. 첫번째는 시리즈로 구성된 싱글비트 플래그 7 개와 리절브 비트 한개, 다음 3 개는 한 바이트로 구성된 넘들.. 헤더길이, 시퀀스번호, 엑크넘버이다. 이 뒤에는 2 옥텟(바이트)으로 이루어진 첵크섬이 따라 붙는다.

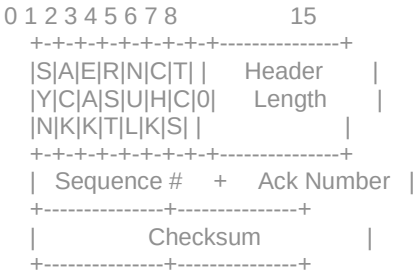


Figure 1, RUDP Header

Control bits

컨트롤 비트는 이 패킷이 머하는 넘인지를 구분시켜준다. SYN 비트는 동기화를 위한 세그먼트의 존재를 알린다. ACK 는 헤더가 유효한지를 알려주고, EAK 는 ACK 의 확장 비트이고, RST 는 세션을 리셋시키는 넘이다. NUL 은 null 세그먼트라는데???, TCS 는 상내를 알려준다. CHK 는 0 과 1 로 되는데 0 일 경우는 첵크섬이 헤더만 계산한 것이고, CHK 가 1 이면 헤더와 데이터영역을 모두 계산한 것이다. 요런 저런 컨트롤에 필요한 세그먼트를 몇가지 모아 놓은 것이데... 이런걸 flow control 을 위한 control bit 라고 하고.. 이런게 있으면 플로우 컨트롤을 지원한다 라고 말할 수 있다.

Header length

사용자 데이터를 포함하지 않은 헤더만의 크기를 나타낸다. 덩어리 패킷을 받았을 때, 시작부터 이 길이만큼을 제거하면 사용자 데이터의 시작이다. EACK, NUL, RST 비트가 셑 되어있으면 사용자 데이터 영역이 따라올 수 없다. 유저 데이터가 포함될 때는 항상 ACK 플래그가 셑 되어있다.

Sequence number

모든 패킷은 시퀀스 넘버를 가져야한다. 커넥션이 처음으로 열릴 때 초기화 시킨 랜덤넘버를 채운다. 이 번호는 커넥션이 열릴 때 SYN 플래그를 탑재한 패킷에 넣는다. 데이터를 전송할 때 마다 시퀀스 넘버를 증가 시킨다.

Acknowledgment Number

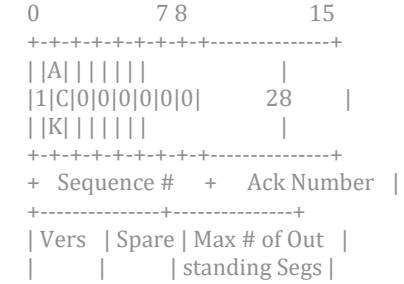
엑크넘버는 전송단위를 구분하는 구분자로 받는 측에서 마지막 받은 시퀀스 넘버를 이용한다.

Checksum

첵크섬은 모든 RUDP 헤더에 계산되어 포함된다. CHK 비트의 값에 따라 전체 혹은 헤더만 계산하여 넣는다. 여기서 사용되는건 UDP 나 TCP 에서 사용하는 16-bit one's complement of the one's complement sum 방식을 사용한다.

1.3.2 SYN 세그먼트

SYN 은 두 호스트간에 연결할 때 시퀀스 넘버를 동기화 할 때 사용한다. 이 세그먼트를 포함한 패킷에는 연결에 필요한 협상정보(negotiable parameters)를 포함한다. 설정가능한 모든 파라미터는 상대방이 알아야 하므로 협상을 위하여 포함시킨다. SYN 은 유저 디파인 데이터를 포함 할 수 없다. SYN 세그먼트는 연결을 자동으로 리셋시키는데 사용될 수 도 있다. (대충 연결되어 있는데 다시 싱크 날리면 연결정보를 재설정하는가 보다.)
아래 그림은 SYN 세그먼트이다.



