American Journal of Networks and Communications
2015; 4(4): 95-103
Published online August 3, 2015 (http://www.sciencepublishinggroup.com/j/ajnc)
doi: 10.11648/j.ajnc.20150404.13
ISSN: 2326-893X (Print); ISSN: 2326-8964 (Online)

Dependable Community-Cloud Framework for Smartphones

Arnold Adimabua Ojugo[1], Fidelis Obukowho Aghware[2], Rume Elizabeth Yoro[3], Mary Oluwatoyin Yerokun[4], Andrew Okonji Eboka[4], Christiana Nneamaka Anujeonye[4], Fidelia Ngozi Efozia[5]

[1]Dept. of Math/Computer, Federal University of Petroleum Resources Effurun, Delta State, Nigeria
[2]Dept. of Computer Science Education, College of Education, Agbor, Delta State, Nigeria
[3]Dept. of Computer Sci., Delta State Polytechnic, Ogwashi-Uku, Delta State, Nigeria
[4]Dept. of Computer Sci. Education, Federal College of Education (Technical), Asaba, Delta State, Nigeria
[5]Prototype Engineering Development Institute, Fed. Ministry of Science Technology, Osun State, Nigeria

Email address:

arnoldojugo@yahoo.com (A. A. Ojugo), ojugo_arnold@yahoo.com (A. A. Ojugo), aghwarefo@yahoo.com (F. O. Aghware), rumerisky@yahoo.com (R. E. Yoro), an_drey2k@yahoo.com (A. O. Eboka), agapenexus@hotmail.co.uk (M. O. Yerokun), anujeonyechristy@gmail.com (C. N. Anujeonye), fenngo31@yahoo.com (F. N. Efozia)

To cite this article:

Arnold Adimabua Ojugo, Fidelis Obukowho Aghware, Rume Elizabeth Yoro, Mary Oluwatoyin Yerokun, Andrew Okonji Eboka, Christiana Nneamaka Anujeonye, Fidelia Ngozi Efozia. Dependable Community-Cloud Framework for Smartphones. *American Journal of Networks and Communications*. Vol. 4, No. 4, 2015, pp. 95-103. doi: 10.11648/j.ajnc.20150404.13

Abstract: Cloud computing enable users to access ubiquitous, on-demand, convenient and shared resource (apps and storage) – as rapidly released by a provider with minimal managed effort. The increased growth of user access to mobile smartphones from 42.5% in 2013 to 78.9% by 2013 and the advent of Androids has made smartphones a preferred choice over PCs due to its design, portability, speed, functionality and Internet access ease – all of which continues to pose significant risk to user data security with high vulnerability to attacks. With its implication to work related functions and biz issues, it exposes sensitive data to adversaries. The study thus, describes a support tool named PushCloud that lets users account the ability to sign-in and perform backup functions on contacts, messages, picture files, documents, videos and recorded voice amongst others. Its other benefit is in the fact that it pools together cloud service providers and allows users a cross platform with minimal price difference. The system helps address security related issue from a user's end via AES-256 encryption on an integrated cloud model, explores its storage capability to guarantee data recovery with a remote server (BDC) for back- and front-end data storage ease.

Keywords: Stochastic, Immunize, Network, Vertices, SIS, SIR, Function, Search Space, Solution, Models

1. Introduction

Today, Android is become a leading platform for mobile devices with its open source feat that distinguishes it from most other mobile platforms such Blackberry, Windows Phone and iOS (Morril, 2010). It is not a specification or distribution of traditional Linux, neither is it a collection of replaceable components or chunk of software ported on a device. Its open source platform is built by Google with OS, middleware, and apps for mobile systems based on Linux kernel that enables developers to write apps majorly in Java with support for C/C++ (Bray, 2010). Its major success is its license that allows third-party porting developments to it. Since its release, it has been constantly improved either in feats, supported hardware, and also extended to new device types besides the originally intended ones (Maia et al, 2010). Recent efforts are to enhance real-time capabilities as employed in a variety of embedded systems (Tapas Kumar and Kolin, 2010).

1.1. Android Platform (AP)

Pernel et al (2013) and Agam (2011) in "Google Android and Linaro Android SDK" note AP is an eco-system layer of app component implemented on mobile (smartphone) hardware as thus (see fig 1):

a. Linux OS provides basic functionality such as security, process/memory management and networking to support vast device drivers. It handles human machine interfaces, file systems, network access etc. Its kernel is modified by Google to use low memory killer, specific inter-process communication system, kernel log feats, shared memory system and many other changes as developed. It runs on standard Vanilla Linux, merging specific changes into its kernel. Recent release aimed at real-time Linux kernel is v4.0.3 (Ice Cream Sandwich).

b. Library with Google's *libc* called *Bionic*, media/graphics (OpenGL|ES), browser-webkit and light-database SQLite. DVM (Dalvik Virtual Machine) completely differs from Sun's JVM and uses register based byte code to conserves memory, max performance and can instantiate many of its apps multiple times, with each app having its own private copy running. DVM uses Linux for memory management and multi-threading to support the Java language.

