A RECONFIGURABLE CONTEXT-AWARE SECURITY FRAMEWORK FOR MOBILE CLOUD COMPUTING

by

Saurabh Dey

Submitted in partial fulfilment of the requirements
for the degree of Doctor of Philosophy

at

Dalhousie University
Halifax, Nova Scotia
July 2018

To my beloved Parents,

and

my wife Meenakshi

Table of Contents

List of Tables

List of Figures

Abstract

Cloud computing has become an important solution to today's data processing problems due to its significant advantages in terms of storage, scalability, and cost. Inclusion of mobile devices in a cloud computing environment is an emerging paradigm and is referred to as mobile cloud computing. This paradigm enables mobile devices, such as smartphones and laptops, to utilize the computation power and storage in the cloud. However, mobile cloud computing involves several modes of communications that are governed by varied security standards, which makes them susceptible to a variety of different attacks. If the communication is compromised, then data security will be at risk and the services enabled by cloud infrastructures will be disrupted.

The primary objective of this thesis is to propose a novel reconfigurable context-aware security framework for mobile cloud computing, which can be deployed in a cloud of cloud environment to provide an additional layer of security. In many application areas security mechanisms are highly heterogeneous, while the cloud server is common to these applications. Therefore, we contend that the proposed framework should be deployed at the cloud premises. The context-aware framework provides varied techniques to improve the quality of service and security of mobile cloud computing using three significant modules. The cognitive module serves as an access control layer to perform clustering-based learning and traffic filtration. The adaptive module operates as a software application. It is responsible for initiating virtual machines, offering cloud services, and self-healing of potentially compromised clouds. The authentication module performs light-weight mutual authentication using locations, timestamp, and message digests.

We have used a novel clustering algorithm KD(K-Means and Density-based spatial clustering of applications with noise) to train the cognitive module with data sets consisting of current and previous inter-packet delays. Distance-based identification of anomalous traffic is performed to ensure effective filtration. The adaptive module is designed with a probabilistic model and validated using the PRISM model checker. Three proposed message digest based authentication schemes are evaluated using the protocol analyzer, Scyther. Our experimental results indicate that the proposed framework can withstand various attacks.

List of Abbreviations and Symbols Used

AE	specifies an asymmetric encryption algorithm used in AMLT
$Auth\_Key_i$	authentication key used in MDA
BF	Bit Flipping operation used in AMLT
DS	digital signature of cloud used in MDA
E	Encryption operation
KD	K-Means DBSCAN
MD	Message digest generated in MDA by hashing two message digests
MD_{cloud}	message digest used in MDA, it consists cloud policy and cloud certificate
MD_{user}	message digest used in MDA, it consists user policy and user certificate
Pk_{pub_cloud}	cloud's public key used in MDA
SE	Specifies a symmetric algorithm
SI	state identifier used in MDA
T_k	symmetric key used in MDA
V_{actual}	number of potential vulnerabilities
$V_{potential}$	number of potential vulnerabilities
V_{score}	vulnerability score
$\#CF$	column reference used in MDA
δe	Indicates the symmetric encryption alg. used in AMLT selected by client, such as AES
δh	Hash function choice parameter for AMLT, and MDLA, which defines the type of hash function used by both parties. E.g. SHA1, MD5 etc.
δl	Indicates location vector of the mobile device. It is a logical OR of the latitude and longitude
δp	indicates the choice of pseudo random number generator. E.g. Blum Blum Shub etc. used in AMLT, and choice of PRNG used in MDLA
δt_{new}	client's current timestamp used in MDLA
δtc	Timestamp of the cloud client used in AMLT

δts	Timestamp of the cloud server used in AMLT
δt	defines the current time-stamp of the mobile device during registration used in MDLA
ϵ	A flag, which indicates if client registration is expired
λ	Flag specifies, whether incoming traffic is accepted
ϕ	indicates the geographical location of mobile device during registration used in MDLA
ϕ_{new}	client's current location used in MDLA
ϕ_{prev}	previous location of the mobile device used in MDLA
$auth_{key_i}$	fresh authentication key used in MDLA
$c - rp$	client record pointer, Column reference in big table at the cloud server where each client's information is stored in AMLT operation
$c - rp_{scd}$	Secured client record pointer, which is encrypted using server public key
$h\{cid\}$	hashed client id used MDLA
$h\{pwd\}$	hashed password used MDLA
$hash\{password\}$	hashed password used in MDA
$hash\{userID\}$	hashed userID used in MDA
k	indicates number of clusters created by K-Means, and number of nearest-neighbour in kNN
$k_{prv-server}$	private key of the key pair used by cloud server in AMLT
$k_{pub-server}$	public key of the key pair used by cloud server in AMLT
$k_{sym-C2S}$	symmetric key used by cloud client used in AMLT
$k_{sym-S2C}$	symmetric key used by cloud server used in AMLT
m_{auth_req}	authentication request
m_{auth_res}	authentication response
$m_{re\_reg\_req}$	re-registration request from cloud server to mobile client used in MDLA
m_{reg_req}	registration request used in MDLA, and AMLT

m_{reg_res}	registration response message used in MDLA, and AMLT
$m_{reg\_upd\_req}$	Registration update request message from cloud to the client
md_{client}	Message digest or hashed message of each mobile client's certificate generated by the cloud server used in scheme MDLA, and AMLT
p_{mn}	Representation of $profile$ used in traffic filtration
$password$	user password used in MDA
$prim_{key_i}$	subsequent primary keys used in MDLA
$prim_{key}$	symmetric key termed as primary key used in MDLA
prv_{cloud}	cloud's private key used in MDLA
ref	column reference used in MDLA
$sec_{key_{new}}$	new secret key used in MDLA
sec_{key}	secret key used in MDLA
$userID$	userID used in MDA
AES	Advanced Encryption Standard symmetric key algorithm
AMLT	Authentication based on Message digest, Location, and Timestamp
BSI	British Standard Institution
CPN	Customer Premises Network
DBSCAN	Density-based spatial clustering of applications with noise
DCS	Digital Cross-connect System
DTMC	Discrete-Time Markov Chain
EPC	Electronic Product Code

FOS	Fast Orthogonal Search
FTP	File Transfer Protocol
HTTP	Hypertext Transfer Protocol
IaaS	Infrastructure as a Service
ICT	Information and Communications Technology
IDG	International Data Group
IEEE	Institute of Electrical and Electronics Engineers
IMSI	International Mobile Subscriber Identity
IoT	internet of things
IPD	Inter Packet Delay
IPDV	Inter Packet Delay Vector
kNNdist	k-Nearest Neighbour Distance
MAC	Media Access Control
MDA	Message Digest based Authentication
MDLA	Message Digest and Location based Authentication
OS	Operating System
PaaS	Platform as a Service
PCA	Principal Component Analysis
PRISM	Probabilistic Symbolic Model Checker
PRNG	Pseudo Random Number Generator
RFID	Radio Frequency Identification
RSA	Rivest, Shamir, and Adelman Algorithm
RSU	Road Side Unit
SaaS	Software as a Service

SHA1	Secure Hash Algorithm 1
SOM	Self Organized Map
SONET	Synchronous Optical Networking
SSE	Sum of Squared Error
SSO	Single sign-on
TCM-KNN	Transductive Confidence Machines for K-Nearest Neighbours
TCP/IP	Transmission Control Protocol/ Internet Protocol
USIM	Universal Subscriber Identity Module
V2I	Vehicle to Infrastructure
V2V	Vehicle to Vehicle
VANET	Vehicular Ad-hoc Network
WBAN	Wireless Body Area Network
WSN	Wireless Sensor Networks
WWW+	World Wide Wisdom

Acknowledgements

The journey of my Ph.D. was a roller coaster ride, which could not have been successfully completed without the support from both of my supervisors Dr. Srinivas (Srini) Sampalli, and Dr. Qiang Ye. I would like to express my sincere gratitude to them for accepting me as a Ph.D. student, sharing their domain knowledge, motivating me at every stage of the study, and providing constructive feedback on my research work. Without their support this thesis would not have been completed. My sincere gratitude goes to my committee members, Dr. Nur Zincir-Heywood, and Dr. Musfiq Rahman for providing valuable feedback, which helped me improve my research.

My sincere appreciation goes to Dr. Yacine Ghamri-Doudane, from University of La Rochelle Institute of Technology, France, for agreeing to be my external examiner. His valuable questions, and suggestions made this thesis better.

Further, I would like to thank my friend Dr. Raghav V. Sampangi and my colleagues at the MYTech Lab, who have supported me in this academic journey.

Last but not least, I would like to thank my family - my wife Meenakshi, who is my support system. Without her love and sacrifices my Ph.D. would not have been possible. I express my sincere gratitude to my parents, Swapan and Chitralekha for giving me life, hope, and inspiration. I extend my thanks to elder brother, Samrat, and sister, Shatabdi, who believed in me and always made me feel special. I would also like to thank all my family members for their blessings.

Without thanking God, the supreme power this list cannot be ended. Finally, I thank God for giving me the opportunity to obtain the highest degree.

Chapter 1

Introduction

1.1 Overview

Mobile cloud computing facilitates anytime, anywhere remote accessibility to users with a plethora of services, such as IaaS (infrastructure as a service), PaaS (platform as a service), and SaaS (software as a service) etc. [82, 30]. These services have helped many small and mid size business organizations in reducing their infrastructure costs as the pay per use model is less expensive than purchasing new hardware or storage devices. Since mobile devices lack sufficient computational power and battery life, it is desirable to perform much of the resource intensive tasks at the cloud server. The mobile device can then be used mainly as a medium for transferring data or as a viewer for visualizing the final results. Mobile cloud computing can help a user perform a complicated task, such as big data analytics remotely irrespective of the geographical location.

IDG (International Data Group) enterprises recent study on cloud usage [45] indicates that 72% of the organizations have at least one cloud application running or they are utilizing some portions of the cloud infrastructure. The market of cloud usage is growing and many business players are leaning towards cloud computing.

International Data Corporation [2] has released a quarterly data of the cloud revenue for private and public clouds based on the data available from IDC's Worldwide Quarterly Cloud IT Infrastructure Tracker. This tracker provides information about the number of servers, storage disks and Ethernet switches that are deployed in the cloud environment. According to IDC, the vendor revenue of sales of cloud IT infrastructure products grew 21.9% year over year to $29.0 billion in 2015. Table 1.1 highlights the cloud IT infrastructure vendor revenue and market share growth for the fourth quarters of 2014 and 2015.

Cloud services have been growing rapidly over the past years. Gartner predicts by

Table 1.1: IDC, April 2016 fourth quarter revenue data for 2014 and 2015 [41]

Vendor	4Q15 Revenue (US$M)	4Q15 Market Share	4Q14 Revenue (US$M)	4Q14 Market Share	4Q15/4Q14 Revenue Growth
1. Hewlett Packard Enterprise	$1,304	15.80%	$1,112	15.60%	17.20%
2. Dell	$845	10.20%	$667	9.40%	26.70%
2. Cisco	$802	9.70%	$591	8.30%	35.50%
4. EMC	$759	9.20%	$634	8.90%	19.70%
5. IBM	$352	4.30%	$345	4.80%	2.10%
ODM Direct	$1,927	23.40%	$2,138	30.00%	-9.90%
Others	$1,328	27.40%	$1,640	23.00%	37.80%
Total	$8,248	100%	$7,126	100%	15.70%
Revenues are in Millions, Excludes double counting of storage and servers					

2020, cloud adoption strategies will impact more than 50 percent of the IT outsourcing contracts [49]. Individuals and organizations are moving towards cloud based platforms for cutting edge technologies, faster deployment, on-demand scalability, and usage based payments. According to Gartner, (Tab 1.2), the cloud market has approximately increased from $209 billion to $246 billion in 2017. The overall growth of revenue indicates the increased usage of cloud resources.

Although the cloud market is largely dominated by software as a service, infrastructure as a service has gained a lot of attention lately. Demand of computational power, storage space, efficiency etc. are becoming the key requirements of individuals considering the use and growth of resource constrained smart devices, and internet of things (IoT). The cloud market segment is seeking highly scalable cloud based infrastructure resources, and consequently, infrastructure as a service has a noticeable rate of growth. In a published article, KPMG [32] presents Wikibon's analysis on the revenue growth of public cloud services (Fig. 1.1), which shows SaaS, and IaaS have a growing market compared to other types of cloud services.

Table 1.2: Worldwide Public Cloud Services Forecast (Millions of Dollars)[49]

	2016	2017	2018	2019	2020
Cloud Business Process Services	40,812	43,772	47,556	51,652	56,176
Cloud Application Infrastructure Services	7,169	8,851	10,616	12,580	14,798
Cloud Application Services	38,567	46,331	55,143	64,870	75,734
Cloud Management and Security Services	7,150	8,768	10,427	12,159	14,004
Cloud System Infrastructure Services	25,290	34,603	45,559	57,897	71,552
Cloud Advertising	90,257	104,516	118,520	133,566	151,091
Total Market	209,244	246,841	287,820	332,723	383,355

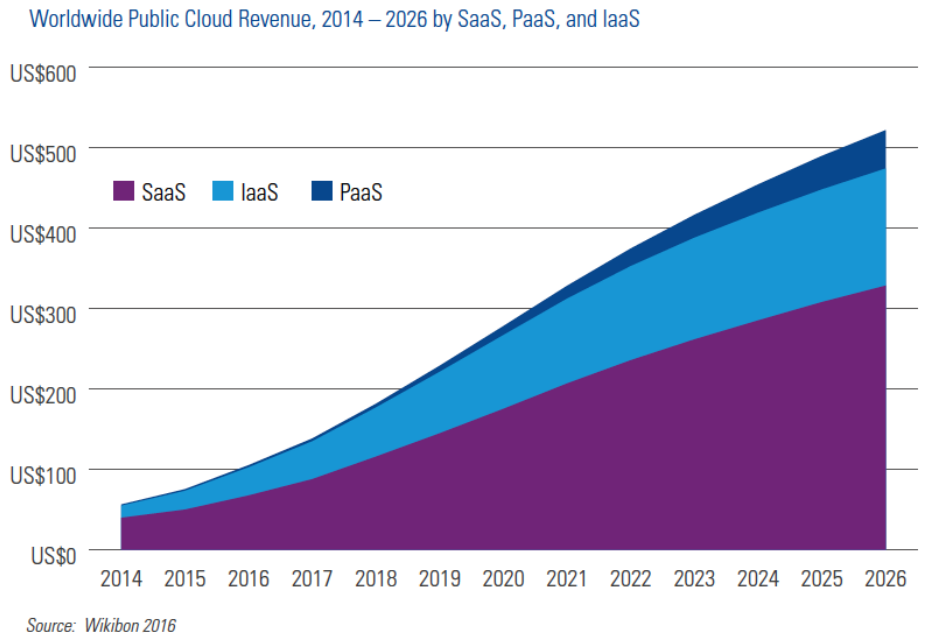

Figure 1.1: Public cloud vendor revenue projections

In addition, the usage of mobile devices, such as smartphones, tablets, and sensors is increasing every year. Fig. 1.2 [110] shows the total number (in millions) of smartphone users and projected smartphone users worldwide from 2014 to 2019. The increased usage of smartphones and cloud resources has triggered the use of mobile cloud computing. Online gaming, audio-video streaming, online storage, web browsing, social networking are various forms of mobile cloud computing that are performed by smartphones or tablets. According to a study conducted by Cisco (2016), by 2020 the mobile data traffic is expected to reach 30.6 Exabytes (one Exabyte = 1018 bytes) per month at a CAGR (compound annual growth rate) of 53% from 2015 to 2020.

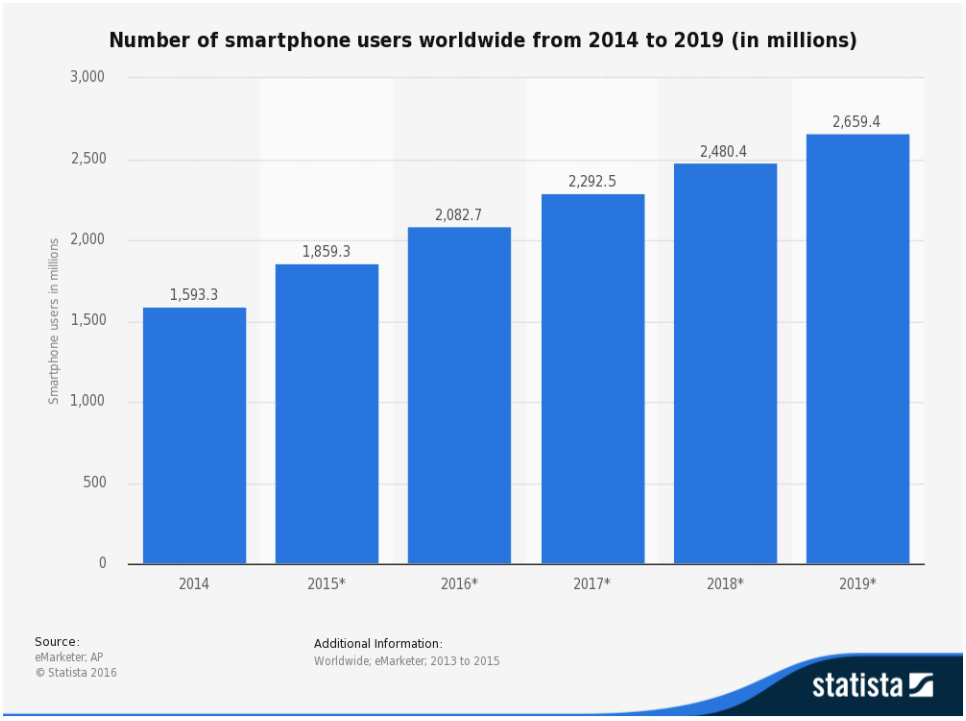

Figure 1.2: Smartphone users (in millions) worldwide from 2014 to 2019 (Statista.com 2016) [41]

Mobile cloud computing has been considered as one stop solution for today's data processing need. Mobile cloud computing could be deployed to integrate citywide critical information infrastructure that could provide services to a large set of customers in a cost-effective way. However, security and privacy stand out as critical issues in the design and deployment of mobile cloud computing. These are especially important in smart city operations, such as mobile health care, power distribution etc. that

are characterized by sensitive and time-critical information transfers, thereby requiring not only guaranteed information transfer security but also data and user privacy. This design challenge stems from several factors. Firstly, a common characteristic of mobile devices is their severe resource constraint, since such devices may consist of small sensors or chips with limited computational power and bandwidth. This makes the deployment of sophisticated and advanced cryptographic algorithms infeasible. Secondly, deploying security has always been a bigger challenge in mobile devices as compared to their wired counterparts, due to their inherent free space and broadcast transmission, as well as protocol vulnerabilities. Thirdly, non-uniform traffic, unpredictable topology changes, differing node densities, high degree of mobility, and high bit-error rates dictate communication between mobile devices and the cloud. Despite these challenges, addressing the security and privacy issues of mobile cloud computing is highly desirable to guarantee a secure and robust functioning of a mobile cloud computing environment, which is a core technology of a smart city.

The research proposes a multi-tier context-aware framework with a scalable cloud of cloud model for enhancing the security of mobile cloud computing. The context-aware framework consists of three reconfigurable modules. The cognitive module performs two phase traffic filtration using pattern matching, and clustering techniques. The adaptive module performs management of inner cloud i.e. virtual machines and hypervisors. The authentication module performs mutual authentication between mobile clients and cloud server. To highlight the significance of the research, smart city framework and its various applications are considered as a case study. The proposed framework is designed to support large scale heterogeneous and dynamic mobile topologies to ensure security, reliability, and privacy of the system.

1.2 Existing Challenges in Mobile Cloud Computing

Cloud computing is a well-defined framework, which is existing in the technology domain from past couple of years. Offloading data to unlimited storage, remote accessibility, anytime availability, data encryption, affordability etc. have made the cloud computing framework a business need. However, researches indicate, cloud computing has various security threats, which are covered in Chapter 2. In this section, some broad security challenges are presented to highlight the importance of

cloud security research.

(a) Data Loss - Offloading data to the cloud server is one of the primary features of cloud computing, which enables resource constrained devices, such as a mobile phone to perform resource intensive tasks without installing any additional components. Oftentimes, concurrent transfer of data from mobile nodes or sensors to cloud storage requires data transformation, which triggers the possibility of data loss. A small portion of data loss might cause serious issue, if the data is sensitive [127]. In addition, if disaster recovery is not implemented or data backup is not performed, then any serious damage to the server may cause significant data loss [11, 63].

(b) Vendor Lock-in - Cloud service providers use their proprietary techniques and methods to store and compress data at the remote server [91, 103]. This is performed to ensure data security, data transformation, and resource utilization. However, the reverse transformation of the cloud client data to universally accepted format is not always easily available. As a result, oftentimes the cloud client cannot switch the service provider, which binds the client to the service provider for an extended period. This is a serious issue, and the cloud providers should follow a common data format, transformation and storage techniqes.

(c) Insider Threat - Cloud clients' data, and cloud infrastructure are accessible by cloud employees, and that provide opportunities for various nefarious activities [71]. Clients use cloud infrastructure for processing and storing various sensitive data. If data is compromised by the cloud employees, then clients will not have enough knowledge of the data breach, and the consequences could be catastrophic in terms of privacy and sensitivity of any information. Trustworthiness of cloud employees depend on the hiring process, which is a policy issue.

(d) Ignorant Cloud Users - All cloud clients are not well conversant with security and privacy settings of the cloud infrastructure, and that makes the clients accounts vulnerable to various types of attacks. Oftentimes people access cloud resources using untrusted networks, and devices, which compromise any sensitive data transfer or users accounts information. Due to ignorance or lack of technical

knowledge, users refrain from updating security settings on the cloud server, and that acts as a weak-link. In addition, if cloud users share their account with others, then the privacy gets compromised, and sometimes that leads to a security issue [47, 67, 6, 115].

(e) Denial of Service - In denial of service attack, the server ports and network bandwidth are occupied by illegitimate requests for an extensive period of time[16, 50]. This issue degrades the performance of a cloud server, and oftentimes the server goes down.

(f) Malware Injection - Malware injection on the cloud server may initiate various types of attacks. An intruder could launch a service module on a cloud instance to eavesdrop all incoming requests and gain knowledge about a client and extract sensitive information. Malware injection may block or deny a legitimate client request [104]. In addition, an adversary may launch a malware service module on the cloud server to provide services (SaaS, PaaS) to a legitimate client and launch man-in-the-middle attack.

(g) Man in the Middle - Man-in-the-middle attack is one type of security threat, where data is relayed or possibly altered during communication by an adversary. Now-a-days, numerous sensitive information are transmitted from mobile devices to cloud servers, such as health records, financial transactions, traffic data etc., which require higher degree of security and privacy. Any weak-link in the communication channel, or the cloud server could expose the data to an adversary, who is actively eavesdropping. Therefore, periodic security audit and data integrity check are essential [27].

Data loss could be avoided by selecting lossless compression techniques, vendor lock-in could be avoided by using fast and free interoperable techniques, insider threat could be avoided by screening cloud employees, and cloud users could be educated to understand the security and privacy need. In order to detect and prevent malware injection, denial of service, and man-in-the-midddle attack, an intrusion detection system or security framework could be employed, which can protect any mobile cloud computing environment. Therefore, the primary focus of this thesis is to design a

security framework, which could be adopted by any cloud provider or mobile cloud computing environment, such as wireless body area network-based smart health system, sensor network-based smart grid infrastructure etc.

1.3 Mobile Cloud Topologies & Apps

Mobile cloud computing could be conceptualized as a network topology or an application, which involves cloud infrastructure, and mobile devices, such as sensor nodes, laptops, mobile phones, tablets, etc. RFID (Radio Frequency Identification), WSN (Wireless Sensor Networks), WBAN (Wireless Body Area Network), VANET (Vehicular Ad-hoc Network) etc. are examples of some important mobile cloud topologies, which could be used as core technologies in the design of various critical infrastructure solutions. To address issues, such as security, reliability and privacy, it is important to understand the different possible configuration and protocols involved in the mobile cloud computing environment.

Radio Frequency Identification

In an RFID-based system, a reader reads data from a tag and sends it to the server for processing, which follows EPC (Electronic Product Code) as RFID standard [93]. The RFID reader could be a mobile device and the server could be part of a cloud infrastructure, which qualifies as a type of mobile cloud computing architecture that uses TCP/IP (Transmission Control Protocol/ Internet Protocol), EPC etc. This architecture may suffer from different types of security issues that are associated with RFID systems and might cause erroneous data processing. Fig. 1.3 shows a hypothetical mobile cloud computing setup with multiple RFID networks. An erroneous or altered data may cause serious security issues.

In Fig. 1.3, "RFID Network 2" is designed for child's safety, where each child is attached to a RFID tag. Data sent from RFID tags are received by the RFID reader installed in the school bus. The RFID reader then sends the data to the cloud server for processing using a middleware, which could be a mobile device. If data is compromised during the transmission, the cloud server may receive incorrect information about a child. This is a serious issue for any critical infrastructure, such as health services. Therefore, it is important to address these issues.

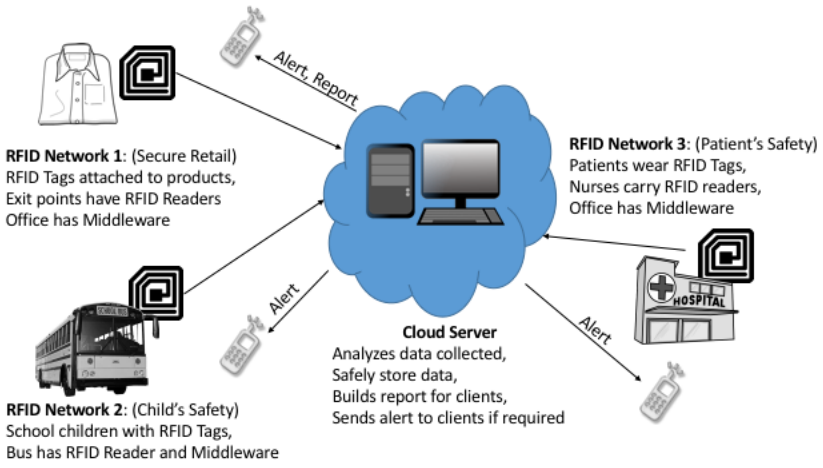

Figure 1.3: Example of Mobile cloud computing topology with multiple RFID networks

Wireless Sensor Networks

Wireless sensor networks can be deployed to collect and transmit environment information of heterogeneous applications, such as monitoring temperature in a data centre, measuring humidity, temperature, and pH in agriculteral land etc.[101, 68].

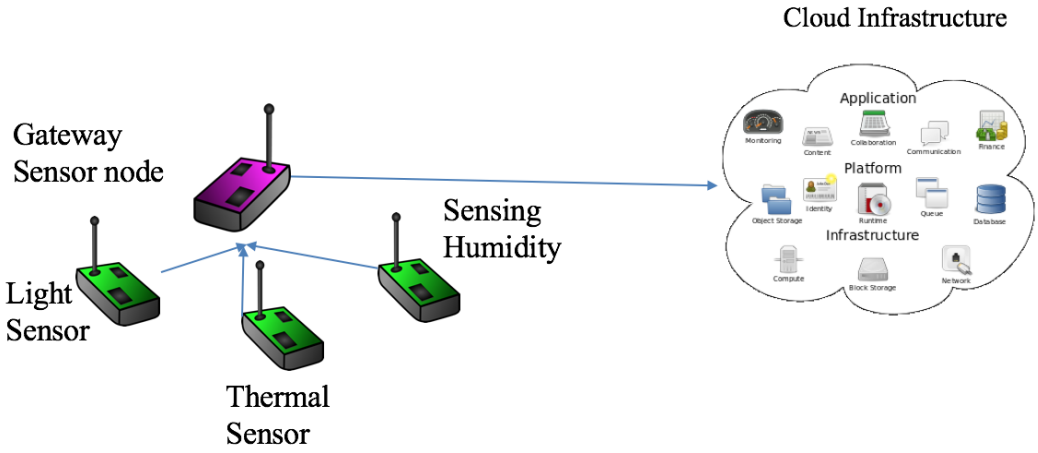

Figure 1.4: Example of Mobile cloud computing topology with wireless sensor network

A WSN can be conceptualized as a mobile cloud computing environment, where raw data is collected using various sensor nodes, and transmitted to a remote cloud

server for further processing using one or more intermediate gateway node(s) (Fig.1.4). Numerous application areas can be designed using wireless sensor networks, such as water, and air monitoring system [114]. Sensor nodes of a WSN collecting sensitive information, such as nuclear power plant data require protection against unauthorized access. Compromising a WSN controlling critical information infrastrucure may cause service disruption, such as power failure, and shutting off water supply.

