Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 2, Issue. 9, September 2013, pg.263 – 267

RESEARCH ARTICLE

CONTRAIL – ARCHITECTURE FOR MOBILE DATA SHARING APPLICATIONS

Lavanya.K[1], Devipriya.C[2], Saradha.S[3], Sangeetha Priya.M[4]

[1] M.E. (CSE), Sri Eshwar College of Engineering, Coimbatore, India
[2] M.E. (CSE), Sri Eshwar College of Engineering, Coimbatore, India
[3] M.E. (CSE), Sri Eshwar College of Engineering, Coimbatore, India
[4] M.E. (CSE), Sri Eshwar College of Engineering, Coimbatore, India

[1] lavanya030891@gmail.com

Abstract— Collaborative applications running on 3G devices often rely on cloud-based servers for computation and storage. A peer to-peer approach to building these applications can provide benefits such as enhanced privacy and bandwidth efficiency. We propose a system which is based on asynchronous network architecture that uses the cloud to relay messages between 3G devices. System employs selective receiver-specific filters at sending devices to ensure that only relevant data consumes precious bandwidth. Proposed framework offers pull-based communications primitives suitable for mobile devices that are often either inactive or subject to poor network connectivity. This system enables robust mobile applications without making assumptions about the security of individual cloud providers. We have implemented this system with Windows cloud environment and demonstrate sample applications executing across Android mobile phone. Sample application shares the multiple contents like text, video, images and emails. In this paper, we introduce Contrail, a new communication architecture for mobile data-sharing applications that eliminates the drawbacks of the client-server approach while retaining its positive attributes.

Keywords: - Client-Server; Virtual Environment; Contrail; Synchronization

I. INTRODUCTION

Smartphone connected to ubiquitous 3G networks are enabling new and diverse applications. One class of applications consists of data sharing services, where data generated by a device (location updates, photos, video clips and real-time feeds) is shared with other devices based on specific policies. These policies are usually based on the generated content; for example, a tourist backpacking through Europe with a Smartphone may want to broadcast her location to friends who live nearby, receive tweets that mention her name, share videos that her family wants to see, or even receive a feed from her home's security camera if it detects movement. Currently, such services are structured as client-server applications; all data generated by a device is uploaded via 3G to a central server which provides it selectively to other devices. The centralized approach allows data to be uploaded just once by a sender for multiple recipients, without requiring any of them to be online at the time.

As an additional benefit, client-server applications do not require inbound connections into devices, allowing cell ISPs to provide greater security by blocking such connections. However, the centralized approach can be inefficient if devices generate substantial data that no other device wishes to receive. In the example of the location-sharing application, the device will continually upload its location to the cloud even if nobody lives nearby, wasting network resources and battery life in the process.

A second problem with the centralized approach relates to privacy: users expose all data to application servers and the third-party cloud providers hosting them, and potentially even to other customers of the same cloud. Since the centralized server has to apply sharing policies on the data, simple encryption is not a viable solution. Privacy-preserving computing on encrypted data holds great promise, but is problem-specific and not yet widely deployed.

In this paper, we introduce Contrail, a new communication architecture for mobile data-sharing applications that eliminates the drawbacks of the client-server approach while retaining its positive attributes. Specifically, Contrail's goal is to enable applications that have the following properties:

Efficiency: Once Data is uploaded only if it is explicitly requested by some other device.

Non-Intrusiveness: if it is explicitly requested from some other device, then data arrives to device

Privacy: Data exchange between users cannot view by other device.

II. RELATED WORKS

For mobile applications, a framework is created as virtual environment used for execution. This application controls the deployment and execution [5]. For virtual environment, service provider should be trusted by user. The cost of communication is high for virtual environment [6] or machine. This has live migration between the devices for mobile applications, no need of modification in code.

In this paper Android mobile is used with cloud server and leads to migration. This migration is used to reduce cost for existing applications. Collaborative computing [8] is used only for period of time, which focuses only on inside of devices. Collaborative computing used to transmit packages with the greater probability and less bandwidth.

Earlier work is to consume only less power for mobile devices using remote execution [11]. Cyber foraging represents the performance of the mobile client interactions. Partitioning is done automatically to reduce the coding efforts with serialization, safety and portability. Zap is a tool which used for testing the integrated devices. In process migration, migrates the live application of execution on network. Storing of snapshot image in distributed storage system and then migrate the device or system. Live migration achieved by workloads movement in cluster and the changes are updating continuously to another device without considering the cost of communication overhead.