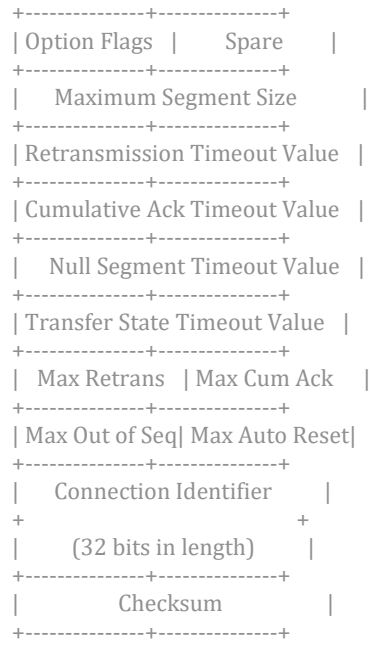


Figure 2, SYN segment

Sequence Number
이 연결을 위한 초기 시퀀스 번호를 포함하고 있다. (랜덤하게 생성한다.)

Acknowledgment Number
이 필드는 ACK 가 설정되어 있을 경우만 유효하다. 이 케이스에서 , 필드는 상대측에서 보낸 SYN 세그먼트의 시퀀스 넘버가 포함된다.

Version
RUDP 버전. 초기 버전은 1 이다.

Maximum Number of Outstanding Segments
위 정보는 ACK 없이 보내져야한다. 이 정보는 받는 측에서 플로우 컨트롤에 사용된다. 이 값은 초기에 설정되면 연결이 유지되는 동안에는 변경될 수 없다. 이 정보는 협상대상이 될 수 없다.

Options Flag Field
이 플래그는 2 바이트짜리다. 아래 보이는것 처럼 디자인되어있다.

BIT#	Name	Description
0	not used	---
1	CHK	데이타 섹섬 인에이블. 이게 1 로 설정되어 있으면 헤더와 데이터를 포함한 섹섬을 포함한다. 협상가능한 파라미터
2	REUSE	이 비트는 오토리셑ㄷ되는 동안에 이전에 협상된 파라미터의 사용 여부를 결정하기 위하여 꼴 설정되어야 한다. 1 로 설정 되면 이전 껄 쓰고 따라붙는 항목을 무시하고, 0 이면 새걸쓴다
3-7	Spare	

Maximum Segment Size
이 값은 상대측에서 SYN 을 보낼 때 내가 받는 정보이다. 이 값은 서로 다른 값을 가질 수도 있다. 서로가 연결 협상을 하고 있는 동안 서로가 받게되며, 각각은 상대에게 이 값보다 크게 패킷을 전송하면 않된다. 이 숫자는 RUDP 으 헤더의 크기도 포함한다. 이 정보는 협상대상이 될 수 없다.

Retransmission Timeout Value
패킷이 응답하지 않을 경우 재전송까지 기다리는 시간값. 밀리세컨단위. 레인지는 0 - 65536. 협상가능한 값으로 양쪽이 모두 같은 값에 동의해야한다.

Cumulative Ack Timeout Value
연속된 세그먼트중에 하나를 보내지 않았을 경우 다음 세그먼트를 보내기까지 사이 값에 해당하는 타임아웃 값. 밀리세컨단위. 레인지는 100-65536. 양쪽이 동의한 값에 협상이 가능하다. 추가적으로 이 값은

무조껀 *Retransmission Timeout Value* 보다는 작아야한다.

Null Segment Timeout Value
데이터 세그먼트를 보내지 않았을 경우, 널 세그먼트를 보내기 위한 타임아웃. 널 세그먼트는 킾 얼라이브와 같은 메카니즘이다. 밀리세컨단위. 레인지는 0-65536. 협상가능한 파라미터.

Transfer State Timeout Value
자동 리셋이 발생하기전의 연걸을 위하여 저장되어 있어야 하는 상태 정보의 타임아웃. 밀리세컨단위. 레인지는 0-65536. 협상가능한 파라미터. 만약 이 값이 0 이면 상태정보의 저장 없이 바로 재연결이 이루어진다.

Max Retrans
재 전송이 연속적으로 이루어질 때 몇번을 할것인지를 알려주는 최대값. 레인지는 0-255. 이 값이 0 이면 계속 재전송을 한다. 협상가능한 파라미터.

Max Cum Ack
The maximum number of acknowledgments that will be accumulated before sending an acknowledgment if another segment is not sent. 레인지는 0-255. 이 값이 0 이면 데이터, 널, 리셋 세그먼트를 받았을 경우 acknowledgment segment 를 바로 보낸다. 협상 가능한 파라미터.