Wireless Body Area Network

Typically in WBAN, wireless sensor nodes are deployed in human body to gather health parameters such as, body temperature, pulse rate etc. The collected information could help to setup a health monitoring system that sends information periodically from sensor nodes to a base station, and then from the base station to a hospital unit [102]. The aspects of WBAN, such as reliability and power consumption, are defined by the IEEE 802.15.6 standard [74].

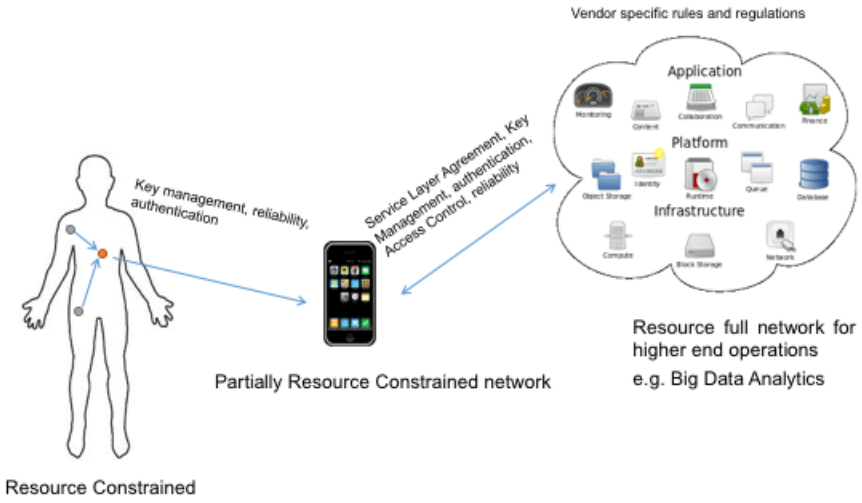

Figure 1.5: Example of Mobile cloud computing topology with wireless body area network

In WBAN (Fig. 1.5), a mobile device, such as a cell phone could be used as the base station and a cloud infrastructure could be used as a hospital unit. The

cloud server is used to process or store the biometric parameters. Since, heath parameters are highly sensitive, WBAN requires protection from unauthorized access or modification. Fabricated health parameters and transmission delay in WBAN may cause loss of human life. Privacy in healthcare is another major concern, therefore, a WBAN for smart healthcare should be implemented with state-of-the-art mobile technologies, and security protocols.

Vehicular Ad-hoc Network

VANET could be used for road traffic management, where each vehicle acts as a fast moving mobile node. A specific type of VANET is V2I (vehicle to infrastructure) [26] wherein, a vehicular on-board chip sends vehicle information to a RSU (road side unit), and the RSU forwards it to the cloud infrastructure for processing. It could be considered as a mobile cloud computing environment, since all the vehicles act as mobile nodes and communicate with different RSUs at different timestamps to send information to the cloud infrastructure for processing. VANET has a highly unstabilized network architecture, which imposes security threat to the network. Fig. 1.6 displays a typical VANET configuration, where V2V (vehicle to vehicle) and V2I are shown. Compromising VANET data may cause traffic disruption, which may lead to other serious problems, such as delay in hospitalization of patient due to traffic congestion.

Mobile Apps

In a mobile cloud computing environment, mobile devices run applications by utilizing resources from multi-tenant remote cloud servers [111]. Mobile platform involved in the mobile cloud known as a thin client as it stores data on the cloud server and accesses cloud services. One such popular cloud service is SaaS, which allows a thin client to perform complex computations on a remote server through light-weight mobile interfaces or apps. There are apps available to download, which allow a user to access other types of cloud services, such as IaaS, PaaS etc. Cloud based gaming is becoming popular now-a-days, which allows remote players to participate in a gaming environment [29]. Offloading gaming processes on cloud servers reduce the resource utilization in mobile devices. Mobile apps allow multiple remote players to connect

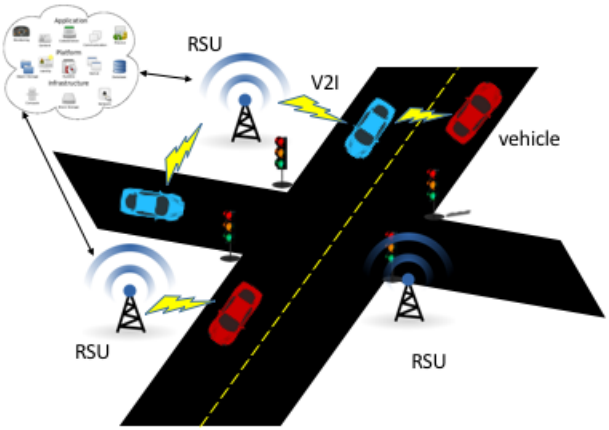

Figure 1.6: Example of Mobile cloud computing topology with vehicular ad-hoc network

simultaneously to cloud based gaming engines and establish gaming networks. These apps are designed to download only specific features that are required to run the games. Furthermore, these apps often perform financial transactions, which require enhanced security and privacy settings in the apps. In the context of smart city, mobile apps play a vital role in critical infrastructure information. Apps can provide real-time traffic updates, weather information etc. from cloud server, and can connect to wearables, smart home sensors, and hospitals etc. In other words, apps enable a thin client to obtain various services, which can improve the quality of life. However, weak encryption algorithms, shady privacy policies, and apps from unknown publishers can cause serious damage to a thin client.

1.4 Motivation

Emerging wireless technologies have introduced a plethora of communication technologies, and devices, such as sensors, smart phones, wireless base stations, RFIDs etc., which are used in mobile cloud computing. However, many of these communication technologies are susceptible to various types of attacks, such as Man-in-the-middle attack and DDoS attack etc. The motivation of this research is derived

from various case studies and probable future scenarios that include existing security threats in mobile cloud computing, and smart city applications.

1.4.1 Securing Mobile Cloud Computing

The focus of our research is security in mobile cloud computing, which can be defined as a computing environment, where mobile devices communicate with cloud services for data processing. The aforesaid services are crucial for a mobile user who accesses cloud infrastructure for data (e.g. big data) analytics and visualization. Since mobile devices lack sufficient computational power and battery life, it is desirable to perform much of the resource intensive tasks at the cloud server. The mobile device can only be used as a medium for transferring data or as a viewer for visualizing the final result. Mobile cloud computing can help a user perform a complicated task, such as big data analytics remotely irrespective of geographical location. Security plays a major role in this kind of environment as it may cause a potential damage to privacy, if user information or transmitted data are compromised. Therefore, it is very important to address security and data integrity issues between a mobile device and cloud server during authentication and data transmission.

1.4.2 A Case Study on Smart City Initiatives

Smart City Initiative

There is no single standard definition for a smart city. One of the key focus of smart city is, sustainable economic development with investments in traditional communications, ICT (Information and Communications Technology), human, and social capital [22]. Every city has its particular need, which should be satisfied to improve the city's health. If a city lacks in health care, education and sanitation, then the smart city objective of that particular city should include improvement of those areas. It is important to understand, how a city could be transformed into a smart city with the current economic and social standards. BSI (British Standard Institution) provides smart city development guidelines in PD 8101 [51] for the city planners. These guidelines help city planners to address key issues in smart city development and minimize the cost of smart city development. However, these guidelines are applicable for cities

in context of United Kingdom.

The BSI (British Standard Institution) [21] provides PAS 181 Smart city framework as a guideline to develop and deliver smart city strategies. The smart city framework indicates various guidelines. The leaders and executives should have a clear vision of the city development. Furthermore, an effective communication and collaboration between stakeholders and other cities are required to integrate the different physical, and digital services. Transformation of the operating model is essential to meet business needs and improve the quality of life. A phased smart city development is advisable to consider feedback from the citizen and stakeholders.

On April 26-28, 2016 many countries participated in the EUREKA Innovation Week [1] to discuss about smart city development and global collaboration. Canada partnered with European countries to co-innovate in smart city solutions and to explore smart city market opportunities. While many developed countries are participating in smart city planning, a large number of developing countries are struggling with the basic ICT infrastructure. Statistics published by The World Bank shows, the internet penetration in a developing country, such as India is 18%, whereas in a developed country, such as Canada is 87.1% [14]. In many countries, smart city initiative is affected by the socio-economic condition, which needs to be improved to automate critical service delivery and to implement smart city solutions.

The definition of a smart city varies from country to country and largely depends on the socio-economic condition of a country. In many developing countries, ICT development is lagging, thereby preventing the use of technologies in the critical infrastructure. The primary focus of a smart city development is to harness the ICT, drive economic growth, and improve the quality of life.

Smart city applications in the context of a developed country involve use of ICT for efficient operations of city's critical infrastructure, such as power distribution, water supply, waste management, pollution control, and traffic management etc.

Applications

A plethora of applications could be deployed in a smart city environment. Smart city applications involve various forms of mobile cloud computing technologies and applications. Some of the smart city applications are listed below.

Smart Home

A number of applications could make a home smarter, such as automation of home appliances and utilities. A timer based home lightning system could be installed to turn on or turn off lights depending on the time of the day. An automated garden watering system could activate the water sprinkling process depending on the values of soil moisture sensor. An automated security alarm could inform the city security services if there is any trespassing in the house [33].

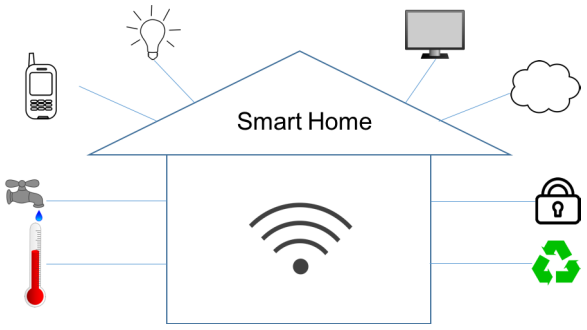

Figure 1.7: An example of smart home applications

Smart Power Supply

Growing urban population demands an expansion of power distribution network. In addition, modern health care and commercial setup require an uninterrupted power supply. Therefore, it is important to isolate faulty areas and re-route the power distribution remotely. Smart power ensures a remote monitoring of the power distribution network to identify pre-fault areas, perform energy savings and provide on demand power supply [3]. Furthermore, real-time grid monitoring can detect power leakage or temporary service disruption.

Smart Water Supply

Water consumption is increasing with the population growth. To safeguard water resource from contamination, theft or leakage water resource monitoring is needed [57]. Effective use of mobile technologies and cloud infrastructure can provide useful information, such as water consumption per household, and commercial establishment, which can be used to achieve a consumer specific water distribution system. A

real-time monitoring can detect leakages in the water distribution network, and alert the water supply board. In short, a smart water supply ensures a safe and optimized water distribution in a smart city.

Smart Traffic Management

Smart traffic management is one of the important concepts of a smart city design, which is partially implemented in many cities around the world. Smart traffic management ensures an efficient traffic movement, less congestion, and reduction of road accident cases [4]. Deploying mobile technologies, such as ip camera, sensors can provide real-time traffic updates to cloud server, which can be used in traffic analytics. A smart traffic system can predict future collisions and identify safe routes.

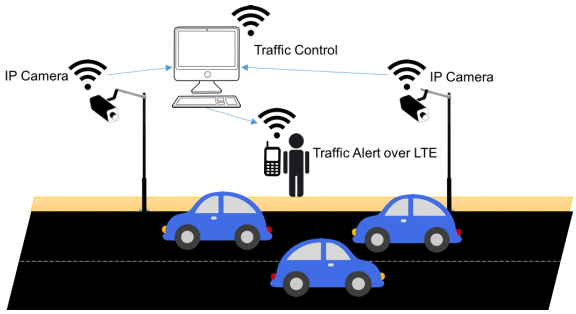

Figure 1.8: An example of smart traffic management

Smart Healthcare System

Smart healthcare integrates various emerging wireless technologies to provide medical care to patients anytime-anywhere. A patient can stay at home while his/her vitals can reach to the Doctor with the help of mobile technologies. A Doctor can provide medical care to a patient at distant location over video monitor or virtual assistance [55]. In addition, data analytics on digital health records provide useful insights into health care, such as determining the corelation between depression and diseases, identifying common health issues in aging population etc. Hospitals and cities are in the process of implementing smart healthcare system to provide quality patient care in a timely manner.

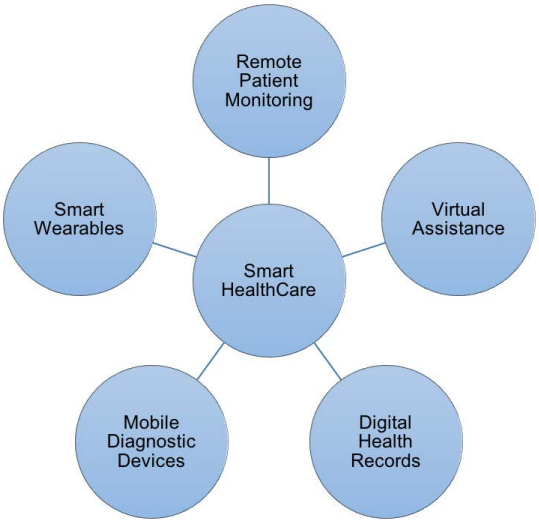

Figure 1.9: Dimensions of a smart health care system

Discussion

To improve the quality of urban life mobile cloud computing could be deployed to deliver various critical services, such as water supply, power distribution, traffic management, health care etc. A perfectly arranged ICT enabled critical service delivery model in an urban area brings in the concept of a "smart city". Efficient and optimized service delivery with a centralized control and real time monitoring of critical infrastructures are the key features of a smart city. Although the mobile nodes and cloud computing infrastructure of a smart city framework help in efficient service delivery, the arrangement does not guarantee a robust smart city operation. Weak security mechanisms and poor privacy policies could cause serious damages to a smart city framework. In this section, three major issues are identified. Firstly, service disruptions due to compromised cloud. If a DDoS attack is launched on the central cloud infrastructure of a smart city framework, then it could shut down all the services that the smart city offers. Secondly, unavailability of critical resource due to compromised smart city application. If a smart grid is compromised, then there could be power failure for a long period. Finally, information leakage or alteration due to compromised node. If a node in a WBAN is compromised it could cause alteration or leakage of personal health records.

From the aforesaid smart city definition and applications, it is clear that the combination of mobile technologies and cloud infrastructure or mobile cloud computing plays a major role in smart city planning, implementation and operation. Critical issues, such as security, reliability and privacy in mobile cloud computing could disrupt the critical services of a smart city. Therefore, it is important to study mobile cloud computing topologies and apps to identify potential areas of improvements of smart city applications.

In context of smart city, to prevent critical service disruption, and to maintain privacy, a standard security framework is required, which could support various types of mobile cloud computing environments, such as WBAN (Wireless Body Area Networks) for smart health care, VANET (Vehicular Ad-hoc NETworks) for smart traffic, RFID (Radio Frequency IDentification) for critical assets‘ tracking etc. A smart city applications are heterogeneous in terms of communication protocols and service criticality, the topologies are static and dynamic depending on the applications, and oftentimes participating nodes are resource constrained in nature. Therefore, it is difficult to deploy a standard security framework for all the applications running in a smart city. Considering the fact that smart city applications are heterogeneous in nature, we propose a context-aware security framework to ensure security and privacy of a smart city operation. The modules of the framework are deployed on different layers of the cloud server to avoid alteration on the applications or clients' networks. The context-aware feature of the framework ensures application specific multi layer security, cognitive learning and cloud selection. Chapter 4, describes the framework in detail.

1.5 Thesis Contributions

We consider various autonomous operations of smart city applications as one of the primary motivations of this research. Since major cities around the globe are initiating smart city framework, and mobile cloud computing is the underlying technology of critical information infrastructure service delivery, any vulnerbalities in the mobile cloud computing environment could cause major service disruptions. Lack of stardizations, interoperability of cloud vendors, data loss, data integrity, data security are the key issues found in mobile cloud computing environment, which could

affect any smart city or critical infrastructure operations. In addition, DDoS, and MITM attacks on WBAN, VANET, sensor network-based smart power grids etc. can cause catastrophic damages. Therefore, the security aspect of mobile cloud computing draws our special attention.

The thesis contributes towards the proposal and design of a security framework, which could be supported by multiple cloud vendors to ensure interoperability, layered security, reconfigurability to support cloud vendors' infrastructures, and clients' security needs. Motivated by Artificial Intelligence, and self-healing systems, we propose a security framework named "reconfigurable context-aware security framework for mobile cloud computing", to use in a cloud of cloud model [39]. The three modules of the framework ensure security of sensitive information, and maximum uptime of the cloud server.

The cognitive module, which is inspired by learning system and artificial intelligence ensures filtration of incoming traffic without the need of human intervention, writing access control lists, ports and IP monitoring etc. Cluster based approach (KD algorithm) is proposed for training the system, and pattern based incoming traffic filtration. Instead of TTL, ports, IP addresses and payload size, we use vectors consisting previous and current inter packet delays as the training set [24]. This helps to identify if there is a change in pattern of inter packet delays of a client requests, and a possible DDoS attack could be prevented.

The adaptive module, which is inspired by self-healing service delivery models ensures maximum uptime of cloud services by preventing cloud resources from possible MITM and DDoS attacks. Cross VM attacks and malware injection could cause data leakage from a VM [100, 61]. Therefore, we propose a VM switching, inner cloud or hyperviosr switching using the adaptive module. We conceptulaize the system as a DTMC (Discrete Time Markov Chain) process, and evaluated its functionality by exploring the probability of VM shutdown during a possible attack.

The authentication module is designed to use device independent features for mututal authentication, which is different than device specific and one way authentication [120, 5, 98, 125]. The research proposes three novel authentication schemes, which can perform mutual authentication of mobile client, and cloud server using message digest or hashed information, location vector, and timestamp [37, 38, 40]. The

authentication schemes use symmetric key encryption during authentication process, which consume less mobile resources. In addition, message digest based authentication eliminates the need of entering user ID and passwords, or any authentication pattern. We evaluate the authentication schemes by determining the vulnurability score (V_{score}) using protocol analyzer *Scyther*, which indicates the proposed authentication schemes can withstand various potential attacks.

The modules of the context-aware framework are loosly coupled, which makes the system reconfigurable, and scalable. The framework allows the cloud vendors to modify the three modules, such as, adding a new traffic filtration layer in the cognitive module, updating encryption and hashing algorithms for the proposed authentication schemes, or replacing a proposed authentication scheme with a new authentication. In addition, the VM switching logic of adaptive module can be modified depending on the security and service requirements of the cloud provider. Furthermore, the reconfigurability feature of the context-aware framework allows the cloud vendors to retain their existing security infrastructure.

1.6 Outline

The following chapters in this thesis are organized as follows. Chapter 2 focuses on the key points from some of the literature we used in our research as related work, chapter 3 highlights the research objectives and states the proposed measures. In chapter 4, we describe our proposed context-aware framework, and we explain the performance evaluation in chapter 5. Chapter 6 summarizes our research outcome through discussion. Finally, chapter 8 states the concluding remarks and highlights the scope for future work.

1.7 Chapter Summary

- With emerging wireless technologies, developing economies, and societies, the concept of smart city has evolved.
- Mobile cloud computing acts as the core technology of a smart city.
- Mobile cloud computing provides a scalable architecture, which could be utilized

in data analytics, storage or complex computations.

- Mobile cloud computing provides the services of a typical cloud computing environment, such as SaaS, PaaS, and IaaS etc. The ease of use, mobility, and capabilities of the end hosts (e.g. sensors, smartphones etc.) makes the mobile cloud computing special.

- For an effective operation of a smart city, a robust, reliable, and secure mobile cloud computing environment is desirable.

- Mobile cloud computing involves in a critical operation often deals with sensitive information, which needs to be protected against unauthorized access or alteration.

- The security and privacy of a mobile cloud computing environment deployed in a smart city operation need to be addressed.

- The thesis proposes a context-aware framework for mobile cloud computing or any smart city application.

- The context-aware security framework could be adopted by any cloud provider or mobile cloud computing environment, such as wireless body area network, smart grid etc. This framework has three modules:

  - The cognitive module performs two phase traffic filtration using clustering techniques and pattern matching.

  - The adaptive module manages the inner clouds and virtual machine initialization.

  - The authentication module performs mutual authentication between the mobile client and cloud server.

Chapter 2

Related Work

2.1 Overview

This chapter is dedicated to discuss the literature study used in the research. Depending on the context, the literatures are categorized into different themes. The first section provides literatures focusing on issues related to mobile cloud computing, since mobile cloud computing is the underlying technology of various applications, such as VANET, WBAN, or a smart city framework. The proposed context-aware security framework is conceptualized based on many well-established research domains, such as intrusion detection systems, cognitive learning, self-healing netowrks etc. Therefore, in the second section, we provide literatures on intrusion detection systems. The third section covers literatures on cognitive learning, which inspired us in the design of cognitive module. The fourth section covers literatures related to classification and cluster analysis, as clustering techniques are used for training the proposed learning system, which performs traffic filtration. The last section of this chapter covers literatures on self-healing networks as our design of adaptive module is inspired by the concept of self-healing.

2.2 Security Issues

This section covers some existing studies and researches that are carried out in the area of cloud computing security and mobile cloud computing security. Although, we consider RFID, WBAN, and VANET to describe our motivation of proposing the framework, it is essential to know, what are the different types of security threats exist in a cloud computing environment and what are the solutions suggested by other researchers.

2.2.1 Security Issues in Cloud Computing

The study done by Popovic et al. [94] outlines the security and privacy issues faced by cloud service providers. Cloud resources run on remote locations, and clients access resources through virtual machines. Although each client runs an individual virtualized environment, sharing the same physical resources can impose a security threat. In addition, each cloud service provider uses different storage structures, which may prevent the clients from switching from one vendor to the other. There is no data integrity standard followed by the cloud service providers and the ownership of control for encryption/decryption is not standardized.

Behl [17] provides a general study on security challenges in cloud computing, which covers various issues beginning from insider threats at the cloud premises to outside malicious attacks including data loss, and service disruption. Threats from employees of a cloud service provider is more of a policy issue, which could be addressed by changing the recruitment process. Many technical issues, such as data loss and service disruption may occur due to the maximum number of open interfaces provided by the cloud server. Our concept of cloud of cloud provides data abstraction to some extent by separating cloud infrastructure that interfaces with the clients and clouds that provide services to the clients. Hence, reduces the chance of data loss or service disruption.

Mathisen *et al.* [82] explain the different policy issues related to cloud computing. If employees of the cloud hosting company are careless and cannot be trusted, then it implies that there is an inherent vulnerability in the service offered. Similar to Popovic et al. this study also mentioned a potential major future concern with vendor lock-in [94]. Since there is no common standard for APIs (application programming interfaces), storage images and disaster recovery, portability from one vendor to another is an expensive operation. Other than these issues, software used in cloud computing is also an important part, which is vulnerable to external attacks. So, by avoiding the use of only open source software in server virtualization, the vulnerability can be reduced.

Yandong et al. [124] focus on different types of clouds such as, public, private and hybrid cloud, and the security issues associated with the public cloud. Public cloud is accessible to a large majority of the general population without much restrictions.

This, coupled with the fact that public cloud has plenty of user information, implies that the public cloud is susceptible to attacks. Furthermore, virtualization of the platform introduces security threats. Regulating administrator privileges is required for better authentication, security and privacy. If the virtualization platform is compromised, it implies that a majority of the virtual machines are under attack, which is a potential threat to data security.

Chopade et al. [24] describe a specific type of cloud security threat, which is known as DDoS (distributed denial of service). In the paper, they provide distance based DDoS attack detection technique. DDoS attack is more harmful than a normal DoS attack and a cloud infrastructure could be brought down by overwhelming requests from multiple locations. Our cloud of cloud concept and the cognitive learning module can help in minimize the probability of getting affected by a DDoS attack. In addition to DDoS, there are number of security issues exist within the cloud server, such as security of hypervisors, virtual machines, etc.

To address the above-mentioned issues, it is essential to have a standard cloud computing design and usage policy, employee trust, proprietary software for virtualization, and finally physical security of server. However, data transmitted from a mobile device to a cloud server is still susceptible to several security issues, which will be addressed in the following section.

2.2.2 Security Issues in Mobile Cloud Computing

Mobile cloud computing is introduced to offload resource intensive computations to a cloud server. Therefore, in addition to mobile device and communication related security issues, mobile cloud computing also suffers from cloud related security issues. Huang et al. [58] provide general overview of mobile cloud computing and highlight some of the security challenges that are found in that domain. Mobile cloud computing has many advantages that include saving battery power of mobile device, providing additional processing power for searching, data mining and storage etc. However, it suffers from security issues, such as security and privacy of user data in the public cloud. Huang et al. propose a secure data processing framework that involves trust management and isolation of private data.

Kim et al. [69] describes the mobile cloud security issues related to malicious programs that are injected in a mobile cloud virtualization environment. These malicious programs can harm a virtualized machine, and spread to the other. An affected VM can cause information leakage and data loss, which is a serious problem for a cloud user.

In addition to data loss, and information leakage, authentication is another major issue found in the mobile cloud computing. Many studies highlight authentication related issues between the mobile devices and the cloud servers. Use of weak encryption techniques and selection of weak keys make an authentication process weak. Yoo et al. [125] propose cellular automata based lightweight scheme, which is used for multi-user authentication in the cloud environment. The authentication is performed using a one-time password, which makes the session establishment secure. However, transmitting the seed without encryption makes their scheme weak and vulnerable to attacks.

Jana et al.[61] describe the various types of security threats that can appear in a mobile cloud computing environment. Accessing enterprise data through mobile devices brings in privacy and security issues, e.g. location services of a mobile device helps an adversary to predict the user's behaviour and movement. They focus on various threats related to AAA (Authorization, Authentication, and Accounting), such as malicious insiders, sniffing, spoofing, identity theft etc. As a control measure they suggest digital signature as used in SAML (Security Assertion Markup Language), security gateways, SSL, encryption, hashing. With the help of fishbone analysis, they present, security and privacy threats associated with a mobile cloud computing environment.

Nguyen et al. [86] provide a deep learning approach to detect cyberattacks in mobile cloud computing environment. They claim to achieve an accuracy rate of 97.11% in detecting attacks. The research performs feature analysis by extracting abnormal attributes from packets. They use PCA (Principal Component Analysis) technique for dimension reduction and selection of optimal features. The learning process is an offline operation and includes multiple layers. The detection phase is online and the implementation in real devices are pending. The learning phase of this scheme is longer, and requires pre-processing of data and features, which increase the

running cost.

Revar et al. describe the usefulness of single sign-on in a mobile cloud computing environment [98]. Single sign-on (SSO) is used on the top layer of cloud. They verified the single sign-on authentication scheme on an Ubuntu server. In SSO, a single user name and password combination is used to access multiple cloud applications. This scheme does not address exactly how a mobile device or any cloud compatible device will access the cloud using SSO.

Ahmad et al. [5] propose a security framework that is based on the subscriber identity module. The authentication is done at boot-time, and is based on USIM (universal subscriber identity module) response to a random challenge issued by the cloud authentication service. The cloud authentication service has a database to store a list of services, mapping of users and their related services, mapping of IMSIs (International Mobile Subscriber Identity) . This scheme works only with mobile devices that support USIM, which is a drawback considering mobile devices that do not support USIM, such as sensors or laptop computers. In addition, there are a few security threats associated with USIM based authentication [112]

Choi et al. [120] describe authentication using profiling. This includes the user information and the service information. However, implementing complex security algorithms and protocols remain a challenge considering the limited resources of mobile devices, such as basic smartphones or sensors. High-end smartphones have good processing power and memory, which could support complex security mechanisms.

Rassan et al. [97] present a biometric authentication for mobile cloud computing, where the mobile device camera acts as a finger print reader. A captured image goes through a pre-processing stage for gray scale conversion, blur effect reduction, normalization, segmentation etc. Extracted features from the core points of the image act as authentication entity, which can validate the mobile device. However, this research is limited to authenticate the client, and requires a human user to enter the biometric. Therefore, biometric authentication is not suitable for mobile cloud computing environment, where intelligent devices and sensors communicate with the cloud infrastructure without any human intervention.

