Video surveillance based on cloud storage

D. A. Rodríguez-Silva, L. Adkinson-Orellana,
F. J. González-Castaño, I. Armiño-Franco
Network and applications Dept.
Gradiant
Ed. Citexvi, 36310 Campus Vigo - Spain
{darguez,ladkinson,javier}@gradiant.org

D. González-Martínez
Telematics Department
Universidade de Vigo
ETSI Telecomunicación, 36310 Campus Vigo - Spain
david.gonzalez@gti.uvigo.es

Abstract—**Traditional video surveillance systems require infrastructures including expensive servers with specific capabilities to process images and store video recordings. We propose to apply solutions based on Cloud Computing with its many advantages as scalability, flexibility, costs reduction or ubiquitous access. This paradigm is based on infrastructures managed by service providers where all the information from the users is processed and stored. Surveillance systems produce and need to store a huge amount of data and, furthermore, to execute specific image analysis in real-time in order to detect events. The proposed video surveillance system based on Cloud Computing will collect multimedia streams generated by surveillance cameras, optimize their transmissions taking into account the network status and store them in a cloud storage system in a efficient way. In addition, the system provides a web portal for the users to configure the cameras and manage the video recordings.**

Video analysis; surveillance system; cloud storage, security

I. Introduction

Traditional CCTV video surveillance systems are being progressively replaced by fully networked digital systems composed of cameras with IP connectivity, network recorders and web-based user interfaces for managing and visualizing the video streams, which are generated and stored directly in digital format. This has created the opportunity to make video surveillance systems "smarter", thanks to the possibility of running automated computer vision analysis on those video streams, which release human operators from intensive and tedious monitoring tasks. "Smart" video surveillance systems are based upon machine learning and computer vision techniques, being able to automatically detect a wide variety of events of interest for different applications, such as security, traffic monitoring, customer behavior analysis in commercial areas, etc.

Smart video surveillance systems usually need to store both video sequences and metadata information related to detected objects and events. In large scale systems with many cameras, storage requirements rapidly increase, posing scalability and reliability problems. For this reason, current client-server architectures may be too expensive for medium-sized enterprises. In addition to hardware costs, they also have high maintenance needs, and require skilled personnel to deploy and configure them. This proposal advocates for the use of cloud solutions to overcome those problems. The main goals are the following:

- To provide a cloud storage environment for efficient large scale data storage considering security aspects such as privacy, reliability and fault tolerance.
- To provide a Cloud Computing environment to run complex video analysis algorithms on demand, in order to adapt the system dynamically to user and computation needs.

The proposed video surveillance system based on Cloud Computing collects and analyzes video streams generated by video surveillance cameras, optimizes the transmission of the video data taking into account network status, and stores the video and generated metadata in a cloud storage system, in a secure and efficient manner. In addition, the system will provide a web-based user interface for management and configuration purposes. The selected cloud storage service to integrate the solution is Amazon S3, because it provides several advantages as simple integration and data encryption.

II. Related Work

Current commercial smart surveillance systems are based on a central server that performs both processing and storage tasks for all cameras [1][2][3]. In some cases [1], part of the processing can be performed in advance in a distributed manner, by using smart cameras or intermediate processing units. However, improvement in scalability is conditioned by the features of specific camera models or other specific purpose hardware, which tend to be expensive for the customer.

Regarding Cloud Computing, there exist solutions to store and serve multimedia content as Amazon Cloud Front [4], which allows developers and businesses to easily distribute content to end users with low latency, high data transfer speeds, and no commitments. There are also some companies as Logitech that offer cameras able to connect to external storage systems like Dropbox to store files [5]. However, current solutions do not consider mechanisms based on buffers to continuously upload data from clients since they are mainly focused in allowing an efficient download. There are more specific solutions such as cloud surveillance or VSaaS (Video Surveillance as a Service) [6] based on IP cameras, without the need of local DVR surveillance equipment. Nevertheless, this system has very limited video analytic capabilities, behind the state of the art of current solutions. Security is a crucial topic in Cloud Computing, specially in scenarios like ours [13,14] where the transmitted data can contain sensitive information. Solutions that encrypt any data that leave the client are common, such as encrypted storage on the cloud [9] or secure document edition SaaS [10], and there also exist solutions focusing on encrypted DaaS (Databases as a Service) [11] or encrypted domain processing [12], which protect data from service providers. Our proposal ensures integrity and confidentiality by securing the communication among its components and applying encryption to stored data. It also uses authentication techniques based on digital certificates.