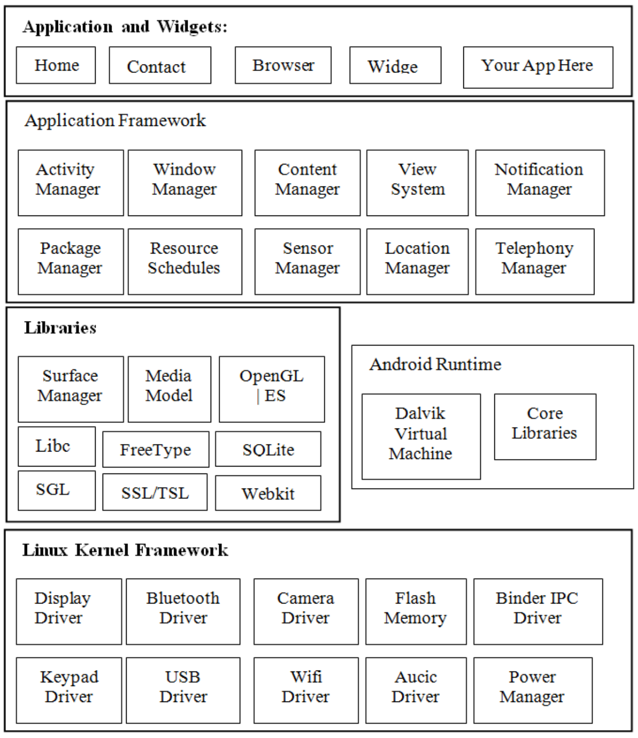


***Fig. 1.** Android OS Platform.*

DVM uses *bionic* (not compatible with *glibc*) so that its native libraries are faster to implement with small custom *pthread* to support services such as system and logging capabilities. Writable data segments are small so as to be loaded into memory with each process. This keeps code size small so that Linux loads only once, all read-only pages. *Bionic* is used: (a) to avoid inclusion of GPL code at user space level in its platform where BSD is used, and (b) for small memory footprint devices with high speed CPUs at relatively low frequencies. *Bionic libc* does not handle C++ exceptions (though omitting such lower level exceptions pose no problem as Java is Android's primary language. It handles exceptions internally). *Bionic* has no priority inheritance for mutexex as implemented in *glibc*. Available in its kernel and accessed via own library in system calls, its lack of priority inversion disqualifies it for real-time capability as applied in robotics/automotive. Google's reason for a complete new VM from scratch as accomplished with DVM's register-based byte code is to reduce patent infringement risk. Thus, existing real-time apps modified for JVM cannot easily be ported to DVM.

c. Application Framework provides higher-level services to apps such as Java classes amongst others. Its use can vary between/with varying implementation.
d. Application/Widget are Android routine distributed apps such as email, SMS, calendar, contacts and Web browser.

1.2. Literature Review

In embedded systems (automotive or robotic), its ability to meet deadlines, time constraints is a critical specification part in its design as such systems must response to stimuli within a certain pre-specified real-time constraints. Thus, the reliability of software has not to focus only on the functional failures but require and detailed evaluation of the ability of the system to meet these timing specifications (Bhupinder and Vijay, 2010).

From a device mainly used for phone calls and messages, the mobile phone (smartphone) is become a multi-purpose device. Though favored by its size, there exists thermal constraints, battery consumption and computational powers that limits it usage and capabilities. Cloud computing has the potential to transform the IT-industry. Thus, Harmen (2012) investigated the possible increase in speed of smartphones by offloading computational heavy app functions via cloud computing. He developed an app that was used to conduct computational heavy tests, and the results showed that it is not beneficial to use cloud computing to carry out these types of tasks; it is faster to use the smartphone.

Pernel et al (2013) In their test for real time behavior and performance on the Android platform – so as to make clear if Android usage be advised in open real-time environments – used for evaluation, a test suite of four performance tests namely: thread switch latency, interrupt latency, sustained interrupt frequency, and semaphore acquire-release timing in contention case, and one behavior test to checks the mutex locking behavior. Their test results showed that the Android in its current state cannot be qualified to be used in real-time environments. Finally we provide some potential solutions for using Android in such environments.

2. Cloud Computing Technology

Cloud computing is the underlying infrastructure that helps scale services exponentially and flex resources rapidly in response to variable supply and demand. Hurwitz et al (2010) It hinges on terminologies and technologies such as:

a. Cloud Services are the actual apps employed by a user to perform one or two tasks. Such as using Snapfish to share photo online, Force.com to create niche market services, NetSuite for ERP services, amongst others.
b. Multi-Tenancy Cloud services are either at software or at

infrastructure layer. Thus, many instances of software and platform it runs are made available to serves many clients. With shared resources, providers have access controls and security for a protected environment for each user.

c. Enterprise-Services (software/infrastructure) is designed to serve an enterprise' specific internal needs not limited to and includes data security, integration, configurability, access, reliability and availability.
d. Global-Services (software/infrastructure) designed for external, arbitrary and non-secure user. Software is native, multi-tenant and designed with Web 2.0 to be scalable and relies on software-based resiliency.
e. Private/Internal cloud connotes enterprise-class service with virtualized and automated infrastructure. While different from cloud-based infrastructures, they both share similar feats, and benefits from same technologies that help cloud services providers rapidly scale.
f. Elasticity allows flexibility to meet user preferences and needs on a near real-time basis, in response to supply and demand triggers. It is also ability of a service or infrastructure to adjust to users' fluctuating demands in service by automatically provisioning of resources as well as by moving the service to be executed on another part of the system.

2.1. Cloud Computing Services

Hamrén (2012) Cloud services are grouped as thus:

a. On-demand self-service: A user can unilaterally provision computing services such as network storage and server time as needed automatically without interference.
b. Network access is available of network accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms (e.g. mobile phones, tablets, laptops).
c. Pooling allows a provider's resources to serve many users on a multi-tenant model, with various physical and virtual resources, dynamically assigned/reassigned to meet users' needs irrespective of location dependence. Users have no control over the exact location of provided resources but are able to specify location at a higher level of abstraction (like as country or datacenter). Resources include storage, processing, memory, and network bandwidth.
d. Elasticity is the release of services as automatically scaled rapidly outward and inward on users' demand. To users, such capability should appear to be unlimited and can be appropriated in any quantity at any time.
e. Service resource measure can be automatic, controlled and optimized to leverage metering capability at some level of abstraction appropriate to service type such as storage, processing, bandwidth and active user accounts. Internet resource usage can be monitored, controlled and reported to provide the needed transparency for both provider and users of the utilized service.

