HTTP Communications: A Case Study of Online Captures with PTTAC

Azahara CAMACHO*, Mercedes G. MERAYO and Manuel NÚÑEZ
Departamento de Sistemas Informáticos y Computación, Universidad Complutense de Madrid, Madrid, Spain
*Corresponding author

Keywords: PTTAC, Passive testing, HTTP, Communication protocol.

Abstract. This paper presents the process of testing a typical scenario of an HTTP communication between a regular user of Internet and a server. For this experiment we have used the Passive Testing Tool for Asynchronous Communications (PTTAC). This tool automatizes a formal framework to perform passive testing for systems where there is an asynchronous communications channel between the tester and the system under test. In this work, we show the problem that motivated this research, the main features of the tool and a case study where we have used PTTAC for testing a communication protocol.

Introduction

Communication protocols are the central piece of message transmission. As a collection of rules, they are able to establish the order of the data flow between different elements, regardless the operating system, the application or the type of device used for the connection. However, their application imply security problems, such as interception and exploitation. In order to prevent these problems, it is necessary the use of techniques that allow us to check the protection of the communications. It is widely recognized that testing methodologies help to check the systems by detecting errors in an effective way.

In testing we can distinguish between two main approaches: active and passive. Active testing requires the interaction of the tester with the System Under Test (SUT). Nevertheless, the most of the systems avoid the access to them owing to security issues or because the system running 24/7. Under these conditions an active testing approach cannot be applied. In this cases we need to use a passive testing paradigm that can be used to check the behavior of a system during its operation. Formal passive testing is already a well-established line of research and several extensions of the original frameworks [1, 2, 3] have dealt with issues such as security and time [4, 5, 6]. For these reasons, these are the most appropriate methodologies to evaluate the correctness of communications.

However, there are few interactive tools which apply these techniques automatically. Its manual application is a complex and error prone process and therefore, the effectiveness of these approaches is reduced. In this paper we analyze the use of PTTAC a tool that automatizes a passive testing technique [7]. It is based on the formal passive testing framework for systems in which there is an asynchronous communications channel between the tester and the system [8]. It can be applied in real-time which implies an important benefit: the detection of an error can be immediately notified to the operators of the system and it allows that appropriate measures can be taken.

With the aim of highlighting the relevance of this tool, we performed a case study in which the communication between two actors is analyzed. This is managed according to the rules of the HTTP protocol [9]. In order to evaluate the efficiency of the automation of the tool and the underlying testing technique, we have checked a set of traces in which we injected different faults. They simulate the effect of a possible attack from an external actor during the communication. This attack could be perpetrated by an unauthorized sniffer that gathers private information while the authorized actors do not detect it.

The rest of the paper is structured as follows. In Section II we introduce PTTAC and a brief description of the framework in what it is based on. In Section III we present the case study with the
actors and the requirements to be checked with the tool. Finally, Section IV presents our conclusions and provides some lines for future work.

**PTTAC – Passive Testing Tool for Asynchronous Communications**

In this section we describe the core of our work: the framework and the tool that has been used in the case study of this paper.

**Framework**

In order to understand the features of the tool, we present a brief description of the framework that implements. In this approach, a trace being observed is checked in order to determine if it satisfies a property or a group of properties that are required to be fulfilled by the SUT. The proposed technique considers that there exists an asynchronous channel/network between the monitor and the SUT. It makes that the monitor might not directly observe the interface of the SUT and, therefore, there can be differences between the trace observed and the trace produced by the SUT: outputs can be delayed. To manage this situation, all the possible observations of the original trace are captured by an automaton [8] and its application to the observed trace allow us to detect any error.

**Tool**

PTTAC is a Java GUI which implements the previous framework. This tool presents four different functionalities:

- **Properties Manager**, it allows the user to manage (create, update, delete) properties associated with a system to test.
- **Automata Generator**, it generates the automata associated to the properties of the system.
- **Capture Window**, it allows the user to select the network to analyze and the protocol which it will use as pattern to test the trace.
- **Testing Manager**, it performs the process of checking a trace against the properties of the selected protocol.