Madhusudhan et al. [79] extend the work proposed by Lee et al. [73]. They

highlight that the authentication scheme for roaming service in global mobility networks proposed by Lee et al. suffers from vulnerability issues. The authentication scheme uses one way hash function and exclusive-OR operation, however, Madhusudhan et al. find the scheme does not perform local password validation. In addition, fails to withstand against man-in-the-middle and replay attacks. Madhusudhan et al. propose a modified version of the authentication scheme for smart card, and use Diffie-Hellman key exchange protocol for generating the secret key shared between the communication entities. Their analysis show the scheme is secure. Their scheme assumes exchange of registration parameters through a secure channel, which is similar to our authentication scheme MDA (Message Digest-based Authentication) [37]. However, their authentication scheme is not suitable for all mobile cloud computing environment. Entering ID and passwords are not a feasible solution for an automated agent participating in mobile cloud computing, such as wireless body sensors.

2.3 Intrusion Detection Systems

The purpose of the proposed context-aware framework is, detection and prevention of anomalous traffic in a mobile cloud computing environment. Therefore, exploring the existing intrusion detection systems help in understanding the issues, and challenges. This section covers selected literatures related to intrusion detection systems.

Kabir et al. [66] present an efficient intrusion detection system for distributed systems, which reduces packet inspection time, and false positive rate. The packet inspection time is reduced by creating primary patterns from Snort rules, which are obtained by prefix indexing and random indexing of Snort rule signatures. These primary patterns are used for incoming traffic inspection. Their study indicates 73% reduction of false positive. In our research, we try to eliminate rule based intrusion detection systems as it is difficult to maintain rules for various client networks' generated traffic in mobile cloud computing environment. This is achieved by populating a knowledge base with registered clients information, and perform a dynamic or periodic update operation depending on feature-based system learning, or change in the client information, such as location of the mobile client etc.

Pan et al. [92] present an IDS system, which does not require to write rules. They propose an IDS for power system or smart grid to prevent possible cyberattacks.

Their solution involves data mining technique, which eliminates the need of coding patterns in the system. A common path, which is a sequence of system states with respect to timestamps are recorded, where each state consists of features that are collected from sensors. Classification is done by comparing states of the common paths and observation. This system addresses one of the smart city application areas discussed in this thesis. Our approach is more generalized and can be applied to any mobile cloud computing application.

Moloja et al. [85] propose a cloud intrusion detection and prevention system for mobile voting devices. Their research involves a client agent and a cloud analysis engine. The client agent monitors and sends user, and sensor input to the cloud analysis engine, which performs malware scanning. This research is useful in the context of voting, where presence of any malware in the mobile system can compromise the security of voting process. Since the system performs remote scanning of only malware in the mobile device, it is more applicable to resourceful devices, such as modern smartphones, tablets etc., where malware can be installed. However, this intrusion detection system cannot be deployed for all types of mobile cloud communication, such as WBAN, VANET etc., where resource constrained mobile nodes, such as sensors are used. Therefore, there is a need of an intrusion detection system, which can be used for all mobile nodes, and which can prevent various threats, such as denial of service, man-in-the-middle etc.

Almi'ani et al. [8] propose an intelligent intrusion detection system using clustered SOM (self organized map) . This research considers publicly available NSL_KDD dataset. They perform codification before feeding the data into SOM neural network. The minimum distance calculation between the SOM neuron and input vector of features activates the winner neuron, and the weights are modified. This process continues until the training process terminates. During the detection phase, Almi'ani et al. use K-Means clustering algorithm on SOM neurons. Since only 5% to 10% internet traffic are malicious, they consider largest cluster as normal connection. For any test connection, only one cluster is activated, and if a test connection activates the largest SOM cluster then it is considered as normal connection.

2.4 Cognitive Learning

This section provides some examples of cognitive based researches, which show how systems are improved by employing cognitive learning process. Supervised and unsupervised learning during the initial phases help a system to predict a future event with higher precision.

Wang [119] describes the concept of cognitive computing and WWW+ (world wide wisdom) to satisfy the need of intelligence in computing. Every node in the WWW+ is treated as a cognitive computer, which thinks and learns similar to a brain by inferences and perceptions. Primarily this method includes machine learning, and mathematics.

Metta et al. [84] describe learning system in a multi-agent robot control system. In a multi-agent environment, where each agent performs specific task, it is very common that the agents conceal errors, and in long run it cause the robot control system to fail. Therefore, it is important to use adaptive agents to ensure that the environment damaging point is not reached. In the study, they use automata based approach and they verify traces using model checking tool. Their case study involves air hockey, where vision, control, and motion agents are used to manage the control setup.

Badia et al. [13] describe a cognition based model in multimedia content delivery, wherein the cognitive process is exploited utilizing the parallel processing power of the network infrastructure. Users' demand for wireless services are growing and the network infrastructure is also becoming more complex, where just changing policy cannot help the multimedia transmission over network channel. Therefore, they propose a solution to use cognitive approach and artificial intelligence to evaluate the video transmission encoding rates to provide improved end user experience.

Siang et al. [107] use cognitive learning method in a gaming environment. They describe how players learn in a computer gaming environment without transferring knowledge beforehand. This involves cognitive learning, which takes place during the players' interaction with the gaming environment. They show a case, where before learning, a player takes action only when a particular event occurs. However, after learning, the same player gets ready and takes action just by getting a hint of the event occurrence.

Benjamin [18] presents VMSoar, a cognitive agent that is based on Soar cognitive architechure. VMSoar understands legal and illegal activities by creating attacks on copies of virtual machines in VMWare. VMSoar can detect Windows NT vulnerabilities, and their new research focus is on finding vulnerabilities in Windows XP. VMSoar learns by launching attacks, and the research claims it is useful to detect vulnerabilities in systems and networks. However, the effectiveness of VMSoar depends on the new rules created and number of attacks it performed.

Similarly, we use the concept of pattern recognition and cognitive learning to learn from incoming traffic and prevent any future attack. The cognitive learning module uses knowledge base and probabilistic approach to decide, which traffic needs to be filtered before processing. In addition, the cognitive learning system helps in improving the quality of service by ensuring maximum uptime of the cloud server.

2.5 Classification and Cluster Analysis of Network Data

This section provides literatures associated with classification and cluster analysis applied on network data. Clustering is an unsupervised learning technique, which performs grouping of unlabeled data. Classification is performed on labeled data.

Nithisha et al. [87] use cluster analysis for intelligent traffic forwarding in router. They considered each flow as an object that consists of packets with source IP, Destination IP, source port, destination port, and service type information. In the study, over a certain period, flows are recorded and clustered into different groups. They have used K-Means algorithm to find the best clusters, and labeled them based on application type. These group of flows are sent to the router for forwarding to the destination. The research is useful, as it can maximize the efficiency of a router.

Ring et al. [99] present a research on learning similarities between IP addresses. In their research, from network data context information is extracted, and IP addresses are grouped based on context. The research focuses on unidirectional communication, and therefore, only source IP, destination port, and protocol are used as context. They have used DBSCAN density based clustering to group only the high similarity IP addresses.

These researches are close to our study, since we are interested in multi-stage traffic filtering. In addition, similar to these studies, we extract features from the

incoming traffic and use inter packet delay as our vector parameter for clustering.

Alqahtani et al. [9] presents a comparative analysis of classification techniques for cloud intrusion detection system. They consider Decision Tree (J48), Naive Bayes, OneR, and K-Nearest Neighbour (KNN) classification techniques and investigate their performance by applying these classification algorithms on the results obtained from SnortIDS, SuricataIDS, FL-SnortIDS, and FL-SuricataIDS. Their findings indicate that in most cases KNN has the highest precision and sensitivity.

Li et al. [76] propose TCM-KNN (Transductive Confidence Machines for K-Nearest Neighbours) data mining algorithm for anomaly detection. They claim TCM-KNN is cost effective and generates low false positives. The research uses KDD Cup 1999 dataset for the study. Their research uses genetic algorithm for data reduction by instance selection, which improves the performance. Filter method in feature selection makes the algorithm efficient.

McGaughey et al. [83] propose a technique for encrypted network traffic classification. Oftentimes, organizations allow encrypted traffic for their employees, which makes it difficult to identify any hidden malicious traffic. Researches indicate scanning ports, IP address are not effective, and packet analysis is expensive. This study performs feature selection from encrypted traffic using FOS (fast orthogonal search) algorithm. The research selects a subset of features from primary feature set to reduce the error in the classification. Use of FOS in feature selection and KNN for classification of test set provide minimal errors and higher true positive rate.

KNN is a supervised learning technique, where features are known and classification is done based on distance calculation. However, our research involves unsupervised learning in system training as the target data is unknown. Therefore, only KNN is not suitable for our system. We have used the distance calculation during the test phase.

Vrbský et al. [118] present a clustering-based solution for optimizing base stations' positions in order to improve quality of service in a smart grid data network. Fault identification, and efficient delivery of service are primary functionality of a communication network used in a smart grid. Placement and number of base stations in a data network play major role in reliability of communication. Their research includes, analysis of four clustering techniques, which are K-means, K-medoids, Agglomerative

Hierarchical Clustering, and Affinity Propagation. Their study exhibit that K-means and K-medoids are the most appropriate clustering techniques for smart grid scenario in terms of the cluster head positions. K-means provides better result in terms of time complexity. The centroids of the clusters represent the base stations, which communicate with the access points and intelligent electronic devices. This research shows that strategic placements of base stations are possible with K-means clustering.

We use similar concept in the training phase to identify a group of centroids that represent legitimate connections. Before performing the clustering process, an optimal value of k is computed and then all data points are assigned to clusters. These k centroids are used to validate any incoming connection based on distance calculation.

Shakya et al. [106] apply a hybrid approach on KDD Cup 1999 dataset for noise reduction, feature selection, clustering, and classification. Their research shows DBSCAN (Density-based spatial clustering of applications with noise) is an effective way of removing noise from the dataset. From 100000 KDD Cup 1999 data points, DBSCAN removes 141 records as noise points. Presence of noise can affect the outcome of K-means clustering. Classification of clustered data indicates if there is any intrusion. Their research shows an accuracy of 96.922%.

During the feature-based training phase of the context-aware framework, we employ a hybrid approach KD (K-Means, DBSCAN) to achieve faster processing and better accuracy. The training phase involves grouping of dataset (registered clients' incoming traffic data) into k clusters. Since the dataset is not very large for each client, and has only two dimensions, by employing K-Means algorithm, we achieve faster partitioning of the training dataset. To determine effective number of groups for the data points, we compute optimal k value using Elbow method. In addition, we obtain more accurate centroid values by cleaning the generated clusters using DBSCAN density-based clustering algorithm. To increase the efficiency and quality of outlier detection using DBSCAN, we use kNNdist function for computing a suitable value of neighbourhood distance. The clean clusters and the computed cluster centroids are the final outcome of the training phase.

2.6 Self-healing Networks

This section focuses on research that shows how network can be rearranged automatically based on different situations. This study helps us understand the concept of automatic rearrangement or self-healing, which could be applied in our framework to dynamically select inner clouds and security mechanisms.

Noh [88] describes there is an important need of network restoration, if it is not done then it may cause many services to fail from running. Self-healing network could be a solution. However, different networks have different architectures, which demands the need of applying different self-healing solutions. He discusses about an end-to-end network architecture depending on functional domains, which consists of CPN (customer premises network) , access network, and transport network. Example of CPN could be any home network. The access network mainly collects traffic from CPN and transmits to other nodes in the network. The transport network domain mainly focuses on optimization of physical link usage. The research describes self-healing concept for the different network architectures. E.g. CPN failure indicates customer downtime, so by increasing the reliability of the equipment used in the customer premises and by using diverse incoming traffic path and diverse path to access nodes the problem could be solved. By using self-healing ring architecture in the access network segment, reliability could be improved. Self-healing at the transport network segment could be achieved by rerouting, and bandwidth optimization.

Wu [121] describes a cost-effective Passive Protected DCS (digital cross-connect system) self-healing network architecture that uses the concept of passive protected network. By the concept of self-healing, minimization of network cost and restoration time is achieved. He discusses, two types of self-healing network architectures. The first one is, the centralized SONET (synchronous optical networking) DCS self-healing network architecture, where controller is informed for any route failure, which then searches the possible route. The second one is, a distributed self-healing network architecture, where external controller is used with each DCS, which finds best route on link failure.

Both these researches motivate us to use the concept of self-healing network in the cloud infrastructure for making necessary rearrangements in order to survive from a possible attack or to improve the quality of service. In the context-aware framework,

the adaptive module acts as a controller and serves clients' requests. If an inner cloud is compromised, then the adaptive module destroys the cloud instance and recreates an identical cloud instance with different security configuration.

2.7 Chapter Summary

- In this chapter, we have listed selective literatures that we have considered during the research.
- Depending on the context, the literatures are categorized into different themes.
- Section 2.2.1 highlights security issues related to cloud computing. Studies indicate that virtualized environment, sharing the same physical resources can impose a security threat. Other security issues include vendor lock-in, data loss, threat from cloud employees etc.
- Huge client base and limited restrictions make public cloud vulnerable to various attacks.
- A DDoS attack on cloud provider interface can disrupt major services.
- It is essential to have a standard cloud computing design and usage policy, employee trust, proprietary software for virtualization, and finally physical security of server.
- Section 2.2.2 focuses on security issues specific to mobile cloud computing. A strong authentication scheme is required for cloud and client validation. However, researches indicate existing authentication schemes either easily breakable or miss mutual authentication property.
- Device specific features, such as USIM, biometric are applicable only to mobile phones, and cannot be generalized to other mobile clients, such as sensors, RFIDs etc.
- Section 2.3 presents various intrusion detection systems. Most of the existing systems are rule based, which require administrator(s) to frequently update and configure rules for protecting the system from possible new threats. We try to

eliminate the need of an administrator, by designing a learning-based system, which performs dynamic update of the knowledge-base utilizing feature and pattern-based filtration.

- Section 2.4 presents concept of cognitive learning and artificial intelligence. These researches help us conceptualize the context-aware framework.

- Section 2.5 provides literatures associated with classification and cluster analysis applied on network data.

- Existing researches indicate the significance of K-Means clustering in partitioning of data. In addition, studies highlight that KNN has high precision and sensitivity.

- KNN is a supervised learning technique, where features are known and classification is done based on distance calculation. However, our research involves unsupervised learning in system training as the target data is unknown. Therefore, only KNN is not suitable for our system.

- DBSCAN is a density-based clustering algorithm, which minimizes noise points. DBSCAN with K-Means produce clean clusters.

- For the training phase of the context-aware framework, we employ a hybrid approach KD (K-Means, DBSCAN) to achieve better accuracy and faster processing. During the K-Means clustering, to generate unique clusters, we identify optimal value of k using Elbow method. For DBSCAN, to improve the outlier detection, we compute neighbourhood distance using kNNdist function.

- Section 2.6 focuses on research that shows how network can be rearranged automatically based on different situations. From these studies, we adopt the concept of self-healing, which we use in the design of the adaptive module.

Chapter 3

Research Objectives

This chapter defines the research objective and states the proposed measures.

3.1 Research Objective

The research objective is threefold. Firstly, standardization of mobile cloud computing security framework, which is adoptable by all smart cities across the world. Secondly, securing critical services of a smart city from various attacks such as, DDoS, MITM, ransomware etc. Thirdly, proposing a framework, which is deployable in any mobile cloud computing environment without performing major modifications in the existing mobile client infrastructure, such as WBAN, VANET etc.

3.1.1 Standardization

Mobile cloud computing is the core technology of a smart city solution. Critical services of a smart city involve heterogeneous mobile cloud computing environments, such as WBAN in smart health care, VANET in smart traffic, wireless sensor networks in smart water etc. Each of these applications has its own communication protocols, security standards, and shortcomings. In a smart city environment, all these applications run under a single smart city framework to ensure efficient service delivery, and interoperability. Robust operation, security, and data privacy are the major concern of a smart city framework. Therefore, in spite of, network and application specific standards, these applications should follow a common security standard while interacting with each other in a smart city framework.

3.1.2 Securing Smart City Services

Smart city services are critical, such as power distribution, water supply, traffic management, health care etc. Efficient delivery of these services is top most priority of

a smart city. In addition, services such as health care, banking etc. demand preservation of data privacy. Therefore, it is important to secure the services of a smart city from unauthorized access, and service disruption. A smart city uses mobile cloud computing, which includes heterogeneous applications or mobile cloud topologies, such as WBAN, VANET, WSN etc. as client side topologies, and one cloud infrastructure. Therefore, the objective of the research is to propose a security framework for mobile cloud computing that supports heterogeneous mobile cloud topologies and applications to ensure maximum uptime of the smart city services.

3.1.3 Preserving Existing Architecture

The third research objective is to propose a framework, which requires only minimal changes in the existing architecture. WBAN based Smart health care, VANET based smart traffic, WSN based smart power etc. use different mobile cloud topologies and security standards, which are difficult and expensive to alter. Therefore, the research objective is to propose a security framework, which is deployable at the centralized cloud premises of any mobile cloud computing environment, such as a smart city.

3.2 Proposed Measures

To achieve the research objectives, we propose a context-aware security framework for mobile cloud computing. The framework uses the concept of cloud of clouds, where a master cloud serves as an interface with smart city applications for accepting requests, and multiple inner clouds provide actual services to those applications. The cloud of clouds arrangement introduces an abstraction of service layer, which helps in preventing many attacks.

The proposed security framework accepts the heterogeneous smart city applications or mobile cloud topologies, such as WBAN, VANET etc. and then performs two phase screening by the cognitive module, which filters out unauthorized incoming traffic. Depending on the outcome of the cognitive module, the adaptive module chooses appropriate inner cloud to provide specific service to the client request. The adaptive module ensures maximum uptime of the cloud infrastructure and manages the virtual machine initialization. The authentication module performs mutual authentication and grants access to legitimate communicating entities. The context-aware security

framework could be used as security standard for smart city solutions or any mobile cloud computing environment.

In addition, the proposed framework requires modification only at the cloud infrastructure, which minimizes the overhead related to client network alteration.

3.3 Chapter Summary

- The research objective is threefold. Standardization of mobile cloud computing security framework, which is adoptable by all smart cities. Securing critical services of a smart city, and minimal modifications of existing mobile client infrastructure.
- The context-aware security framework uses the concept of cloud of clouds, which introduces service layer abstraction.
- The context-aware security framework could be used as security standard for smart city solutions or any other mobile cloud computing environment.
- The cognitive module performs two phase traffic filtration.
- The adaptive module manages the inner clouds and virtual machine initialization.
- The authentication module performs mutual authentication between the mobile client and cloud server.
- The proposed framework requires modification at the cloud premises, which minimizes the cost of client network modification.

Chapter 4

Context-Aware Framework

4.1 Key terms and Definitions

This section contains key terms and definitions of useful concepts, which are applied in our research, such as cloud computing, machine learning, probability model etc.

Cloud Computing

In order to comprehend this thesis, we provide brief descriptions of different terminologies associated with cloud and its services.

1. **Public Cloud:** Public cloud does not require initial investment and offers a plethora of services to general mass. However, it does not provide significant amount of security for the data and control over the cloud infrastructure [126].

2. **Private Cloud:** Private cloud as the name indicates is contrary to public cloud. Private clouds are owned by specific business organizations and are mainly used for internal services. Private cloud can be classified as another form of traditional server. It provides customized security for data and more control over the cloud infrastructure [126][53].

3. **Hybrid Cloud:** Hybrid cloud is a combination of both private and public cloud, which is used by an organization for improving its resource performance. The organization places its non-important components on the public cloud and controls the computing environment through its private cloud [7].

4. **Virtualization:** Virtualization is the core technology used in cloud computing environment. Cloud server provides access to a pool of hardware, network and storage resources through virtualization [95]. A virtual platform is created

to serve a client with customized configuration and requirements, and by initializing many such virtual machines with limited number of actual physical resources, the cloud server can provide its services to multiple concurrent users.

5. **PaaS (platform as a service):** Platform as a service allows product deployment on the cloud, which requires distributed resources [10]. A web based product might require to access various resources or infrastructure that are distributed over multiple locations. In order to deploy these product, PaaS provides platform that has access to the cloud based distributed resources.

6. **IaaS (infrastructure as a service):** Using infrastructure as a service, any individual or an organization can lease cloud resources to accomplish their job [105], such as utilizing storage servers (e.g. renting EMC2 data servers as a storage) provided by a cloud service provider.

7. **SaaS (software as a service):** Software as a service is the most commonly used cloud service, where clients use open or proprietary cloud based software e.g. cloud based antivirus or word processing software etc.

In addition to SaaS, PaaS, and IaaS, cloud service providers offer network-as-a-service (NaaS) [117] by providing access to virtualized network resources and desktop-as-a-service (DaaS) [44] considering the demand of availability of desktop resources on mobile devices.

Machine Learning

The thesis proposes a context-aware framework, which comprises of three modules. The cognitive module of the framework uses machine learning for traffic filtering. Therefore, it is crucial to understand the concept of learning, which includes classification, and clustering.

1. **Classification:** Classification is a supervised machine learning technique, which includes a set of known observations for training the system. Rules are applied in classification, and any future observation is classified based on these rules and training set. Machine learning became popular in network and cloud security,

and therefore, in cloud security research, controlled or known network traces are labeled as training set to classify new set of traffic traces [19].

2. **Clustering:** Clustering is a form of unsupervised learning, where grouping of observations is performed based on similarity measure. Unlike, classification, clustering does not require any training. In clustering, an unknown data set is explored to form finite number of clusters, where the data points in each cluster share common attributes [48].

3. **Euclidean Distance:** Euclidean distance between two points or vectors in a n dimensional space is a length of straight line segment, which connect the two points or vectors. Consider, an Euclidean space of n dimension, then the Euclidean distance between any two vectors $X = \{x_1, x_2, x_3, ..., x_n\}$, and $Y = \{y_1, y_2, y_3, ..., y_n\}$ is given by Eq. 4.1. The Euclidean distance in two dimension is same as Pythagorean theorem. [15, 77].

$$\begin{aligned} dist(X,Y) &= \sqrt{(x_1-y_1)^2 + (x_2-y_2)^2 + (x_3-y_3)^2 + ... + (x_n-y_n)^2} \\ &= \sqrt{\sum_{i=1}^{n}(x_i-y_i)^2} \end{aligned} \tag{4.1}$$

In our research, we have used Squared Euclidean distance Eq. 4.2 as it provides same result in clustering, and minimizes square root calculation complexity.

$$dist(x,y) = \sum_{i=1}^{n} ||(x_i - y_i)||^2 \tag{4.2}$$

4. **Manhattan Distance:** The Manhattan distance between two vectors $X = \{x_1, x_2, x_3, ..., x_n\}$, and $Y = \{y_1, y_2, y_3, ..., y_n\}$ in a n dimensional space is defined as the sum of the differences between the components of the two vectors [109].

$$dist(X,Y) = \sum_{i=1}^{n} |(x_i - y_i)| \tag{4.3}$$

5. **K-Means:** K-Means is a partition based clustering technique, which performs grouping of non-labeled data. The objective of K-Means is to minimize the

within group distance, and maximize the between groups distance [122, 123, 75]. K-Means algorithm starts with a set of data points $X = \{x_1, x_2, x_3, ..., x_n\}$, which are the input data. The data should be normalized before processing for clean clusters. An initial value of k is selected, which indicates the number of clusters. As a starting point, k points from the clusters are chosen as initial centroids $C = c_1, c_2, .., c_k$. All the data points in X are assigned to each of the clusters by finding minimum Euclidean distance from the centroid of the cluster. (Eq. 4.4).

$$dist(x_i, c_i)_{min} = \sum_{i=1}^{k} \sum_{j=1}^{n} ||(x_j^{(i)} - c_i)||^2 \tag{4.4}$$

The centroid c_i of each cluster is recomputed. The minimum distance calculation, assignment of points to clusters, and computation of centroids steps are repeated until it reaches a steady-state.

6. **DBSCAN:** DBSCAN (Density-based spatial clustering of applications with noise) is a density based clustering technique of n points, where pairwise distance $dist(x_i, x_j)$, where $i, j \in n$ is calculated and dense region is formed. This dense region is known as *eps*-neighbourhood and *eps* is the radius of the region. $minPts$ are minimum number of points required to form a cluster. [28, 106, 46].

   A point x_c is known as a core point, if it has a neighbourhood $Nbhd_{eps}(x_c) \geq minPts$. A point x_b is known as border point, if it is directly density reachable from x_c, such that $x_c \in Nbhd_{eps}(x_c)$, and $Nbhd_{eps}(x_b) < minPts$ (Fig. 4.1) A point x_o is an outlier, if it is not reachable from x_c[46].

   We have used DBSCAN in our research for detecting outliers and cleaning clusters.

7. **kNN:** kNN (k-Nearest Neighbour) algorithm is a simple and fundamental classification algorithm, where a test set is classified based on distance calculation from the training set. kNN uses a set of sample points $S_{sample} = \{s_1, s_2, ..., s_n\}$ and set of features $F = \{f_1, f_2, ..., f_m\}$ as a training set $s_i f_j$, where $i = \{1, 2, .., n\}$ and $j = \{1, 2, ..., m\}$. Distance calculation, such as Euclidean distance (Eq. 4.5), is considered to classify a test set $S_{test} = \{s'_1, s'_2, ..., s'_n\}$.

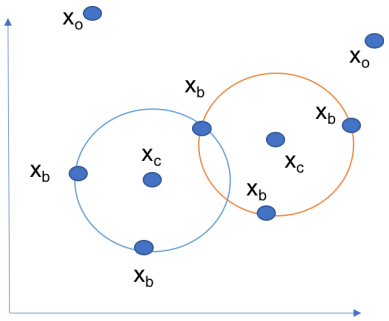

Figure 4.1: Core points, border points, and outliers

$$dist(s_i, s'_l) = \sqrt{(s_i f_1 - s'_l f_1)^2 + (s_i f_2 - s'_l f_2)^2 + .. + (s_i f_m - s'_l f_m)^2} \quad (4.5)$$

kNN uses a distance metric, and a value of k for classification. To avoid any tie situation, an odd value is chosen for k, and a smaller number can avoid including noise points. If $k = 1$ then a test point is classified more accurately to its nearest point [96].

8. **kNNdist:** kNNdist (k-Nearest Neighbour Distance) calculates k-Nearest neighbour distances, which is useful to find optimum *eps* value for DBSCAN. The kNNdist plot provides a knee, which can be considered as optimum *eps* value [54].

Concept of Probability Model

One component of context-aware framework is the adaptive module, which is designed to make some decision based on input provided. In order to understand the functionality and design of adaptive module, we need to know concepts of probability model.

1. **Markov Chain:** Consider a state space $S = \{s_0, s_1, s_2, ..., s_n\}$. A stochastic process (X_n) is called a markov chain, if $n \geq 0$, and all states $s_0...., s_i, s_j \in S$. The transition is possible from current state s_i to the next state s_j independent of the past states $s_{(i-1)}....s_0$. A transition probability $P_{(ij)}$ provides the likelihood of single state transition from s_i to s_j. The transition probability $P_{(ij)}$ is presented in a square matrix P, where each row sums to 1 [108].

$$X_{(n+1)} = PX_n \tag{4.6}$$

4.2 Overview of Context-Aware Framework

The research proposes a security framework for the mobile cloud computing, which is termed as the context-aware framework [39]. This framework consists of three major modules: cognitive module, adaptive module, and authentication module shown in fig. 4.2.

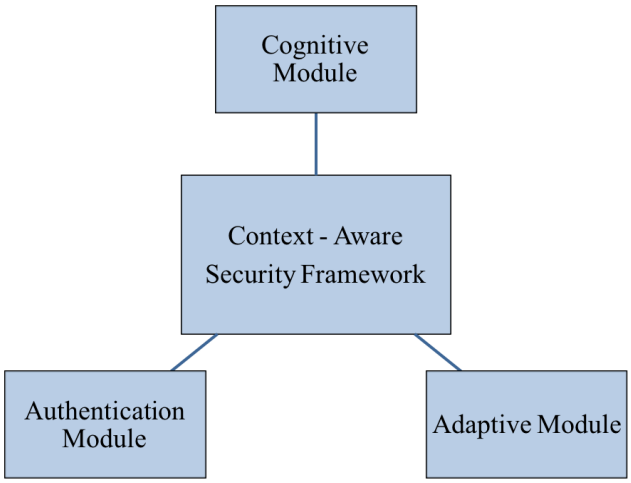

Figure 4.2: Modules of Context-Aware Framework [39]

In addition, the research proposes a cloud of cloud model for mobile cloud computing environment. Fig. 4.3 represents a pictorial overview of the cloud of cloud model that consists of an outer cloud or master cloud and multiple inner clouds' instances. In the context of smart city, a set of inner clouds could be used to support smart traffic, while another set of inner clouds could be utilized for smart grid systems. We claim that our system is reconfigurable, therefore, to reduce the infrastructure overhead, a set of inner clouds could be used for all applications supported by a mobile cloud computing environment, such as a smart city. In a hypothetical scenario, we consider that each inner cloud in the cloud of cloud model serves a particular type of client, such as if one cloud is responsible for serving cellphone traffic, then other is responsible for serving VANET traffic etc. The outer cloud is used to manage the

incoming and outgoing traffic, while the inner clouds provide the actual services, such as PaaS, SaaS, IaaS to the mobile clients. The modules of context-aware framework are deployed in specific layers of the cloud of cloud model.