2.2. Service Model

Hamrén (2012) Service models are grouped into three as:

a. Platform as Service (PaaS) is the capability provided to the user to deploy onto a cloud infrastructure, user-created or acquired apps using programming languages, libraries, services and tools as supported by a cloud provider. The user only has control over his deployed apps and possibly configuration settings for app-hosting environment.
b. Software as Service (SaaS) are services and apps provided to users on cloud infrastructure, accessible from client's devices via a thin client interface (email, web browser), or via a program interface. User has no control of underlying infrastructure such as network, operating system, servers, storage, or individual apps; but is limited to user-specific application configuration settings.
c. Infrastructure as Service (IaaS) is capability provided to user such as processing, storage and other resources, so he can deploy and run arbitrary software to include operating systems and apps. User only has control over operating systems, storage, deployed apps and limited control of selecting network devices (like firewalls and SSH embedded within the organization wishing to engage in cloud services. All of which is aimed at improved data security and integrity from the client-end).

2.3. Deployment Models

Deployment models deal with various forms of intrusion from adversaries with malicious intent towards data. Thus, data security must be ensured. Ureigho (2012) and Ojugo et al (2012a) various methods to improve intrusion detection on cloud infrastructure exist but clouds are deployed as:

a. Private cloud is exclusive to an organization with multiple users. It is owned, operated and managed by organization, third party, or both; and may exist on or off premises.
b. Community cloud is exclusive to a specific community of users from an organization with shared concerns such as mission, security requirements, policy, and compliance considerations. It may be owned, managed, and operated by one or more of the organizations in the community, a third party, or both; and may exist on or off premises.
c. Public cloud is made for open use by the public. It may be owned, managed, and operated by a business, academic, or government organization, or some combination of them. It exists on the premises of the cloud provider.
d. Hybrid cloud combines two/more cloud infrastructures (private, community or public) with unique entities, but bound together by standardized or proprietary technology to enables data and application portability. For example, cloud bursts for load balancing between clouds.

3. Advanced Encryption Standard (AES)

AES is a specification for the encryption of electronic data established by the U.S. National Institute of Standards and Technology (NIST) in 2001. It is based on the Rijndael cipher developed by two Belgian cryptographers, Joan Daemen and Vincent Rijmen via proposal to NIST at the AES selection process. Rijndael is a family of ciphers with different key and block sizes. For AES, NIST selected three members of the Rijndael family, each with a block size of 128-bits, but three different key lengths: 128, 192 and 256 bits. AES 192/256 is approved for top-secret data by most Governments as closely aligned with public crypto.

Ureigho (2012) Crypto knowledge in the public and foreign intelligence domains has skyrocketed, and a vulnerability that the NSA can exploit is possibly a vulnerability that someone else can exploit. Thus, drafting of AES focuses on choosing a candidate standard that though may be broken given any amount of time and data, but will prove intractable for a time. Adversaries can only break crypto when they have the keys no matter how mathematically secure the crypto is. Most adversaries focus more on key retrieval via methods like brute force by attacking the endpoints that generate the keys. Though not as hard as it seems if we consider how many user and corporate machines get infected with malware alongside the sort/range of key-related backdoors are planted in popular software), and a simple subpoena may get keys in some situations. As more user data moves toward cloud, backdoors in public services (voluntarily provided or not) are going to make the job of key recovery even easier.

Hurtwiz et al (2010) AES was initiated in 1997 by National Institute of Standards and Technology (NIST), a unit of U.S. Commerce Department search to find a robust replacement for the Data Encryption Standard (DES) and to a lesser degree Triple DES. The specification called for symmetric algorithm (same key for encryption/decryption) using block encryption of 128-bits size, supporting key sizes of 128, 192 and 256 bits, as a minimum. The algorithm was required to be royalty-free for use worldwide and offer security of a sufficient level to protect data for the next 20 to 30 years. It was to be easy to implement in hardware and software, as well as in restricted environments (for example, in a smart card) and offer good defenses against various attack techniques.

Its selection process fully subjected to public scrutiny and preliminary analysis by the world cryptographic community to decide. This will ensure the best possible analysis design via its full visibility. On this submission that saw Rijndael were other cipher families also subjected to more extensive analysis namely: (a) MARS by IBM Research team, (b) RC6 by RSA Security, (c) Serpent by Ross Andersen, Eli Biham and Lars Knudsen, (d) Twofish by a large team of researchers including Counterpane's respected cryptographer, Bruce Schneier.

Implementations were tested extensively in C and Java for speed, reliability in cryptosystem, key usage, algorithm set-up time and its resistance to various attacks (both in hardware- and software-centric systems). Detailed analysis was provided by global crypto-community, and in 2000, NIST announced Rijndael (standardized in 2001 by the Secretary of Commerce and approved Federal Information Processing Standard). Thus, all sensitive, unclassified documents will use Rijndael as AES. Table 1 shows a number of key combinations and key size.