The first functionality of the tool is the Properties Manager that allows the user the definition, modification or removal of a property that is associated with a system. After this step, the Automata Generator provides an interface in which the user can generate the automaton of a selected property. This is an essential step in the process of testing due to the fact that it is the model that it will use to check the correction of the observed trace. Once the automata have been built, the Capture Window must be used to capture the traces of interest. This functionality offers the possibility of selecting different communication protocols (HTTP, FTP…) and the network that we are going to use as monitor. Finally, while the trace is being captured, it is necessary to check whether the SUT that we are testing is correct or it has an abnormal behavior. The Testing Manager performs this process and informs the tester about the progression.

The PTTAC class diagram is depicted in Fig. 1. The main class of the project is `MainDialog` which communicates all the subclasses with the runner of the GUI, the `GUITRunner` class. Each of the functionalities previously introduced has a principal class that is composed of different subclasses. The Properties Manager class is `TabProperty`; the Automata Generator class is `TabAutomaton` which has the subclass `GalleryAutomaton`; the Capture Window class is `TabCapture` and the Testing Manager class is `TabTest` which depends on the `AutomatonTree` class.
Case Study

After the analysis of the tool that we have used in our work, we describe the case study in which it has been applied this passive technique, with the main aim of evaluating the usefulness of the proposed framework and as a consequence, the effectiveness of the implemented tool. We applied our approach to traces corresponding to the communication protocol HTTP. Due to the extended use and study of this protocol, capturing traces with errors is almost impossible. For this reason, we decided to apply mutation testing techniques [10] for obtaining a mutated specification of the protocol in order to evaluate real traces against an incorrect HTTP specification [11]. Different properties were obtained from this mutated specification. If the tool is correctly implemented, it should inform us about the differences between the observed traces (correct) and the mutated specification in form of properties (incorrect).

According to the original specification of the HTTP protocol, we designed a property that represents the exchange of message for getting a resource. This property expresses the answers of the server in case that the client requests a resource. These must be one of the following: 2xx (acknowledge receipt of the request), 3xx (redirection to continue the request), 4xx (error in the request) or 5xx (error in the server). And the only possible responses that are not allowed correspond to the codes 1xx (useful information that the client needs to know for continuing the communication) because they are only sent under exceptional conditions.
By means of the application of mutation techniques, we modified the specification with the following transformation: the answer 2xx was replaced by 1xx. Then, we generated a property associated with this erroneous behavior. After the definition of the property the corresponding automaton is generated. Figure 2 shows the obtained automaton.

![Figure 2. Automaton generated.](image)

Then, we captured HTTP traces to check the effectiveness of the proposed technique for detecting the kind of errors related to this property using the Capture Window. The flow of messages are checked in real-time by the Testing Manager until it finds an error. In Figure 3 there is a screen capture of the result. The trace was obtained from a newspaper webpage where we requested a resource. PTTAC detected the error of checking the real trace against the mutated property.

![Figure 3. Testing manager.](image)

**Conclusions and Future Work**

In this paper we have presented a case study based on the analysis of online capture of HTTP traces with the tool PTTAC introduced in a previous work. This tool implements a framework for checking the correctness of a system in which exists an asynchronous communications channel between the monitor of the tester and the system under test. The innovative part of this work is the possibility of testing environments where the tester cannot interact with a system, as happens in the case of communication protocols. The definition of properties and its mutation to represent incorrect behavior of the HTTP protocol allowed to test captured traces and determine the effectiveness of the tool for detecting faults corresponding to the unfulfillment of properties designed by the testers with real HTTP traces.

There are some enhancements that we would like to implement in the next version of PTAAC. The current tool is able to capture online traces from different protocols such as HTTP or FTP. We think that it could be interesting to extend this range of possibilities with other types of protocols, not only...
for those oriented to communications by computers, but also for those oriented to communication between smartphones, medical devices or wearables.

Acknowledgement

Research partially supported by the Spanish projects ESTuDio (TIN2012-36812-C02-01) and DArDOS (TIN2015-65845-C3-1).

References


