CENG 492

TEST SPECIFICATIONS REPORT

SECURE VIDEO STREAMING PROXY SERVER

TEAM PRIME'

Ant OngunKefeli

DoğanPoyraz

YurdakulGöksuOrhun

OnurCemSırlı

April 24, 2011

Contents

1. Introduction

This document is the Test Specification Report of Secure Video Streaming Proxy Server project, which is held by Prime' group.

1.1. Purpose of the Document

The purpose of this document is to give information about testing methods that we will use while we prepare our final package and environment of each of our test methods. Using the tests mentioned in this document, we are aiming to produce a project as bug-free as possible. Also, this document will be guideline for our project's reliability, availability and serviceability issues that may come up during our remaining work and tests.

1.2. Scope of the Document

This document includes our testing plans and strategies for the Secure Video Streaming Proxy Server project. Our testing strategies, testing staff and group organization, testing schedule and some detailed information about our strategies and procedures are included as well. Mainly, this document is a guideline for remaining work of our project.

1.3. Major Constraints

This project is executed by four senior students in METU Computer Engineering department. Team members are working on the project besides their lectures and other homework. Therefore, time emerges as a huge problem and constraint for remaining work and testing of the project.

In addition to time shortage, our team had some problems with the sponsor company. We could not get the support that was to be offered, and the hardware required and promised such as web servers with static IP address. Eventually, this situation is the largest constraint obstructing our project's development.

1.4. Definitions, Acronyms and Abbreviations

- SVSPS:Secure Video Streaming Proxy Server
- Codec: Coding Decoding
- GUI: Graphical User Interface
- HTTP: Hypertext Transfer Protocol
- MPEG: Moving Pictures Experts Group
- VLC: VideoLAN Client

1.5. References

- Software Requirements Specification, by Prime', Fall 2010
- Detailed Design Report, by Prime', Fall 2010
- Myers, G.J. (2004). The Art of Software Testing

2. Test Plan

2.1. Software to be Tested

2.1.1. Prime' Media Server

This is the main component that collects the streams on one server and sends them to web servers according to load balancer's directives. This software and machine will be tested as a whole against failure of incoming streams, recovery on source failure and delay and mission success on target web server changes directed by load balancer.

2.1.2. Prime' Load Balancer

This component has the most important mission in the system. It tries to balance the load on web servers by changing, adding, removing or dedicating web servers to stream process of a particular video. Like the media server, load balancer software and machine will be tested as a whole. It will be tested against sudden load increases (load spikes), continuous high load, continuous low load, very high demand for a particular video. These conditions will be created by us and we will observe how system responds to these incidents, how efficient it manages or can it manage the situation well.

2.1.3. Web Servers

These components' job is pretty simple. They will simply get the video, which is assigned by the load balancer and send them to the clients. We will conduct various tests on these components as their malfunction would cause observable effects to the user side. First of all, individual tests will be performed to check the functionality on a smaller scale. These tests will mainly consist ofperformance, functionality and data quality tests. Later on, the component will be tested while it is in an environment which consists of a number of its peers. Similar tests will be conducted to individual testing.

2.2. Testing Strategy

2.2.1. Unit Testing

Our media server, web servers and load balancer will be tested individually to begin with. During these tests, we will prepare special testing scenarios to cover as much as logical paths that the components follow during their execution workflow. Although it isnot mathematically possible to test all the scenarios, carefully designed test should be inclusive enough.Before integration, unit testing should be sophisticated enough as after this phase we will mostly assume that any error encountered will not be due to individual system component malfunction.

2.2.2. Integration Testing

After testing each module individually, we will connect thempair wise and then make connections by hand. First we will connect media server to web servers and test that pair, then we will connect web servers and load balancer and test them. If both pairs work well we will connect all three together and continue testing as a whole.

2.2.3. Validation Testing

This test will be run in order to see if the project has met its previous designs and expectations as mentioned in earlier reports such as SRS, SDD. We will try to create a system as close to designed ones as possible.

2.2.4. High-order Testing

We are planning to run some higher level tests on the system since its job is mostly to organize extraordinary situations.

- **Stress Tests**
- **Performance Tests**
- **Alpha and Beta Tests**

2.3. Test Metrics

We will be using the following metrics:

- Number of test cases executed
- Number of bugs detected
- Number of bugs fixed
- Number of priority bugs fixed

2.4. Testing tools and environment

We are not planning to use any testing tool other than our media server, web server and load balancer tools. Instead we will create different conditions on these components.