III. CLOUD SURVEILLANCE SYSTEM

A. *General architecture*

As described before, the proposed cloud-based architecture allows a smart and fully scalable video surveillance system supporting a pay-per-use business model. It is composed of the following components (Fig. 1):

- Client: it collects video data from one or several IP cameras, managing their transmission to the cloud processing servers in an efficient manner, by controlling buffer input/output rate.
- Processing server: it receives and processes video data from the clients, controlling their contribution rate assisted by its own temporarily storage, and, optionally, running computer vision algorithms in order to detect events and trigger alarms. It sends the processed information to the cloud storage server through a secure connection.
- Storage server: it stores all the information generated by the cameras and processing servers.
- Web server: it provides a web portal to manage the system, allowing to access the information stored in the cloud, as well as to configure video analytic rules, alarms, etc.

The proposed solution addresses the dynamic deployment of specific processing servers adapted to the computational needs in every moment, thus avoiding situations where lack of computational resources can be a problem. The processing server will negotiate with the clients the video transmission rate, in order to guarantee the stability of the system, for instance in case of temporary cloud storage failure.

Finally, privacy is an issue of special relevance in video surveillance systems. In order to safeguard confidentiality, communications will be secured using the SSL protocol, and asymmetric encryption will be applied to stored data. The modular architecture proposed in this paper is suitable to add new security mechanisms, by adding new modules to carry out the security enhancements.

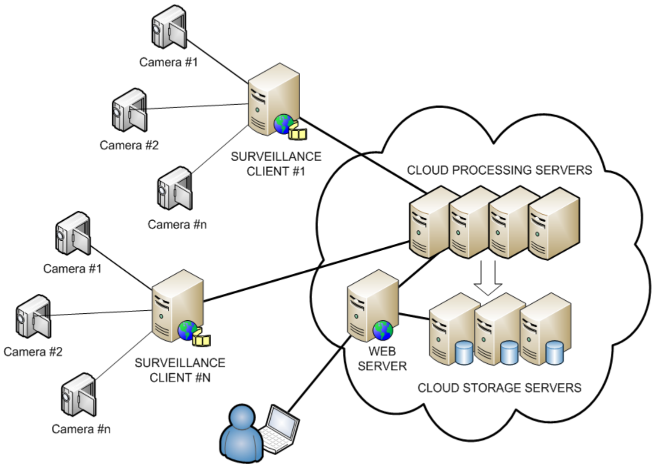

Figure 1. Architecture of the proposed cloud video surveillance system.

B. *Implementation*

The modular design of the system provides several advantages like the possibility of replacing the cloud storage service in case of failure or low performance, or updating the client module to support new acquisition devices. Java is the main technology to implement the modules of the system.

The camera chosen in our tests was an IP Cam AXIS M1011 that supports an HTTP-based application programming interface named VAPIX [7]. VAPIX provides functionality for requesting images, controlling network camera functions (PTZ, relays etc.) and setting/retrieving internal parameter values.

As previously said, the cloud storage service chosen was Amazon S3 [8]. Its main features are the following:

- Possibility to write, read, and delete objects ranging from 1 byte to 5 terabytes of data each. The number of objects is not limited in principle.
- Each object is stored in a bucket and retrieved via a unique, developer-assigned key. A bucket can be stored in one of several regions. It is possible to select a region to optimize latency, minimize costs, or address regulatory requirements.
- Objects stored in a region never leave it unless a transfer is ordered. For example, objects stored in the EU (Ireland) region never leave it.
- There are authentication mechanisms to ensure that data is safe from unauthorized access. Objects can be made private or public, and rights can be granted to specific users.
- Options for secure data upload/download and encryption of data at rest are provided for additional data protection.
- It employs standard-based REST and SOAP interfaces designed to work with any Internet-development toolkit. The default download protocol is HTTP.

Next we describe the implementation details and the performance of the modules.

Client

The client has two main functions. On one hand it handles the data generated by the IP cameras by means of a script that allows configuring them (mainly image quality) through the VAPIX API. This data is received through a socket and stored in a local cache (internal buffer). On the other hand, the client will use REST over HTTPS to communicate with the processing server in order to send a predefined accumulated amount of data to it.

