Vincent Yeung
6.033 Section #8
Saltzer/Garfinkel 1pm

6.033 Design Project – Executive Summary

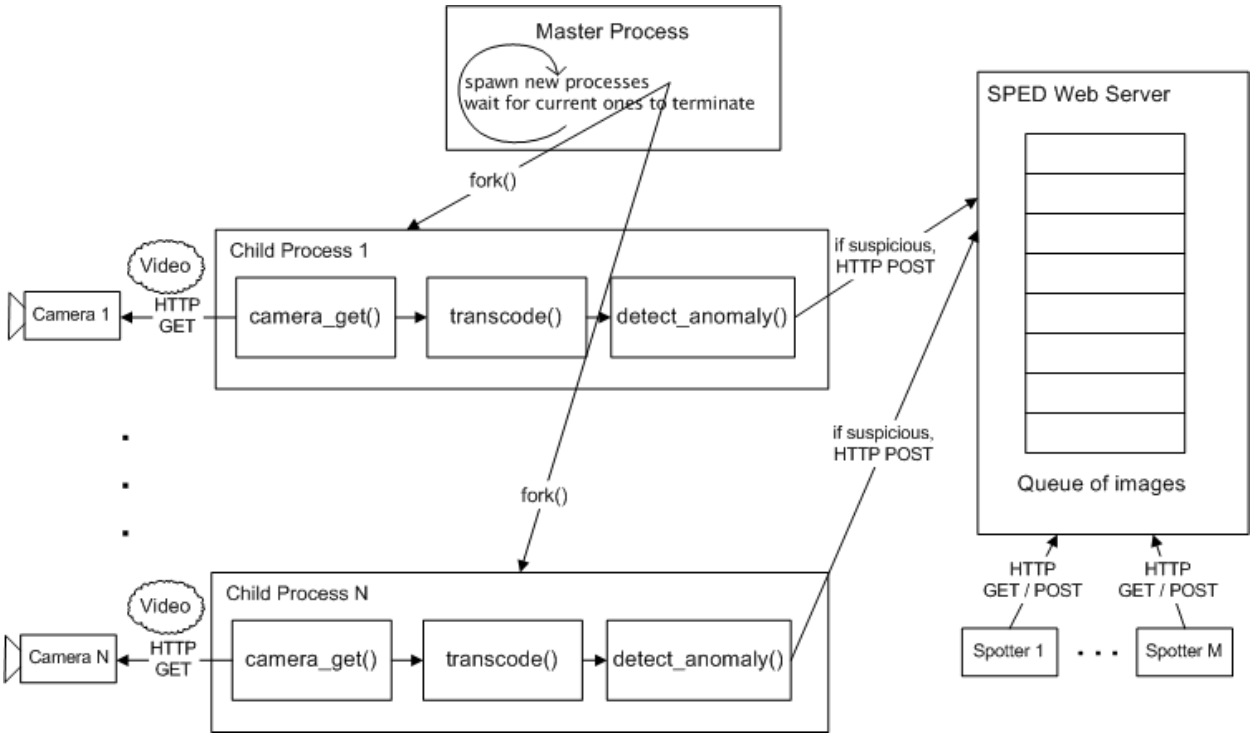
Purpose

This report describes a software system used for video surveillance. The design of the system had three primary goals: fault toleration, scalability, and graceful performance degradation under heavy workloads. Other issues such as performance optimization were not treated with the highest priority in this design as the hardware was assumed to be fast enough to handle reasonable workloads within the surveillance system's original specifications.

Design Overview

The system consists of a central master process that is responsible of spawning and maintaining child processes, each of which obtains a short video clip from a camera, passes it through the transcoder, runs the AI analysis, and obtains the most suspicious frame of the video and its "threat level." If the threat level is high, the process transmits the suspicious frame to the web server process, which will store the image in a database. The web server serves images from this database to human spotters over the Internet for further review. If the spotters decide that the image appears suspicious, the system will trigger its alarm.

The following diagram summarizes the operation of the software system:



Video Processing

The master process creates a new child process for every camera. Each child process does the following in its lifespan:

1. If there is no open connection to its camera, initiates one using HTTP GET and reads a fixed amount of data from the stream.
2. Executes the transcoder routine to convert the stream into JPEG.

3. Executes the AI routine to identify the most suspicious frame and its threat level.
4. If the threat level is above a fixed threshold, send the suspicious frame to the web server.
5. Terminates.

To ensure that erroneous child processes, such as those in infinite loops, know to stop consuming system resources, each process has a timed alarm that will terminate the process when triggered.

The master process is signaled when a child process terminates, at which point it spawns a new child process on the same camera. The master process also uses status information it receives about the terminated child to determine if the termination was caused by a timeout from the alarm. If consecutive child processes on the same camera time out, the master process will decide there is an overload and wait using an exponential back-off scheme before spawning new processes on the camera.

Web Server

The web server uses a SPED design, which is appropriate in this system for two reasons. First, compared to many other web server designs, SPED is simple. Second, SPED's biggest drawback, process blocking during disk I/O, never occurs in this system because all data is stored and accessed from RAM and not a hard disk.

The video processing threads described in the previous section alert the web server about suspicious images using HTTP POST. When a frame is determined to be suspicious by the software, it is attached with its originating camera number, current time, and estimated threat level and sent to the web server. The web server has a task listening to these updates, among the other requests it receives. It receives the image and assigns it a unique identifier. The server then stores the information in a simple linked-list queue.

Human spotters initiate a connection with the web server using HTTP GET and are assigned a spotter number. Upon a new image request, the server will select the first image starting from the front of the queue that has not already been served to the spotter making the request. Spotters' responses are sent via HTTP POST, and the web server will use them to change the threat level of the image. The threat level increases or decreases by a constant amount, or remains unchanged, depending on whether or not the spotter thinks there is suspicious activity, or if he/she is uncertain. If the updated threat level goes above a high threshold, the system alarm is triggered; if it goes below a low threshold, the image is discarded. Finally, if an image has received more than a threshold number of responses and still remains in the "uncertain" threat level range, it is also removed from the database.

Feasibility

The system achieves fault isolation by having a separate process for each camera. In addition, since each process only works on its camera for a short period of time, other errors such as infinite loops can be easily detected with the use of timed alarms. The system is scalable because only a few global numerical parameters need to be changed to support additional cameras, and the design employs an exponential back-off waiting scheme to help the system adapt and recover from occasional overload.