These applications work in pervasive environment with workload consisting of low or limited bandwidth. Today’s cloud framework is more focus on the privacy of data, cost for securing the data with large storage. Between the network devices QoS should be maintained using the protocols with integrated services such as RSVP [14]. This protocol used to carry the requests to virtual machine and that fetch the requests to proceed in network. QoS [15] applications choose service level by a mechanism called Integrated Services for traffic delivery, which deliver the packets even though in traffic. Only between the fixed nodes QoS data are transmitted.

III. ANDROID ENVIRONMENT

Virtual environment is created for the Android application [7] that exchanges any type of data to another device. This can be done with Android mobile of version2.1 and with cloud or virtual environment which is used as storage system. This can be done only in online mode. Previous model has two issues that centralized approach can be inefficient if devices generate substantial data that no other device wishes to receive. In the example of the location-sharing application [9], the device will continually upload its location to the cloud even if nobody lives nearby, wasting network resources and battery life in the process. A second problem relates to privacy: users expose all data to application servers, and potentially even to other customers of the same cloud. Since the centralized server has to apply sharing policies on the data, simple encryption is not a viable solution. Privacy-preserving computing on encrypted data is used, but is problem-specific.

3.1 Experimental Background

Contrail assumes that devices trust each other. For example, when one device sends data to another device, it trusts the recipient to not redistribute that data. Similarly, we assume that devices do not impersonate each other. We expect this trust to result from social contracts; for example, if the owner of the sending device knows the owner of the receiving device. In concrete terms, we assume the existence of a social graph where the existence of a link between two users implies that they trust each other completely. We assume that it is unsafe to reveal data to a cloud provider in unencrypted form. We do assume, however, that cloud providers are reliable and do not lose in light data. We assume that individual cloud providers can become unavailable due to new `failure modes', such as quota overages, payment delays, increased rates and changing business alliances. We assume that at least one operational and feasible cloud provider exists at any given point of time.

Contrail consists of two primary subsystems: a client side module that executes on each device, and a messaging layer that resides in the cloud. Each client-side module periodically initiates a TCP/IP connection to the cloud-based messaging layer via 3G (or a Wi-Fi hotspot). Contrail's basic operation can be described in simple terms: data sent from one device to another device is first uploaded to the cloud via one device-to-cloud connection, and subsequently pulled by the recipient device via another such connection.

These device-to-cloud interactions are the only network-level connections that occur in the system; for ease of exposition, we assume no out-of-band interactions between devices via channels such as Bluetooth. Applications on devices use Contrail by linking to a library that exposes several functions for sending and receiving data (see Figure 3.1.1). Internally, this library uses IPC to communicate with the client-side module running on the device. A single client-side module runs on each device in a separate process, and is shared by multiple applications.

3.2 Cloud Tier

In the Cloud tire module we built a cloud execution model in which process and data's are executed across multiple servers. The first tier consists of front-facing web servers that accept and manage connections from clients. Application code executes in a second tier of stateless compute nodes, with all persistent data stored within a separate storage tier. The stateless nature of the compute tier allows such platforms to easily scale out code written from inexperienced developer's application; each incoming request can be load balanced to any compute node, allowing throughput to be ramped up simply by adding more machines to the system. The storage tier is separately scaled out to partition and replicate data in order to provide fault-tolerant and scalable storage.

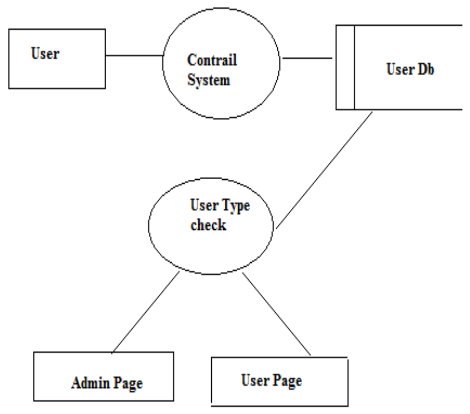

Fig 3.1.1 Contrail Login

3.3 Blog Storage

The blog storage is responsible for accepting connections from devices. As described earlier, the client module on each device periodically initiates a connection for pushing data or pulling it to a web server, which we will call the gateway for that device during that connection.

