Model Checking the SIP Protocol with Spin

– Course Project *Model Checking of Software*, WS 2017/18 –

Prof. Stefan Leue
University of Konstanz
Stefan.Leue@uni-konstanz.de

1 Overview

Through this mini-project, the participants of the Model Checking of Software course are supposed to gain some practical experienc with state-of-the-art model checkers in modeling and verifying concurrent systems. Our choice of the system to be modeled is Session Initiation Protocol (SIP), a communication protocol for establishing, maintaining, and terminating multimedia sessions such as those used by Voice over Internet applications. Participants will use Spin, a popular explicit state model checker, and its input language Promela to (1) model certain aspects of the SIP protocol, to (2) specify high level properties that the model should satisfy, and to (3) verify whether the specified properties are indeed satisfied.

In this short note, we will briefly introduce the Spin model checker (Sec. 2) and the SIP protocol (Sec. 3). Organizational issues will also be discussed in the remainder of the note.

2 SPIN

The Spin model checker [2] can be used to simulate and verify concurrent systems written in its input language Promela. A Promela model consists of a set of `proctype` definitions. Each proctype is a class of processes whose instances run autonomously and concurrently when the model is executed. Instances of a proctype can be created statically or dynamically at run time. Processes communicate with each other via either shared variables, or synchronous rendez-vous communications, or exchanging messages over bounded message buffers. Figure 1 shows a simple example of a Promela model where two clients attemp to send files to one another using the only bus available for one client at a time. The exclusive access to the bus is achieved by synchronzing with the server using two signals `request` and `release`. Interested readers may use the random or guided simulation features of Spin to check if the model satisfies certain properties, such as (1) whether the access to the bus is granted to only one client at a time; or (2) whether every client will eventually obtain the bus access; or (3) whether there may be a deadlock, in which case the server and both clients will wait for signals from each other forever and thus be unable to execute further.

```
mtype = {request, finish, file};

chan toServer = [0] of {mtype};
chan buf = [5] of {mtype};

proctype client(byte cid){
  do
  :: toServer!request ->
      empty(buf) ->
        do
        :: buf!file;
        :: break;
        od;
      toServer!finish;
  :: buf?file;
  od;
}

active proctype server(){
  do
  :: toServer?request -> toServer?finish;
  od;
}

init{
  atomic{run client(0); run client(1)}
}
```

Figure 1: A Promela model.

The Spin model checker can be downloaded for free at the following link:

`http://spinroot.com/spin/Man/README.html`

Windows users can directly use the precompiled executable. Excutables for Linux systems are also available, although we advise to compile the source code on your machine, which can be as simple as executing `make` without changing a single line of the makefile. You need a C/C++ compiler universally available (by adding the locatioin to the *PATH* environment variable) in order to run Spin.

You may also want to use XSpin or iSpin (new GUI for Spin), a graphic user interface for the model checker, written as a Tcl/Tk script. For this purpose, you may need to download and install software to interpret Tcl/Tk, such as ActiveTcl on Windows. See detailed instructions on the Spin Website. Special attention is needed when making necessary changes to the XSpin script, e.g., to change the location of the script, the location of the C/C++ compiler, etc.

In case you encounter troubles during the installation of the software, please feel free to contact the tutor.

3 SIP

SIP is a client-server based communication protocol used to set up and tear down peer-to-peer multimedia sessions. A SIP user, called a user agent, can behave both as a client or a server even at a same time: When the user initiates a session by sending requests, the user acts as a client. Otherwise, the user acts as a server and sends back response messages. Figure 2 depicts one typical scenario, among greatly many others, in which a media session is sucessfully setup and eventually torn down. The arrows between user agents and proxies represent exchanged messages of different types serving different purposes. More detail of the SIP protocol can be found in the respective RFC specification [1].

SIP has a layered structure as shown in Figure 3. In the project, we should concentrate on the interactions between user agents. Therefore, we shall take an approach to abstract from all other aspects such as encoding, routing, among others, and keep the focus on the two highest layers, namely, the transaction user and the transaction layer.

The role of the transaction user is to initiate a session, cancel initiating a session, to respond to requests from other users, or to terminate sessions, etc. Each request together with its ensuing responses constitutes a transaction, which is managed by the transaction layer at both the client side and the server side.

