PlusCAL Ticketing System

The goal of this project several-fold, which part building on top of the other. Create a subfolder per sub-question to keep them separated.

Part 1: Verifying a honest system: 4 pts

In the first part you need to analyze the given model, describing a simplified ticket system between a honest server and several honest clients. There are several flaws in the model that need to be fixed and a part of the client needs to be written before it can be effectively used to analyze the system.

You need to modify it in order to allow: - The clients to buy tickets using their initial money - Change the specification / apply additional restrictions such that the system eventually stops. That is from some point on, all clients are satisfied with the tickets they have and will neither seek to buy additional tickets, nor will they seek refund for the ones they already have.

You need to ensure proper execution, by establishing invariants and temporal properties. For instance, with all honest partners, every client should end up in a situation where the number of tickets plus his remaining money equals the initial money. Depending on the changes on the model, it might be a good idea to separate the definition of how your system behaves and the properties into two separate files.

Also try to make the model more realistic with respect to client behavior.

If you can, try to reduce the complexity of the model (the number of states) without modifying its "the interesting part of the behavior". If you do this, please add a short description to your report.

Part 2: Introducing a malicious client: 4 pts

Now you need to change the specification by introducing a malicious client. This client does not seek to acquire tickets, but simply wants to find ways to scam the system. In this part you may not change the honest clients, nor the server. Only introduce a new process that uses the same communication routes as the honest client, but chooses the content of the messages such that he ends up with more money than he initially has. Bank IDs and IPs of honest clients are hidden to the malicious client, he may only interact with the server.

Part 3: Fixing and extending the protocol : 12 pts

Security 6pts

Change the communication protocol and the message sequence between the (honest) clients and the server such that they are safe. That is the malicious client from part 2 no longer succeeds with his scam, and moreover there can not be any other easy attack path.

Explain and justify your design choices in the report (and possibly in the comments); Note which variables are considered a secret or publicly available.

“Proof” that your approach works by modelling a large variety of client behaviors.

Extending the protocol 6pts

Introduce additional steps in the protocol / make the protocol more expressive

- Allow for cancellation / reservations of sets of seats
- Introduce additional steps requesting the list of available seats to avoid asking for seats that are already taken (at least most of the time)
- Introduce a “reservation” step before payment
- Allow users to change ip address without compromising security
- Bonus: Allow for messages to get lost without loosing correctness or safety

Deliverables

You need to hand in the (documented) code and a short report (<3 pages) given a rough explanations of the features you have added or changed as a git repo. For your invariants for instance, write down in plain english (or french) what you wish to check with it. All of this needs to be contained in one folder that also holds the .git subfolder showing the temporal evolution of your project and who did what.