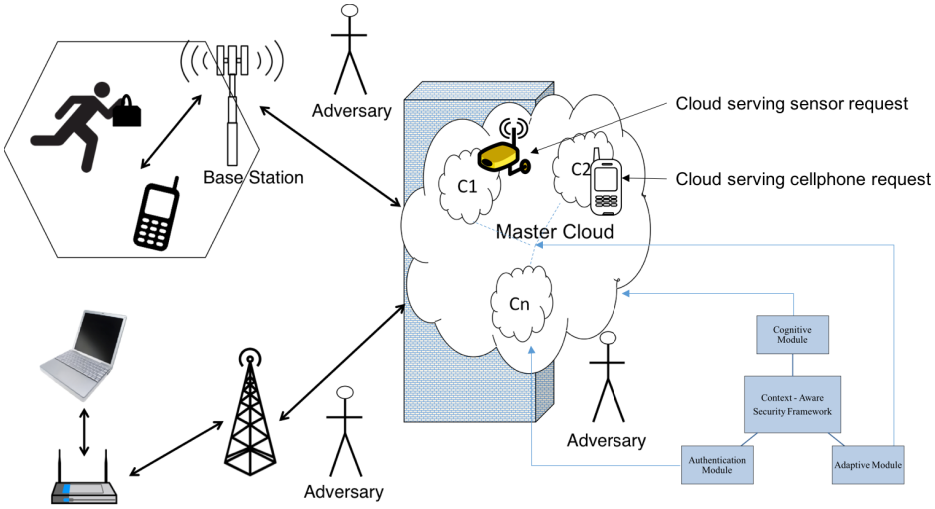

Figure 4.3: Proposed Cloud of Cloud Model with Context-Aware Framework [39]

4.3 Architecture of Cloud of Cloud Model

The research proposes two separate designs for cloud of cloud model. These designs are ideal for mobile cloud computing environment running the proposed context-aware framework. Figure 4.4 presents a cloud of cloud design that uses type 1 hypervisor [12, 43, 81]. In this particular design, one hypervisor configuration is used on the physical hardware to separate the application software that runs the client virtual machines. The application package or software running on the hypervisor initiates multiple sets of virtual machines with different configurations and application layer security. In this setting, each set of virtual machines are considered as an inner cloud that serves a set of mobile clients. The inner clouds run separate configurations to minimize crosstalk between set of virtual machines.

We propose another cloud of cloud model that uses type 2 hypervisor architecture [81, 36]. In this construct, a hypervisor runs on top of host operating system. In the proposed model, multiple hypervisors are used on top of host operating system, and each hypervisor initiates a set of virtual machines. Therefore, in this setting, an inner cloud is formed with a set of virtual machines and their associated hypervisor (Fig.

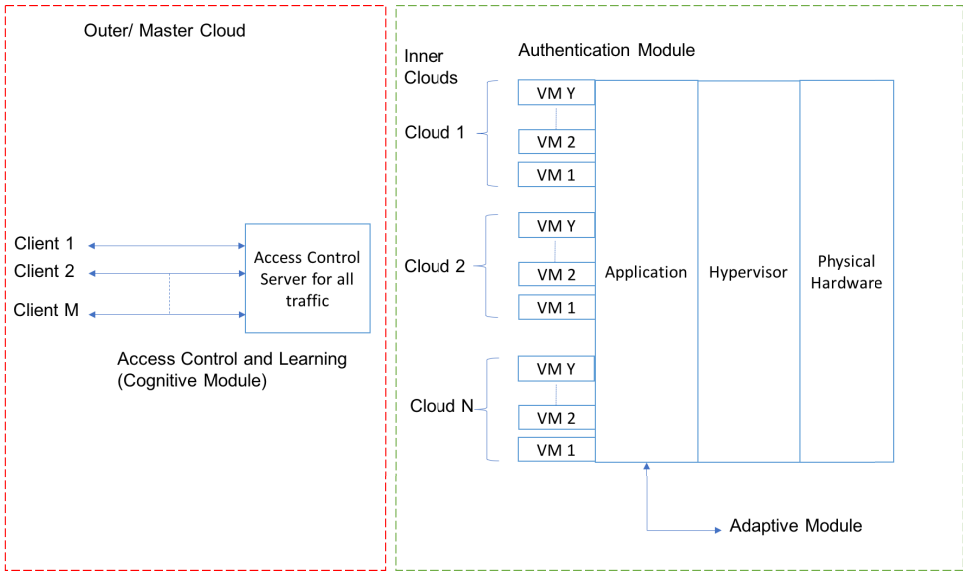

Figure 4.4: Architecture of Cloud of Cloud Model based on Type 1 Hypervisor

4.5). In real-world implementation, a similar design could be achieved by deploying a set of type 2 hypervisors, such as KVM, VMware Workstation, and virtual server etc. This specific implementation of cloud architecture minimizes the probability of getting affected by hypervisor related attacks. If there is a security issue with one hypervisor and if it gets compromised at some point, then the other type 2 hypervisors will ensure the risk-free operation of the cloud services due to the heterogeneous nature. The set of type 2 hypervisors could be selected based on the energy consumption, performance, and vulnerability [64, 36, 59, 113].

The cognitive module is deployed in the outer cloud or master cloud, and it is used to filter the incoming traffic. The cognitive module could be implemented as an access control server. In this research, we have conceptualized the cognitive module as a learning system, which utilizes clustering technique for incoming traffic categorization.

The adaptive module is deployed at the cloud of cloud model as an application package or software, and it is used to direct the traffic to a hypervisor, and virtual machine. In addition, the adaptive module initiates and terminates virtual machine instances depending on configuration or security changes. For type 1 hypervisor-based cloud of cloud model, the adaptive module is conceptualized as a small software, which maintains logical partitioning of inner clouds, selection of an inner cloud depending

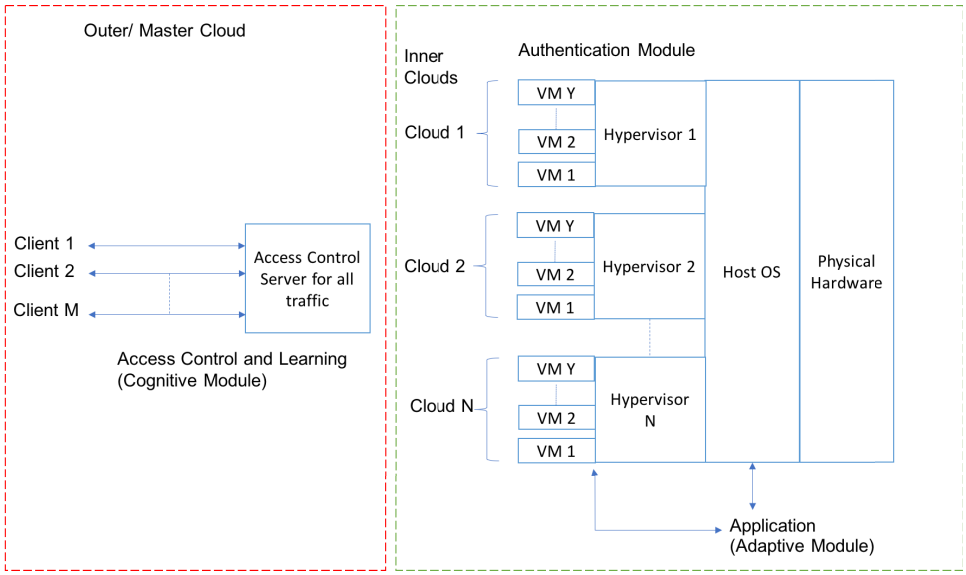

Figure 4.5: Architecture of Cloud of Cloud Model based on Type 2 Hypervisor

on threat level detection and performs VM initialization. For type 2 hypervisor-based cloud of cloud model, the adaptive module is conceptualized as two software applications, where one runs on Host operating system for hypervisor or inner cloud selection, and another runs on each of the hypervisors for VM initializations.

The authentication module is deployed in each inner cloud and is customized depending on the security requirement of the mobile cloud computing application, and resource availability of the mobile nodes used in the communication. In this research, we have proposed three authentication schemes, which could be deployed in the authentication module for the completeness of the context-aware framework. Since, we claim the system is reconfigurable, any of the modules could be reconfigured to accommodate a new or existing authentication scheme, and/or learning systems.

These three reconfigurable modules ensure the security of any mobile cloud computing environment, such as a smart city application. These modules can operate individually, however, to establish an effective security model for mobile cloud computing, we bind these modules.

4.4 Cognitive Module

The cognitive module of the context-aware framework is deployed in the master or outer cloud as an access control server to safeguard the inner clouds or virtual machines from any malicious incoming traffic. This module is reconfigurable, and can be customized based on the security need of the cloud provider. A two-phase screening of this module ensures that only authorized clients gain access to the cloud resources. The cognitive module comprises a pattern matching engine, a short term and long term learning system, and a knowledge base (Fig. 4.6). The cognitive module ensures filtration of incoming traffic by a series of operations. Acceptance or rejection of an incoming request is specified by a flag λ .

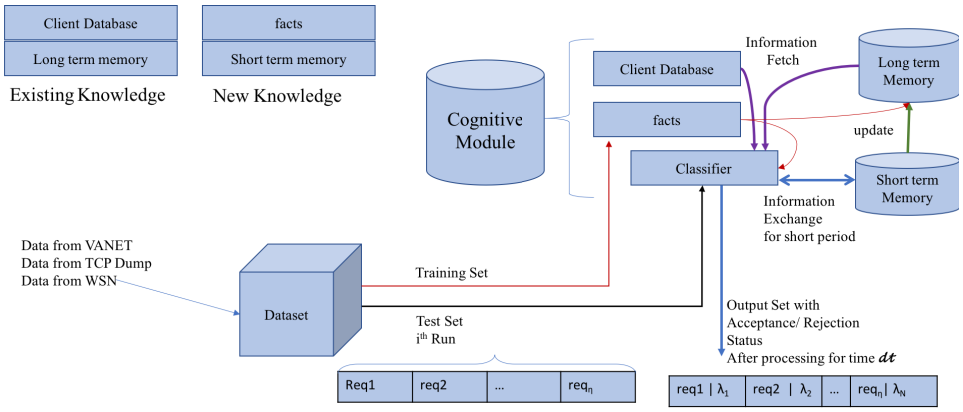

Figure 4.6: Architecture of Cognitive Module

The client database stores limited information about registered cloud clients, e,g. client's location during registration and authentication, client ID, client's OS (operating system) during registration and authentication, timezone etc. The cognitive module operates at the outer layer of the cloud infrastructure, therefore, only limited information is stored for faster processing and security. The long-term memory or long term learning system is used to store information about the session, and possible changes in the communication, such as valid location change of a client, permanent and temporary change of access device(s), security threats associated with a client's information etc. Short term memory is used to store any new information for a short period of time, which helps processing of an identical future client request. Short term memory updates the long-term memory, if there is a permanent change in the client's information, or if there is a persistent threat data. In addition, the cognitive module

consists of classifier that provides rule based classification or clustering techniques for traffic analysis. Since the modules of context-aware framework are reconfigurable, any new classifier could be introduced for better results.

The main function of the cognitive module is to monitor patterns in the incoming traffic. A pattern is defined as a tuple $[profile, feature]$ of prerequisite information extracted from the incoming request. The $Phase1$ screening performs $profile$ based filtration, where $profile$ indicates client's OS, and coarse location. If $Phase1$ detects anomalousness, then $Phase2$ is activated, where $feature$ based filtration is performed using clustering techniques.

4.4.1 Phase 1: Screening Profile

$profile$ indicates client's profile information stored in the cloud server that includes client's OS, and coarse location. In this phase, the cognitive module first checks the client's ID in the database, and extracts the required information, which are location and OS. Persistent screening of $profile$ is performed for all client requests in an established session.

A $profile$ is represented by p_{mn}, where value of mn varies between 00 and 11, and defines whether an incoming request has known OS and known location. Three probability values are considered to define the outcome of this phase. If 0% match is found, then the probability of legitimate traffic will be "0". A 100% match indicates the traffic is legitimate, therefore, the probability value will be close to "1". 50% match indicates the incoming traffic could be legitimate or malicious, therefore, the probability is "0.5". $Phase1$ accepts the input requests from requesting source and fetches to $Phase2$ after first level of screening. Presence of profile $p01$ in incoming request indicates the request does not have known OS. However, the location is known, and therefore, the details are updated in the short-term memory and the request is transferred to $Phase2$ for further analysis. Tab. 4.1 represents possible scenarios and outcome of $Phase1$.

4.4.2 Phase 2: Screening Feature

The research highlights various mobile cloud computing topologies, which generate clients' network specific traffic data, e.g. traffic from VANET are dissimilar than

Table 4.1: Screening Profile Scenarios and Outcome

ProfileValue	Client OS	Coarse Location	Scenario	Outcome
p_{00}	unknown	unknown	Possible man-in-the-middle attack	0%$match$, rejected
p_{01}	unknown	known	Possible man-in-the-middle attack or client has switched to other device in the same local area network	50%$match$, short-term memory update, fetched to phase 2
p_{10}	known	unknown	Possible man-in-the-middle attack or client is moving between cities with the same device during a communication	50%$match$, short-term memory update, fetched to phase 2
p_{11}	known	known	Valid client	100%$match$, long term memory update, fetched to phase 2 for further analysis

traffic generated in WBAN. The heterogeneous nature of incoming traffic restricts the usage of certain data fields. VANET data contains precise location information, however, data generated from WSN contains the sensor nodes readings. The data field contains rich information, and feature extracted from data can provide insight in traffic filtration. However, due to heterogenous data fields, it is difficult to generalize feature selection for filtration process. In order to solve this problem, we have considered inter packet delay (IPD) as the $feature$ of incoming traffic. For a registered client, the arrival time of packets at destination or cloud server are monitored and inter packet delays are computed, which are used to train the system (Fig. 4.7). The values of inter packet delays depend on various factors [31]. Software implementation, deployed hardware, speed of communication channel, queuing, and network

failure can impact the arrival time of packets at destination.

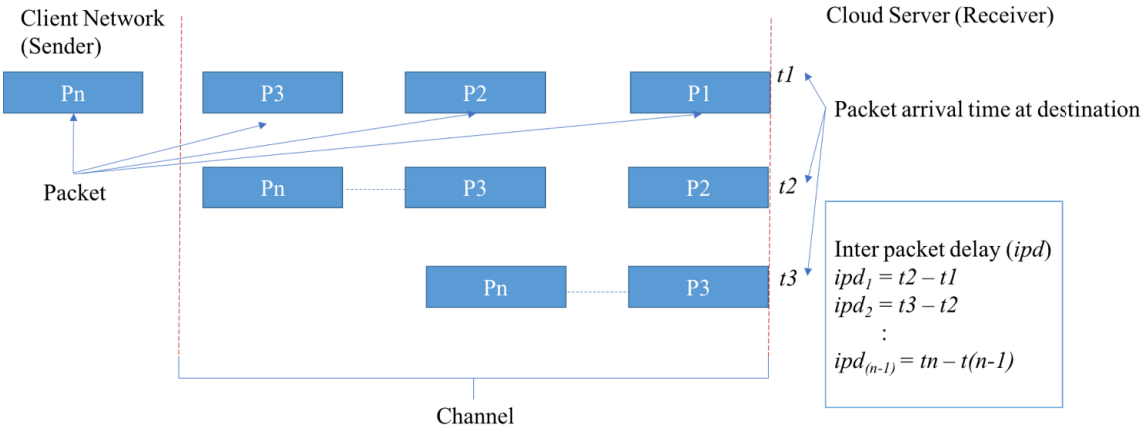

Figure 4.7: Inter packet delay computation at destination

*Phase*2 performs *feature* based filtration, where an incoming client request is tested with the training set, and based on similarity measure the traffic is either rejected or accepted. The operation of this phase depends on training of the system. The values of inter packet delays are unknown, and the values are not labeled. Therefore, the system requires unsupervised learning for creating groups or clusters. Cognitive module uses a set of existing data to create optimal number of groups, which are used by *Phase*2 during the filtration process.

System Training

During the training phase, for a client ID (*cid*), a large sample of incoming traffic (X) is extracted and the inter packet delays are computed based on the arrival time. The system must be trained for each client in order to understand client specific traffic pattern. Since the inter packet delays are just some unlabeled real numbers that do not contain any metadata, rule based classification cannot be performed. Therefore, we use cluster based approach to create natural grouping of the data. Fig. 4.8 shows the pictorial overview of the clustering process, which has three steps.

- Extract a large sample from the incoming traffic for a specific client, and compute the inter packet delays, which are considered as set of data points, such that $X = \{x_1, x_2, x_3, .., x_n\}$.

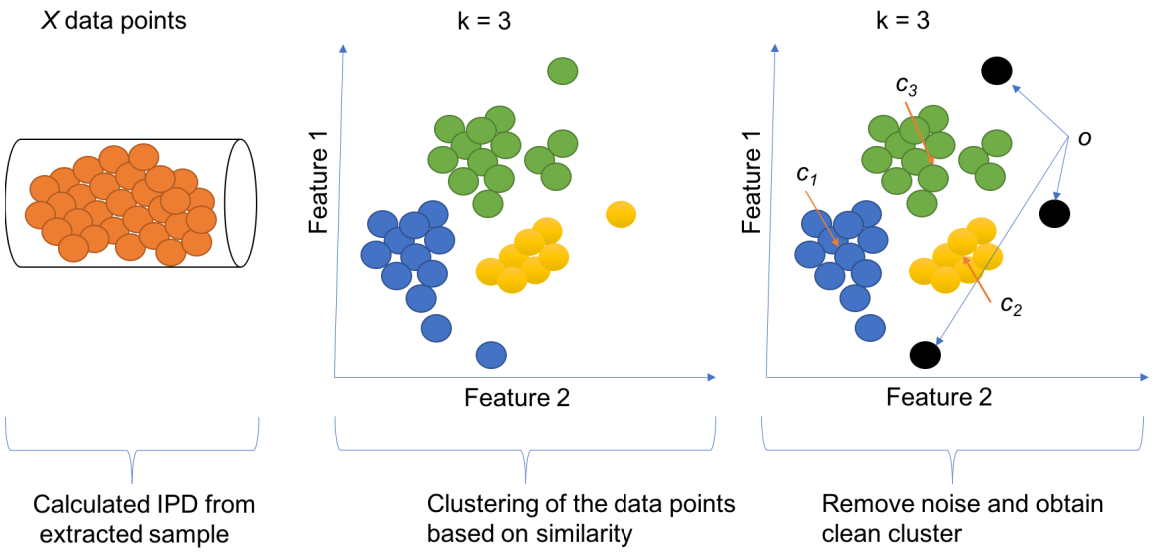

Figure 4.8: Steps in inter packet delay based clustering

- Partition the data points into optimum number (k) of groups or clusters based on similarity.
- Clean each cluster and remove the outliers (o). Obtain the centroid (c_i) of the clean clusters.

We propose KD algorithm for clustering, which is a combination of "K-Means", and "DBSCAN (Density-based spatial clustering of applications with noise)" clustering algorithm. K-Means is widely accepted partitioned based clustering technique, which provides k clusters of input data points based on similarity measure. From the data points k number of initial centroids or cluster means are chosen that are relatively far from each other. Then, each remaining data point is placed in a cluster that contains the nearest centroid. This process continues until all data points are assigned to clusters. Finally, the cluster centroids are computed. However, clusters formed using K-Means contain noise points or outliers, which may affect training of the system. Therefore, we use DBSCAN to clean each of the generated clusters.

The values of inter packet delays are not directly used into the system as input data points. Our study indicates due to network related delays, and efficiency of source or transmitter the arrival time of packets at distance gets affected, and therefore, by using only IPDs it is difficult to find a pattern in the data set. In order to obtain a pattern, we have considered vectors of IPDs, which are termed as

inter packet delay vectors (IPDVs) . Each vector is formed by combining the previous and the current inter packet delays, such as $IPDV_1 = \{ipd_0, ipd_1\}, IPDV_2 = \{ipd_1, ipd_2\}, ..., IPDV_n = \{ipd_{(n-1)}, ipd_n\}$. This transformation of data can introduce various patterns in two consecutive vectors, such as {short delay, long delay} & {long delay, short delay}, or {long delay, short delay} & {short delay, long delay} etc.

In the proposed KD (Alg. 1), we use a set, X of n IPDVs as input data points. The output of KD is a set of clean clusters, and their associated centroids, which are stored in the long-term memory for testing of new incoming traffic. Once the system is trained for all the registered clients, the $Phase2$ can perform fast traffic filtration.

Traffic Filtration

Traffic filtration in $Phase2$ refers to the testing of new incoming traffic with the existing knowledge. When a new set of traffic arrives (ideally 3 packets) from a specific client, the two inter packet delays (ipd_0, ipd_1) are computed based on the arrival time of packets at the destination. One IPDV ($x_{new} = \{ipd_0, ipd_1\}$) is formed, which contains a pattern {short delay, long delay} or {long delay, short delay}, or {long delay, long delay} or {short delay, short delay}. For faster traffic filtration process, one IPDV from the incoming traffic is computed, and therefore, only 3 packets are monitored at a time for distance-based traffic validation.

For simplicity, we consider measuring the squared Euclidean distances between the incoming vector (x_{new}) and the centroids c_i of the k clusters (Eq. 4.7). If the calcualted distance ($dist(x_{new}, c_i)$) is less or equal to the distance of the farthest data point from the centroid ($dist(x_{farthest}, c_i)$), then the new incoming traffic vector will be placed in that cluster, and the incoming traffic set will be considered as valid. If Eq. 4.7, generats a longer distance, then the traffic is rejected, and short-term memory is updated.

$$dist(x_{new}, c_i) = \sum_{i=1}^{k} ||x_{new} - c_i||^2 \tag{4.7}$$

If more than one IPDV is considered for testing, more number of incoming traffic packets need to be monitored for IPD computations, which will decrease the efficiency of the filtration, and will require multiple distance calculations.

Algorithm 1 KD (K-Means DBSCAN) Algorithm

INPUT:

#1. Sample set $X = x_1, x_2, x_3, ..., x_n$ for K-Means, where x_i is an IPDV

#2. *eps*, and $minPts$ for DBSCAN

OUTPUT:

k clean clusters with centroids c_i

PROCEDURE:

Step 1. Select possible optimum value for k based on "Elbow" method

Step 2. Choose initial centroids c_i from the k clusters

Step 3. Compute distance between each point $x_1, x_2, x_3, ..., x_n$ and the cluster centroids $\sum_{i=1}^{k} c_i$

Step 4. Place each point to a cluster, if it is nearest to that cluster centroid $min \sum_{i=1}^{k} \sum_{j=1}^{n} ||x_j^{(i)} - c_i||^2$

$||x_j^{(i)} - c_i||^2$ represents squared Euclidean distance between a point $x_j^{(i)}$ in cluster i and the cluster centroid c_i

Step 5. Continue the process until all points are assigned to the clusters. K-Means minimizes the within-cluster variance

Step 6. Re-compute the centroids of each cluster

Step 7. Select an optimum value for *eps* neighborhood based on "kNNdist (k-Nearest Neighbour Distance)" plot. In addition, assign $minPts = 4$ as one of the inputs to DBSCAN algorithm. $minPts$ specifies the dense region

Step 8. Select each cluster from the k clusters, and compute the pair-wise distance between the data points.

$dist(x_i, x_j)$, where $i, j \in m$. (m specifies the total data points in the cluster under observation) The *eps* distance neighbourhoods are formed, and the core points, directly reachable points, noise points are determined

Step 9. After ignoring the noise point(s), centroid c_i from each clean cluster is computed

4.5 Adaptive Module

The adaptive module is inspired by the self-healing network research, where dynamic arrangements of network components are performed to ensure maximum network service uptime. This module performs the selection of cloud infrastructure dynamically depending on the security need of a communication session. We use the concept of cloud of cloud shown in Fig. 4.4, and Fig. 4.5, where a master or outer cloud acts as an envelope and encapsulates number of smaller clouds $Cloud1, Cloud2, .., Cloudn$.

4.5.1 Module Design as Simple Decision Tree

The adaptive module is conceptualized as an application running on the cloud premises, which controls the initialization of virtual machine(s), and selection of security configuration. The operation of this module depends on the values that it receives from the cognitive module. For a specific client, each incoming traffic is filtered by the cognitive module and passed to the adaptive module. A flag value λ is attached by the cognitive module that triggers the operations of the adaptive module. We describe the functionality of this module using a simple decision tree diagram (Fig. 4.9).

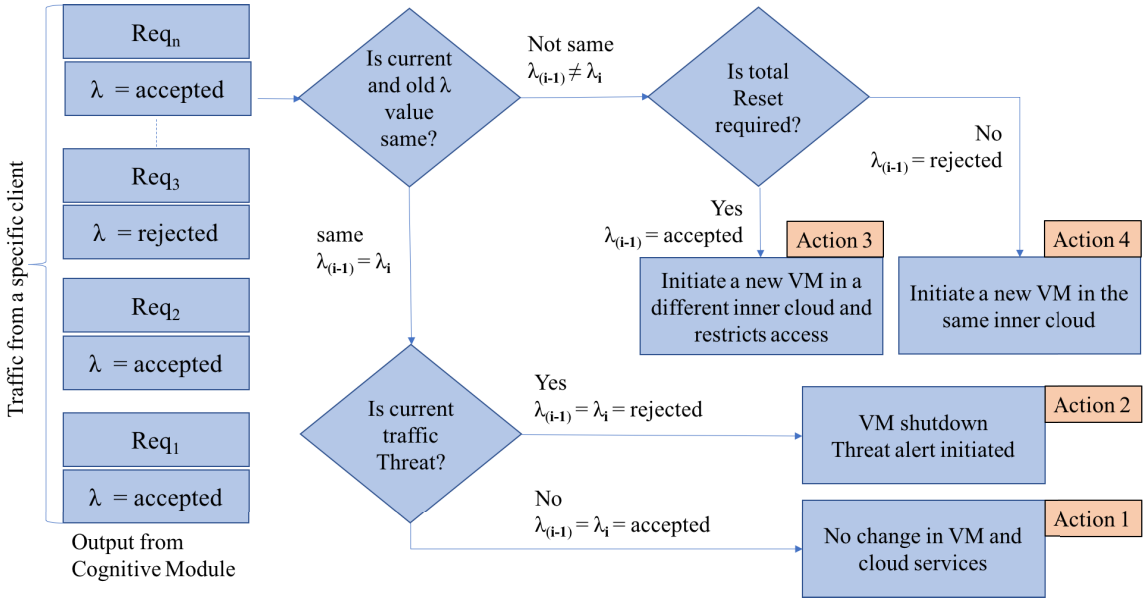

Figure 4.9: Decision tree diagram of the Adaptive module

If the received λ_i indicates the incoming traffic is "accepted", then adaptive module checks the previous $\lambda_{(i-1)}$ value. If both are same, then the module does not

perform any specific operation other than transferring the traffic to the appropriate VM. However, a mismatch of two consecutive λ values cause a series of operations. If $\lambda_{(i-1)} \neq \lambda_i$, then depending on the value of $\lambda_{(i-1)}$, the module switches the VM and restricts the client access level.

If $\lambda_{(i-1)}$ ="rejected" and λ_i = "accepted", then the threat level is marked as "low" and as a security measure the system initiates a new VM in the same inner cloud. Although the new flag value is "accepted" the system switches to new VM to ensure termination of any existing malicious process, which could cause leakage of sensitive information related to the client or hypervisor. The switching of ongoing processes from one VM to other is performed seamlessly with the help of a state recorder that keeps track of all the essential processes running in all VMs.

If $\lambda_{(i-1)}$ ="accepted" and λ_i = "rejected", then the threat level is marked as "severe" and as a security measure the system initiates a new VM in a different inner cloud for the next accepted traffic. To avoid any inter VM attack the entire inner cloud switching is carried out. If the value of $\lambda_{(i+1)}$ is "rejected" as well, then the VM is shutdown and "threat alert" is initiated. This indicates the specific client network or communication channel is compromised and the cloud VM should not process any future incoming traffic for the specific client without performing a re-registration or re-authentication. If the incoming traffic is legitimate, and the system updates both λ_i, and $\lambda_{(i+1)}$ value as "rejected", then two consecutive rejections will lead to VM shutdown. This may affect the efficiency of service delivery, however, from security point of view, considering a legitimate traffic as malicious is less harmful (false positive) than considering a malicious traffic as legitimate (false negative).

