

MINIATURE SIP FOR EMBEDDED SYSTEMS

Leonardo Maccari Rufino

Antônio Augusto Fröhlich

`{leonardo,guto}@lisha.ufsc.br`

`http://www.lisha.ufsc.br/~guto`

October 14, 2010

1. Introduction



- Session Initiation Protocol (SIP):
 - Application layer protocol;
 - Responsible for creating, modifying, and terminating a session;
 - Establishes calls through networks via IP protocol;
 - Widely used in embedded systems;
 - Problem: resource-constrained embedded systems.

1. Introduction



- Proposal:
 - Adaption of the SIP protocol in order to enable its utilization in resource-constrained embedded systems.
 - Target: embedded systems do not have one person controlling it.
 - Does not use some features, requests, and headers fields present in a full SIP version.
 - Nevertheless, the modifications have not compromised the protocol functionality.
 - Uses the Embedded Parallel Operating System (EPOS).

2. Proposed Implementation for the SIP Protocol

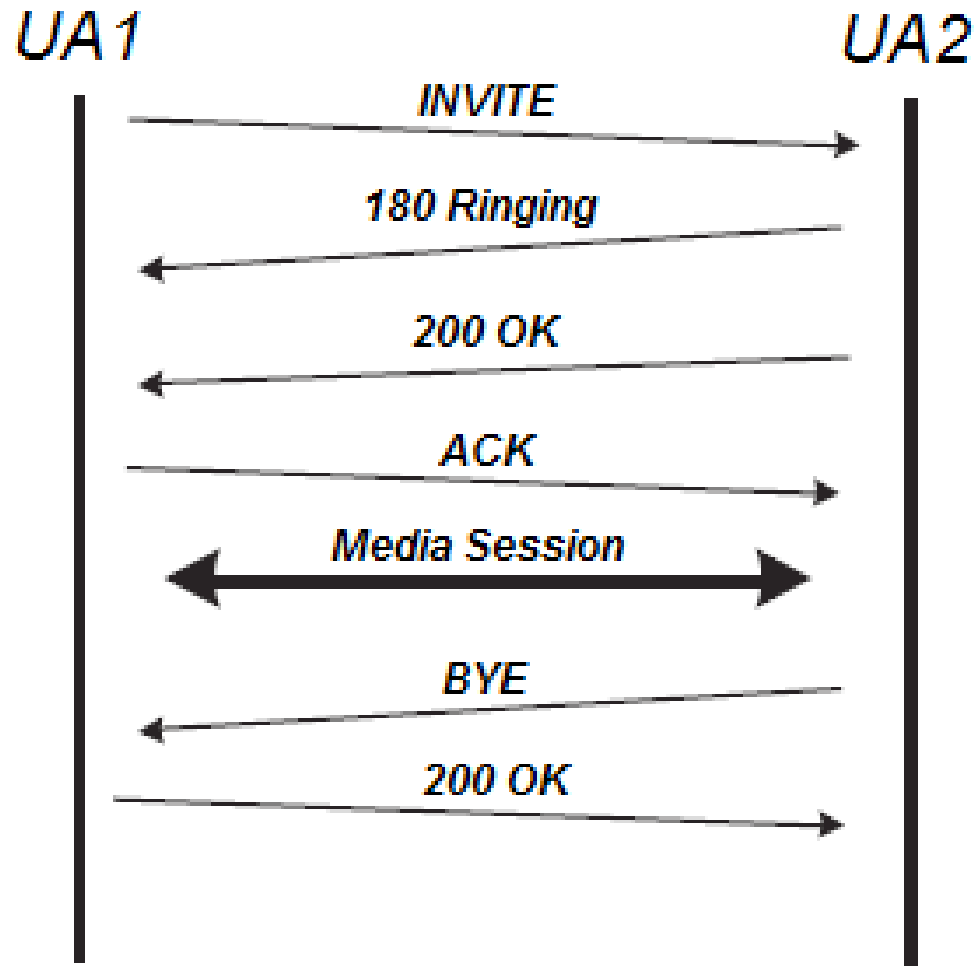


- In order to obtain a miniature version of the SIP protocol able to run in a resource-constrained embedded system, without compromising its functionalities, only a subset of request and header fields were used.