Table 1. *Key combination and size.*

Key Size	Possible Combinations
16-bits	65536
32-bits	$4.2 * 10^9$
56-bits (DES)	$7.2 * 10^{16}$
64-bits	$1.8 * 10^{19}$
128-bits (AES)	$3.4 * 10^{38}$
192-bits (AES)	$6.2 * 10^{57}$
256-bits (AES)	$* 10^{77}$

There is an exponential increase in possible combinations as key size increases. DES is a symmetric crypto algorithm with a key size of 56 bits that has been cracked in the past using brute force attack. The argument that a 128-bit symmetric key is computationally secure against brute-force attack is proved thus: if (a) fastest supercomputer of 10.51 Pentaflops = 10.51 x 10^{15} flops (floating point operations per second), (b) flops per combination check is 1000, (c) combination checks per second = (10.51 x 10^{15}) / 1000 = 10.51 x 10^{12}, and (d) number of seconds in one Year = 365 x 24 x 60 x 60 = 31536000, then the number of year required to crack 128-AES is given by:

$$Years\ to\ crack\ 128bit\ AES = \frac{3.4 * 10^{38}}{31536000(10.51 * 10^{12})}$$
$$= 1.02 * 10^{18} = 1\ Billion - Billion\ years$$

Table 2. *Key combination and time to crack.*

Key Size	Time to Crack
56-bits (DES)	399secs
128-bits (AES)	$1.02 * 10^{18}$ years
192-bits (AES)	$1.872 * 10^{37}$ years
256-bits (AES)	$3.31 * 10^{18}$ years

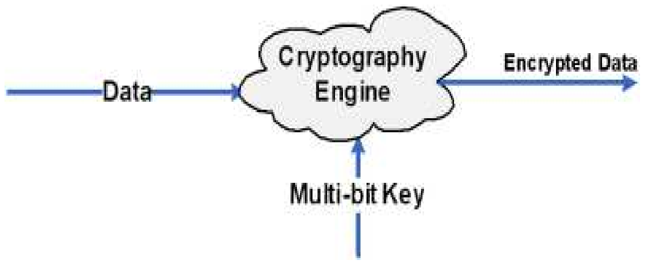


Fig. 2. *Multi-bit key cryptographic algorithm.*

AES's strength is generally expressed in length of the numeric 'key' used to scramble/unscramble messages. The study aims to display the strength of the AES against brute force attacks with different key sizes and the time it takes to successfully mount a brute force attack factoring future advancements in processing speeds. A cryptographic algorithm requires multi-bit key to encrypt the data as shown in Fig. 2.

3.1. Brute Force Attack on AES

The key length used in the encryption determines the practical feasibility of performing a brute-force attack, with longer keys exponentially more difficult to crack than shorter ones. Brute-force attack systematically checks all possible keys till correct key is found. Brute force attack on a 5-bit key is as in fig 3. This shows it takes max of 32-rounds to check every possible combination starting with 00000. Given sufficient time, a brute force attack is capable of cracking any known algorithm.

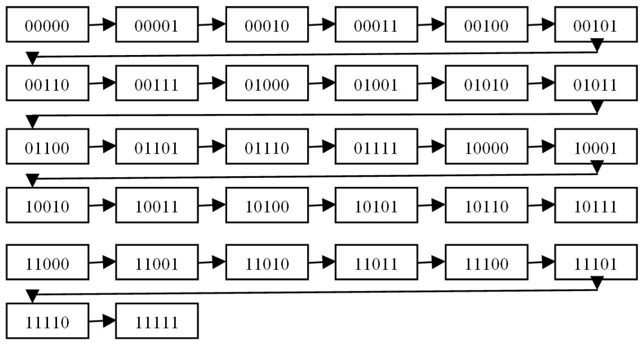


Fig. 3. *Brute force attack on 5-bit key.*

3.2. Advanced Encryption Standard in Browsers

The cipher AES-256 is used among other places in TSL/SSL across the Internet. It's considered amongst the top ciphers and in theory, it is less penetrable with its extensive combinations of keys, which is and remains massive.

Kangas (2012) SSL/TLS provide the majority of security in the data transmitted over the Internet today. Most users are unaware of the degree of security and privacy inherent in a secure connection ranging from almost none to a really good enough for US government TOPSECRET data. The cipher and encryption technique is what varies and thus provides the variable level of security needed. There are a large number of different ciphers. Some are very fast and very insecure. Some are slower and very secure. Others are weak (export-grade).

AES is a successor cipher and encryption technique to DES and was standardized in 2001 after a 5 year review. Currently, one of the most popular algorithms used in symmetric key cryptography (used for actual data transfer in SSL and TSL). It is also the gold-standard encryption method and many security-conscious organizations require its employees to use AES-256 (256-bit AES) for all communications.

This study highlights AES role in SSL, which web browsers and email programs support it, how to implement the 256-bit AES encryption on all secure communications, and more.

AES is FIPS (Federal Information Processing Standard) certified with no known brute-force attack success (except some side channel timing attacks on processing of AES that are not feasible over a network environment, not applicable to SSL in general). AES security is strong enough to be certified for use by most governments for top secret information.

Its design and strength of all key lengths (128, 192 and 256 bits) are sufficient to protect classified information up to the SECRET level. TOPSECRET data requires the use of either 192 or 256 key lengths. AES in products intended to protect national security systems and data must be reviewed/certified by NSA prior to their acquisition and use. (Hathaway, 2003)

It is often debated if 128-bit AES is computationally secure against brute-force attack. Governments and businesses place a great deal of faith in the belief that AES is so secure that its security key can never be broken. From table 2, it takes the fastest supercomputer, 1 billion-billion years to crack AES 128-bit via brute force attack. If we assume that a computing system existed that can recover DES key in 1sec, it take same machine approximately 149 trillion years to crack 128-bit AES. Though, difference in cracking AES-128bit and AES-256 is minimal as any breakthrough in 128-bit will probably render 256-bit tractable also. AES remains safe against brute force attacks contrary to belief and arguments. Its key size for encryption always be large enough despite the considerable advancements in processor speeds based on Moore's law as in fig. 3.