The client has a control module with the following functions:

- Analyzing the amount of video data existing in the internal buffer to detect when it is necessary to send them to the processing server.
- Controlling the input/output rate of the internal buffer to increase/decrease the size of the data packages to be sent to the processing server.
- Attending the responses from the processing server to control the rate and quality of the images generated by the IP cameras. There exist three types of responses:
    - *OK*: no changes are needed.
    - *Decrease rate*: the client must reduce the amount of data sent to the processing server during a short period of time.
    - *Reduce quality*: corresponding to a more critical situation, where the client must reduce the amount of data sent during a longer time. The quality will be reduced until an *OK* message is received.

Fig. 2 shows the flow diagram of the client considering the different response types.

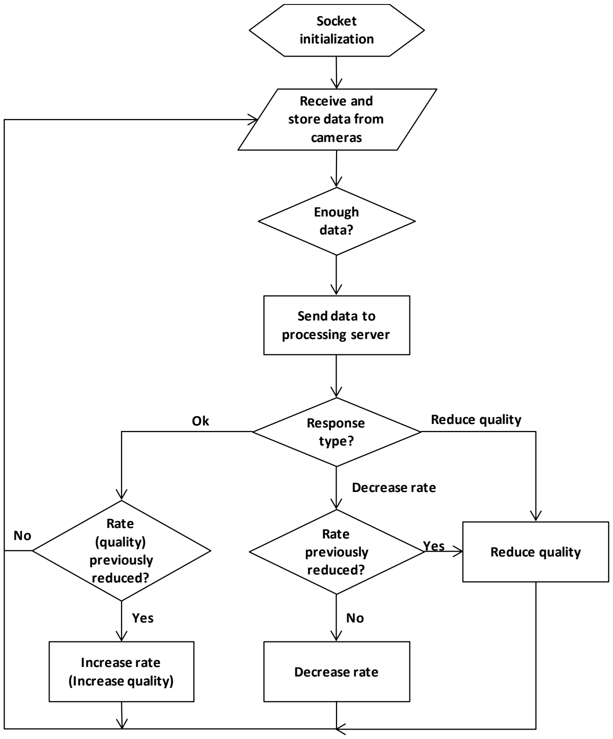

Figure 2. Flow diagram of the client.

Processing server

This component is in charge of managing all the traffic received from the clients through a socket, as well as processing, classifying and storing the data locally, to be sent afterwards to the cloud storage system. This server has several modules:

- Video processing module: It analyzes data seeking events, following complex algorithms with defined rules.
- Local storage module: It stores received data in a local disk temporarily and controls disk usage considering two thresholds, preventive and critical (with risk of total occupation). When the first threshold is reached, the module finds out which clients are transmitting at higher rate and notifies the control module to send a *Decrease rate* message to them. If the second threshold is reached a more strict action is required and a *Reduce quality* message is sent to the clients. When disk usage drops, the clients can transmit again at a normal rate with a normal quality.
- Control module: It stores received data as files through the local storage module, indexing them efficiently in order to facilitate their retrieval. Afterwards, when possible, the files are sent to the cloud storage server, with an XML metadata file containing the necessary information to easily identify the stored information. This module also controls the size of the files according to the size of the data packages received from the clients, and checks local disk usage in order to indicate the clients if they have to modify their sending rate.
- Remote storage module: this module communicates with the permanent storage server through a Java API in order to upload the video and metadata files of each client, as well to download required ones when necessary.

Fig. 3 shows the flow diagram of the local storage module, where the two types of thresholds are considered.

Storage server

This server offers permanent storage to keep video data integrity. In our case this service is provided by Amazon S3. It has a data structure based on buckets and objects. A bucket has a unique identifier and it is composed by an unlimited

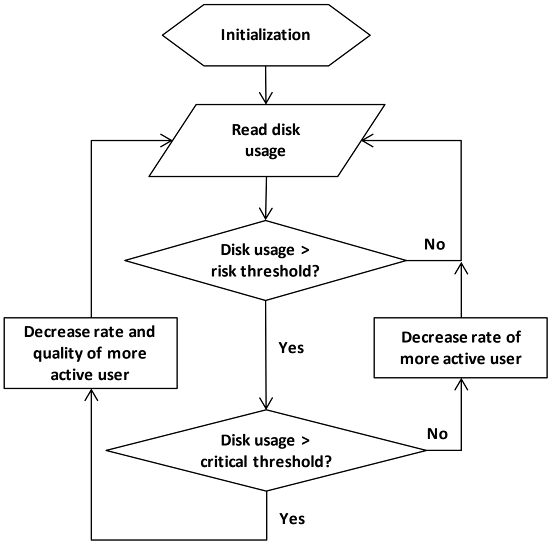
Initialization
Read disk usage
Disk usage > risk threshold?
No
Yes
Decrease rate and quality of more active user
Decrease rate of more active user
Disk usage > critical threshold?
No
Yes