2.1 Requests



- Implemented:
 - *INVITE*;
 - *ACK*;
 - *BYE*.



2.1 Requests

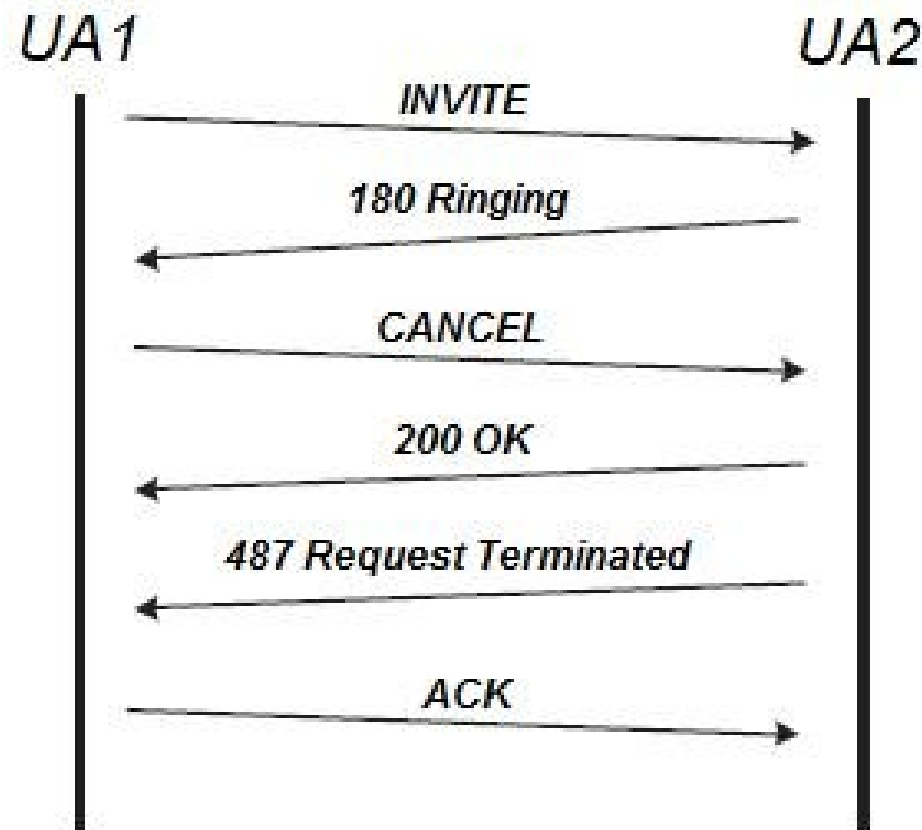


- Not implemented:
 - *CANCEL*;
 - *OPTIONS*;
 - *REGISTER*;
 - Located in separate *RFCs*:
 - *REFER, SUBSCRIBE, NOTIFY, MESSAGE, UPDATE, INFO, PRACK, ...*

2.1 Requests



- **CANCEL:**
 - Used to cancel a request previously sent by a UAC (User Agent Client).



2.1 Requests



- *CANCEL*:
 - When an invitation to start the session comes from the embedded system:
 - There is no need to cancel the attempt call
 - When the attempt is made by the opposite side:
 - As the system has no user, there is no need to send provisional responses (e.g. ringing) and the connection should be accepted or rejected instantly.
 - Then, the *CANCEL* method can be suppressed without impact on functionality:
 - It has no effect on a request to which a UAS (User Agent Server) has already sent a final response;
 - Being applicable in situations where the server side can take a long time to respond to a request.

