

Anhang III zu den Vorlesungsunterlagen:

Beispiele aus dem RFC 3261 der IETF  
(SIP-Standardisierung)

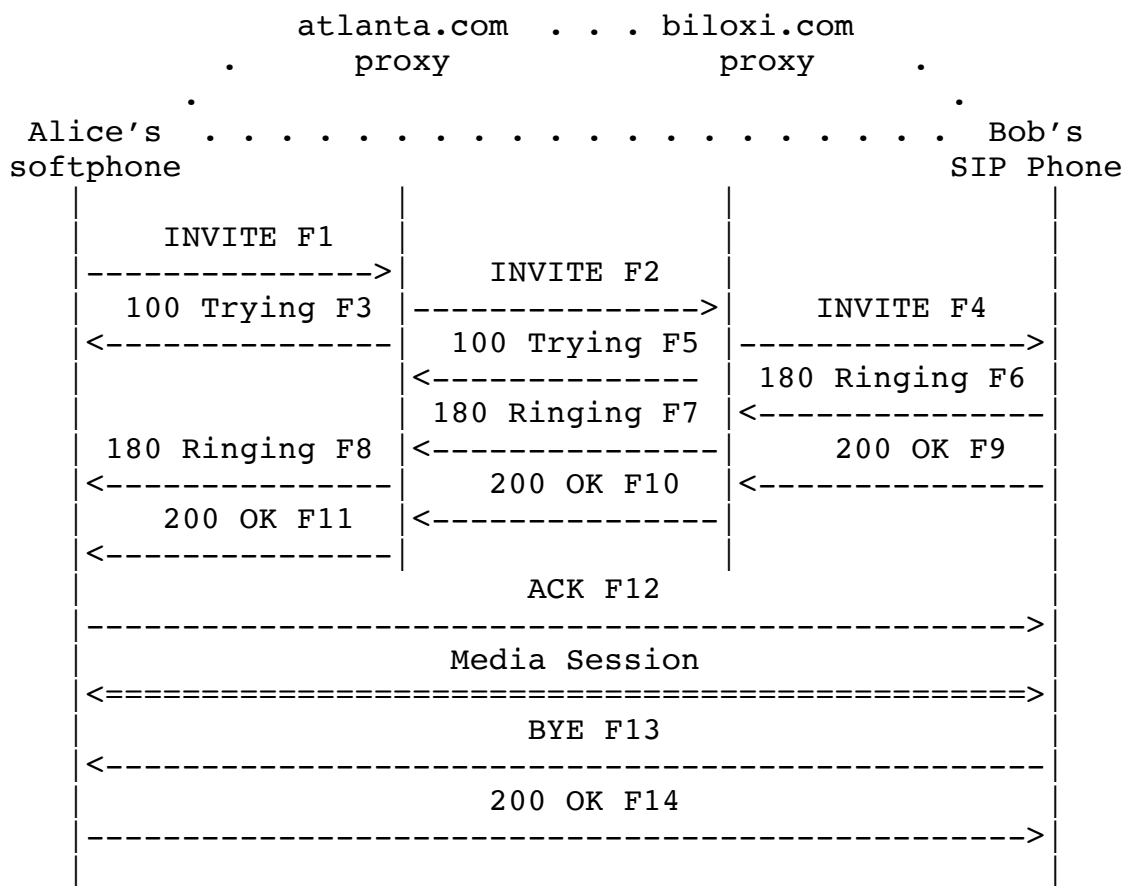


Figure 1: SIP session setup example with SIP trapezoid

```

INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bK776asdhds
Max-Forwards: 70
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710@pc33.atlanta.com
CSeq: 314159 INVITE
Contact: <sip:alice@pc33.atlanta.com>
Content-Type: application/sdp
Content-Length: 142
  
```

(Alice's SDP not shown)

The first line of the text-encoded message contains the method name (INVITE). The lines that follow are a list of header fields. This example contains a minimum required set. The header fields are briefly described below:

## Table of Contents

1	Introduction .....	8
2	Overview of SIP Functionality .....	9
3	Terminology .....	10
4	Overview of Operation .....	10
5	Structure of the Protocol .....	18
6	Definitions .....	20
7	SIP Messages .....	26
7.1	Requests .....	27
7.2	Responses .....	28
7.3	Header Fields .....	29
7.3.1	Header Field Format .....	30
7.3.2	Header Field Classification .....	32
7.3.3	Compact Form .....	32
7.4	Bodies .....	33
7.4.1	Message Body Type .....	33
7.4.2	Message Body Length .....	33
7.5	Framing SIP Messages .....	34
8	General User Agent Behavior .....	34
8.1	UAC Behavior .....	35
8.1.1	Generating the Request .....	35
8.1.1.1	Request-URI .....	35
8.1.1.2	To .....	36
8.1.1.3	From .....	37
8.1.1.4	Call-ID .....	37
8.1.1.5	CSeq .....	38
8.1.1.6	Max-Forwards .....	38
8.1.1.7	Via .....	39
8.1.1.8	Contact .....	40
8.1.1.9	Supported and Require .....	40
8.1.1.10	Additional Message Components .....	41
8.1.2	Sending the Request .....	41
8.1.3	Processing Responses .....	42
8.1.3.1	Transaction Layer Errors .....	42
8.1.3.2	Unrecognized Responses .....	42
8.1.3.3	Vias .....	43
8.1.3.4	Processing 3xx Responses .....	43
8.1.3.5	Processing 4xx Responses .....	45
8.2	UAS Behavior .....	46
8.2.1	Method Inspection .....	46
8.2.2	Header Inspection .....	46
8.2.2.1	To and Request-URI .....	46
8.2.2.2	Merged Requests .....	47
8.2.2.3	Require .....	47
8.2.3	Content Processing .....	48
8.2.4	Applying Extensions .....	49
8.2.5	Processing the Request .....	49

8.2.6	Generating the Response .....	49
8.2.6.1	Sending a Provisional Response .....	49
8.2.6.2	Headers and Tags .....	50
8.2.7	Stateless UAS Behavior .....	50
8.3	Redirect Servers .....	51
9	Canceling a Request .....	53
9.1	Client Behavior .....	53
9.2	Server Behavior .....	55
10	Registrations .....	56
10.1	Overview .....	56
10.2	Constructing the REGISTER Request .....	57
10.2.1	Adding Bindings .....	59
10.2.1.1	Setting the Expiration Interval of Contact Addresses .....	60
10.2.1.2	Preferences among Contact Addresses .....	61
10.2.2	Removing Bindings .....	61
10.2.3	Fetching Bindings .....	61
10.2.4	Refreshing Bindings .....	61
10.2.5	Setting the Internal Clock .....	62
10.2.6	Discovering a Registrar .....	62
10.2.7	Transmitting a Request .....	62
10.2.8	Error Responses .....	63
10.3	Processing REGISTER Requests .....	63
11	Querying for Capabilities .....	66
11.1	Construction of OPTIONS Request .....	67
11.2	Processing of OPTIONS Request .....	68
12	Dialogs .....	69
12.1	Creation of a Dialog .....	70
12.1.1	UAS behavior .....	70
12.1.2	UAC Behavior .....	71
12.2	Requests within a Dialog .....	72
12.2.1	UAC Behavior .....	73
12.2.1.1	Generating the Request .....	73
12.2.1.2	Processing the Responses .....	75
12.2.2	UAS Behavior .....	76
12.3	Termination of a Dialog .....	77
13	Initiating a Session .....	77
13.1	Overview .....	77
13.2	UAC Processing .....	78
13.2.1	Creating the Initial INVITE .....	78
13.2.2	Processing INVITE Responses .....	81
13.2.2.1	1xx Responses .....	81
13.2.2.2	3xx Responses .....	81
13.2.2.3	4xx, 5xx and 6xx Responses .....	81
13.2.2.4	2xx Responses .....	82
13.3	UAS Processing .....	83
13.3.1	Processing of the INVITE .....	83
13.3.1.1	Progress .....	84
13.3.1.2	The INVITE is Redirected .....	84