Figure 3. Flow diagram of local storage module.

number of objects. Each client will use a bucket that will contain the video files generated from the cameras associated to it and the metadata file describing their content, basically the start and end dates of the video fragments. The Java API offered by Amazon S3 allows the processing server to upload and download encrypted files, as well as organizing objects and buckets easily.

Web server

The web server hosts a web portal based on Java Servlets. This portal is accessible via HTTPS and provides a user interface that allows managing the data stored in the system, including the following operations:

- Recover the whole video stored
- Recover a video specifying the start and end date
- Delete a video totally or partially

If the user decides to recover a video, it will be retrieved from the storage server by means of the processing server, giving the user the possibility of downloading or playing the video via streaming.

IV. PERFORMANCE TEST

Our development can be compared with traditional surveillance systems, such as those that save the recordings in local storage or a remote server (FTP, NFS, iSCSI...). The main advantages of our system are resilience and scalability. A system where the client stores the recordings directly in the storage provided by Amazon S3, using a small buffer to prevent data loss, is clearly more resilient than a traditional one. It can prevent data loss using the local buffer if the cloud storage provider goes down during a limited period of time. This time can be used by the system to substitute the fault storage provider by an operational one.

The experiment aimed at demonstrating performance improvements when the processing server controls contribution rates. Thanks to it, the time during which the system can support a failure in the cloud storage increases, even keeping the client transmission rate if the service goes down momentarily. We thus compared the performance of the system whether o not the processing server is present. We configured the test so as to let us clearly observe the behavior of the system in a small time slot. The scenario simulates a failure in the storage provider: the service goes down 3 minutes after the beginning of the test, when the system has reached a stable state. Then, the service is kept down for 40 seconds and then goes up again.

Fig. 4 illustrates the behavior of the system when the processing server is present. Initially the client storage rate is stable and close to the maximum transmission rate supported by the client connection with the storage server. When the storage service goes down, the processing server detects the lack of service because its buffer content starts to grow. Then the processing server sends a request to the client to reduce the transmission rate in order to grant the service and avoid data loss. When the storage service is reestablished, the service provider lets the client increase the transmission rate gradually, until the maximum is reached again. The system overcomes the communication failure without mishap.

Next, Fig. 5 illustrates the behaviour of the system when the processing server is not present. In this case, the client sends data directly to the storage provider and it depends exclusively on the capacity of his own buffer to support Amazon S3 failures. When the service goes down and the client buffer is full, data is irremediably lost. This is a highly problematic situation in a surveillance system, where all data should be hold securely.

Obviously, the problem presented in this scenario can be solved using a larger client buffer, or setting an even lower video quality that in the previous scenario, but these solutions imply degradation in terms of cost or performance. The processing server also allows managing different cloud providers and may provide more complex algorithms to prevent data loss.

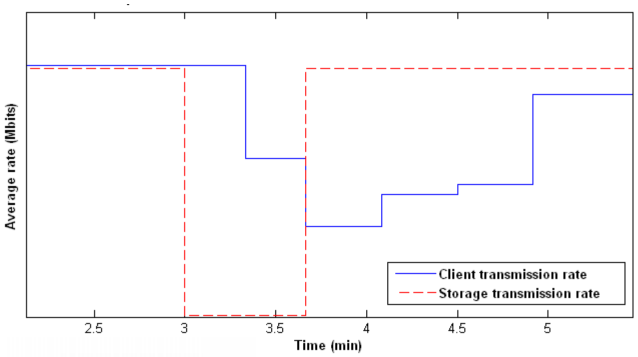

Figure 4. System behavior facing a failure in Amazon S3 service.