If this is the first time that the device is connecting to the cloud infrastructure, the gateway executes a `join' operation on a compute node, which then creates a queue in the storage tier for that device. The name of this queue is simply the Device ID of the connecting device. The purpose of the queue is to hold incoming data items and when the gateway receives a data message from the client module on the device, it executes a `send' operation at a compute node with the message as a parameter.

For each destination specified on the data message, the compute node locates the queue of the destination device using its Device ID as the queue name and appends the message to it. Before it appends the message to each destination device's queue, it changes the header so that it contains just that device as the sole destination. Alters sent to the device from other devices.

In cloud environment group users can be created by giving details and shared the details of the file in an organisation. A single user can create the group and fill the details of other user. User validates their details and stored in cloud storage. These details are stored in database. These can be created by generating report for validate users. These group users can have username and password. A user in a group should generate a report that details of the group and user. They can store any type of files which the storage in their group database.

IV. CLOUD MESSAGE API

A messaging layer is that which resides in the cloud and runs as an API. An application creates an end-point by calling the Open Port function, specifying a PortID and a filter installation callback function. Once the application opens a port, other end-points {i.e., other applications with open ports} can try to install filters on it, in order to receive data from it. These filters are delivered to the application via the filter installation callback. When a filter is delivered, the application can either accept or reject it, by returning true or false from the respective callback.

To actually send data to other end-points, the application calls the Publish function with an item as a parameter. This results in all the installed filters being evaluated on the item. The evaluation of the filters is performed by the Contrail library, within the application's own process. If the file is matched by one or more filters, it is transferred to the client-side module via IPC by library, along with a list of destinations, corresponding to the end-points that installed.

4.1 Operations

Cloud application should be created for storing the users and their details with the user login page. For each user has a storage which can access in offline, mainly this app is used to store files, videos, others, etc., which is overloaded by android mobiles. When the users enter the login with username and password, it enters into home page of cloud application. This app consists of photo viewer, gallery, etc., where the user can store their videos and photos. This also has news reader which can be of updated news stored from the mobile. For this contrail system is used as cloud application and retrieves from the repository system by user's account.

4.2 Experimental Setup

A software application in general is implemented after navigating complete life cycle method of a project. Various life cycle processes such as requirement analysis, design phase, verification, testing and finally followed by the implementation phase results in a successful project management. The software application which is basically a web based application has been successfully implemented after passing various life cycle processes mentioned above.

As the software is to be implemented in a high standard industrial sector, various factors such as application environment, user management, security, reliability and finally performance are taken as key factors throughout the design phase. These factors are analyzed step by step and the positive as well as negative outcomes are noted down before the final implementation.

Security and authentication is maintained in both user level as well as the management level. The data is stored in Access 2000 as RDBMS, which is highly reliable and simpler to use, the user level security is managed with the help of password options and sessions, which finally ensures that all the transactions are made securely.

The application's validations are made, taken into account of the entry levels available in various modules. Possible restrictions like number formatting, date formatting and confirmations for both save and update options ensures the correct data to be fed into the database. Thus all the aspects are charted out and the complete project study is practically implemented successfully for the end users.

4.3 Communication

Between mobile devices communication said to be reliable in our framework it is plays an important role in this application. The transferring of packets may take place and that the packet size may vary for different application. This may transfer the packets frequently and then update of database should also be frequent. Then user can transfer the files or data from database to their mobile phone. This may process at various speeds and contrail system will be used to store and retrieve the data using request by user. These will be stored in cache-buffer is updated frequently which can be re-fetched by the user.

Contact Provider is used in mobile phone for maintaining the contact list where the user can send the file to other users. These said that synchronisation is needed for the contact list to avoid confusion. Application data can be transferred to other mobiles and that based on the On-demand basis. Data can be requested by the application and it is send by the coherence protocol. These data are synchronised for transferring as packets with the destination and the source.

Virtual environment is that store the data as pool of the cloud storage. These pools as storage are maintained by the third parties like hosting organisations, which they contain only the large data which cannot be processed manually. These data are replicated and can take backup when it is in offline mode. This can be of distributed storage where many number of users can store the data and retrieve their own data by contrail environment with separate storage area in the cloud. This can be of distributed system which can be distributing the data to many systems at same time.