13.3.1.3	The INVITE is Rejected .....	85
13.3.1.4	The INVITE is Accepted .....	85
14	Modifying an Existing Session .....	86
14.1	UAC Behavior .....	86
14.2	UAS Behavior .....	88
15	Terminating a Session .....	89
15.1	Terminating a Session with a BYE Request .....	90
15.1.1	UAC Behavior .....	90
15.1.2	UAS Behavior .....	91
16	Proxy Behavior .....	91
16.1	Overview .....	91
16.2	Stateful Proxy .....	92
16.3	Request Validation .....	94
16.4	Route Information Preprocessing .....	96
16.5	Determining Request Targets .....	97
16.6	Request Forwarding .....	99
16.7	Response Processing .....	107
16.8	Processing Timer C .....	114
16.9	Handling Transport Errors .....	115
16.10	CANCEL Processing .....	115
16.11	Stateless Proxy .....	116
16.12	Summary of Proxy Route Processing .....	118
16.12.1	Examples .....	118
16.12.1.1	Basic SIP Trapezoid .....	118
16.12.1.2	Traversing a Strict-Routing Proxy .....	120
16.12.1.3	Rewriting Record-Route Header Field Values .....	121
17	Transactions .....	122
17.1	Client Transaction .....	124
17.1.1	INVITE Client Transaction .....	125
17.1.1.1	Overview of INVITE Transaction .....	125
17.1.1.2	Formal Description .....	125
17.1.1.3	Construction of the ACK Request .....	129
17.1.2	Non-INVITE Client Transaction .....	130
17.1.2.1	Overview of the non-INVITE Transaction .....	130
17.1.2.2	Formal Description .....	131
17.1.3	Matching Responses to Client Transactions .....	132
17.1.4	Handling Transport Errors .....	133
17.2	Server Transaction .....	134
17.2.1	INVITE Server Transaction .....	134
17.2.2	Non-INVITE Server Transaction .....	137
17.2.3	Matching Requests to Server Transactions .....	138
17.2.4	Handling Transport Errors .....	141
18	Transport .....	141
18.1	Clients .....	142
18.1.1	Sending Requests .....	142
18.1.2	Receiving Responses .....	144
18.2	Servers .....	145
18.2.1	Receiving Requests .....	145

18.2.2	Sending Responses .....	146
18.3	Framing .....	147
18.4	Error Handling .....	147
19	Common Message Components .....	147
19.1	SIP and SIPS Uniform Resource Indicators .....	148
19.1.1	SIP and SIPS URI Components .....	148
19.1.2	Character Escaping Requirements .....	152
19.1.3	Example SIP and SIPS URIs .....	153
19.1.4	URI Comparison .....	153
19.1.5	Forming Requests from a URI .....	156
19.1.6	Relating SIP URIs and tel URLs .....	157
19.2	Option Tags .....	158
19.3	Tags .....	159
20	Header Fields .....	159
20.1	Accept .....	161
20.2	Accept-Encoding .....	163
20.3	Accept-Language .....	164
20.4	Alert-Info .....	164
20.5	Allow .....	165
20.6	Authentication-Info .....	165
20.7	Authorization .....	165
20.8	Call-ID .....	166
20.9	Call-Info .....	166
20.10	Contact .....	167
20.11	Content-Disposition .....	168
20.12	Content-Encoding .....	169
20.13	Content-Language .....	169
20.14	Content-Length .....	169
20.15	Content-Type .....	170
20.16	CSeq .....	170
20.17	Date .....	170
20.18	Error-Info .....	171
20.19	Expires .....	171
20.20	From .....	172
20.21	In-Reply-To .....	172
20.22	Max-Forwards .....	173
20.23	Min-Expires .....	173
20.24	MIME-Version .....	173
20.25	Organization .....	174
20.26	Priority .....	174
20.27	Proxy-Authenticate .....	174
20.28	Proxy-Authorization .....	175
20.29	Proxy-Require .....	175
20.30	Record-Route .....	175
20.31	Reply-To .....	176
20.32	Require .....	176
20.33	Retry-After .....	176
20.34	Route .....	177