Adaptive module could be implemented as an application and deployed on the host OS to control the hypervisors (for cloud of cloud model based on type 2 hypervisors) or VMs (for cloud of cloud model based on type 1 hypervisors). Since the module is an application and reconfigurable, it could also be deployed in a container based virtualization, such as Docker [65] to control the execution of Apps.

4.6 Authentication Module

The authentication module is one of the three components of the context-aware framework, which ensures only legitimate client gain access to the cloud resources, and only a legitimate cloud server establishes session with a client. The authentication module is deployed at cloud and client end. However, the resource intensive tasks are carried out only at the server end to minimize mobile client's processing and battery power usage. The authentication module could be implemented as a firmware application or configuration in mobile devices, such as any WSN sink nodes [25]. At the cloud premises, the authentication module could be implemented as an application, which can run alongside with the adaptive module.

The authentication module ensures a mutual authentication between the two communication parties. Different types of authentication schemes could be used in the authentication module depending on the type of security and performance need. In the initial design of the framework, we consider our earlier proposed authentication schemes MDA (Message Digest Based Authentication) [37, 40] and MDLA (Message Digest and Location based Authentication) [38], which could be deployed as lightweight authentication schemes. Later, these authentication schemes are extended to AMLT (Authentication based on Message digest, Location, and Timestamp) .

Motivation

In MDA, MDLA, and AMLT a registered mobile client is authenticated by the cloud server upon verifying the information stored in the cloud server database. The requests messages are encrypted with multiple keys, and message digests of cloud server and mobile devices are used for integrity check.

The primary motivation of proposing these authentication schemes, is to perform mutual authentication between a mobile client and a cloud server without the need of entering passwords or biometric parameters. This eliminates the need of human presence during the authentication process. Most of the existing systems authenticate only the mobile client, and assumes the server is legitimate, which may not necessarily be true. An adversary may pose as a cloud server, and offers malicious services. Therefore, performing mutual authentication is essential.

In a mobile cloud computing environment, the location of client nodes are usually dynamic. If cloud resources are accessed over untrusted networks, then there is a possibility of man-in-the-middle or replay attacks. With these authentication schemes, we try to provide security using dynamic and multiple keys, which are computed on both ends without sharing over the communication channel. In all three schemes, the authentication is achieved by validating the encrypted message digest.

Since mobile devices participating in mobile cloud computing are heterogenous in nature, and oftentimes resource constrained, such as body sensors, it is crucial to reduce the number of resource intensive operations during each authentication process. This is achieved by employing message digest as authentication parameter, symmetric-key encryption, and providing flexibility of encryption or hash algorithm selection.

The following sections will cover all the three authentication schemes in detail.

4.6.1 MDA: Message Digest Based Authentication

Ahmad et al. [5] mentioned that IMSI (International Mobile Subscriber Identity) can be used as a mode of authentication. However, not every mobile device has the built-in IMSI chip. For example, many tablet computers, laptops, and sensors are not equipped with IMSI chips. Therefore, an IMSI based authentication scheme is not appropriate for mobile cloud computing. Our proposed scheme, MDA [37, 40], does not depend on IMSI and it does not require any additional hardware infrastructure. MDA is applicable to a variety of different mobile devices, including those that do not have IMSI chips. MDA is composed of two phases: Registration and Authentication. The details of these phases are presented as follows.

Registration Phase

The registration process (Fig. 4.10) of a mobile device or mobile user to a cloud server is a one-time process wherein the user ID and the password are setup and some encrypted files are exchanged. Setting up of an account or registration involves exchange of user id, password, and mobile client's information. During this phase, the mobile device's user ID and password are created to access cloud server. Cloud Server stores $hash\{userID\}$, $hash\{password\}$, and user's mobile device information

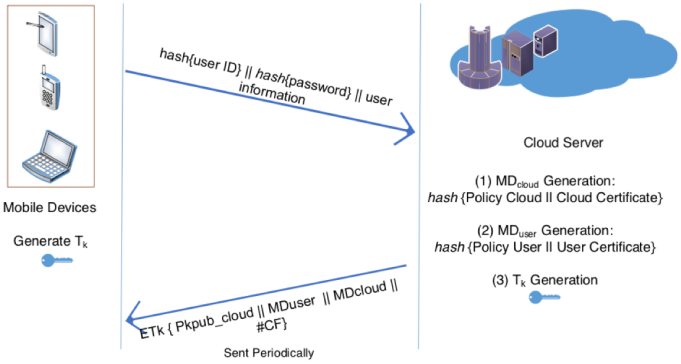

Figure 4.10: Registration Process of mobile device with cloud server [40]

in big table for efficient lookup. It generates two hashed messages or message digests. MD_{user}, which consists of user policy (cloud resource usage policy, and user access level) and User certificate, and the other message digest is MD_{cloud}, which consists of cloud policy (user add policy, cloud resource restriction/accessibility information), and Cloud certificate.

Upon generating both message digests, cloud server creates an encrypted message to transmit this information to the mobile device. The encrypted message is: $E_{T_k}\{Pk_{pub\_cloud} \| MD_{user} \| MD_{cloud} \| \#CF\}$, where Pk_{pub_cloud} is the cloud's public key, MD_{user} and MD_{cloud} are the generated message digests, and $\#CF$ is the column reference, which refers to the cloud authentication database for that particular cloud user information. These information are sent from cloud server to mobile device after encrypting with key T_k that is generated in both the mobile device and cloud server by XOR-ing (Exclusive OR) hashed $userID$ and hashed $password$ (Eq. 4.8). The proposed authentication scheme is applicable once the MD_{user}, MD_{cloud}, and $\#CF$ are transferred to the mobile device during the registration process.

$$T_k = hash\{userID\} \oplus hash\{password\} \tag{4.8}$$

Authentication Phase

At the beginning of each session, the mobile user or mobile device and the cloud server need to authenticate each other in order to start transferring actual data. The

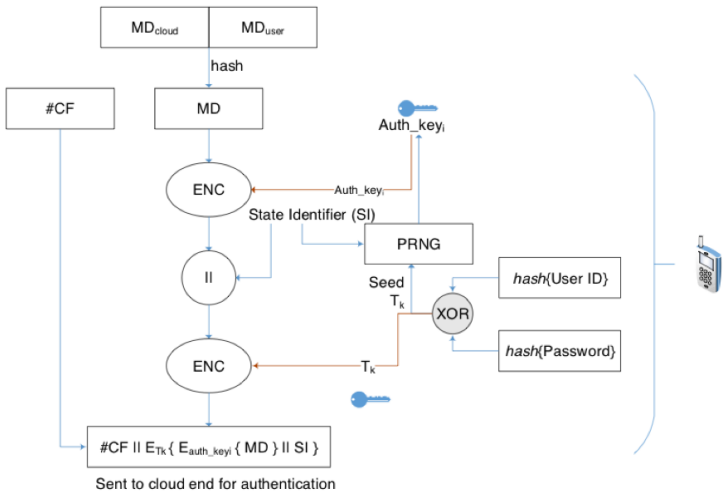

Figure 4.11: Mobile device sends connect request to cloud server during mobile device authentication by cloud server [40]

authentication process is divided into two sets of operations namely, cloud authenticating the mobile device, and mobile authenticating the cloud.

Cloud Authenticating Mobile Device:

When a mobile device wants to send an authentication request to the cloud server, it generates a key T_k, using hashed $userID$ and hashed $password$ (Eq. 4.8). The key T_k, works as the seed for the PRNG (pseudo random number generator) to generate an authentication key, $Auth\_Key_i$. This authentication key $Auth\_Key_i$ is required to encrypt the message digest MD, which is generated by hashing MD_{cloud} and MD_{user} (Eq. 4.9). The $Auth\_Key_i$ is the n^{th} sequence of bits generated by PRNG that is specified by the state identifier SI. Key T_k is used to encrypt the state identifier SI, and the encrypted message digest $E_{Auth\_Key_i}\{MD\}$.

Finally, the encrypted message, $E_{T_k}\{E_{Auth\_Key_i}\{MD\}\|SI\}$ is sent to the cloud server (Fig. 4.11) along with the column reference $\#CF$. The message sent from mobile device to cloud server is $\#CF\|E_{T_k}\{E_{Auth\_Key_i}\{MD\}\|SI\}$

$$MD = hash\{MD_{cloud}\|MD_{user}\} \tag{4.9}$$

Upon receiving the authentication request message, cloud server performs decryption operations (Fig. 4.12). The cloud server searches the specific hashed $userID$

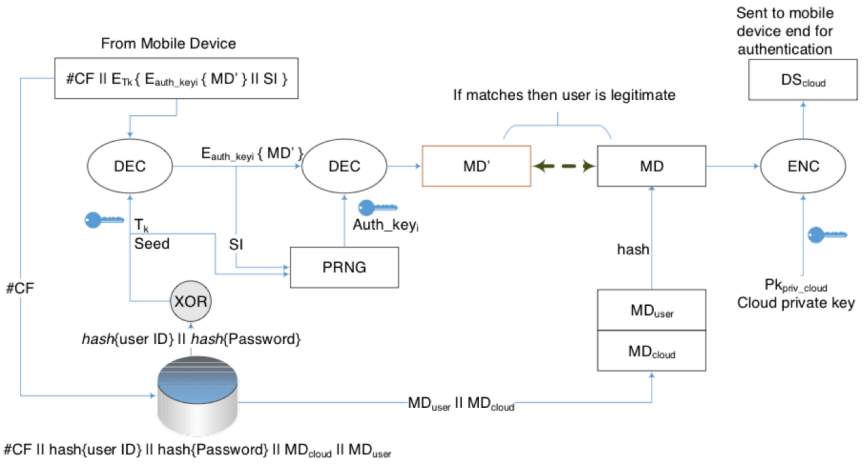

Figure 4.12: Cloud Server authenticates mobile device once it receives a request from mobile device end [40]

and hashed *password* in the cloud authentication database based on the shared column reference $\#CF$. Once the hashed $userID$ and hashed *password* are found, T_k is generated (Eq. 4.8) by XOR operation. The generated T_k at the cloud server decrypts the message $E_{T_k}\{E_{Auth\_Key_i}\{MD\}\|SI\}$ to obtain the state identifier (SI), and encrypted message digest $E_{Auth\_Key_i}\{MD\}$. In addition, the key T_k is used as the seed to the PRNG, and the retrieved SI is used to specify the n^{th} sequence of bits obtained from PRNG as the authentication key, $Auth\_Key_i$. The authentication key, $Auth\_Key_i$ decrypts the encrypted message digests $E_{Auth\_Key_i}\{MD\}$ and obtains MD, which is then matched with the existing message digest stored at the cloud server. Both the message digests will match only if the user is legitimate.

Mobile Device Authenticating Cloud Server:

Once the mobile device is authenticated, the cloud server sends its digital signature, which consists of MD encrypted with cloud's private key Pk_{priv_cloud}, to the mobile device. After receiving the digital signature DS, the mobile device decrypts it with cloud's public key Pk_{pub_cloud}, stored in the mobile device. If the decrypted MD matches with the message digest MD' stored in the mobile device, then it can be stated that the cloud server is legitimate (Fig. 4.13). On successful execution of the authentication process, both mobile device and cloud server establish a session

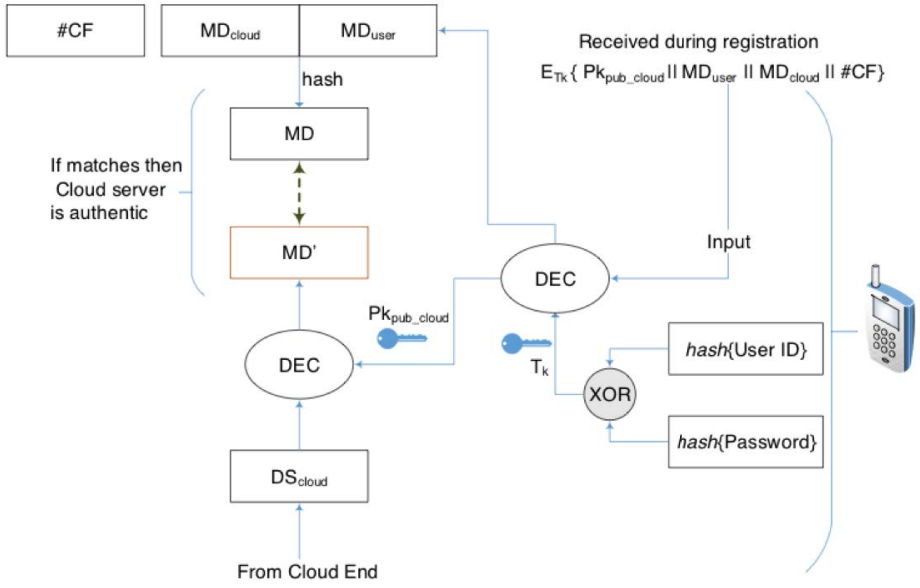

Figure 4.13: Mobile device authenticates cloud server once it receives Digital Signature of the cloud server [40]

for data transmission.

4.6.2 MDLA: Message Digest & Location Based Authentication

MDLA is a novel authentication scheme, which validates a mobile client and a cloud server participating in a mobile cloud computing [38, 41]. This scheme is independent of device specific properties, such as USIM (universal subscriber identity module) or MAC (media access control) address. Three key phases - registration, authentication and update constitute the operation of MDLA scheme. The registration phase stores mobile clients credentials to the cloud server and generates a seed key and a message digest for the initiation of the authentication phase. The authentication phase is the core phase of MDLA. The steps in the authentication phase run each time when a mobile client wants to establish a session with the cloud server. This phase ensures that data transmission is possible, if and only if both parties are legitimate and if the mobile client is registered with the cloud server. There are three different types of updates performed by the proposed scheme to ensure confidentiality and access

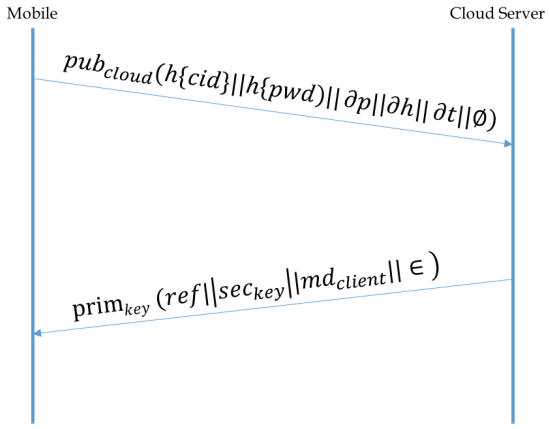

Figure 4.14: Registration of mobile client with the cloud server [41]

control updating the client registration: re-registration, updating the authentication: re-authentication, and updating the keys: key generation.

Registration Phase

Similar to MDA, the registration phase of MDLA is an account setup phase, where a mobile client registers with a cloud server and exchanges setup information such as client id, password, contact information etc. Typically, a certification authority certifies a cloud server, therefore MDLA focuses on certifying the registered mobile client. During the registration phase, the cloud server obtains mreg-req from the mobile client and stores in the big table residing at the cloud server, which contains hashed client id, hashed password, timestamp, location information, etc. The mobile client sends a registration request message encrypted with the clouds public key pubcloud (Eq. 4.10). Fig. 4.14 highlights the registration process.

$$m_{reg\_req} = E_{pub_{cloud}}(h\{cid\}\|h\{pwd\}\|\delta p\|\delta h\|\delta t\|\phi) \tag{4.10}$$

$h\{cid\}$, $h\{pwd\}$ are the hashed client id and hashed password, respectively. δp defines the choice of PRNG (pseudo random number generator), δh defines the type of hash function the mobile device is running. δt defines the current time-stamp of the mobile device at the time of registration, and ϕ indicates the geographical location (latitude and longitude) of the mobile device at the time of registration. Upon receiving the encrypted message, the cloud server decrypts it with the cloud's

private key prv_{cloud}. Then the server stores the received parameters in the server's big table specific for each of the mobile clients. After storing the parameters, the cloud server creates a client certificate with the client's account credentials and generates a message digest md_{client} of the client's certificate. In addition, it generates a random number as secret key sec_{key}, which is used during the authentication phase for the cloud server validation. The $h\{cid\}$, $h\{pwd\}$ are XOR-ed and used as the seed for the PRNG in order to generate a symmetric key termed as primary key $prim_{key}$ (Eq. 4.11). During the registration phase, the location of the mobile client (ϕ) sent from the mobile device, is used as the state identifier for the PRNG to identify a stream sequence as $prim_{key}$. For all the subsequent primary keys $prim_{key_i}$, previous location of the mobile device ϕ_{prev} is used as the state identifier for the PRNG.

$$h\{cid\} \oplus h\{pwd\} \rightarrow PRNG \xrightarrow{\phi_{prev}} prim_{key_i} \tag{4.11}$$

In addition, the cloud server creates a registration expiry period ϵ for the mobile client, which forces a re-registration. The cloud server decrements the expiry period, which becomes true when the value reaches "0". The re-registration helps in updating password, expiry period, and re-generation of client's certificate. Upon completion of key generation (sec_{key}, $prim_{key}$), message digest (md_{client}), and expiry period (ϵ) the cloud server sends a response back to the mobile client encrypting with primary key $prim_{key}$ (Eq. 4.12). This response m_{reg_res} includes the big table column reference (ref) for efficient lookup.

$$m_{reg\_res} = E_{prim_{key}}(ref||md_{client}||sec_{key}||\epsilon) \tag{4.12}$$

The mobile client generates the primary key $prim_{key}$ using the client's $h\{cid\}$, $h\{pwd\}$, ϕ (Eq. 4.11) and decrypts the received message. The mobile client then stores the four received parameters for the authentication phase.

Authentication Phase

The authentication phase is the core phase of MDLA. The steps in the authentication phase run each time when a mobile client wants to establish a session with the cloud server. This phase ensures that data transmission is possible, if and only if both parties are legitimate and if the mobile client is registered with the cloud server.

We achieve mutual authentication during this phase by exchanging and verifying authentication parameters, such as md_{client} and ϕ. For the authentication phase, the following preconditions must be satisfied.

- The mobile client is registered with the cloud server and received message digest md_{client} from the cloud server during registration.
- The mobile client knows his/her cid and pwd to access the cloud services.
- The cloud server is synchronized with the mobile client, which means the cloud server has δp, δh, δt, ϕ, $h\{cid\}$, and $h\{pwd\}$ for each registered mobile client.
- The mobile client has received ref, and sec_{key} from the cloud sever.

For sending the authentication request, each time the mobile device generates the primary key $prim_{key_i}$ (Eq. 4.11), and a fresh key known as authentication key, $auth_{key_i}$ (Eq. 4.13) by using mobile client's current timestamp δt_{new} as the seed, and mobile client's current location ϕ_{new} as the state identifier for the PRNG. The state identifier is used to specify a PRNG stream sequence as the $auth_{key_i}$. The mobile client sends an authentication request to the cloud server by sending the authentication message m_{auth_req} (Eq. 4.14).

$$\delta t_{new} \rightarrow PRNG \xrightarrow{\phi_{new}} auth_{key_i} \tag{4.13}$$

$$m_{auth\_req} = E_{prim_{key_i}}(E_{auth_{key_i}}(md_{client}) \| \phi_{new} || \delta t_{new}) || ref \tag{4.14}$$

As soon as the cloud server receives m_{auth_req} from the mobile client, it reads the column reference ref and locates the mobile client's information in the server's big table (Fig. 4.15) to get the $prim_{key}$, and ϵ. The job of the column reference is to perform efficient lookup at the big table. If ref is modified or removed during transmission the cloud server decrypts the encrypted portion of the message by performing a trial and error method with all the stored primary keys. The cloud server first verifies with server's table if the registration is still valid for the client's entry by checking the expiry period, ϵ. If the value for ϵ is "0", the cloud server rejects the authentication request and sends a re-registration (described in the update phase)

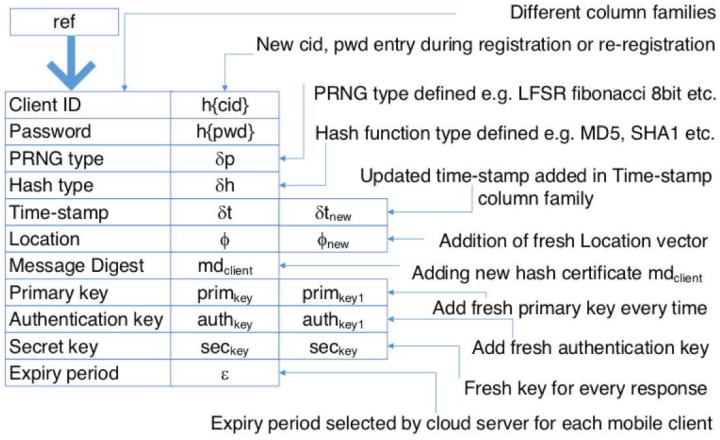

Figure 4.15: Security parameters' entry to the cloud server big table [41]

notification to the mobile client. For a valid registration, the cloud server decrypts the encrypted portion of the authentication message m_{auth_req} using $prim_{key}$ to obtain ϕ_{new}, and δt_{new} used in generating the $auth_{key_i}$ (Eq. 4.13). The $auth_{key_i}$ decrypts the rest of the encrypted message and obtains md_{client}, which is verified with the stored message digest. A successful match indicates that the mobile client is legitimate.

Once the mobile client is authenticated by the cloud server, it is time for the cloud server to be authenticated by the mobile client. The cloud server sends an authentication response message m_{auth_res} (Eq. 4.15).

$$m_{auth\_res} = E_{sec_{key}}(md_{client}||sec_{key_{new}}) \tag{4.15}$$

The response message contains md_{client}, which is encrypted with the pre-shared key sec_{key}. In addition, the cloud server picks up another random number as new secret key and sends it to the mobile client to use it for the next authentication request. The mobile client receives the response message m_{auth_res} from the cloud server and decrypts it using the stored sec_{key}. The received md_{client} is matched with the stored message digest. The successful match ensures the cloud server is legitimate. During this process the old sec_{key} stored in the mobile device is replaced with the new one received from the cloud server.

Update Phase

For a specific mobile client, re-registration takes place when the expiry period ϵ becomes "0". During the reregistration process the cloud server sends $m_{re\_reg\_req}$ (Eq. 4.16) to the mobile client, which contains a registration request encrypted with sec_key. Once the mobile client receives the $m_{re\_reg\_req}$ from the cloud server, it decrypts the message with the stored sec_{key} and retrieves the md_{client}, and registration request. At first, the mobile client validates the cloud server by matching the message digest received with the stored message digest. If the cloud server is legitimate, the mobile client reads the re-registration request and sends m_{reg_req} (Eq. 4.10) to the cloud server. A re-registration process expects the mobile device to fetch a new $hpwd$, and a new timestamp δt. The cloud server re generates a fresh client certificate, message digest md_{client}, and a new expiry period ϵ during re-registration. The cid, δp, and δh remain unchanged.

$$m_{re\_reg\_req} = E_{sec_{key}}(md_{client}||re\_registartion\_request) \tag{4.16}$$

Updating the authentication refers to a re-authentication process, which is triggered if there is a sudden connection loss after the session establishment. The reason for connection loss could be a network error, value of ϵ becoming "0" during an ongoing session, or some type of forced termination of the connection. The re-authentication is same as the authentication process, and is initiated by sending the m_{auth_req} (Eq. 4.14) to the cloud server in order to prevent any rogue client to access the cloud server during a session re-establishment. Updating keys refers to the key generation process. Keys are updated frequently to make sure even if a key is compromised, the subsequent messages are still secure. We use three major keys in our authentication scheme termed as primary key - $prim_{key}$, authentication key - $auth_{key}$, and secret key - sec_{key}. The primary key generation (Eq. 4.11) accepts the $h\{cid\}$ XOR-ed with the $h\{pwd\}$ as the seed to the PRNG. The variability of the $prim_{key}$ is introduced by using the previous location ϕ_{prev} of the mobile device as state identifier to the PRNG. The $auth_{key_i}$ generation (Eq. 4.13) accepts the current timestamp δt_{new} as the seed to the PRNG and current location of the mobile device ϕ_{new} as the state identifier. The $auth_{key}$ is fresh for every authentication request due to the use of two variables, the current time-stamp and the current location. The secret key sec_{key} is

a random number chosen by the cloud server during the registration phase and sent to the mobile client. An old key is replaced with a newer key $sec_{key_{new}}$ during every authentication response that is sent from the cloud server to the mobile client. Every authentication request ensures a sec_{key} update.

4.6.3 AMLT: Authentication based on Message digest, Location, and Timestamp

We propose a novel authentication scheme, AMLT (Authentication based on Message digest, Location, and Timestamp) to validate the mobile client and the cloud server participating in the mobile cloud computing, which is independent of USIM. The proposed scheme, AMLT, is an updated version of the previously proposed method "MDLA (message digest and location based authentication)"[38]. Technically, AMLT uses message digest generated by the mobile client, location of mobile client, timestamp of mobile client and that of cloud server as principle parameters for authentication. In detail, multiple keys based on these parameters are generated, which are used to secure the exchanged messages. Furthermore, the update of keys with minimum correlation to the previous keys introduces the unpredictability to the system, which makes the system less vulnerable to attacks.

AMLT is a 3-phased scheme, wherein the phase of Initial Handshake is required for account setup. Initial Handshake is a one-time process and it requires asymmetric encryption. This is followed by the phase of Mutual Authentication, which ensures a mutual authentication of registered client and the cloud server. In addition, as an added security measure, during the third phase of Time-based Re-registration, a forced re-registration takes place. The first phase of the scheme is Initial Handshake, which stores the mobile client's credentials to the cloud server. In addition, this phase generates a seed key and a message digest for the initiation of the authentication phase. A mobile client registered through Initial Handshake can access the resources at the cloud server by the successful completion of mutual authentication of cloud server and mobile client. The details of these phases are described as follows.

Initial Handshake

The Initial Handshake phase binds a mobile client with a cloud service provider. This phase includes sharing a mobile client's information with the cloud server (Fig. 4.16). In a typical scenario, a cloud server is certified by a certification authority, therefore, in Initial Handshake phase, we concentrate on certification of the mobile client. The mobile client generates its certificate as a hashed message md_{client}(Eq. 4.17), and sends a registration request m_{reg_req} to the cloud server (Eq. 4.18) along with some key parameters.

$$md_{client} = H\{c\text{-}id|c\text{-}pwd\} \tag{4.17}$$

$$m_{reg\_req} = AE_{k_{pub\text{-}server}}(md_{client}|\delta e|\delta h|\delta p|\delta tc|\delta l) \tag{4.18}$$

Initial Handshake phase uses asymmetric encryption AE and the request message is sent to the cloud server by encrypting with the server public key $AE_{k_{pub-server}}$. The request message m_{reg_req} is decrypted at the server end, using cloud's private key $AE_{k_{prv-server}}$. The decrypted message provides configuration about the client, which the cloud server stores in its database for that specific client, and the record is marked by client record pointer $c - rp$. Mobile client and the cloud server run identical hash function, and PRNG (pseudo random number generator) by sharing type of hashfunction field δh, and choice of PRNG δp field. In addition, the mobile client fetches its current location, referred as location vector, and timestamp δtc to the cloud server for synchronization. Once the cloud sever approves the request, it generates a symmetric key $k_{sym-S2C}$, using $PRNG_{\delta p}$ (Eq. 4.19). The client's timestamp is used as the seed value to the PRNG and location vector is used as the state identifier the i^{th} generation. In addition, bit-flipping operation is introduced on the location vector to increase the degree of unpredictability.

$$k_{sym\text{-}S2C} = PRNG_{\delta p}(H_{\delta h}(\delta tc), BF(\delta l)) \tag{4.19}$$

$$c\text{-}rp_{scd} = AE_{k_{pub\text{-}server}}(c\text{-}rp) \tag{4.20}$$

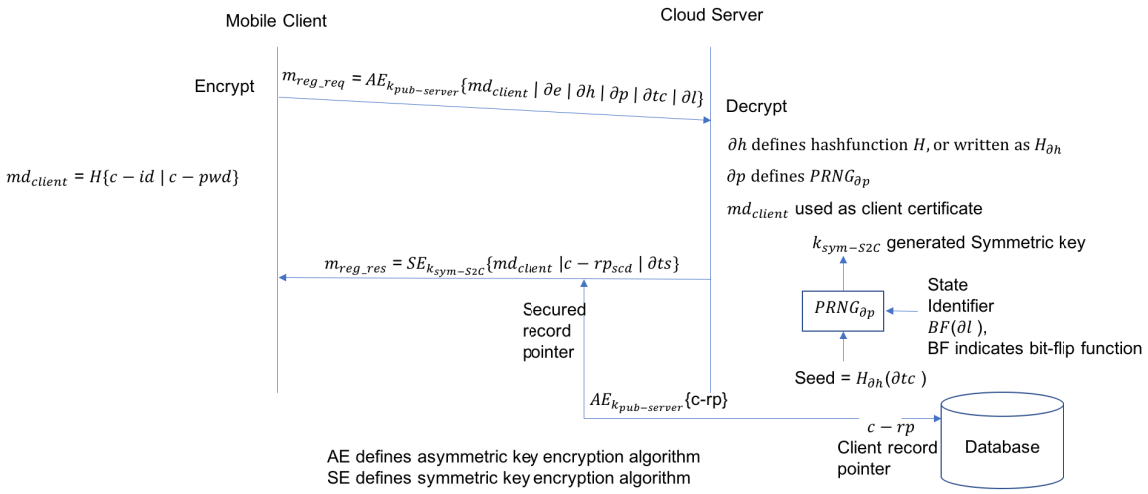

Figure 4.16: Initial Handshake in AMLT

$$m_{reg\_res} = SE_{k_{sym-S2C}}(md_{client}|c\text{-}rp_{scd}|\delta ts) \tag{4.21}$$

These operations take place at the server side, and therefore, do not consume client side resources. Furthermore, the cloud server generates secured record pointer (Eq. 4.20) for the client, which ensures efficient table lookup during the authentication phase. A secured registration response message is sent to the mobile client or cloud client at the end of the Initial Handshake phase, which indicates a successful registration. Eq. 4.21, shows that symmetric key encryption is used to secure the response message. In addtion to md_{client}, the cloud server shares the secured client record pointer, and cloud's current timestamp δts with the mobile client. A standard public key encryption is used for transferring parameters during the registration request process. During the registration response, a chosen symmetric key encryption algorithm is used, which is determined by δe . Initial Handshake is a one time or infrequently occurring process, therefore, the use of asymmetric encryption in registration request has much lesser impact on resource utilization.