Max Out of Seq
EACK 를 보내기 전까지 누적된 시퀀스 넘버가 잘못된 패킷의 최대 값. 레인지는 0-255. 이 값이 0 이면 EACK 를 바로 보낸다. 협상가능한 파라미터.

Max Auto Reset
연결을 리셋하기 전까지 수행된 연속적인 오토 리셋의 최대값. 레인지는 0-255. 이 값이 0 이면 토리셋은 수행되지 않고, 바로 리셋되어 버린다. 협상가능한 파라미터. 이 카운터는 연결이 새로 열리면 클리어된다.

Connection Identifier
새로운 연결이 설정되면 현재 연결사이에서 구분될 수 있는 유일값을 주고 받는다. 양쪽에서는 이값을 저장하고 있어야한다. 오토리셑이 수행되면 상대측에서는 저장해 두었던 오리지날 ID 를 전송하여 오토리셋이 수행되는 동안 사용한다.

1.3.3 ACK Segment

ACK 세그먼트는 연속적인 세그먼트 사이에 사용된다. 이는 RUDP 헤더에 다음 시퀀스 번호와 액크넘버를 포함하고 있다. ACK 세그먼트는 세그먼트를 구분할 때 보내어지나, 전송할 때 데이터를 포함하여 전송이 가능하다. DATA세그먼트와 NULL 세그먼트는 항상 ACK 필드를 포함해야 하며 Acknowledgment Number 를 채워야한다. 독립적인 ACK 세그먼트는 6바이트로 구성되어있다.

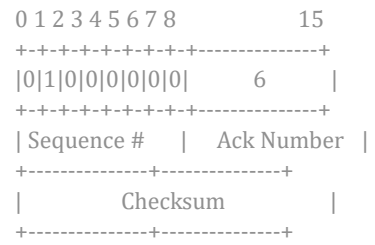


Figure 3, Stand-alone ACK segment

1.3.4 EACK segment

EACK 세그먼트는 out of sequence(시퀀스 번호가 잘못됨)를 받았을 경우 보내는 acknowledge segments 이다. 이 세그먼트는 잘못된 시퀀스 번호를 포함한게 몇개였는지 정보가 포함되어있다.

항상 ACK 세그먼트아 함께 보내지며, 마지막에 제대로 전송된 시퀀스 번호를 넣어서 보낸다. 헤더의 길이는 확정되어 있지 않다(variable). 최소 길이는 7바이트이며 최대 길이는 maximum receive queue length 값에 의존한다.

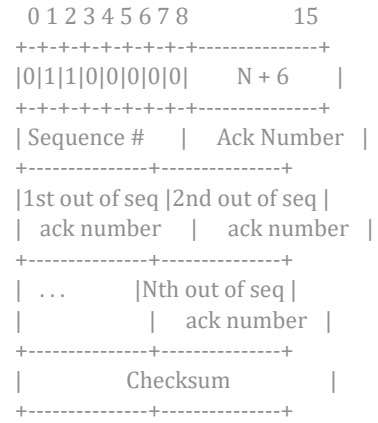


Figure 4, EACK segment

1.3.5 RST segment

RST 세그먼트는 연결을 닫거나 리셋할 때 사용된다. RST 세그먼트를 받았을 경우, 전송자는 반드시 새로운 패킷을 보내는 것을 중지해야한다. 그러나 이미 보내어지고 있던 패킷은 모두 보내어야한다. RST 세그먼트는 구분자로써 사용되니 어떠한 데이터도 포함해서는 않된다.

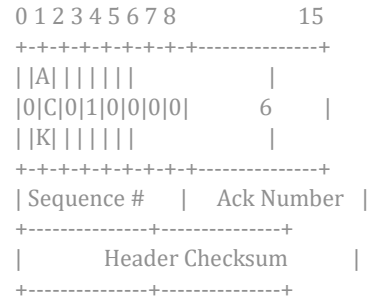


Figure 5, RST segment

1.3.6 NUL segment

NUL 세그먼트는 상대측이 아직도 살아있나를 결정하는데 사용된다(일종의 Keep-Alive). 그래서 일걸 keep-alive 라고 부르기도 한다. NUL 세그먼트를 전송받으면, 그 즉시 ACK 세그먼트로 응답해야 하며, 다음 시퀀스 번호를 넣어서 연결이 유효함을 알려주어야한다. 그리고 받은 NUL 세그먼트는 그냥 버린다. NUL 은 반드시 ACK를 포함해야하며 유저데이터는 절대 포함되어서는 않된다.