20.35	Server .....	177
20.36	Subject .....	177
20.37	Supported .....	178
20.38	Timestamp .....	178
20.39	To .....	178
20.40	Unsupported .....	179
20.41	User-Agent .....	179
20.42	Via .....	179
20.43	Warning .....	180
20.44	WWW-Authenticate .....	182
21	Response Codes .....	182
21.1	Provisional 1xx .....	182
21.1.1	100 Trying .....	183
21.1.2	180 Ringing .....	183
21.1.3	181 Call Is Being Forwarded .....	183
21.1.4	182 Queued .....	183
21.1.5	183 Session Progress .....	183
21.2	Successful 2xx .....	183
21.2.1	200 OK .....	183
21.3	Redirection 3xx .....	184
21.3.1	300 Multiple Choices .....	184
21.3.2	301 Moved Permanently .....	184
21.3.3	302 Moved Temporarily .....	184
21.3.4	305 Use Proxy .....	185
21.3.5	380 Alternative Service .....	185
21.4	Request Failure 4xx .....	185
21.4.1	400 Bad Request .....	185
21.4.2	401 Unauthorized .....	185
21.4.3	402 Payment Required .....	186
21.4.4	403 Forbidden .....	186
21.4.5	404 Not Found .....	186
21.4.6	405 Method Not Allowed .....	186
21.4.7	406 Not Acceptable .....	186
21.4.8	407 Proxy Authentication Required .....	186
21.4.9	408 Request Timeout .....	186
21.4.10	410 Gone .....	187
21.4.11	413 Request Entity Too Large .....	187
21.4.12	414 Request-URI Too Long .....	187
21.4.13	415 Unsupported Media Type .....	187
21.4.14	416 Unsupported URI Scheme .....	187
21.4.15	420 Bad Extension .....	187
21.4.16	421 Extension Required .....	188
21.4.17	423 Interval Too Brief .....	188
21.4.18	480 Temporarily Unavailable .....	188
21.4.19	481 Call/Transaction Does Not Exist .....	188
21.4.20	482 Loop Detected .....	188
21.4.21	483 Too Many Hops .....	189
21.4.22	484 Address Incomplete .....	189

#### 8.1.1.3 From

The From header field indicates the logical identity of the initiator of the request, possibly the user's address-of-record. Like the To header field, it contains a URI and optionally a display name. It is used by SIP elements to determine which processing rules to apply to a request (for example, automatic call rejection). As such, it is very important that the From URI not contain IP addresses or the FQDN of the host on which the UA is running, since these are not logical names.

The From header field allows for a display name. A UAC SHOULD use the display name "Anonymous", along with a syntactically correct, but otherwise meaningless URI (like sip:thisis@anonymous.invalid), if the identity of the client is to remain hidden.

Usually, the value that populates the From header field in requests generated by a particular UA is pre-provisioned by the user or by the administrators of the user's local domain. If a particular UA is used by multiple users, it might have switchable profiles that include a URI corresponding to the identity of the profiled user. Recipients of requests can authenticate the originator of a request in order to ascertain that they are who their From header field claims they are (see Section 22 for more on authentication).

The From field MUST contain a new "tag" parameter, chosen by the UAC. See Section 19.3 for details on choosing a tag.

For further information on the From header field, see Section 20.20. Examples:

```
From: "Bob" <sips:bob@biloxi.com> ;tag=a48s
From: sip:+12125551212@phone2net.com;tag=887s
From: Anonymous <sip:c8oqz84zk7z@privacy.org>;tag=hyh8
```

#### 8.1.1.4 Call-ID

The Call-ID header field acts as a unique identifier to group together a series of messages. It MUST be the same for all requests and responses sent by either UA in a dialog. It SHOULD be the same in each registration from a UA.

In a new request created by a UAC outside of any dialog, the Call-ID header field MUST be selected by the UAC as a globally unique identifier over space and time unless overridden by method-specific behavior. All SIP UAs must have a means to guarantee that the Call-ID header fields they produce will not be inadvertently generated by any other UA. Note that when requests are retried after certain



failure responses that solicit an amendment to a request (for example, a challenge for authentication), these retried requests are not considered new requests, and therefore do not need new Call-ID header fields; see Section 8.1.3.5.

Use of cryptographically random identifiers (RFC 1750 [12]) in the generation of Call-IDs is RECOMMENDED. Implementations MAY use the form "localid@host". Call-IDs are case-sensitive and are simply compared byte-by-byte.

Using cryptographically random identifiers provides some protection against session hijacking and reduces the likelihood of unintentional Call-ID collisions.

No provisioning or human interface is required for the selection of the Call-ID header field value for a request.

For further information on the Call-ID header field, see Section 20.8.

Example:

Call-ID: f81d4fae-7dec-11d0-a765-00a0c91e6bf6@foo.bar.com

#### 8.1.1.5 CSeq

The CSeq header field serves as a way to identify and order transactions. It consists of a sequence number and a method. The method MUST match that of the request. For non-REGISTER requests outside of a dialog, the sequence number value is arbitrary. The sequence number value MUST be expressible as a 32-bit unsigned integer and MUST be less than  $2^{31}$ . As long as it follows the above guidelines, a client may use any mechanism it would like to select CSeq header field values.

Section 12.2.1.1 discusses construction of the CSeq for requests within a dialog.

Example:

CSeq: 4711 INVITE