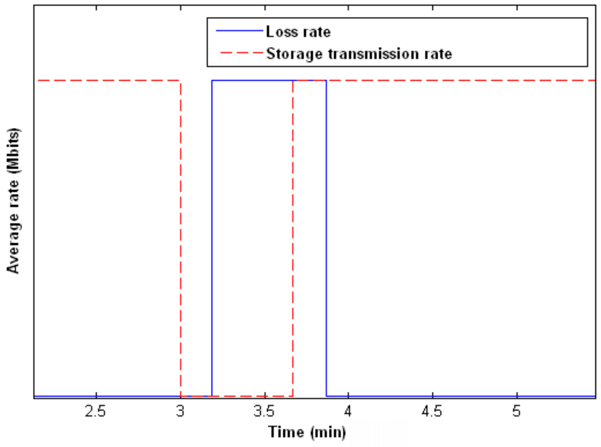

Figure 5. Behavior of the system facing a failure in Amazon S3 if the processing server is not present.

V. Conclusion

Modern “smart” video surveillance systems have several advantages, but the traditional client-server architectures used nowadays pose severe limitations, mainly related to scalability and storage problems. To cope with them we propose a modular video surveillance system based on Cloud Computing technologies that provides a highly reliable solution that may work with different storage providers at the same time. The client manages a set of IP cameras and communicates in a secure way with a processing server, which handles contribution rate and detects security events. The system relies on a mechanism based on local and processing server buffers to minimize the loss of data recorded by the cameras. This permits to deal with traffic peaks without reducing the quality of video. We have used Amazon S3 as cloud storage server because its API allows an easy integration. Our tests demonstrate that the proposed cloud video surveillance system can face a network problem or a failure in Amazon S3 by regulating the rate and quality of the data sent by the clients.

The modular architecture proposed in this paper takes advantage of the Cloud Computing paradigm and is suitable for the future advances in surveillance, such as the incorporation of multi-sensor and biometric solutions or 3D video.

As future work we plan to automatize switching between cloud providers considering aspects like costs, delays or bandwidth.

Acknowledgment

This research has been partially supported by the CloudMeUp grant (IDI-20101357), funded by CDTI, Spain.

References

[1] AgentVi (accessed February 26, 2012). http://www.agentvi.com/

[2] NiceVision (accessed February 26, 2012): http://www.nice.com/video/analytics

[3] VGuard (accessed February 26, 2012): http://www.vguardinternational.com/cms/

[4] Amazon Cloud Front (accessed February 26, 2012): http://aws.amazon.com/en/cloudfront/

[5] Logitech Alert Security System (accessed February 26, 2012): http://www.logitech.com/en-us/349/9256

[6] Cloud surveillance (accessed February 26, 2012): http://www.cloudsurveillance.com

[7] Amazon S3 Storage Service (accessed February 26, 2012): http://aws.amazon.com/es/s3/

[8] VAPIX API website (accessed February 26, 2012): www.axis.com/techsup/cam_servers/dev/cam_http_api_index.php

[9] S. Kamara and K. Lauter, “Cryptographic Cloud storage,” Workshop on Real-Life Cryptographic Protocols and Standardization, 2010.

[10] L. Adkinson-Orellana, D. A. Rodríguez-Silva, F. J. González-Castaño, and D. González-Martínez, “Sharing Secure Documents in the Cloud. A Secure Layer For Google Docs,” Proc. of 1st International Conference on Cloud Computing and Services Science (CLOSER 2011), Noordwijkerhout (Netherlands), May 5-10, 2011.

[11] X. Tian, X. Wang, and A. Zhou, “DSP re-encryption: A flexible mechanism for access control enforcement management in DaaS,” Proc. CLOUD’09, Bangalore (India), 2009, p. 25–32

[12] D. A. Rodríguez-Silva, F. J. González-Castaño, L. Adkinson-Orellana, A. Fernández-Cordeiro, J. R. Troncoso-Pastoriza, and D. González-Martínez, “Encrypted Domain Processing for Cloud Privacy. Concept and Practical Experience,” Proc. of the 1st International Conference on Cloud Computing and Services Science (CLOSER 2011), Noordwijkerhout (Netherlands), May 5-10, 2011.

[13] K. Kraus and R Reda, “Security Management Process for Video Surveillance Systems in Heterogeneous Communication Networks,” Proc. of the IFIP Wireless Days, 6th IFIP Network ControlConference, Dubai, Nov. 2008.

[14] K. Kraus, O. Martikainen, and R. Reda , "High performance Security Management Processing in Advanced Intelligent Video Surveillance,” Proc. of the 7th International Conference on Informatics and Systems (INFOS), March 2010.