Mutual Authentication

The Mutual Authentication phase is the core phase of AMLT. The steps in the Mutual Authentication phase run each time when a mobile client wants to establish a session with the cloud server. This phase ensures that data transmission is possible, if and

only if both parties are legitimate and if the mobile client is registered with the cloud server. We achieve mutual authentication during this phase by exchanging and verifying authentication parameters, such as md_{client}.

During the authentication phase, the registered mobile client or cloud client, generates a secret key $k_{sym-C2S}$, using the PRNG (Eq. 4.22). The seed value for the PRNG is obtained by XORing message digest or client certificate with hashed server timestamp, which makes the seed value dynamic for every key generation. In addition, bit-flipping operation is performed on the server timestamp to generate a state identifier, which considers the i^{th} PRNG generation as the client's symmetric key $k_{sym-C2S}$.

$$k_{sym\text{-}C2S} = PRNG_{\delta p}(H_{\delta h}(\delta ts) \oplus md_{client}, BF(\delta ts)) \tag{4.22}$$

$$m_{auth\_req} = SE_{k_{sym\text{-}C2S}}(md_{client}|\delta tc|\delta l)|c\text{-}rp_{scd} \tag{4.23}$$

A secure session establishment is the primary objective of the Mutual Authentication phase, therefore, a secure authentication request is sent from the client side encrypting with symmetric key, $k_{sym-C2S}$ (Eq. 4.23). A standard symmetric key encryption algorithm, SE is used in this process by both the parties. The pre-shared algorithm indicator δe guarantees that the cloud server runs client specific encryption algorithm. Upon receiving the authentication request, the cloud server decrypts the secured record pointer $c - rp_{scd}$ using cloud's private key, and locates the client information in the cloud database. Once the client information is found, the cloud server checks the registration expiry ϵ of the client, and if the client is found registered, then the server begins client authentication request processing. The cloud server generates $k_{sym-C2S}$, using its pre-shared timestamp, δts, and decrypts m_{auth_req}. Then the cloud server updates its database with the new client timestamp, δtc, and location vector, δl, information. The obtained md_{client}, is verified with the stored message digest, and a successful match establishes the authenticity of the mobile client.

Once the mobile client is authenticated by the cloud server, it is time for the cloud server to be authenticated by the mobile client. Therefore, the cloud server sends a secure authentication response message m_{auth_res} (Eq. 4.24) to the mobile client, which is encrypted using the server key, $k_{sym-S2C}$. The mobile client receives the

response message m_{auth_res} from the cloud server and decrypts it using the generated server secret key $k_{sym-S2C}$. Mobile client uses, the same client timestamp, δtc, and location vector, δl, to compute the key $k_{sym-S2C}$. The received md_{client} is matched with the stored message digest. The successful match ensures the cloud server is legitimate. During this process the old server timestamp stored in the mobile device is replaced with the new one δts received from the cloud server.

$$m_{auth\_res} = SE_{k_{sym-S2C}}(md_{client}|\delta ts) \tag{4.24}$$

Time-based Re-registration

The phase of Time-based Re-registration refers to the updating of registration information and re-authentication.

For a specific mobile client, re-registration takes place when the expiry period, ϵ indicates that the client registration is expired. During the re-registration process the cloud server sends a registration update request, $m_{reg\_upd\_req}$ (Eq. 4.25) to the mobile client, which contains mobile client certificate, md_{client}, and a registration update request, $req_{reg-upd}$ encrypted with server symmetric key, $k_{sym-S2C}$. Once the mobile client receives the re-registration request, $m_{reg\_upd\_req}$, it generates symmetric key $k_{sym-S2C}$ (Eq. 4.19) and decrypts the message. At first, the mobile client validates the cloud server by matching the message digest received with the stored message digest. If the cloud server is legitimate, the mobile client reads the re-registration request and sends m_{reg_req} (Eq. 4.18) to the cloud server. A re-registration process expects the mobile device to fetch a new md_{client}, and a new timestamp δtc. The cloud server adds a new expiry period ϵ during re-registration. The other parameters, such as δtc, δp, δh, and δe could remain unchanged.

$$m_{reg\_upd\_req} = SE_{k_{sym-S2C}}(md_{client}|req_{reg\text{-}upd}) \tag{4.25}$$

Updating the authentication refers to a re-authentication process, which is triggered, if there is a sudden connection loss after the session establishment. The reason for connection loss could be due to network error, value of ϵ becoming "expired" during an ongoing session, or some type of forced termination of the connection. The re-authentication is same as the authentication process, and is initiated by sending

the m_{auth_req} (Eq. 4.23) to the cloud server in order to prevent any rogue client to access the cloud server during a session re-establishment.

4.7 Chapter Summary

- The chapter presents a security framework for mobile cloud computing.
- The security framework is called context-aware for its adaptable nature.
- The context-aware framework works with a cloud of cloud model that consists of one master or outer cloud and n inner clouds.
- The master cloud is basically an access control server placed at the outer layer of the cloud infrastructure to filter incoming traffic.
- The inner clouds are conceptualized as a set of hypervisors, where each hypervisor initializes multiple virtual machines to serve clients.
- The security framework has three modules. Cognitive module, adaptive module, and authentication module. These modules are deployed on different layers of the cloud of cloud model to provide a multistage security.
- Cognitive module filters out the unauthorized incoming traffic based on pattern matching, and clustering techniques.
- Cognitive module has two phases. $Phase1$ performs $profile$ verification, and $Phase2$ performs $feature$ verification.
- $feature$ verification is performed using clustering techniques. The research proposes KD algorithm.
- Output of cognitive module is the traffic set, and a flag λ.
- An accepted traffic is processed by adaptive module and an appropriate inner cloud is assigned to serve the request.
- The adaptive module destroys a compromised inner cloud instance, and recreates a new inner cloud instance with different security configuration.

- The authentication module is deployed at each inner cloud, and is re-configurable depending on the application or mobile topology.
- Any standard and suitable authentication scheme could be used in the authentication module depending on the requirement, and resource availability.

Chapter 5

Performance Evaluation

5.1 Overview

A research can be performed by performing different types of studies, such as surveys, field study, experiments, computer simulation etc. [23]. The three modules of the context-aware framework are evaluated using performance analysis, model analysis, simulation, and formal analysis. The evaluation methodology ensures reliability and security of the framework. The design of the cognitive module is evaluated using performance analysis, which involves attack recognition, and reliability. The adaptive module is verified by model analysis, which is performed using PRISM (probabilistic symbolic model checker) [72][39]. The proposed authentication module is evaluated using simulation, and formal security analysis with determining the degree of vulnerability [37, 38, 40, 41].

5.2 Reliability and Accuracy of Context-Aware Framework

The reliability and accuracy of the proposed context-aware framework depends on the reliability and performance of the three individual modules. To validate the performance of the framework two experiments are required. In one experiment, a fixed input traffic set is processed multiple times by the context-aware framework and the observations are recorded. In another experiment, different input traffic sets are processed by the framework for a given period and the observations are recorded. For a better result, all three modules must be validated individually. The context-aware framework is reconfigurable, and in addition, a learning system is associated, therefore, upgrading the modules and adding more features improves the accuracy of the framework.

The following assumptions are made as part of the evaluation methodologies. Let us consider,

Input number of request traffic = η

Number of legitimate traffic requests in input set = η_l

Number of malicious traffic requests in input set = η_m

Detected malicious traffic by context-aware framework= ρ

Detected malicious traffic, which is actually legitimate = ρ_l

Detected malicious traffic, which is actually malicious = ρ_m (True Positive)

Detected legitimate traffic by context-aware framework= θ

Detected legitimate traffic, which is acutally legitimate = θ_l (True Positive)

Detected legitimate traffic, which is acutally malicious = θ_m

A specific experiment or RUN indicated by = τ

Considering the aforesaid assumptions, Tab. 5.1 defines a confusion matrix with the values for false positive (FP), false negative (FN), true positive (TP), and true negative (TN) to highlight whether the system has detected the malicious traffic correctly. E.g. θ_m represents a false negative condition, where a malicious traffic is not detected as malicious. In the case of malicious traffic detection, ρ_l represents a false positive condition, where a legitimate traffic is detected as malicious.

Table 5.1: Confusion Matrix

		Predicted Cases		
		Detected Legitimate	Detected Malicious	Total
Actual Cases	True Legitimate	θ_l	ρ_l	η_l
	True Malicious	θ_m	ρ_m	η_m
	Total	θ	ρ	η

Tab. 5.2 represents the mock values from experiment 1, and Tab. 5.3 represents the mock values from experiment 2. Based on the experimental results, accuracy, precision and recall are calculated, which help in determining the performance of the framework and the evaluation of the learning system.

Accuracy

Accuracy of the cognitive module or the context-aware framework is tested for two conditions (Equation 5.1).

Condition 1: Same input traffic set η is used for n runs, i.e. η input set is passed through the framework from RUN τ_0 to τ_n to test effect of learning on accuracy

Table 5.2: Experiment 1 - Same input traffic set is processed for n RUNS

Input set	RUN	Legitimate traffic in input set	Malicious traffic in input set	Detected malicious traffic	Detected legitimate traffic
η	τ_0	η_l	η_m	ρ_0	θ_0
η	τ_1	η_l	η_m	ρ_1	θ_1
η	τ_2	η_l	η_m	ρ_2	θ_2
η	τ_3	η_l	η_m	ρ_3	θ_3
η	τ_4	η_l	η_m	ρ_4	θ_4
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
η	τ_n	η_l	η_m	ρ_n	θ_n

of the framework. If the error-percentage reduces over time, then the framework is considered to be accurate.

Condition 2: Different input traffic set η_i is used for n runs, i.e. η_0 to η_n input set are passed through the framework from RUN τ_0 to τ_n to test effect of learning. Testing this condition ensures the system is capable of processing different types of input traffic sets. This also ensures the system is reliable and robust.

$$error - percentage = \frac{|\rho - \eta_m|}{|\eta_m|} \times 100 \tag{5.1}$$

Reliability

To find how reliable the context-aware framework is, precision of malicious traffic detection is calculated using experiment 1 data. If the precision value remains constant or exhibit minimal changes for same input traffic set η, then the system is considered as reliable.

$$precision = \frac{\rho_m}{\rho_m + \rho_l} \tag{5.2}$$

Table 5.3: Experiment 2 - Different input traffic set is processed for n experimental RUNs

Input set	RUN	Legitimate traffic in input set	Malicious traffic in nput set	Detected malicious traffic	Detected legitimate traffic
η_0	τ_0	η_{l_0}	η_{m_0}	ρ_0	θ_0
η_1	τ_1	η_{l_1}	η_{m_1}	ρ_1	θ_1
η_2	τ_2	η_{l_2}	η_{m_2}	ρ_2	θ_2
η_3	τ_3	η_{l_3}	η_{m_3}	ρ_3	θ_3
η_4	τ_4	η_{l_4}	η_{m_4}	ρ_4	θ_4
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
η_n	τ_n	η_{l_n}	η_{m_n}	ρ_n	θ_n

In addition, the probability distribution of the detection of malicious traffic is given by recall. For a fixed input traffic set η for n experimental RUNs, if the probability of detection of malicious traffic varies, then the system is considered as non-reliable. This could also happen due to the learning effect of the system.

$$recall = \frac{\rho_m}{\rho_m + \theta_m} \tag{5.3}$$

As necessary condition of these experiments, all the three modules' functionality must be tested.

5.3 Validation of Coginitve Module and Results

The functionality of the cognitive module depends on the existing knowledge base and the algorithms used for traffic filtration. The propose system, uses KD algorithm in learning, which could be replaced by other algorithms depending on the requirements. Cognitive module performs traffic screening in two phases for $profile$, and $feature$ based filtration. The following sections provide the steps involved in the training and

testing of the cognitive module.

5.3.1 Building and Usage of Client Profile

$profile$ indicates client's profile information available in the cloud server that includes client's OS, and coarse location. This information could be obtained during the registration process, and updated upon completion of a successful session. The implementation of this module consists of building database for registered client. Tab. 5.4 indicates a sample user database, which consists of location values and OS used during registration and last update. The sample databse is a subset of user information stored at the cloud premises. For this database (Tab. 5.4), we have considered only parameters required by the cognitive module. The "Cid", "TimeStamp", "XAxis", and "YAxis" values in the sample database are populated from another study performed by Jetcheva *et al.*, where actual movement of buses in the Seattle (Washington area King County Metro bus system) are recorded[62]. Apart from representing the structure of client database and type of data, the study of Jetcheva *et al.* is not used in the validation process of congnitive module.

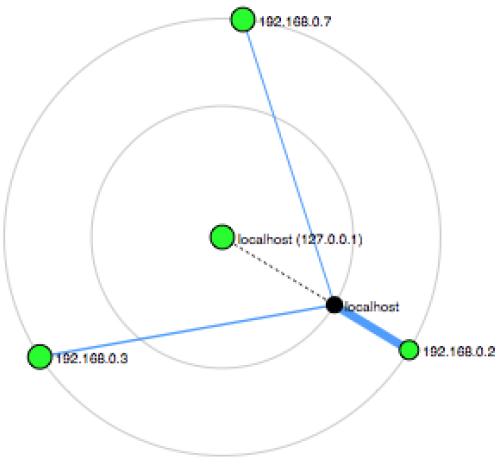

Figure 5.1: Wireless LAN setup with devices running Android, OSx, iOS, Linux. Green colour indicats devices have fewer than 3 open ports

The coordinate values in Tab. 5.4 are used for location based filtering. For simplicity, the sample OS values are randomly assigned. During $Phase1$ screening, $profile$ information performs the basic filtration. From any incoming traffic the

Table 5.4: Sample Data from UserDatabase used in Phase 1

Cid	TimeStamp	XAxis	YAxis	City	InitialOS	OSUpdate
1001	2182812626	45667.5	153846.9063	Seattle	ios 10.0.x	ios 10.0.x
999	2182812628	50225	163284.9688	Seattle	ios 10.0.x	ios 10.0.x
996	2182812627	41868.375	161553.5781	Seattle	OSx 10.4	OSx 10.5
995	2182812623	39538.625	158561.5313	Seattle	Ubuntu 14.04	Ubuntu 14.05
992	2182812634	57607.25	155747.5469	Seattle	OSx 10.4	OSx 10.5
987	2182812848	45324.25	152641.6094	Seattle	ios 10.0.x	ios 10.0.x
986	2182812625	43827.875	155169.2031	Seattle	ios 10.0.x	ios 10.0.x
960	2182812772	56641.625	142338.875	Seattle	Ubuntu 14.04	Ubuntu 14.05
955	2182812624	41725.75	165150.0313	Seattle	OSx 10.4	OSx 10.5
950	2182812649	47674.375	151243.1875	Seattle	Ubuntu 14.04	Ubuntu 14.05
948	2182812649	45454.375	156132	Seattle	ios 10.0.x	OSx 10.4
943	2182812735	55806.875	156083.9688	Seattle	Ubuntu 14.04	Ubuntu 14.05
941	2182812641	51523.5	163249.8281	Seattle	OSx 10.4	OSx 10.4
936	2182812633	44044.625	154803.0469	Seattle	Ubuntu 14.04	Ubuntu 14.05
922	2182812628	48083	156257.9688	Seattle	ios 10.0.x	ios 10.0.x
918	2182812627	46724.625	161394.5	Seattle	OSx 10.4	OSx 10.4
915	2182812643	45627.125	152175.2969	Seattle	OSx 10.4	OSx 10.4
914	2182813459	52976.625	145469.3125	Seattle	ios 10.0.x	ios 11.0.x
912	2182812617	38429	170112.0313	Seattle	Ubuntu 14.04	ios 11.0.x
903	2182812643	47086.75	153455.9844	Seattle	OSx 10.4	OSx 10.4
948	2182812617	45712.5	156284.9531	Seattle	OSx 10.4	OSx 10.4
950	2182812617	46874.625	151257.5781	Seattle	ios 10.0.x	ios 10.0.x
903	2182812614	47424.25	153463.3125	Seattle	ios 10.0.x	ios 10.0.x
915	2182812614	45618.375	151775.6094	Seattle	OSx 10.4	OSx 10.4
941	2182812613	51523.5	163249.8281	Seattle	OSx 10.4	OSx 10.4
999	2182812601	50225	163284.9688	Seattle	ios 10.0.x	ios 10.0.x
992	2182812601	57607.25	155747.5469	Seattle	ios 10.0.x	ios 10.0.x
936	2182812600	44044.625	154803.0469	Seattle	OSx 10.4	ios 11.0.x
1001	2182812599	45633.75	154181.0313	Seattle	OSx 10.5	ios 11.0.x
955	2182812597	41577.125	164490.7656	Seattle	ios 10.0.x	ios 10.0.x
995	2182812596	39554	159161.3906	Seattle	OSx 10.5	OSx 10.5
922	2182812595	48083	156257.9688	Seattle	OSx 10.5	OSx 10.5
996	2182812595	41893	162793.5469	Seattle	ios 10.0.x	ios 10.0.x
918	2182812595	46724.625	161394.5	Seattle	ios 10.0.x	ios 10.0.x
986	2182812592	43827.875	155169.2031	Seattle	OSx 10.4	OSx 10.5

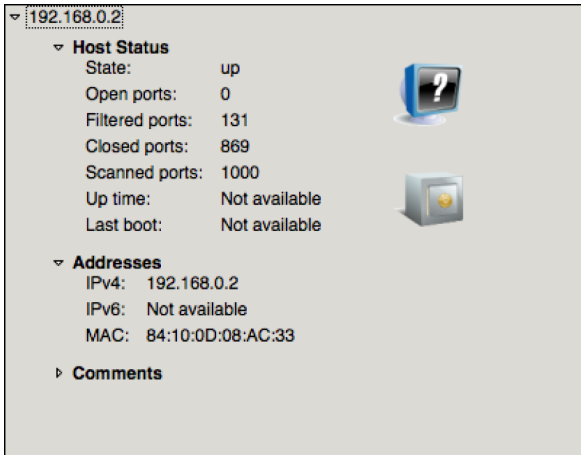

Figure 5.2: Zenmap screen capture. OS detection of remote host running Android

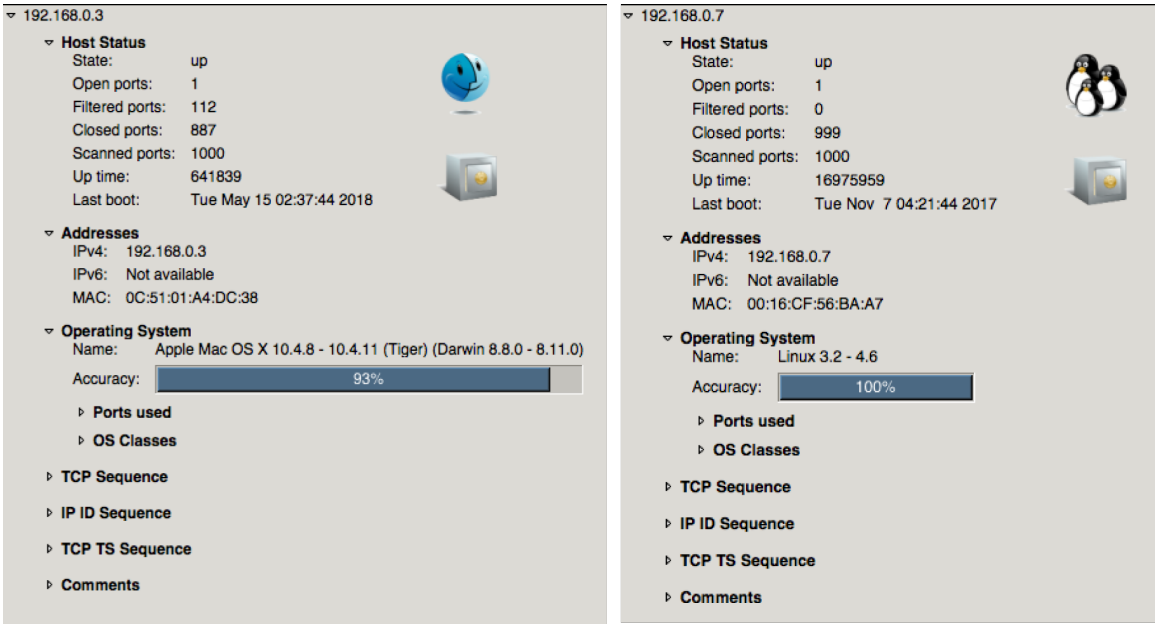

Figure 5.3: Zenmap screen capture. OS detection of remote host runningiOS and Linux

required information are extracted and matched with the database. However, OS and location information cannot be directly extracted from the incoming traffic. If the incoming traffic is from VANET, then the payload contains the location information. However, for other types of network traffic it is difficult to identify the location. There are various open source tools available, which can provide coarse location, such as "Zenmap"[80]. In addition to zenmap, we can also purchase a database from website such as `https://www.ip2location.com/`, then we can map each ip address to a city. If the client system allows, then zenmap can detect the OS and the gateway location, which can provide an approximation of the client's location. Fig. 5.3, and Fig. 5.2 provide screenshots of "Zenmap" experiments in our local area network (Fig. 5.1). We successfully obtained OS information of two devices connected in the LAN. "Zenmap" scanning could not obtain remote OS information from the Android device. This type of problem can be addressed, by granting permissions to cloud-basd applications during the client registration or installation of cloud-based applications on mobile devices. In addition, other tools or services can be utilized to detect client OS and location information.

In this scenario, since location entry is not available for the specific hosts, if incoming traffic obtained from the Linux device matches the database entry ($profile = p_{10}$), then the traffic is considered as partially legitimate or a possible MITM attack (Tab. 4.1). Therefore, $Phase1$ passes it to $Phase2$.

5.3.2 Cluster Analysis of Training Data

The operation of $Phase2$ depends on the effective training of the system. In the previous chapter, we have mentioned that inter packet delay (IPD) is considered as the key $feature$ for this training. In order to obtain the IPDs for system training, we have selected one specific client and one application layer protocol, FTP (File Transfer Protocol). Depending on the type of application and service(s), any application layer protocols can be considered for system training and testing. The training considers inter-packet delay vector as a data point in the training and data filtration, which forms patterns in two consecutive vectors, such as {short delay, long delay} & {long delay, short delay}, or {long delay, short delay} & {short delay, long delay} etc. Therefore, this inter-packet delay-based traffic filtration can be used irrespective of

the type of application layer protocol. As an extension of the performance evaluation, other application layer protocol, such as HTTP (Hypertext Transfer Protocol) can also be used. However, for simplicity and convenience we have considered FTP as the application layer protocol for the training and traffic filtration. For the experiment, we have set up a client-server connection, and transferred relatively large file. Fig. 5.4 shows the screenshot of Wireshark FTP packet capture. During this session, the total number of data packets received is, 741026.

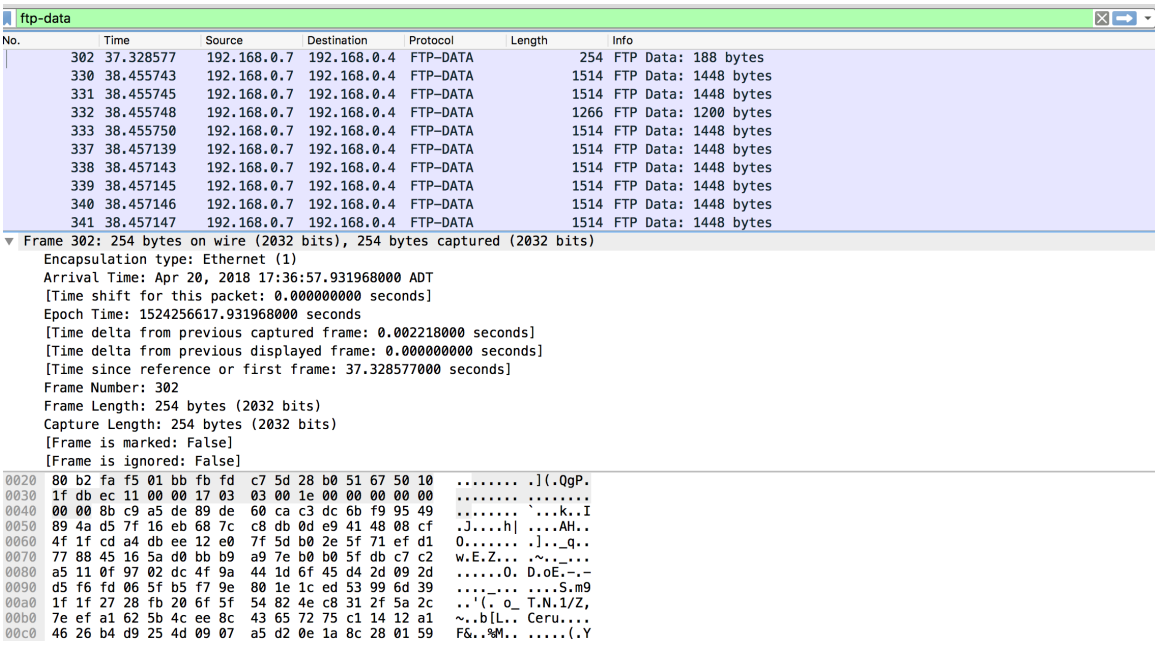

Figure 5.4: Screenshot of Wireshark Capture of FTP Packets

The arrival times of packets at the destination are used to calculate the IPDs (Eq. 5.4). These IPDs are used to construct inter packet delay vectors (IPDVs). After calculating the IPDVs (Eq. 5.5), we prepare the training set.

$$ipd_i = arrival\_time_{packet_i} - arrival\_time_{packet_{(i-1)}} \tag{5.4}$$

$$IPDV_i = \{ipd_{(i-1)}, ipd_i\} \tag{5.5}$$

Among 741026 data points, we extract a random set of $n = 58001$ consecutive data points, ignoring data points at the beginning and end. We have chosen > 50000 data points, to identify useful patterns, and less than 100000 data points for faster processing during the training phase. The value of n could be increased based on

Table 5.5: Sample FTP Data from 58001 data points, and calculated IPDVs

Time	V1	V2	srcIP	dstIP
41.724614	0.000002	0.001652	192.168.0.7	192.168.0.4
41.726266	0.001652	0.000007	192.168.0.7	192.168.0.4
41.726273	0.000007	0.000065	192.168.0.7	192.168.0.4
41.726338	0.000065	0.000003	192.168.0.7	192.168.0.4
41.726341	0.000003	0.001489	192.168.0.7	192.168.0.4
41.72783	0.001489	0.000004	192.168.0.7	192.168.0.4
41.727834	0.000004	0.000123	192.168.0.7	192.168.0.4
41.727957	0.000123	0.000004	192.168.0.7	192.168.0.4
41.727961	0.000004	0.001431	192.168.0.7	192.168.0.4

requirements. Tab 5.5 highlights the structure of the data points, where $V1$ and $V2$, represents $ipd_{(i-1)}$, and ipd_i

Since we have only set of vectors in our training set, we apply unsupervised learning technique (clustering) on our data points. We have implemented proposed KD (Alg. 1) in "R"[116] and created the first set of clusters using K-Means algorithm. To select an optimal initial value for k, we have applied "Elbow" method (Fig. 5.5), which shows after $k = 5$, the sum of squared error (SSE) decreases.