4. Framework and Implementation

Data stored in cloud receives malicious attempts. Clients may not understand security feats provided by Cloud Service Providers. Thus, study proposes a reliable AES encryption, employed at a client's end on community cloud to help protect data, separate from firewalls and other infrastructure in place by cloud providers. We models a security framework to make cloud "dependable" and achieve these:

a. Implement an integrated community-cloud that allows a user choice at sign-in unto the cloud infrastructure with AES-256 encryption at the client's end for improved data integrity against adversary to yield a dependable cloud.
b. Storage support for the integrated cloud with a remote server (completely transparent to user).
c. Data, at client's end is secured via AES-256 (to protect user data and message contents at end-to-end connection within the community cloud). SSL protects the username and password – alongside NAT, firewall and gateway that are implemented within the (Intranet) framework.
d. Sync, selected content in mobile devices to any available cloud technology on the model and ensure they are available anywhere and anytime you want to access or manage them which also takes out anxiety of losing important files, if device is damaged, lost or stolen.
e. Recommend suitable security models from existing benchmarks to improve user confidence in cloud computing services.

4.1. The Nigerian Front for Integrated Mobile Cloud

The framework will bring see many cloud service providers (infrastructure and services) brought together into one single user platform via mobile computing. It is an extension of the miniaturization process and faster computing on Moore's law, bringing about dependable and secure data storage capability to users via portable mobile devices.

4.2. Experimental Model Design Overview

The study provides a community-cloud model and support for users at Federal University of Petroleum Resources Effurun Nigeria. The model achieves this via a native app for mobile device (operable from Android v2.2) with support for web-service to allow Internet connectivity ease and connection to remote server and cloud-provider services via an API call (adapter). The framework will masks all technical nuances between application-model and data-models such as session management, connectivity, authentication and authorization. Its security is handled via AES-256 implemented on SSL/TSL applied to all data as backed by the cloud firewall to ensure security. Its client end-to-end encryption solution uses AES-256 to protect its data; while SSL protects username and password (see fig 4).

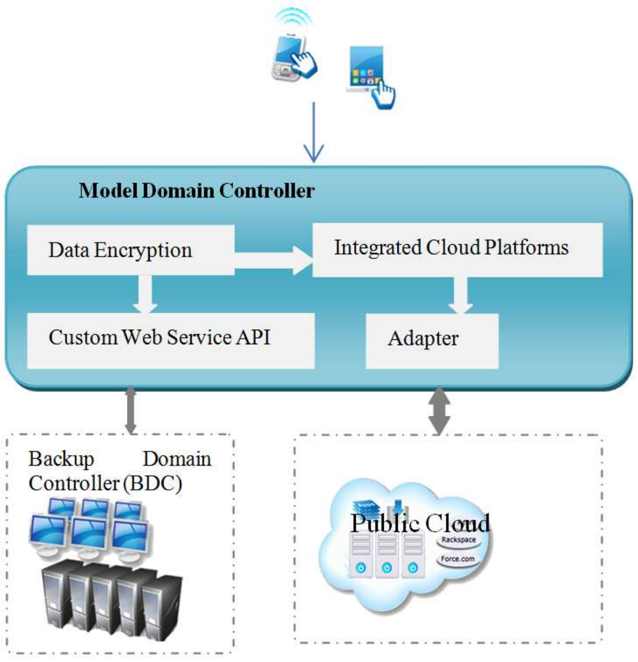


Fig. 4. *Application and Data Model for FUPRE Community-Cloud.*

Tools used for the development of this native app include Android SDK, Apache XAMPP and Google's Android Studio. This native app is enabled and ported on any Android platform from v2.2 with forward compatibility.

The system implements AES-256 encryption as supported by SSL/TSL. It is adopted because: (a) it is computationally, mathematically secure against brute force attacks, (b) quite flexible, (c) its small-size Java codes and support for C(++/#), (d) memory size required is small as ported on AP. Thus, has no effect on smartphone speed and performance, and (e) ease of integration as implemented with Java and support for C-language into its web browsers with ease of connectivity.

Web-browser used includes Safari, Firefox and Netscape – all of which enable AES-256 encryption on SSL/TLS protect data transfer between user and server. However, data transfer over the Internet between the sender and recipient remains unprotected, no matter how good SSL in use is.

4.3. System Implementation / Snapshots

Implementing the app on Android emulator with many of its snapshots as in Appendix, lets a user to download the native app and create an account so as to be able to sign-in and perform backup functions on contacts, messages, picture files, documents, videos and recorded voice amongst other backups and/or synchronization. After which, the user on first backup is notified to choose the cloud platform (whichever choice is made comes with its many benefits but the price difference is minimal). The technical nuances are not discussed such as billing etc. The app then performs AES encryption of user data before they are backed up too either to the remote server (backup domain controller – BDC) or to the integrated cloud provider's server via a cloud service API call (adapter).