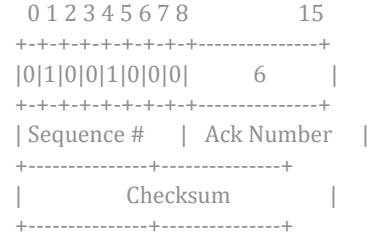


Figure 6, NUL segment

1.3.7 TCS Segment

TCS 세그먼트는 현재 연결의 상태정보를 전달한다.

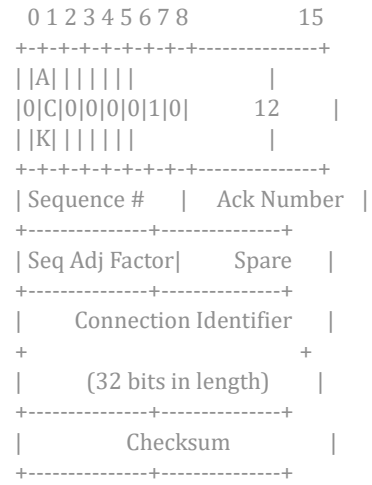


Figure 7, TCS segment

Sequence Number

현재 연결에 선택되었던 초기의 시퀀스 번호...

Acknowledgment Number

마지막 시퀀스 넘버에 대응하는 ack number를 만들어 넣는다.

Seq Adj Factor

This field is used during transfer of state to adjust sequence numbers between the old and current connections.

Connection Identifier

새로 열린 RUDP 연결에서 전 구간에 걸쳐 유일하게 연결을 구분할 수 있도록 해주는 ID값이다. 여러개의 연결이 있을 경우 각각의 연결을 구분해주는데 사용된다.

1.3.8 Detailed Design

A separate internet draft is being prepared which details in connections state transitions in Specification and Description Language (SDL) format. It also contains more details on the internal design of RUDP.

1.3.9 Feature Description

아래의 내용들은 RUDP를 이해하는데 도움이될 몇몇 특징을 적어놓았다. 아래의 세부 내용들은 서버와 클라이언트 간의 연결속에서 세그먼트를 서보 주고받을 때에 사용된 용어 및 동작에 관한 내용들이다.

Retransmission timer

전송측은 설정가능한 전송주기 타이머를 가진다. 이 타이머는 data, nul, or reset 등이 전송되었을 때 마다 항상 초기화되어 가동된다. 만약 송신에 대한 응답을 받지 못하였을 경우.. 타이머가 지정된 시간에 동작하여 재전송하는데 이용된다. 만약 여전히 하나 혹은 여러개의 응답을 받지 못하였을 경우 이 타이머는 재가동될것이다. 추천값은 600 ms 이다.

Retransmission Counter
위에서 나온 재전송 타이머의 최대 반복 횟수에 필요한 카운터이다. 이 것또한 설정가능한 값이며 추천값은 2이다. 만약 카운터 값이 설정된 값을 초과하게 되면 이 연결은 깨진 연결로 간주하고 처리해야한다. 아래의 *Broken connection handling* 에서 이러한 상태를 어떻게 처리하는지 세부적으로 설명하고 있다.

Stand-alone acknowledgments
stand-alone acknowledgment 세그먼도 하나의 세그먼트로 acknowledgment 에 관한 정보만을 포함하고 있다. 시쿤스 번호 필드에는 data, null, or rest의 다음 시퀀스 번호를 포함한다.

Piggyback acknowledgments
수신자가 전송자에게 data, null, or reset 세그먼트를 전송할 때는 마지막 시퀀스 번호에 대응한 엑크를 포함하여 전송하여야 한다.

Out-of-sequence acknowledgments counter
수신자는 시퀀스 번호가 잘못된 세그먼트가 도착하였을 경우 그 수를 세고 있어야 한다. 그 수가 한도를 초과하면 EACK에다가 out-of-sequence의 한도값을 넣어서 송신자에게 보내주어야 한다. 그리고 0으로 리셋한다. 추천값은 3이다. 만약 송신자에게 EACK를 포함한 놈이 도착되면 자신이 보낸 데이터가 수신자에게 전달되어 못하였음을 인지해야 한다.