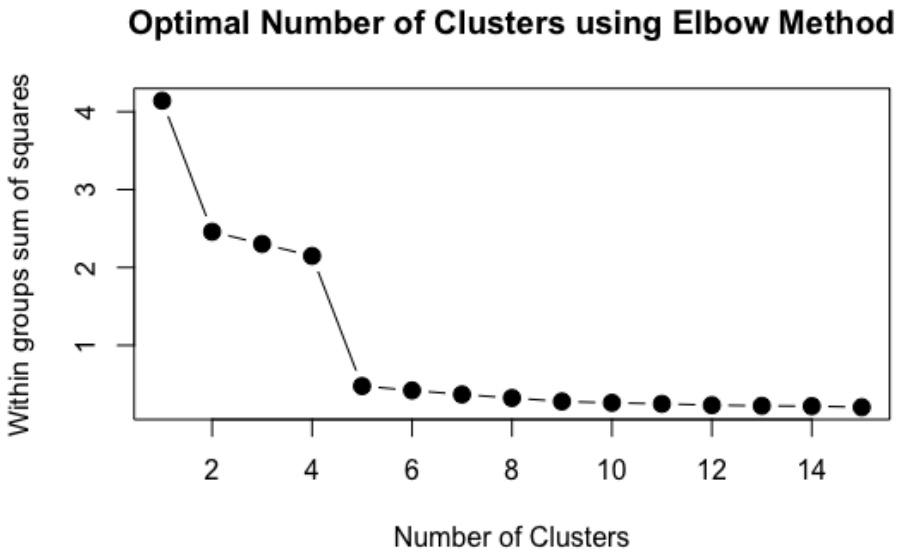

Figure 5.5: Elbow method analysis to find the optimum value of k for K-Means

K-Means with $k = 5$ provides ($between\_SS/total\_SS = 88.5\%$), which is a good

measure and for other higher values of k the measure of the total variance does not change significantly. Our experimental results show that among the 5 clusters, two clusters are having less number of data points (82, and 83), two are having slightly over 2000 data points(2266, 2218), and one cluster has 53352 data points. Tab. 5.6 provides centroids of the clusters generated using K-Means.

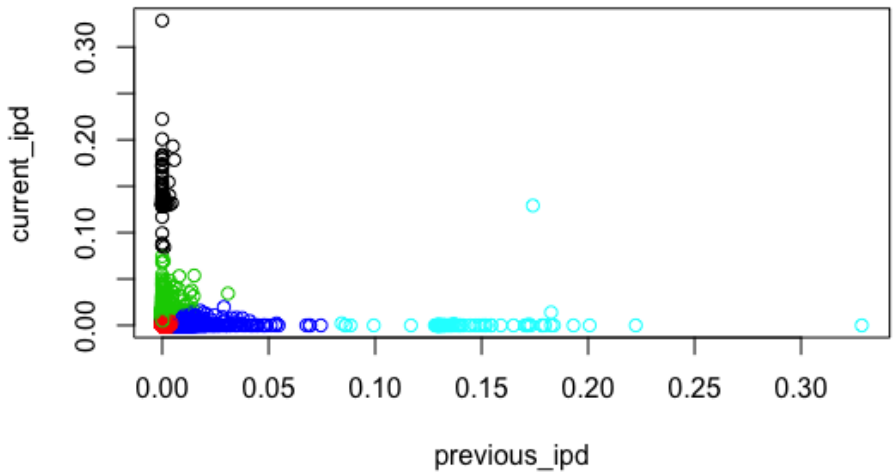

Figure 5.6: K-Means cluster when k = 5

Table 5.6: Centroids of the obtained clusters

Cluster	previousIPD	currentIPD
C1	0.000771893	0.009173547
C2	0.000553803	0.000557287
C3	0.000432842	0.143516342
C4	0.009088005	0.00079173
C5	0.14334306	0.001786663

However, the clusters generated by K-Means (Fig. 5.6) contain outliers, which should be removed to obtain clean clusters. We use DBSCAN density based clustering on each of the generated clusters. DBSCAN accepts *eps*, and *minPts* as input. DBSCAN computes pair-wise distance $dist(x_i, x_j)$ between data points in each cluster and form *eps* distance neighbourhoods (Algo. 1). We have chosen a fixed value 4 for *minPts*, and use kNNdist (k-Nearest Neighbour Distance) plot to find optimal values

for *eps* for each of the clusters [54].

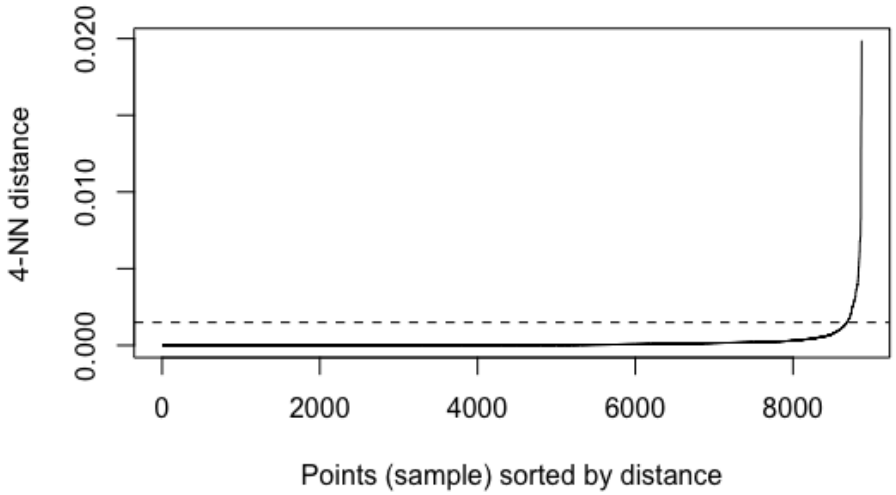

Figure 5.7: kNNdist plot applied on cluster 1 to find the optimum *eps* value for DBSCAN

Table 5.7: Number of outliers detected using DBSCAN and number of remaining data points in clean clusters

	C1	C2	C3	C4	C5
K-Means	2218	53352	82	2266	83
Outliers	42	1439	2	37	3
CleanClusters	2176	51913	80	2229	80

Fig. 5.7 and Fig. 5.8 indicate the sample knee plot and the DBSCAN plot. The dotted horizontal lines at the knee plot indicates the optimum *eps* value for the cluster data points under observation. We have used the obtained *eps* value as input to DBSCAN to find outliers in the cluster. Tab. 5.7 summarize the results from DBSCAN plots, and indicates the total number of outliers in each cluster. Tab. 5.8 provides the centroids of the clean clusters.

5.3.3 Testing for MITM and DDoS Attack

The cognitive module is validated for MITM and DDoS attacks. The $profile$ verification at $Phase1$ performs basic screening, such as location and OS information.

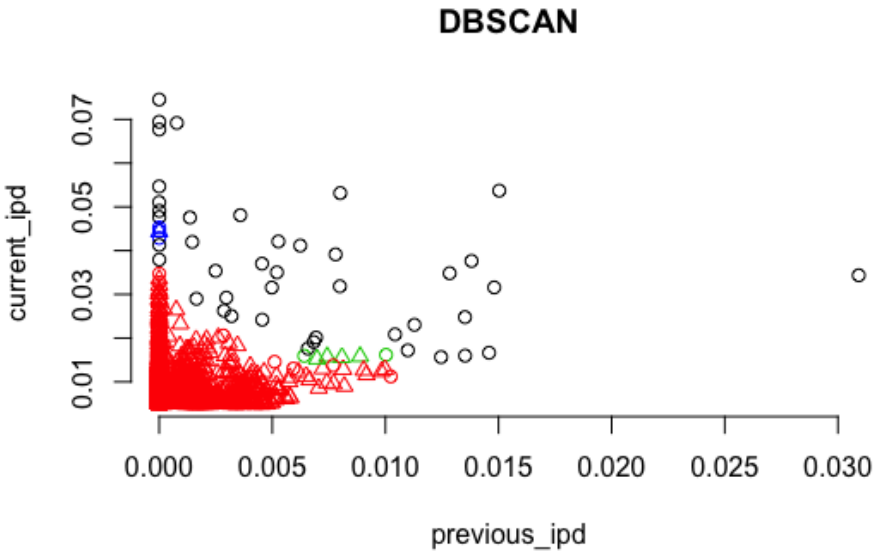

Figure 5.8: DBSCAN Density-based clustering on Cluster 1 data set

Table 5.8: Centroids of the clean clusters

Cluster	previousIPD	currentIPD
C1	0.008614483	0.000763544
C2	0.000390181	0.000522646
C3	0.0004436	0.138640563
C4	0.000631617	0.00814219
C5	0.138883051	0.00024238

Therefore, *Phase*1 does not perform well in terms of detecting MITM and DDoS attack. *Phase*2 performs filtration based on clustering techniques, and we found this method is effective in threat detection.

For detecting anomalous behaviour in the incoming traffic, we perform distance based traffic filtration for a client. Eq. 4.7 is used to calculate the squared Euclidean distance between the new incoming vector and the centroids of each cluster (Tab. 5.6). In addition, distance between the farthest point and the centroid of a cluster is measured. If the calculated distance (Eq. 5.6), D is ≥ 0, then the incoming traffic is considered as legitimate and it is placed in the appropriate cluster.

$$D = \sum_{i=1}^{k} dist(x_{farthest}^{(i)}, c_i) - dist(x_{new}, c_i) \tag{5.6}$$

Table 5.9: Sample IPD data from simulated rogue client

IPD	V1	V2	srcIP
0.099999905	0.099999905	0.049999952	192.168.0.9
0.049999952	0.049999952	0.050000191	192.168.0.9
0.050000191	0.050000191	0.049999952	192.168.0.9
0.049999952	0.049999952	0.549999952	192.168.0.9
0.549999952	0.549999952	0.549999952	192.168.0.9
0.549999952	0.549999952	0.200000048	192.168.0.9
0.200000048	0.200000048	0.359999895	192.168.0.9
0.359999895	0.359999895	0.230000019	192.168.0.9
0.230000019	0.230000019	0.090000153	192.168.0.9
0.090000153	0.090000153	0.089999914	192.168.0.9
0.089999914	0.089999914	0.190000057	192.168.0.9
0.190000057	0.190000057	0.419999838	192.168.0.9
0.419999838	0.419999838	0.410000086	192.168.0.9

In order to validate the system for possible DDoS or MITM attack, we have simulated a rogue client using "Python" socket program and generated set of packets that have very small packet arrival time (Tab. 5.9). We have considered, during a DDoS attack multiple requests are sent within a short period of time, therefore, the packets' arival times are kept very small. For an MITM attack scenario, packets can be generated with different arrival times, however, due to the presence of previous, and current inter-packet delays, a pattern is formed, such as {short delay, long delay} or {long delay, short delay}, or {long delay, long delay}, or {short delay, short delay} and the probability of getting an exact match of pattern between a valid client and rogue client is less. Furthermore, we calculate the distance for each possible rouge points from the centroid, and compute D by considering distance to the farthest point. Tab. 5.10, Tab 5.11, and Tab. 5.12 present sample of our findings with "three" possible rogue client requests. Vectors from probable rogue incoming traffic are computed, and distance, D is measured. "One" of the incoming traffic is not accepted by any clusters due to $D \not> 0$ and flagged as threat.

Table 5.10: Distance calculation for incoming request req1 with vector $\{V1_{req1} = 0.049999952, V2_{req1} = 0.549999952\}$

dist(RogueToCentroid)	dist(FarthestToCentroid)	D
0.303303459	0.00428419	-0.299019269
0.309250746	0.034286644	-0.274964102
0.294916602	0.004272975	-0.290643627
0.167685824	0.03421951	-0.133466313
0.304332164	0.000033671	-0.304298493

Table 5.11: Distance calculation for incoming request req2 with vector $\{V1_{req2} = 0.230000019, V2_{req2} = 0.090000153\}$

dist(RogueToCentroid)	dist(FarthestToCentroid)	D
0.056760261	0.00428419	-0.052476071
0.015291048	0.034286644	0.018995596
0.059078474	0.004272975	-0.054805499
0.055565071	0.03421951	-0.021345561
0.060645592	0.000033671	-0.060611921

Table 5.12: Distance calculation for incoming request req3 with vector $\{V1_{req3} = 0.089999914, V2_{req3} = 0.190000057\}$

dist(RogueToCentroid)	dist(FarthestToCentroid)	D
0.042346528	0.00428419	-0.038062338
0.038269773	0.034286644	-0.003983129
0.040659866	0.004272975	-0.036386891
0.010182996	0.03421951	0.024036514
0.04388917	0.000033671	-0.043855499

5.4 Evaluation of Adaptive Module

The decision making process of adaptive module depends on the data it receives from the cognitive module. The λ_i, and $\lambda_{(i-1)}$ value initiate a specific action by the adaptive module. Since the value of λ depends on authenticity of incoming traffic and functionality of the cognitive module, the state of the adaptive module is stochastic in nature and, therefore, cannot be predicted. We can analyze the process with probability distribution.

5.4.1 Automated Analysis of Probabilistic Model

If we consider each state has equal probability due to unpredictibility of legitimate and malicious traffic, then we can graphically represent (Fig. 5.9) the states using the transition diagram, which is similar to toss of a fair coin [70, 20].

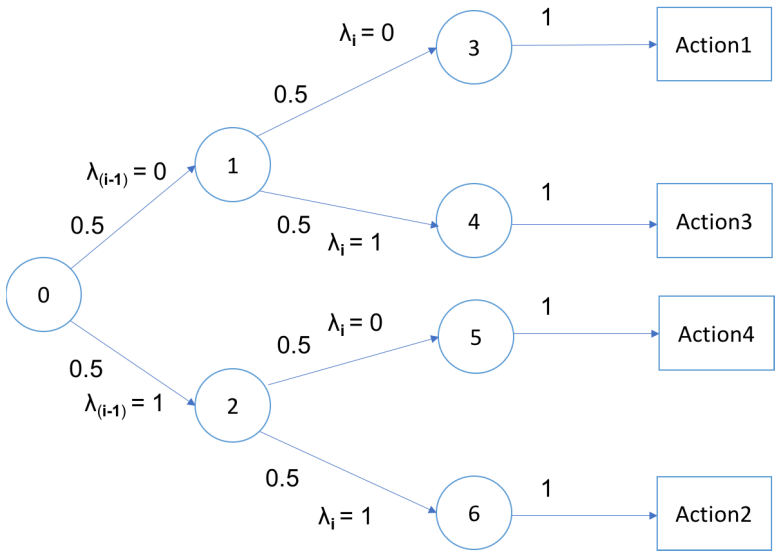

Figure 5.9: Graphical representation of Adaptive module algorithm

We evaluate the design of adaptive module using automated analysis of PRISM [72] probabilistic model checker. Fig. 5.9, clearly shows that the transition of states depends on the probability values, which are 0.5. With the different states of the adaptive module, we build a Markov chain of possible seven states, which form the state space $S = \{s_0, s_1, s_2, ..., s_7\}$ [89, 108]. s_0 represents the initial state
s_1 represents a state when $\lambda_{(i-1)}$ = accepted

s_2 represents a state when $\lambda_{(i-1)}$ = rejected

s_3 represents a state when $\lambda_{(i-1)} = \lambda_i$ = accepted

s_4 represents a state when $\lambda_{(i-1)}$ = accepted, λ_i = rejected

s_5 represents a state when $\lambda_{(i-1)}$ = rejected, λ_i = accepted

s_6 represents a state when $\lambda_{(i-1)} = \lambda_i$ = rejected

s_7 represents the action state

With the probability values displayed in (Fig. 5.9), we can build the transition matrix (T), which is also termed as the probability matrix. In Markov chain, any state can change to the next state with probability T. Therefore, a state change from s_i to $s_{(i+1)}$ is obtained by (Eq. 5.7),

$$s_i T = s_{(i+1)} \tag{5.7}$$

We simulate this model as DTMC (Discrete-Time Markov Chain) probabilistic model, which is useful in our research as the transition to a next state depends only on the current state and not on the entire system [60]. Algo. 2 shows the DTMC model implementation of adaptive module using *PRISM* language.

Using the model implementation we find the probability P of "VM shutdown and threat alert issued", adding the property $P =?[Fs = 7\&a = 2]$, where a denotes the "next state". We obtain $P = 0.228$ based on 1000 iterations. For different properties, in Listing 7.1, we present the summary of our simulation results extracted from $PRISM$. Fig. 5.10 shows sample simulation graph path of length 100 steps.

Probabilistic model design of this module can provide a probability distribution, which could be used in the implementation and operation of the adaptive module for accessing any probable threat in future.

Algorithm 2 Adaptive Module design as DTMC

```
dtmc//Discrete-time Markov chain model

module adaptive

        //lamda previous values
        lp: [0..1] init 0;
        // 0 = accepted
        // 1 = rejected

        //lambda curent values
        lc : [0..1] init 0;
        // 0 = accepted
        // 1 = rejected

        //local state. State space S
        s : [0..7] init 0;

        // action of adaptive module
        a : [0..4] init 0;
        // 1 = no change in VM; Run as usual
        // 2 = shutdown VM requested; Threat Alert issued
        // 3 = initiate a new VM in a different inner cloud and restricts access
        // 4 = initiate a new VM in the same inner cloud

        //state transition based on probability matrix P
        [] s=0 -> 0.5 : (s'=1) & (lp'=0) + 0.5 : (s'=2) & (lp'=1);
        [] s=1 -> 0.5 : (s'=3) & (lc'=0) + 0.5 : (s'=4) & (lc'=1);
        [] s=2 -> 0.5 : (s'=5) & (lc'=0) + 0.5 : (s'=6) & (lc'=1);
        [] s=3 -> 1 : (s'=7) & (a'=1);
        [] s=4 -> 1 : (s'=7) & (a'=3);
        [] s=5 -> 1 : (s'=7) & (a'=4);
        [] s=6 -> 1 : (s'=7) & (a'=2);
        [] s=7 -> (s'=7);

endmodule
```

```
Simulating: P=? [ F s=7&a=2 ]

Simulation method: CI (Confidence Interval)
Simulation method parameters: width=unknown, confidence=0.01, number of samples=1000
Simulation parameters: max path length=10000

Sampling progress: [ 100% ]

Sampling complete: 1000 iterations in 0.082 seconds (average 8.2e-05)
Path length statistics: average 3.8, min 3, max 4

Simulation method parameters: width=0.03419089026484178, confidence=0.01, number of samples=1000

Simulation result details: confidence interval is 0.228 +/- 0.03419089026484178,
based on 99.0% confidence level

Result: 0.228

---------------------------------------------------------------------

Simulating: P=? [ F s=7&a=1 ]

Simulation method: CI (Confidence Interval)
Simulation method parameters: width=unknown, confidence=0.01, number of samples=1000
Simulation parameters: max path length=10000

Sampling progress: [ 100% ]

Sampling complete: 1000 iterations in 0.034 seconds (average 3.4e-05)
Path length statistics: average 3.8, min 3, max 4

Simulation method parameters: width=0.03519403780188987, confidence=0.01, number of samples=1000

Simulation result details: confidence interval is 0.248 +/- 0.03519403780188987,
based on 99.0% confidence level

Result: 0.248

---------------------------------------------------------------------

Simulating: P=? [ F s=7&a=x ]
Property constants: x=4

Simulation method: CI (Confidence Interval)
Simulation method parameters: width=unknown, confidence=0.01, number of samples=1000
Simulation parameters: max path length=10000

Sampling progress: [ 100% ]

Sampling complete: 1000 iterations in 0.02 seconds (average 2e-05)
Path length statistics: average 3.7, min 3, max 4

Simulation method parameters: width=0.035746775210841825, confidence=0.01, number of samples=1000

Simulation result details: confidence interval is 0.26 +/- 0.035746775210841825,
based on 99.0% confidence level

Result: 0.26
```

Listing 5.1: Simulation Results from PRISM

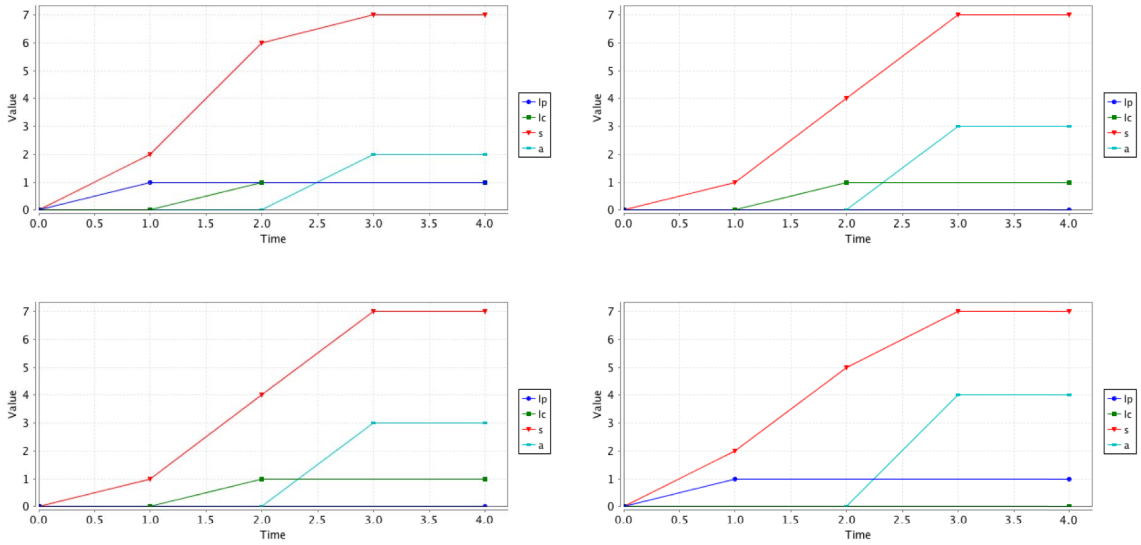

Figure 5.10: Four sample PRISM simulation of Adaptive module designed as DTMC

5.5 Simulation and Formal Security Analysis of Authentication Module

We evaluate the three proposed authentication schemes by formal security analysis. By computing the vulnerability score, V_{score}(Eq. 5.8), and by performing security analysis of the proposed schemes, we determine how secure the schemes are when executing in a single client and cloud communication [37, 40]. Let us consider $V_{potential}$ is the total number of possible vulnerabilities, which are expected to found in the proposed scheme and V_{actual} is the number of actual vulnerabilities recorded. Then, the likelihood of actual vulnerabilities of an authentication scheme defines its vulnerability score (V_{score}) (Eq. 5.8). Lesser vulnerability score indicates that the proposed scheme can prevent more number of attacks. The value of V_{score} lies between 0.0 and 1.0.

$$V_{score} = \frac{V_{actual}}{V_{potential}} \tag{5.8}$$

We configured the protocol analyzer *Scyther*[34] for our validation. We have chosen all types of attacks from the setting option in *Scyther* and in addition we analyzed our schemes for 10 RUNs, In every run the tool tried to use different types of attacks and hackers initial knowledge to compromise the system. The proposed

scheme is validated mainly against replay attacks, and MITM attacks. In addition, one of our earlier proposed scheme MDA is validated for MITM, and reply attacks emulating the operations using Java program.

5.5.1 Emulation and Security Analysis of MDA

MDA Emulation

We have emulated the operations involved in MDA using Java program and validated it for man-in-the-middle attacks, and replay attacks [40]. The proposed scheme MDA uses public key encryption in the registration process, symmetric encryption in authentication process and a standard hashing algorithm in generating the message digests. MDA does not have any specification for algorithms and therefore, any standard combinations of encryption algorithms and hashing algorithms could be used in the operations of MDA. In order to emulate the operations of MDA, we have used RSA (Rivest, Shamir, and Adelman) as the public key encryption algorithm, SHA1 (Secure Hash Algorithm 1) as the hashing algorithm, and AES (Advanced Encryption Standard) as the symmetric key encryption algorithm. Instances of two user defined java classes (CloudServer and MobileClient) are used to run the operations of MDA.

In addition, we configured the emulated MDA setup with two man-in-the-middle objects. These objects are responsible for modifying the authentication request frames sent from a mobile client to the cloud server, and authentication response frames sent from the cloud server to a mobile device. Upon creation of the mobile device object, we perform the registration process by sending the registration request to the cloud server. The cloud server object uses SHA1 algorithm to generate the client and cloud message digests. RSA Keypair generation is performed at the cloud server and the Pk_{pub_cloud} is sent along with the MD_{user} and MD_{cloud}. Both entities, the mobile device and the cloud server use AES encryption with key T_k during the authentication process.

By invoking $run()$ method of authentication thread, 1000 authentication request frames are sent from the registered mobile device to the cloud server as shown in Fig. 5.11. The man-in-the-middle attack object is activated randomly, and it modified the request frames that are sent. Among 1000 frames, 256 frames are modified in transit. The cloud server running MDA scheme has rejected all the modified request

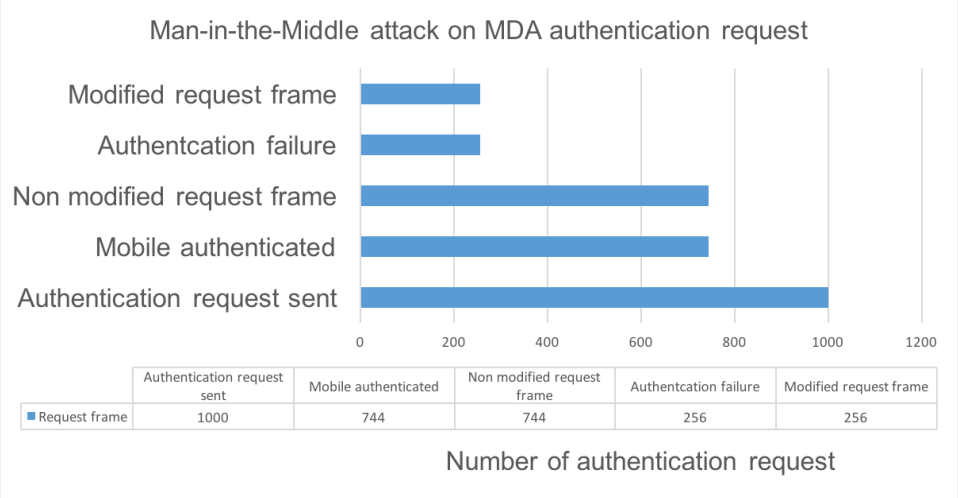

Figure 5.11: Man-in-the-Middle attack on MDA authentication request [40]

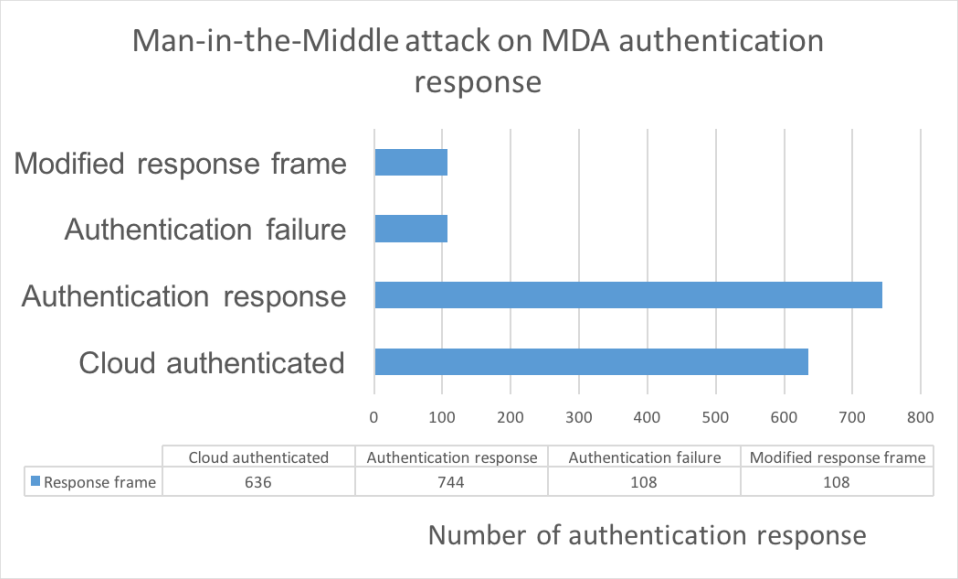

Figure 5.12: Man-in-the-Middle attack on MDA authentication response [40]

frames. Only 744 non modified request frames are processed by the cloud server and 744 authentication response frames are generated and sent back to the mobile device as shown in Fig. 5.12. MDA is a mutual authentication scheme, therefore, the authentication response frames are authenticated by the mobile device. Further, the second man-in-the-middle object has launched random attacks on the response frames. Among 744 response frames, 108 are modified, which are later rejected by the mobile device. Only 636 non modified response frames are accepted.