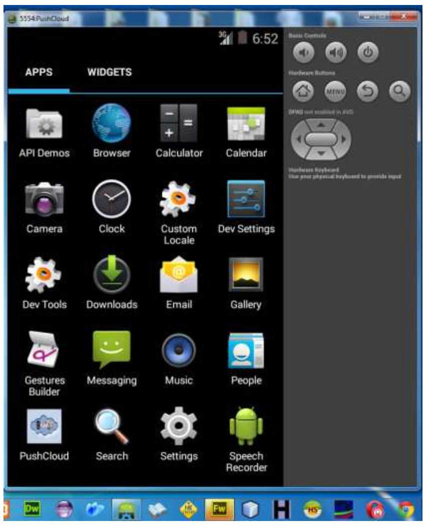
4.4. Rationale for Cloud Implementation

AES is secure, its data encryption is more mathematically efficient, is elegant cryptographic algorithm with a model whose strength resides in the key-length options. Time required for an intruder to crack algorithm is directly proportional to length of key used to secure data transfer or communication. AES allows a choice between 192 and 256-bit implementation, making it exponentially stronger. There are no significant tradeoffs in functionality, speed and memory – as implementation on Android Studio makes it quite portable for target device. Required memory is relatively small and does not affect device speed even with extra functionalities. Target Android OS is v2.2 with forward compatibility to v4.0. This will bring closer to users, cloud technology with its many benefits at a cheap price.

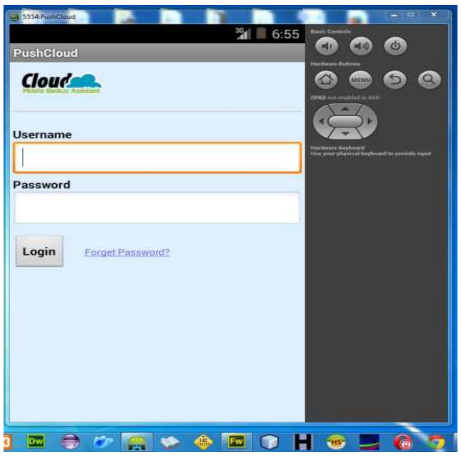
5. Conclusion

The incessant need of users to protect stored, online data continues to foster the field of Data Forensic, which aims at measures to help detect network intrusion alongside keeping such adversaries off-bay via biometrics and cryptography so as to achieve the needed data non-repudiation, confidentiality, security and integrity for client end-to-end transaction. Cloud infrastructure is a system that enables 5-essential feats namely: self-provision, pay-per-use, on-demand resources availability, scalability and resource pooling. It consists of a physical layer of hardware resources necessary to support cloud services (such server, storage and network devices) and an abstraction layer of software-deployed on physical layer to manifests as cloud feats such as virtualization, grid computing, outsourcing and utility computing. Thus, study yields an integrated cloud on Android platform for smartphones as motivated by the need to proffer clients' transaction the much required security.

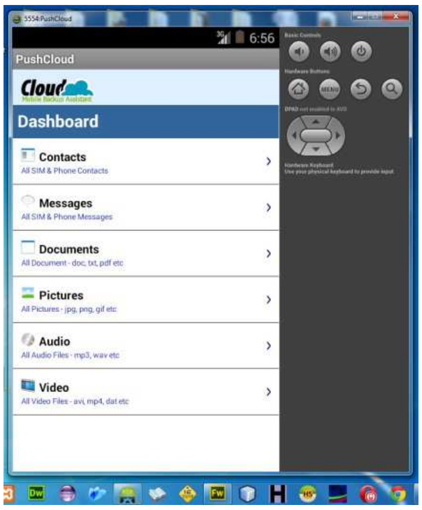
Appendixes



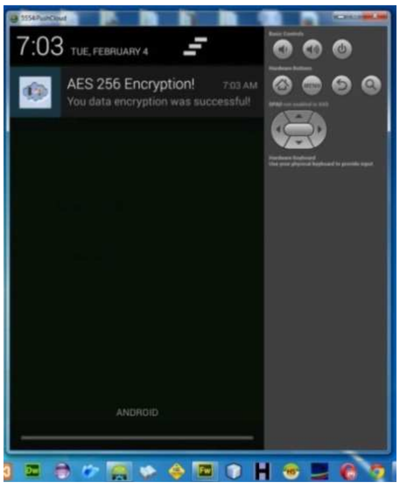
Appendix A. *Snapshot of PushCloud App installed on Android.*



Appendix B. *User Account Login on PushCloud.*

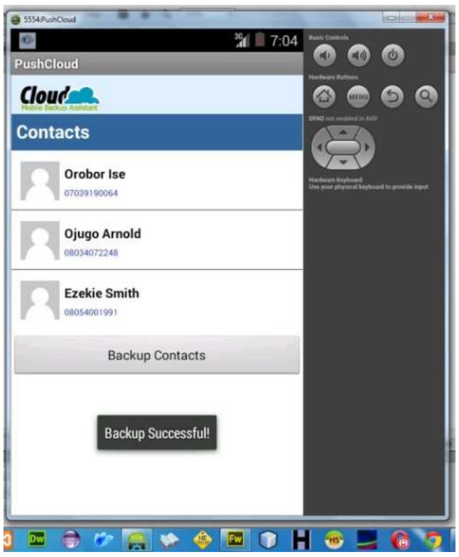


Appendix C. *PushCloud Dashboard with Menu for Contacts.*



AES 256 Encryption! 7:03 AM
You data encryption was successful!

Appendix D. *Contact backup with AES Encryption first.*



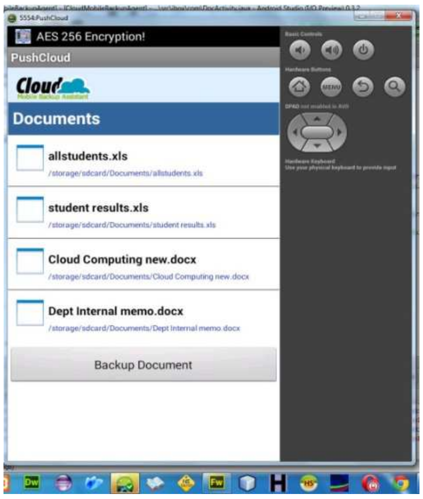
Orobor Ise
07039190064

Ojugo Arnold
08034072248

Ezekie Smith
08054001991

Backup Successful!