There are two kinds of communicating parties in the SIP protocol: user agents and proxies. Proxies may be stateful or stateless. A stateful proxy has also a transaction layer and remembers the state of any particular trans-

```
                     atlanta.com  . . . biloxi.com
                 .      proxy              proxy     .
               .                                       .
       Alice's  . . . . . . . . . . . . . . . . . . . .  Bob's
      softphone                                        SIP Phone
         |                |                |                |
         |    INVITE F1   |                |                |
         |--------------->|    INVITE F2   |                |
         |  100 Trying F3 |--------------->|    INVITE F4   |
         |<---------------|  100 Trying F5 |--------------->|
         |                |<-------------- | 180 Ringing F6 |
         |                | 180 Ringing F7 |<---------------|
         | 180 Ringing F8 |<---------------|     200 OK F9  |
         |<---------------|    200 OK F10  |<---------------|
         |    200 OK F11  |<---------------|                |
         |<---------------|                |                |
         |                       ACK F12                    |
         |------------------------------------------------->|
         |                   Media Session                  |
         |<================================================>|
         |                       BYE F13                    |
         |<-------------------------------------------------|
         |                     200 OK F14                   |
         |------------------------------------------------->|
         |                                                  |
```

Figure 2: A SIP session setup example.

```
---------------------------
|    Transaction User     |
---------------------------
|    Transaction Layer    |
---------------------------
|     Transport Layer     |
---------------------------
|     Syntax/Encoding     |
---------------------------
```

Figure 3: The layered structure of the SIP protocol.

action. A stateless proxy, on the contrary, simply forwards messages from one side to another. In the project, we require only the modeling of stateless proxies. The modeling of stateful proxies is optional.

Participants may assume the SIP protocol to run on reliable communication connections such as those using TCP/IP. Modeling with regard to UDP connections is only optional.

4 Organizational Issues

4.1 Requirements

Modeling. Your Promela model of the SIP protocol must consist of at least two user agents and two proxies. Our main focus is to model all the scenarios related to setting up sessions and terminating sessions. You must model both the client behavior and the server behavior, including

- Client Behavior
    - sending the requests INVITE, ACK and CANCEL for setting up sessions
    - sending the request BYE for terminating sessions
- Server Behavior
    - sending corresponding resposes to requests of client including:
        - accepting a request
        - rejecting a request
        - server error to fulfill a request

Note that a user agent should be able to act as a client or a server at a same time. The behavior of the transaction user layer must be modeled, including client transactions and server transactions. Any user agent should be able to have multiple sessions at a time. Proxies may be stateless.

The optional aspects of the SIP protocol include (1) registration, (2) dialogs (3) querying, (4) modifying sessions, (5) stateful proxies, (6) transport layer behavior, (7) forking and merging requests, among others irrelavant to the above mandatory aspects to be modeled.

Properties. Suggest some high level properties that you want your model to satisfy. These must include at least two safety properties and two liveness properties. Simulate your model using Spin to check whether the properties that you specify may be violated.

Verification. Express your suggested properties in LTL formulae. Use Spin to model check these properties. In case a property is violated, analyze the obtained counterexample and identify the cause of the property violation. Fix the problem by modifying your model, and then verify the property again on the modified model. Repeat this procedure until your model satisfies the property.

Report. Write a report to describe how you conduct the verification of your model, and how you eliminate errors in your implementation in case you obtain property violating counterexamples.

4.2 Groups

Participants may form groups of up to two persons.

4.3 Submissions

The required submissions include (1) the model that you implement, (2) the suggested properties expressed both in natural languages (English) and in LTL, and (3) the verification report. Each group needs only to submit one report. The following are the submission deadlines:

Deadline	**Submission**
December 19	The model that you implement
January 16	The properties
February 6	The final project report, including verification results

During the tutorials, at regular intervals there will be a check of the progress made by each group in modeling the protocol. The first session for checking the progress will be **December 6**. Each group must explain their model having been built so far, report on challenges and difficulties that they will already have confronted and overcome, and suggest their plan to the future implementation.

4.4 Try to understand the following key concepts

- User Agent Client
- User Agent Server
- Transaction

- Session
- Dialog
- Requests
- Responses
- Stateless Proxy
- Initiating a Session
- Canceling a Request
- Terminating a Session

References

[1] SIP: Session Initiation Protocol, RFC 3261. Available at `http://www.rfc.net/rfc3261.html`.

[2] G. Holzmann. *The Spin model checker: Primer and reference manual.* Addison-Wesley, 2004.