Situation is same for environment as well. All components are already working on their planned environment. Our web servers and media server are running on 64 bit windows, while load balancer is running on a 32 bit Linux.

Hardware is only issue for media server since it may do several codec processing at the same time. Except that rest of the project is network communication so hardware won't be an issue for other components since network components are standard on most of the computers.

3. Test Procedure

3.1. Unit Testing

3.1.1. Prime' Media Server

We will conduct the following black-box test cases for media server:

- GUI
  - Login and logout functionalities will be tested with various inputs. The following table describes the input types we will use:

Username	Password
Existing username	Correct password
Existing username	Incorrect password
Non-existing username	Some password
Erroneous input	Erroneous input

  - User management system will be tested.
    - Adding user to empty database
    - Removing the only user from database
    - Adding user to non-empty database
    - Removing a user among multiple users
  - Stream management system will be tested
    - Adding stream to empty stream folder
    - Removing the only streamfrom stream folder
    - Adding stream to non-empty stream folder
    - Removing a stream among multiple streams
    - Crosschecking stats of streamed video with a feed of a receiver

3.1.2. Load Balancer

We will use an incremental approach for load balancer testing. That is, starting from a single user, we will open as many connections as possible and observe the load on web servers. Therefore, we will discover errors of load distribution functionality of load balancer.

3.1.3. Web Servers

We will check availability of each web server by using their IP. This procedure will consist of two parts:

- Sending a stream to the web server
- Receiving the stream from the web server

3.2. Integration Testing

Integration testing will consist of three main parts:

- Media Server-Web Server Integration
  Firstly we will test the integration of media server and web server. Specifically we will start a stream using our media stream and see whether it is received correctly on the web server side.
- Web Server-Load BalancerIntegration
  Load balancer is supposed to distribute the load among web servers. We will connect to load balancer with multiple clients and test if all available web servers are used in redirection process by load balancer.
- Complete System
  Since web servers can be considered as articulation point of our system, we will proceed with previous integration tests sequentially; therefore, test the integration of whole system.

3.3. Validation Testing

We will test each use-case that is defined in our SRS document.

3.4. High-order Testing

3.4.1. Stress Tests

We will test the system with maximum possible connections and observe the delay and transmission times. We will try to run the system without crushing with maximum efficiency.

3.4.2. Performance Tests

We will test our load balancer with different and mostly high levels of load. Increasing the load means increasing the number of clients connecting to the system.

3.4.3. Alpha and Beta Tests

Our product's customers would be mostly companies that use video streaming heavily. It is not possible to install this system to a company just to test it. Moreover, using someone as a client to test the system would not be meaningful because our project is all about internal working of the system. Therefore, we will do the alpha and beta tests ourselves.

4. Testing Resources and Staffing

We do not need any testing resource except team members and our personal computers. We will create test conditions on our computers. But, we may require a few more computers in order to be used as clients. Regarding the client issue, we are considering to use virtual machines to create additional clients/servers. We will use whichever is more suitable at that moment. For personal responsibilities of members, please refer to Part-7 of this document.

5. Test Work Products

In order to execute our tests, we will create some bat files or similar scripts in order to start/stop connections and streams. These files will be executed when required. Another product of our tests might be some log files created automatically by the system on each machine. Except these files, our project tests will not have much product since our tests are based on results not products.

6. Test Record Keeping and Test Log

For testing purposes, we do not plan to use any tools. We will just create the server-client conditions by hand and collect test data. Then we will store the data to a document with all other test results. Finally, we will compare the results and try to see which part of the system is more inclined to a crash or failure.

7. Organization and Responsibilities

Responsibility	Staff
Unit Testing	All
Integration Testing	Onur Cem Sırlı Yurdakul Göksu Orhun
Validation Testing	Onur Cem Sırlı Doğan Poyraz
High-order Testing	Yurdakul Göksu Orhun Ant Ongun Kefeli

8. Test Schedule

Test Name	Execution Date
Unit Tests	26.04.2011-30.04.2011
Integration Tests	01.05.2011-07.05.2011
Performance Tests	08.05.2011-16.05.2011
Feature Tests	17.05.2011-25.05.2011