Cumulative acknowledge timer
응답이 아니거나 잘못된 시퀀스를 받았을 경우 수신자는 타이머를 설정하고 기다려야한다. 이 타이머가 익스파이어 되면 아웃오프시퀀스 경우는 EACK를 전송한다. 다른 경우 스탠드얼론 엑크를 전송한다. 추천값은 300 ms 이다.

Null segment timer
클라이언트는 타이머를 연결이 열기는 순간부터 유지해야한다. 그리고 데이터 세그먼트를 전송할 때 마다 초기화 시켜서 재가동 시켜준다. 만약 클라이언트의 null segment 타이머가 익스파이어되면, 클라이언트는 서버로 null 세그먼트를 전송해준다. 즉 keep alive 체크를 유지해야 한다는 이야기다. 이 세그먼트의 시퀀스 번호가 유효하다면 서버는 이에 응답해야 한다. 서버는 널 세그먼트의 이전의 값과 현재 값 두개를 타임아웃 값으로 유지해야한다. 만약 클라리언트로 부터 널 세그먼트를 수신 받으면 이 타이머는 리셋된다. 만약 이 타이머가 익스파이어되면 연결이 끊어진 것으로 간주한다. 이 부분에 대해서는 아래의 *Broken Connection Handling*을 참조한다. 추천 값은 2 sec 이다.

Auto Reset
양쪽의 연결은 오토 리셑으로 초기화 할 수 있다. 이 오토리셑은 과도한 재전송, 널 세그먼트 타이머 익스파이어, 전송 상태 타이머의 익스파이어등 다양한 상황에서 사용될 수 있다. 오토 리셑은 양측 연결을 초기화하고 각종 타이머값을 초기할 수 있으며, 시퀀스의 초기화 등 연결에 필요한 설정값을 다시 협상하여 재설정 할 수 있다. 오토 리셑 카운터의 최대값을 넘기지 않으면 연결을 유지한 상태에서 설정이 가능하다. 최대값을 넘든다면 당연히 연결설정이 초기화 된다. 추천값은 3이다.

Receiver Input Queue Size
입력 큐 사이즈는 설정 가능한 파라미터이다. 추천값은 32 패킷이다. 입려 큐의 사이즈는 플로우 컨트롤 메카니즘을 따른다. 대충 하나의 데이터 크기는 저 패킷수를 넘지 않도록 조절하라는 이야기인건가?

Congestion control and slow start

RUDP는 이러걸 지원하지 않는단다.

UDP port numbers
RUDP에서는 특정한 포트 번호를 제안하지 않고있다. 쓰고 싶은데로 정해서 쓰면된다. 단 일반적으로 쓰고 있는 디파인된 포트는 피하는게 좋다. RFC 1700 참조.

Support for redundant connections
만약 RUDP 연결이 실패하면 상위 레이어 프로토콜(ULP - Upper Layered Protocol)에서 스트널이 발생할 것이고 전송상태 타이머를 가동한다. 그리고 상위 레이어에서는 새로운 연결을 열어서 이전에 보내던 패킷을 다시 보내게 된다. 패킷이 중복되거나 잃어버리지 않는 것을 보장한다. 만약 ULP에서 새로운 연결로 상태정보를 보내지 않거나 상태 타이머가 익스파이어되면 이 연결은 해제되어 버린다. 이 타이머의 ㄱ밧은 설정 가능한 값이다. 추천값은 1 sec 이다.

Broken connection handling
만약 다음과 같은 현상이 발생하면 RUDP 연결이 끊어졌다고 가정한다.
- 재전송 타이머가 익스파이어되고, 최대값을 초과하였을 때
- 서버의 널 세그먼트 타이머가 익스파이어 됬을 때

만약 위의 두경우중 하나가 발생하고 상태 타이머가 0이 아니면, ULP는 연결이 끊어졌음을 인지하고 API를 통하여 시그널을 날린 후에 전송 상태 타이머를 가동한다. 만악 이 타이머가 익스파이어 되면 오토 리셑이 가동된다. 전송 상태 타이머의 값이 0이 되면, ULP는 연결이 실패한 것으로 간주하고 리셑시킨다.