2.1 Requests



- *OPTIONS*:
 - Used to query a UA (User Agent) or proxy server about their capabilities without the need to place a call with the other party.
 - Example: before a UAC sends an *INVITE* message with the *Require* header field, it can send an *OPTIONS* to make sure that the UAS supports the required fields.
 - This sequence of two messages, *OPTIONS* and *INVITE*, has the same effect of sending *INVITE* twice.
 - In this situation, the first invitation requires a few options.
 - However, if the called party does not support the required extensions, the standard procedure is to reject the request, sending a header field explaining the reason.
 - Therefore, a second *INVITE* can be generated with only the options allowed by the UAS.
 - Wherefore, this request can be removed without any problem.

2.1 Requests



- **REGISTER:**

- Used by a UA to notify a SIP network of its *Contact* URI and a URI which should have requests routed to this contact.
- This work suppressed this method because each UA has a fixed IP address.
- Then, the message exchanges are sent directly to the IPs of end-points desired, without the use of servers.



2.2 Header Fields



- The SIP protocol has 44 header fields described in its standard.
- Among these, only some have been used in this work:
 - *Allow, Call-ID, Contact, Content-Disposition, Content-Length, Content-Type, CSeq, From, Max-Forwards, Record-Route, Require, Route, To, Unsupported, and Via.*

2.2 Header Fields



- All 29 remaining header fields were excluded.
- A major reason for the exclusion is that a portion of them only convey information about a user to another, however there is no need to exchange these data.
 - *Accept-Language, Alert-Info, Call-Info, Content-Language, Date, Error-Info, Organization, Priority, Retry-After, Server, Subject, Timestamp, User-Agent, and Warning.*

2.2 Header Fields



- Some have been deleted from the miniature version because there is no available encoding for them, in other words, their fields would be empty if necessary send them.
 - *Accept, Accept-Encoding, Content-Encoding, MIME-Version, Proxy-Require, and Supported.*

2.2 Header Fields



- This study does not require authentication schemes because it is designed for confined systems, not connected to the Internet.
- Consequently, some headers were excluded.
 - *Authentication-Info, Authorization, Proxy-Authenticate, Proxy-Authorization, and WWW-Authenticate.*

2.2 Header Fields



- The remaining header fields were removed for their own reasons.
- *Expires:*
 - Indicates the time that the message contents is valid, since after a response this field no longer has meaning.
 - As the responses are sent immediately on receiving a request, without having to wait for user actions, this header does not apply.
- *In-Reply-To:*
 - Enumerates the Call-IDs that a call references or returns.

2.2 Header Fields



- The remaining header fields were removed for their own reasons.
- *Min-Expires*:
 - Transmits the minimum expiration time supported by a registrar server in response to a request *REGISTER*.
 - As this work does not use this method, this header also becomes unnecessary.
- *Reply-To*:
 - Specifies a URI that should be used in response to a request.
 - As we are sending messages from fixed IP addresses, there is no need for a system has more than one URI identifying it.

2.2 Header Fields



- The miniature SIP version was implemented in the Embedded Parallel Operating System (EPOS).
 - Small footprint;
 - Complete communication stack, including UDP (used in the transport layer);
 - Support for sensor networks;
 - Without user interaction.
- Hence, with all these changes made in the original version of the SIP protocol, there is a great economy of memory, achieving a final system image with a small size.
- Then, this proposed implementation can be used even in the smallest and simplest embedded systems present today.