Appendix E. *Contacts Backup successful on PushCloud.*



AES 256 Encryption!

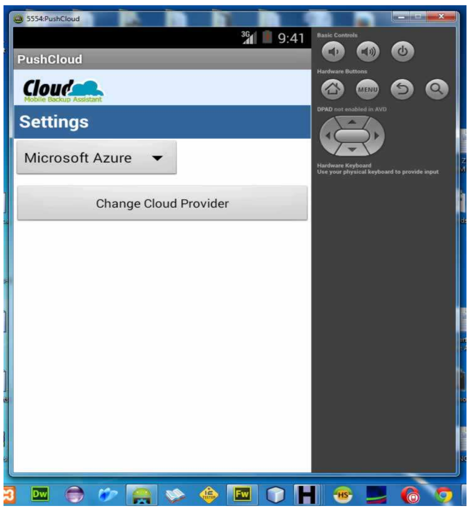
allstudents.xls
/storage/sdcard/Documents/allstudents.xls

student results.xls
/storage/sdcard/Documents/student results.xls

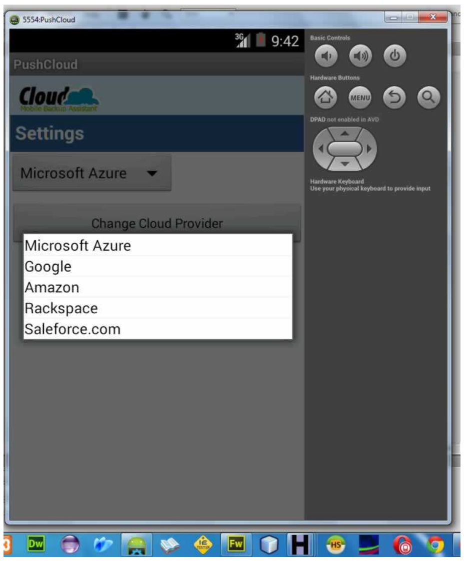
Cloud Computing new.docx
/storage/sdcard/Documents/Cloud Computing new.docx

Dept Internal memo.docx
/storage/sdcard/Documents/Dept Internal memo.docx

Appendix F. *PushCloud Integrated setting for User Choice of Cloud Infrastructure to backup Data.*



Appendix G. *Dropdown Menu of PushCloud Integrated setting for User Choice of Cloud Infrastructure to backup Data.*



Appendix H. *Documents List with Backup Option.*

References

[1] Agam, S., (2011). Google's Android 4.0 ported to x86 processors, retrieved online via: http://www.computerworld.com/s/article/9222323/Google_s_Android_4.0_ported_to_x86_processors.

[2] Alonso, G., Rellermeyer, J and Roscoe, T., (2011). R-osgi: Distributed applications through software modularization, IFIP Lecture Notes in Computer Science (LNCS), 4834, p1–20.

[3] Android, Linaro Android Build Service, https://android-build.linaro.org/ last accessed January 2014.

[4] Bhupinder, S.M and Vijay, K.M (2010). Reliable Real-Time Applications on Android OS, retrieved online via: www.users.ece.gatech.edu/~vkm/Android_Real_Time.pdf.

[5] Bray, T., (2010). Ongoing by Tim Bray-What Android Is, www.tbray.org/ongoing/When/201x/2010/11/14/What-Android-Is.

[6] Chun, B., Ihm, S., Maniatis, P., Naik, M and Patti, A., (2011). Clonecloud: Elastic execution between mobile device and cloud, Proc. of 6th ACM conference on Computer systems, p301–314.

[7] Cohen, R., (2010). The cloud computing opportunity by the numbers. www.elasticvapor.com/2010/05/cloud-computing-opportunity-by-numbers.html.

[8] Cole, B., (2012). Real-time Android: real possibility, really hard to do - or just plain impossible? www.embedded.com/electronics-blogs/cole-bin/4372870/Realtime-Android/real-possibility-really-really-hard-to-do-or-just-plain-impossible.html.

[9] Crysta X, Crysta X. NET, [online] www.crystax.net/nl/android/ndk/7. Last accessed January 2014.

[10] Dinh, H.T., Lee, C., Niyato, D and Wang, P., (2012). A survey of mobile cloud computing: architecture, applications and approaches, Wiley Wiresless Communications and Mobile Computing, http://onlinelibrary.wiley.com/doi/10.1002/wcm.1203/abstract.

[11] Divya, V.L., (2012). Mobile application with cloud computing, Int. J. of Scientific Research Publications, 2(4), p1-6.

[12] Google, Android SDK, http://developer.android.com/sdk/index.html.

[13] Giurgiu, I., Riva, O., Juric, D., Krivulev, I and Alonso, G., (2009). Calling the cloud: Enabling mobile phones as interfaces to cloud applications, Proc. of ACM/IFIP/USENIX 10th Int. Conf. on Middleware. Springer-Verlag, p83–102.

[14] Guo, Y., Zhang, L., Kong, J., Sun, J., Feng, T and Chen, X., (2011). Jupiter: Transparent Augmentation of Smartphone Capabilities through Cloud Computing, ACM Transaction on Mobiheld, Portugal: Cascais, ACM-978-1-4503-0980-6/11/10.