Retransmission Algorithm
재선송은 EACK 세그먼트를 받던가, 혹은 재전송 타이머가 익스파이어되면 수행된다. EACK 세그먼틀 수신하였을 경우 unacknowledged sent queue를 비운다. EACK 세그먼트로 부터 마지막 잘못된 시퀀스 번호와 필요한 ACK 넘버를 추출한다. 그리고 응답하지 않은 큐에 있던 것들을 재전송한다.

Signals to Upper Layer Protocol (ULP)
아래의 시그날들이 API를 통해 ULP로 전송될 수 있다. 이것들은 비동기 적인 이벤트로 ULP에 전달된다.
- 연결 열림 : 전송을 위한 연결이 생성되었음.
- 연결 거부 : 종료 대기상태가 아닌 상태에서 연결이 종료되었을 때
- 연결 종료 : 종료 대기상태에서 종료되었을 때
- 연결 실패 : 재전송 알고리즘에 의해 연결이 끊어진 것으로 판명 되었을 때
- 오토 리셋 : 데이터가 로스되어 새로운 연결을 열었을 때

Checksum Algorithm
16-bit one's complement of the one's complement

FEC
Forward Error Connect(FEC)에 대해 별도로 디파인된건 없다. 알라서 처리하라는 이야기다.

1.4 Feature Negotiation
클라이언트는 협상가능할 파라미터를 포함한 SYN 세그먼트를 전송하여 연결을 초기화 한다.
서버는 어셉트를 하고, 클라이언트가 보낸 파라미터를 그냥 수용하거나 다른 값이 있을 경우는 그 값을 넣어서 SYN에 ACK를 담아서 응답을 보낸다. 클라이언트는 서버가 보낸 파라미터를 선택하고 수락된 연결을 거부 한 후 RST를 날린다. 그리고 필요한 것을 취사선택 한 후 다시 연결을 한다. 오토 리셑의 경우에는 이

과정을 수행할 수 없다.

2.0 Future Potential Enhancements

RUDP는 앞으로 클라이언트/서버의 연결에 시메트릭 모드를 지원할 것이다. 이것은 양쪽 어느곳에서나 연결을 능동적으로 시작할 수 있다.

RUDP는 익스텐드 시퀀스와 엑크널리지 필드를 현재 1 옥텟에서 2 옥텟으로 확장 지원할 것이다. 이것은 전송 윈도우가 현재 255보다 더 커짐을 의미한다.

또한 네트워크를 좀더 효율 적으로 사용할 수 있도록 Nagle Algorithm을 사용할 수 있도록 연구중이다.

3.0 References

[1] Postel, J. (ed.), "Internet Protocol - DARPA Internet Program Protocol Specification", RFC 791, USC/Information Sciences Institute, September 1981.

[2] Postel, J., "User Datagram Protocol", RFC 768, USC/Information Sciences Institute, August 1980.

[3] Postel, J. (ed.), "Transmission Control Protocol", RFC 793, USC/Information Sciences Institute, September 1981.

[4] Velten, D., Hinden, R. and Sax, J., "Reliable Data Protocol", RFC 908, BBN Communications Corporation, July 1984.

[5] Partridge, C. and Hinden, R., "Version 2 of the Reliable Data Protocol", RFC 1151, BBN Corporation, April 1990.

[6] Braden, R., "Computing the Internet Checksum", RFC 1071, ISI, September 1988

[7] V. Jacobson, "Congestion Avoidance and Control," Computer Communication Review, Val. 18, no. 4, pp. 314-329, Aug. 1988.

[8] W. Stevens, RFC 2001 ?TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms?, January 1997

[9] V. Jacobson, "Modified TCP Congestion Avoidance Algorithm", April 30, 1990.

[10] Z. Wang, J. Crowcroft, A Dual-Window Model for Flow and Congestion Control, The Distributed Computing Engineering Journal, Institute of Physics/British Computer Society/IEEE, Vol 1, No 3, page 162-172, May 1994.

[11] Romanow, Allyn, "TCP over ATM: Some Performance Results", ATM Forum/93-784

4.0 Author's Addresses

Tom Bova
Cisco Systems
13615 Dulles Technology Drive
Herndon, VA 20171

Tel: +1-703-484-3331
Email: tbova@cisco.com

USA

Ted Krivoruchka
Cisco Systems
13615 Dulles Technology Drive
Herndon, VA 20171
USA

Tel: +1-703-484-3325
Email: tedk@cisco.com