4.4 Performance

When the data is transferred to mobile from the cloud with the low bandwidth or same bandwidth. This can be offline to transfer the data and packages are store without issues of the virtual environment. For this we introduce the Contrail system which created application package that has the user and group login. This performance can be treated as simulation and that data transfer. In this proposed system cloud – mobile environment migration takes place where the data can be shared between users. Here migration between the two systems said as end to end secure connection. This can be also said as peer to peer communication. They can share the resources between system and with the low bandwidth. The only advantage is to provide user friendly to share the resources and to store and retrieve the data form and to the cloud storage. The Contrail system creates the cloud application to view the news reader, gallery and within the storage of the virtual environment. This can be very useful for user to store the any number of files.

V. CONCLUSION AND FUTURE ENHANCEMENT

As 3G devices grow in power and functionality, the ability to share data seamlessly across them is increasingly valuable. Contrail enables data-sharing applications that are bandwidth- and power efficient, non-intrusive, privacy-aware and supports decoupled communication between devices with non-overlapping periods of connectivity. In this paper, we showed that two primary mechanisms used by Contrail filters and cloud-based relay can be used to construct applications that have these properties. We show that a Contrail implementation on the Windows Azure platform leverages the scaling ability of the cloud to support large numbers of clients while retaining good latency and throughput. The future enhancement is to provide user interface screens upgraded for touch sensitivity, social document editor for creating and editing the document collaboratively.

REFERENCES

[1] Shih-Hao Hung, Chi-Sheng Shih, Jeng-Peng Shieh, Chen-Pang Lee, and Yi-Hsiang Huang "An Online Migration Environment for Executing Mobile Applications on the Cloud" 2011 Fifth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing

[2] P. Chun, B. G. Maniatis, "Augmented Smartphone applications through clone cloud execution," in Proceedings of the 12th Workshop on Hot Topics on Operating System,2009.

[3] A.Rudenko, P.Reiher, G.J. Popek, and G.H. Kuenning, "Saving portable computer battery power through remote process execution" SIGMOBILE Mob. Comput. Commun. Rev,. vol. 2, pp. 19-26, 1998.

[4] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson A. Rabkin, I. Stoica, and M. Zaharia, "Above the Clouds: A berkeley view of cloud computing," EECS Dept, University of California, Berkeley, Tech. Rep. UCB/EECS-2009-28, Feb 2009

[5] M.Satyanarayanan, B. Gilbert, M.Toups, N.Toila, A.Surie, D.R. O'Hallaron, A.Wolbach, J.Helfrich, P.Nath and H.A. Lagar-Cavilla, "Pervasive personal computing in an internet suspend/resume system," IEEE Internet Campus., vol. 11,no. 2,2007,pp.16-25.

[6] S.Osman, D.Subhraveti, G.Su, and J.Nich, "The design and implementation of Zap: A system for migrating computing environments," in Proceedings of the Fifth Symposium on Operating System and Implementation, 2002, pp.361-376.

[7] Android Developers website,[Online], Available: http://developer.android.com/

[8] R.T.Kouzes, J.D. Myers, W.A. Wulf, "Collaboratories: doing science on the Internet," IEEE Computer, vol.29, no.8, pp40-46, Aug 1996.

[9] J.Flinn, D.Narayanan, and M.Satyanarayanan, "Self-tunedremote execution for pervasive computing," in Proceedingsof Hot Topics in Operating System, 2001, pp.61-66.

[10] T. Garfinkel and M. Rosenblum, "When virtual is harder than real: security challenges in virtual machine based computing environments," in Proceedings of the 10th conference on Hot Topics in Operating Systems - Volume 10, 2005, pp. 20–20.

[11] J. Smith and R. Nair, Virtual Machines: Versatile Platforms for Systems and Processes. Morgan Kaufmann Publishers Inc., 2005.

[12] L. Zhang, S. Deering, D. Estrin, S. Shenker, D. Zappala, "RSVP: a new resource ReSerVation Protocol," IEEE Networks Magazine, vol.7, no.5, pp.8-18, Sep 1993.

[13] I. Mahadevan, K. M. Sivalingam, "Architecture and Experimental Results for Quality of Service in Mobile Networks using RSVP and CBQ", ACM/Baltzer Wireless Networks Journal, Vol.6, No.3, 2000.

[14] A. K. Talukdar, B. R. Badrinath, A. Acharya, "MRSVP: A Resource Reservation Protocol for an Integrated Services Network with Mobile Hosts", ACM Journal of Wireless Networks, Vol. 7, 2001.