3. Results

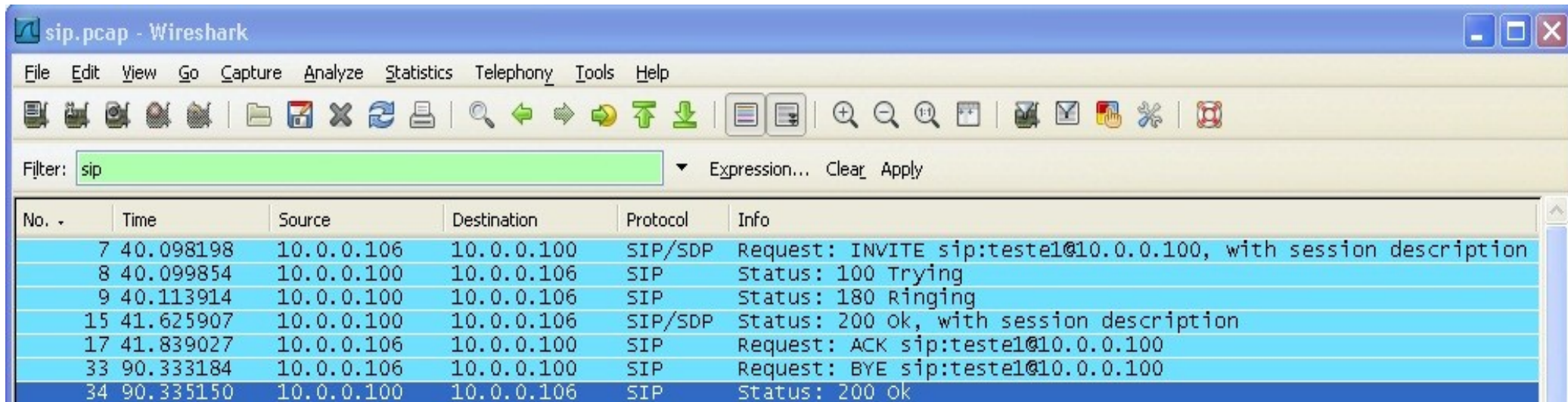


Section	SIP	libosip	Asterisk
.text	81,074	232,914	609,678
.data	6,360	3,220	7,912
.bss	16	1,172	67,972
TOTAL	87,450	237,306	685,562

Size of SIP implementations in bytes

- The final image size, with EPOS and SIP version, reached 118,532 (116 Kbytes), much less than the others, even without considering the several megabytes of operating system overhead.

3. Results

A screenshot of the Wireshark network protocol analyzer interface. The title bar reads "sip.pcap - Wireshark". The menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Tools, and Help. The toolbar contains various icons for file operations, capture, and analysis. The filter bar shows "sip" in a green box. The packet list table below shows a sequence of SIP messages between 10.0.0.106 and 10.0.0.100.

No.	Time	Source	Destination	Protocol	Info
7	40.098198	10.0.0.106	10.0.0.100	SIP/SDP	Request: INVITE sip:teste1@10.0.0.100, with session description
8	40.099854	10.0.0.100	10.0.0.106	SIP	Status: 100 Trying
9	40.113914	10.0.0.100	10.0.0.106	SIP	Status: 180 Ringing
15	41.625907	10.0.0.100	10.0.0.106	SIP/SDP	Status: 200 ok, with session description
17	41.839027	10.0.0.106	10.0.0.100	SIP	Request: ACK sip:teste1@10.0.0.100
33	90.333184	10.0.0.106	10.0.0.100	SIP	Request: BYE sip:teste1@10.0.0.100
34	90.335150	10.0.0.100	10.0.0.106	SIP	Status: 200 ok

Initialize and finalize a session with SIP in Wireshark

- EPOS operating system with the SIP implemented, running on VirtualBox virtual machine.
 - IP: 10.0.0.106
- A machine running Windows XP operating system and X-Lite SIP phone.
 - IP: 10.0.0.100

4. Conclusion



- Adaptations to the SIP protocol to make it feasible in resource-constrained embedded systems.
 - Subset of header and request;
 - Without compromising its functionalities.
- SIP miniature version consumes less than 120Kb of code and data.
- It is able to communicate to any SIP phone, starting and ending a session.
- Future work:
 - Implement the developed version in a real embedded system;
 - To study other necessary protocols for communication, such as RTP.