[15] Gupta, P and Gupta, S., (2012). Mobile Cloud computing: future of cloud, Int. J. Adv. Res. in Electrical, Electronics and Instrumentation Engineering, 1(3), p134.

[16] Hamrén, O., (2012). Mobile phones and cloud computing: A quantitative research paper on mobile phone application offloading by cloud computing utilization, Master’s Thesis, Dept of informatics, Human Computer Interaction SPM 2012.07, UMEA University.

[17] Hurwitz, J., Bloor, R and Kaufman, M., (2010). “Cloud computing for dummies: HP special edition”, Wiley publications, New York.

[18] IBM, (2014). Inside the Linux 2.6 Completely Fair Scheduler, [Online]. www.ibm.com/developerworks/library/lcompletely-fair-scheduler/.

[19] Jeong, S., Zhang, X., Kunjithapatham, A and Gibbs, S., (2010) Towards an elastic application model for augmenting computing capabilities of mobile platforms, Mobile Wireless Middleware, Operating Systems, and Applications, p161–174.

[20] Kalkov,I., Franke, B and Schommer, J., (2012). A Real-time Extension to the Android Platform, In Proceedings of the 10th International Workshop on Java Technologies for Real-time and Embedded Systems, Copenhagen, Denmark.

[21] Krishnan, S., (2010). Programming Windows Azure. O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

[22] Kumar, K. & Lu, Y. (2010). Cloud computing for mobile users: can offloading computation save energy? IEEE Computer Society.

[23] Lu, Y., Li, S and Shen, H., (2011). Virtualized screen: A third element for cloud-mobile convergence, IEEE Multimedia, 18(2), p4–11.

[24] Maia, C., Nogueira, L and Pinho, L.M., (2010). Evaluating Android OS for Embedded Real-Time Systems, Proceedings of 6th International Workshop on Operating Systems Platforms for Embedded Real-Time Applications, Brussels, Belgium.

[25] Marinelli, E.E., (2009). Hydrax: Cloud computing on mobile devices using MapReduce, Masters' Thesis, School of Computer Science, Carnegie Mellon University, Pittsburg, CMU-CS-09-164.

[26] Marrapese, B., (2010). Google ceo: a few years later, the mobile phone becomes a super computer. http://www.itnews-blog.com/it/21320.html.

[27] Mei, L., Chan, W and Tse, T., (2008). A tale of clouds: paradigm comparisons and some thoughts on research issues, IEEE Asia-Pacific Services Computing Conference APSCC'08, p464.

[28] Ojugo. A.A., Orobor, A.I., Yoro, R.E and Aghware, F.O., (2012a). Dependable community cloud model implemented using model view controller: a case of FUPRE, Technical report on cloud Technologies (FUPRE-Tech-03-2012), p11–24.

[29] Ojugo. A.A., Eboka. A.O, Okonta, E.O, Yoro, R.E and Aghware, F.O., (2012b). Implementation issues of VoIP for rural telephony in Nigeria, J. Emerging Trends in Computing and Info System, 4(2), p172.

[30] Ojugo, A.A., Eboka, A.O and Yoro, R.E., (2013a). Technical issues for IP-based network in Nigeria, J. of Wireless Communications and Mobile Computing, 2(2) p43-50.

[31] Ojugo, A.A., Eboka, A.O., Yerokun, M.O., Iyawa, I.J.B and Yoro, R.E., (2013b). Cryptography: salvaging exploitation against data integrity, Int. J. Networks and Communications, 2(2), p47-55, doi: 10.11648/j.ajnc.20130202.14.

[32] Pernel, L, Fayyad-Kazan, H and Timmerman, M., (2013). Android and real time application: take care, J. Emerging Trends in Computing and Info. Systems, 4, Special Issue ICCSII, ISSN 2079-8407.

[33] Rittinghouse, W.J and Ransome, F.J., (2010). Cloud Computing implementation, management and security. Boca Raton, FL: CRC Press

[34] Reese, G. (2009). Cloud Application Architectures. O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

[35] Sarna, D.E.Y., (2011). Implementing and developing cloud computing applications. Taylor and Francis Group, Boca Raton, FL: CRC Press.

[36] Satyanarayanan, M., Bahl, P., Caceres, R and Davies, N., (2009). The case for vm-based cloudlets in mobile computing, IEEE Pervasive Computing, 8(4), p14–23.

[37] Shetty, K and Singh, S., (2011). Cloud Based Application Development for Accessing Restaurant Information on Mobile Device using LBS, Int. J. UbiComp, 2(4), DOI:10.5121/iju.2011.2404 37.

[38] Sung, A., Xu, J., Chavez, P., Mukkamala, S., (2004). Static analyzer of vicious executables, Proceedings of 20th Annual Computer Security Applications Conf., IEEE Computer Society, p326-334.

[39] Tapas Kumar, K and Kolin, P., (2010). Android on Mobile Devices: An Energy Perspective, IEEE 10th International Conference on Computer and Information Technology, Kuala-Lumpur, Malaysia.

[40] Ureigho, R.O.J., (2012). Dependable cloud computing: a framework for secure cloud, Unpublished PhD thesis, Department of Computer Science, Ebonyi State University Abakiliki, Ebonyi State.

[41] Vinutha, S., Raju, C.K and Siddappa, M., (2012). Development of hospital management system utilizing cloud computing and Android OS using VPN connections, Int. J. Sci. Tech. Res, 1(6), p59.

[42] Zhang, X., Kunjithapatham, A., Jeong, S and Gibbs, S., (2011). Towards an elastic application model for augmenting the computing capabilities of mobile devices with cloud computing, Mobile Networks and Applications, 16(3), p270–284.