With MDA emulation, we are able to validate the operations of the MDA scheme and exhibit that MDA can withstand man-in-the-middle attacks. However, our emulation has failed to establish that MDA can prevent replay attacks on request or response frames. The scheme does not have any timestamp component, which makes it difficult to identify if a valid request frame is sent from an adversary at later stage.

Security Analysis

To perform the security analysis of the authentication phase, we focus on evaluating the vulnerability of certain parameters such as, $hash\{userID\}$, $hash\{password\}$, MD_{user}, MD_{cloud}, which are used in our authentication scheme. If the V_{score} value is high, i.e. these parameters are compromised at any level of communication between the mobile device and the cloud server during the authentication process, then we will fail to establish that this scheme is secure. In addition, we need to check the aliveness and synchronization of the mobile device and the cloud server during the authentication. We use the security analyzer *Scyther* to check the vulnerability of each of these parameters [37, 40].

Tab. 5.13 summarizes our claims and how secure the scheme is for each claim. In this section, we provide detailed security analysis of the claims that are validated using *Scyther*.

Claim 1: *$Auth\_Key_i$ remains secret throughout the authentication process.*
$Auth\_Key_i$ is used to encrypt and decrypt the message digest MD to provide multi layer security . A Mobile device sends the message digest MD to the cloud server by encrypting with $Auth\_Key_i$, which is a symmetric key. Both parties, the mobile device and the cloud server can independently generate the $Auth\_Key_i$ using state identifier SI. Since $Auth\_Key_i$ is not exchanged between both the parties, it remains secret.

Claim 2: *State identifier SI is kept secret.*
SI is the state identifier for PRNG used to specify the n^{th} sequence of PRNG as the desired pattern. The mobile device sends SI to the cloud server for specifying the n^{th} sequence of PRNG to generate the $Auth\_Key_i$. SI is sent from mobile device to cloud server after encrypting with key T_k. The claim at the mobile end that SI being secret is validated using *Scyther*. On the other hand, the cloud server does not share

Table 5.13: Security Analysis of the proposed scheme using Scyther during Authentication phase

Role	Sl.Num.	Claim	Status	Comment
Mobile	1	secret $Auth\_Key_i$	Ok	No Attacks
	2	secret SI	Ok	No Attacks
	3	secret $password$	Ok	No Attacks
	4	secret $userID$	Ok	No Attacks
	5	secret MD_{user}	Ok	No Attacks
	6	secret MD_{cloud}	Ok	No Attacks
	7	Alive	Ok	No Attacks
	8	Weakagree	Ok	No Attacks
	9	Niagree	Ok	No Attacks
	10	Nisynch	Ok	No Attacks
Cloud	1	secret $Auth\_Key_i$	Ok	No Attacks
	2	secret $password$	Ok	No Attacks
	3	secret $userID$	Ok	No Attacks
	4	secret SI	Ok	No Attacks
	5	Alive	Ok	No Attacks
	6	Weakagree	Ok	No Attacks
	7	Niagree	Ok	No Attacks
	8	Nisynch	Ok	No Attacks

or send SI, therefore, it is safe at the cloud end.

Claim 3: *password is secret.*

The safety of the *password* depends on the user. Typically no one shares their password, thereby keeping it safe. The password can be a 512 bit string, which is hashed and a copy of the user's *password* is stored at the cloud server during registration process. The mobile device does not send the *password* during authentication process, but it is used at both ends to generate the key T_k. Our *password* safety claim is validated by *Scyther*, but, in reality, it is dependent on the user.

Claim 4: *userID is safe.*

$userID$ is sent to the cloud server only during the registration process, which is required along with the *password* to generate T_k. The $userID$ is hashed and a copy is stored at the cloud server during registration process. The mobile device does not send $userID$ to the cloud server during authentication process. Therefore, $userID$ remains safe. Our $userID$ safety claim is validated by *Scyther*, but, in reality, it is dependent on the user.

Claim 5: *The scheme requires MD_{user} to be a secret.*

MD_{user} is the hashed information related to the user, which may contain policy information and unique information about the user. This information is generated at the cloud server end and sent to the mobile device after a successful registration process. The information is hashed and encrypted with $Auth\_key_i$ before transmitting from the user mobile device to the cloud end. *Scyther* validated our claim that MD_{user} is safe.

Claim 6: *The scheme requires MD_{cloud} is secret.*

MD_{cloud} is the hashed information related to the cloud server, which may contain the cloud server certificate or other policy related information. As with MD_{user}, this information is hashed and encrypted with $Auth\_key_i$ before transmitting from mobile device to the cloud server. As part of the authentication process, the combination of MD_{user} and MD_{cloud} is used at the cloud end to verify whether the mobile device is legitimate. In addition, the combination of MD_{user} and MD_{cloud} encrypted with Pk_{priv_cloud} is used as the cloud server digital signature to authenticate the cloud server by the mobile device. The protocol analyzer (*Scyther*) validated the claim that MD_{cloud} is safe.

Claim 7: *Mobile device and the cloud server remains alive during the execution of the protocol.*

The cloud server is said to be *alive* if it has been using the proposed scheme for the initial $(i-1)$ messages exchanged with the mobile device, when the latter sends the i^{th} message. The protocol analyzer (*Scyther*) validates the *aliveness* claim.

Claim 8: *Guarantees $Weakagree$* [78].

Our proposed scheme guarantees that the mobile device is in *weakagreement* with the cloud server. Therefore, the mobile device is running the proposed scheme with the cloud server; likewise the cloud server is running the scheme with the mobile device. The communication is not affected by adversary during the operation of the proposed scheme. This claim is validated using the protocol analyzer (*Scyther*).

Claim 9: *Assures $Niagree$ between the mobile device and the cloud server* [78].

$Niagree$ or Non-injective agreement ensures that the mobile device and the cloud server agree upon some data exchange during running of the proposed scheme. During the operation of the proposed scheme, the mobile device can send data to the cloud

server safely (and vice-versa). If *Niagree* fails, then we can conclude that there is a man-in-the-middle attack during the operation of the protocol. However, our claim is validated using protocol analyzer (*Scyther*).

Claim 10: *Holds Nisynch during the authentication process* [56].
Nisynch or non-injective synchronization is valid if all actions before the claim are performed as per the proposed scheme description. *Nisynch* ensures that there is no synchronization problem between the mobile device and the cloud server during the authentication process. *Nisynch* is validated using the protocol analyzer, *Scyther*.

Our security analysis results are based on *Scyther* validation, which indicates that MDA supports all of the claims that we have made during the execution of the scheme for multiple runs. The claims are not compromised by any types of attacks during the authentication process. Therefore, the vulnerability score (V_{score}) obtained is 0.

Hashing is a mechanism that makes the proposed scheme secure, and implicitly provides a means for integrity check. Even if any adversary sniffs the message that is sent during the authentication process, it is not possible for the adversary to interpret the message digests, MD_{cloud} or MD_{user} . Furthermore, it is worth noting that MDA is completely based on a technique employing simple $userID$, $password$, and hashing. It does not include any of the system specific properties such as IMSI of mobile phone or MAC (media access control) address of a laptop, which makes the scheme applicable to any type of mobile device without any alteration.

5.5.2 Security Analysis of MDLA

We determine the vulnerability of parameters such as client id, *cid*, password, *pwd*, client certificate digest, md_{client}, timestamp, δt, hash function, δh, choice of PRNG, δp, mobile device location, ϕ, and keys - $prim_{key}$, sec_{key}, $auth_{key}$, using protocol analyzer *Scyther*. If most of the parameters fail the vulnerability test then the value of actual vulnerability V_{actual} will go high, and that will result in a high vulnerability score V_{score}. Tab. 5.14 provides the security status of the claims that we made during the registration phase, and Tab. 5.15 provides security status of claims we made during authentication phase [38, 41].

Claim 1: *Our claim that client id (cid) is secret is true.*
The client id is secret as it is personal for each mobile client and is never transmitted

Table 5.14: Scyther security analysis: Registration phase

Sl.Num.	Claim	Status	Comment
1	secret $prim_{key}$	Ok	No Attacks
2	secret sec_{key}	Ok	No Attacks
3	secret cid	Ok	No Attacks
4	secret pwd	Ok	No Attacks
5	secret δt	Ok	No Attacks
6	secret δh	Ok	No Attacks
7	secret δp	Ok	No Attacks
8	secret ϕ	Ok	No Attacks
9	secret md_{client}	Ok	No Attacks
10	secret ϵ	Ok	No Attacks
11	Alive	Ok	No Attacks
12	Weakagree	Ok	No Attacks
13	Niagree	Ok	No Attacks
14	Nisynch	Ok	No Attacks

in plain text between the parties. The mobile client sends the *cid* during registration phase only after hashing and encrypted with $prim_{key}$. *Scyther* validates the *cid* is secret.

Claim 2: *Our claim that password (pwd) is secret is true.*
The password is secret as it is sent only during registration after hashing. $h\{pwd\}$ is sent from the mobile device to the cloud server after encrypting using $prim_{key}$. *Scyther* validates that *pwd* is secret.

Claim 3: *The primary key $prim_{key}$ is claimed to be secret and validated.*
$prim_{key}$ is a pseudo random number sequence, which is generated using hashed client id $h\{cid\}$ and hashed password $h\{pwd\}$. Both the parameters are XOR-ed and used as seed to the PRNG. The previous location of the mobile device ϕ_{prev} is used as the state identifier. All these information are safe and in addition, variable location of the mobile device and periodic change of password ensure variability in the value of $prim_{key}$.

Claim 4: *The authentication key $auth_{key_i}$ is calimed to be secret and validated.*
$auth_{key_i}$ or the authentication key is generated both ends by using the time-stamp δt_{new} of the mobile device as the seed to the PRNG and mobile's current location vector ϕ_{new} as the state identifier for identifying a PRNG sequence as the key. Both

Table 5.15: Scyther security analysis: Authentication phase

Sl.Num.	Claim	Status	Comment
1	secret $prim_{key}$	Ok	No Attacks
2	secret sec_{key}	Ok	No Attacks
3	secret $sec_{key_{new}}$	Ok	No Attacks
4	secret $auth_{key}$	Ok	No Attacks
5	secret δt_{new}	Ok	No Attacks
6	secret ϕ_{new}	Ok	No Attacks
7	secret md_{client}	Ok	No Attacks
8	Alive	Ok	No Attacks
9	Weakagree	Ok	No Attacks
10	Niagree	Ok	No Attacks
11	Nisynch	Ok	No Attacks

the time-tamp and the location vector are variables, which ensure $auth_{key_i}$ is unpredictable.

Claim 5: *Secret key sec_{key} is secret is the claim we made and is validated.*
The secret key sec_{key} is chosen randomly by the cloud server and sent to the mobile device encrypted during registration process. Every authentication response from the cloud server to the mobile device includes a fresh sec_{key}, which ensures variability and unpredictability of the key.

Claim 6: *The message digest md_{client} is secret as it is hashed message. We validated the message digest.*
The md_{client} is generated by hashing the client's certificate. The client certificate is generated at the cloud server and hashed before sending it to the mobile client. md_{client} is secret since it is a hashed message and the transmission to the mobile client is encrypted by $prim_{key}$ during registration, by $auth_{key}$ during authentication request, and by sec_{key} during authentication response.

Claim 7: *The location vector, ϕ is claimed to be secret and is validated.*
The location vector of the mobile device is the logical OR of latitude and longitude, which is claimed to be safe as it is fetched from the mobile device to the cloud server by encrypting with $prim_{key}$. Even if one transmitted message is captured and decrypted by an adversary, the probability of having a varied location vector in the next transmitted message is high due to the mobility of the client, which maintains the level of security of the system.

Claim 8: *The time-stamp δt is claimed to be secret.*
Our proposed scheme uses fresh timestamp as one of the key parameter in the generation of keys. The time-stamp is sent encrypted using $prim_{key}$ during registration and authentication phase. We have validated that δt is secure using *Scyther*.

Claim 9: *PRNG choice parameter, δp is claimed to be secure and it is validated.*
The PRNG choice parameter, δp, used by both the parties is transferred from mobile device to the cloud server encrypting with $prim_{key}$. This helps in synchronizing the PRNG in both sides. We validated that our claim δp is secret.

Claim 10: *Hash function choice parameter, δh used is claimed to be secure and validated.*
The hash function used by both the parties are defined using parameter δh. This parameter is exchanged during registration phase and re-registration phase between parties encrypted using $prim_{key}$, and both the parties use it for the authentication process. We validated δh and found our claim is true.

Claim 11: *The authentication scheme ensures both parties are Alive during the operation*
The mobile device and the cloud server are said to be *alive* when both the parties use the proposed scheme for initial $(n-1)$ messages exchange and both parties send each other the next n^{th} message. The *aliveness* is validated using *Scyther*.

Claim 12: *Our scheme ensures non-injective agreement $Niagree$ between the mobile client and the cloud server* [78].
Non-injective agreement ensures that the mobile client and the cloud server are in agreement to exchange data with each other during the operation of the proposed scheme. Both parties can safely communicate with each other for the data exchange and this is validated using protocol analyzer (*Scyther*).

Claim 13: *The proposed scheme holds $Nisynch$ during the communication and is validated* [56].
$Nisynch$ or non-injective synchronization is valid if all the operations between the two parties are performed as proposed. We considered there is no synchronization problem in the communication and the $Nisynch$ claim is validated using the protocol analyzer, *Scyther*.

5.5.3 Security Analysis of AMLT

In our research, we performed the formal analysis of AMLT using protocol analyzer *Scyther*[35]. Specifically, we studied the vulnerability of parameters, such as client certificate digest, md_{client}, client timestamp, δtc, server timestamp, δts, hash function choice parameter, δh, choice of PRNG, δp, location vector, δl, client record pointer at the server database, $c - rp$, and keys - $k_{sym-S2C}$, $k_{sym-C2S}$.

Our objective is to explore if the proposed scheme passes the vulnerability test. Table 5.16 provides the security status of the claims that we made during the Initial Handshake phase, and Table 5.17 provides security status of claims we made during the Mutual Authentication phase.

Table 5.16: Scyther security analysis: Initial Handshake

Sl.Num.	Claim	Status	Comment
R1	secret md_{client}	Ok	No Attacks
R2	secret δtc	Ok	No Attacks
R3	secret δl	Ok	No Attacks
R4	secret δts	Ok	No Attacks
R5	secret δp	Ok	No Attacks
R6	secret δh	Ok	No Attacks
R7	secret δe	Ok	No Attacks
R8	secret $c - rp$	Ok	No Attacks
R9	secret $k_{sym-S2C}$	Ok	No Attacks
R10	Alive	Ok	No Attacks
R11	Niagree	Ok	No Attacks
R12	Nisynch	Ok	No Attacks

Table 5.17: Scyther security analysis: Mutual Authentication

Sl.Num.	Claim	Status	Comment
A1	secret md_{client}	Ok	No Attacks
A2	secret $c - rp$	Ok	No Attacks
A3	secret δts	Ok	No Attacks
A4	secret δtc	Ok	No Attacks
A5	secret δl	Ok	No Attacks
A6	secret $k_{sym-C2S}$	Ok	No Attacks
A7	secret $k_{sym-S2C}$	Ok	No Attacks
A8	Alive	Ok	No Attacks
A9	Niagree	Ok	No Attacks
A10	Nisynch	Ok	No Attacks

Claim R1 & A1: *Our claim that client message digest (md_{client}) is secret is true.* The md_{client} is generated by applying one way hash function on concatenated value of client's ID and password, $\{c - id|c - pwd\}$ This message digest is generated at the client side and used as client's certificate. md_{client} is sent encrypted either using cloud's public key or the two secret keys all the times during the communication process. *Scyther* validates the md_{client} is secret.

Claim R2 & A4: *Our claim that client's timestamp (δtc) is secret is true.* Our proposed scheme uses fresh client's timestamp as one of the key parameters in the generation of symmetric key, $k_{sym-S2C}$. The timestamp is sent encrypted using $k_{pub-server}$ during registration request, and using $k_{sym-C2S}$ during authentication request phase. We have validated that δtc is secure using *Scyther*.

Claim R3 & A5: *The location vector, δl is claimed to be secret and is validated.* The location vector of the mobile client is the logical OR of latitude and longitude, which is claimed to be safe as it is fetched from the mobile client to the cloud server by encrypting with $k_{pub-server}$ during registration request, and using $k_{sym-C2S}$ during authentication request phase. Even if one transmitted message is captured and decrypted by an adversary, the probability of having a varied location vector in the next transmitted message is high due to the mobility of the client, which maintains the level of security of the system.

Claim R4 & A3: *Our claim that server's timestamp (δts) is secret is true.* Our proposed scheme uses fresh server's timestamp as the key parameter in the generation of symmetric key, $k_{sym-C2S}$. The timestamp is sent encrypted using symmetric key, $k_{sym-S2C}$ during registration response, and the authentication response. We have validated that δts is secure using *Scyther*.

Claim R5: *PRNG choice parameter, δp is claimed to be secure and it is validated.* The PRNG choice parameter, δp, used by both the parties is transferred from mobile client to the cloud server during registration request, encrypting with $k_{pub-server}$. This helps in synchronizing the PRNG in both sides. We validated that our claim δp is secret.

Claim R6: *Hash function choice parameter, δh used is claimed to be secure and validated.*

The hash function used by both the parties are defined using parameter δh. This parameter is exchanged during registration request process encrypted using $k_{pub-server}$, and both the parties use it for the Initial Handshake and Mutual Authentication processes. We validated δh and found our claim is true.

Claim R7: *Symmetric algorithm choice parameter, δe used is claimed to be secure and validated.*

The symmetric key encryption algorithm selection parameter used by both the parties are defined using δe. This parameter is exchanged during registration request process encrypted using $k_{pub-server}$, and both the parties use it for the Initial Handshake and Mutual Authentication processes. We validated δe and found our claim is true.

Claim R8 & A2: *Client record pointer, $c - rp$ used is claimed to be secure and validated.*

The client record pointer in the server database is denoted using $c - rp$. This value is encrypted by the cloud server using $k_{pub-server}$, and sent to the mobile client during the registration response. The cloud client sends the pre-encrypted record pointer $c - rp_{scd}$ to the cloud server during authentication process. We validated $c - rp$ during Initial Handshake and Mutual Authentication and found our claim is true.

Claim R9 & A7: *The server symmetric key $k_{sym-S2C}$ is calimed to be secret and validated.*

$k_{sym-S2C}$ or the server symmetric key is generated by both ends using hashed client's time-stamp $H_{\delta h}(\delta tc)$ as the seed to the PRNG. In this key generation, a bit flipping operation BF is performed on mobile client's location vector δl, and is used as the state identifier for identifying a PRNG sequence as the key. Both the timestamp and the location vector are variables, which ensure generated $k_{sym-S2C}$ is unpredictable. *Scyther* validation shows our claim is true.

Claim A6: *The client symmetric key $k_{sym-C2S}$ is calimed to be secret and validated.*

$k_{sym-C2S}$ or the client symmetric key is generated by both ends using XORing of md_{client}, and hashed server's timestamp $H_{\delta h}(\delta ts)$, which is used as the seed to the PRNG. In this key generation, a bit flipping operation BF is performed on cloud server's timestamp δts, and is used as the state identifier for identifying a PRNG sequence as the key. Both the time-tamp and the message digest are secure, which

ensure generated $k_{sym-C2S}$ is secure. *Scyther* validation shows our claim is true.

Claim R10 & A8: *The Initial Handshake and Mutual Authentication processes ensure both parties are Alive during the operation*

The mobile client and the cloud server are said to be *alive* when both the parties use the proposed scheme for initial $(n-1)$ messages exchange and both parties send each other the next n^{th} message. The *aliveness* is validated using *Scyther*.

Claim R11 & A9: *Our scheme ensures non-injective agreement Niagree between the mobile client and the cloud server* [78].

Non-injective agreement ensures that the mobile client and the cloud server are in agreement to exchange data with each other during the operation of the proposed scheme. Both parties can safely communicate with each other for the data exchange and this is validated using protocol analyzer (*Scyther*).

Claim R12 & A10: *The proposed scheme holds Nisynch during the communication and is validated* [56].

Nisynch or non-injective synchronization is valid if all the operations between the two parties are performed as proposed. We considered there is no synchronization problem in the communication and the *Nisynch* claim is validated using the protocol analyzer, *Scyther*.

5.6 Chapter Summary

- The evaluation methodology ensures reliability and security of the framework.
- The framework is evaluated using performance analysis to ensure the reliability and accuracy of the functionality of the framework.
- Two experiments are considered as part of the evaluation methodologies:
  - Experiment 1 is configured with a fixed input traffic set, which is processed multiple times by the context-aware framework and the observations are recorded.
  - Experiment 2 is configured with different input traffic sets, which are processed by the context-aware framework for a given period and the observations are recorded.

- $error - percentage$ is calculated for determining accuracy of the framework.
- Precision is calculated to determine the reliability of the proposed framework.
- Cognitive Module is evaluated based on cluster analysis. Proposed KD algorithm is used.
- For training the system, K-Means partitioned based clustering is used to create clusters of n data points. Elbow method is used to determine optimal value of k.
- A pair of IPDs (previous IPD, and current IPD) are considered as vector, which is used as input to K-Means
- DBSCAN density based clustering is used to remove outliers from clusters with optimal eps, and $minPts$ values. Clean clusters centroid is obtained.
- Distance based traffic filtration is performed, where farthest points of each cluster are measured with the new data point.
- Adaptive module is designed as a probability model, where the states are unpredictable due to randomness in the data. The module is implemented as DTMC.
- Adaptive module is implemented and validated using probabilistic symbolic model checker $PRISM$.
- Authentication module is validated using emulation and formal security analysis.
- Scyther is used for formal security analysis, where the authentication schemes are tested for vulnerabilities. V_{score} is computed based on number of actual vulnerabilities (V_{actual}) found in the schemes.

Chapter 6

Discussion

6.1 Overview

The overall urban population across the world is increasing every year, which imposes a threat to the natural resources. In SmartAmerica Expo 2014, the smart cities USA team [52] indicates, by the year 2025, there will be an additional demand of 80 billion metric tons of municipal water. In addition, the consumption of power supply, air etc. will impact the quality of urban life. To overcome this situation and to improve the quality of life, sustainable usage of resources need to be considered.

Cities across the world are transformed into smart cities with the help of ICT to provide customized on-demand critical services, and prevent wastage of critical resources, such as water supply, power etc. Mobile cloud computing is a core technology, which can be used to transform a city into a smart city. Since a smart city deals with critical information infrastructure, such as health care, traffic management, water supply, power distribution, pollution control etc. it is important to address issues and challenges associated with a smart city development and operation.

Mobile cloud computing environment of a smart city takes advantage of widely-deployed mobile devices and high computation power of cloud servers. The technological advancement of mobile cloud computing is rapid and it covers a wide range of application domains that include RFID, VANET, WBAN etc. Although these application domains are well defined and protected in their own way, the faster growth in technology and computation power makes most of the systems vulnerable to attacks. Furthermore, the smart city applications suffer from many restrictions in terms of scaling up resources and computational power, which leaves little room for improvement. A mobile cloud computing environment deployed in a smart city framework uses heterogeneous arrangements, such as smart health care applications use WBAN, smart traffic management use road sensors, ip cameras, and smart home applications use WSN, RFID etc. In many instances these critical applications transfer sensitive

information, which require protection from unauthorized access. Similar to privacy, the security and reliability of smart city applications need to be addressed to avoid disruption of critical services. Therefore, in this thesis, we propose a context-aware framework and a cloud of cloud model to secure mobile cloud computing environment, which can protect the critical services of a smart city. The context-aware framework adds an additional layer of security in the smart city operation by reducing the chances of service disruption caused by anomalous traffic.

6.2 Key Features of Context-Aware Framework

The thesis proposes a context-aware security framework, which could be supported by multiple cloud vendors to ensure interoperability, layered security, re-configurability to support cloud vendors' infrastructures, and clients' security needs. Motivated by Artificial Intelligence, and self-healing systems, we propose a security framework named "reconfigurable context-aware security framework for mobile cloud computing", to use in a cloud of cloud model [39]. The three modules of the framework ensure security of sensitive information, and maximum uptime of the cloud server.

The cognitive module, which is inspired by learning system and artificial intelligence ensures filtration of incoming traffic without the need of human intervention, writing access control lists, ports and IP monitoring etc. Cluster based approach (KD algorithm) is proposed for training the system, and pattern based incoming traffic filtration. Instead of TTL, ports, IP addresses and payload size, we use vectors consisting previous and current inter packet delays as the training set [24]. This helps to identify if there is a change in pattern of inter packet delays of a client requests, and a possible DDoS attack could be prevented.

The adaptive module, which is inspired by self-healing service delivery models ensures maximum uptime of cloud services by preventing cloud resources from possible MITM and DDoS attacks. Cross VM attacks and malware injection could cause data leakage from a VM [100, 61]. Therefore, we propose a VM switching, inner cloud or hypervisor switching using the adaptive module. We conceptualize the system as a DTMC (Discrete Time Markov Chain) process, and evaluated its functionality by exploring the probability of VM shutdown during a possible attack.

The authentication module is designed to use device independent features for

mututal authentication, which is different than device specific and one way authentication [120, 5, 98, 125]. The research proposes three novel authentication schemes, which can perform mutual authentication of mobile client, and cloud server using message digest or hashed information, location vector, and timestamp [37, 38, 40]. The authentication schemes use symmetric key encryption during authentication process, which consume less mobile resources. In addition, message digest based authentication eliminates the need of entering user ID and passwords, or any authentication pattern. We evaluate the authentication schemes by determining the vulnurability score (V_{score}) using protocol analyzer *Scyther*, which indicates the proposed authentication schemes can withstand various potential attacks.

The modules of the context-aware framework are loosly coupled, which makes the system reconfigurable, and scalable. The framework allows the cloud vendors to modify the three modules, such as, adding a new traffic filtration layer in the cognitive module, updating encryption and hashing algorithms for the proposed authentication schemes, or replacing a proposed authentication scheme with a new authentication. In addition, the VM switching logic of adaptive module can be modified depending on the security and service requirements of the cloud provider. Furthermore, the reconfigurability feature of the context-aware framework allows the cloud vendors to retain their existing security infrastructure.

6.3 Benefits and Challenges

The proposed context-aware framework design provides four major benefits. Firstly, the modules of the framework are loosly coupled, and therefore, upgrade of a specific module does not affect the functionality of other modules. Secondly, in the KD algorithm a set of inter packet delay vectors $\{ipd_{(i-1)}, ipd_i\}$ are used instead of inter packet delays as data points, which generates clusters of patterns. Network delay, nodes speed, and network congestion can cause variability in packet arrival times, and therefore, usage of inter packet delay vectors in clustering can minimize filtration errors. In addition, KD algorithm removes outliers from training set, which makes the training set clean, and improves the quality of filtration process. Thirdly, dynamic cloud selection, VM initialization, VM shutdown etc. performed by the adaptive module introduce self-healing of the cloud, which ensures maximum service uptime.

Finally, considering message digest, location and timestamp as primary authentication parameters eliminate the need of password or token based authentication.

The primary challenges of the proposed framework are, performance analysis, implementation, and deployment. The thesis presents various techniques to evaluate the modules, and it is assumed that the modules will function flawlessly if implemented and deployed in a real-world scenario. However, the performance analysis presented in this thesis are limited to specific validation techniques, and data sets, which may fail to support current cloud infrstructure, and available mobile client networks, such as RFID, WBAN etc. We have claimed that the proposed context-aware framework is reconfigurable and could be upgraded depending on the security requirements. However, once deployed, the system may not be altered easily. The cognitive module uses cluster-based approach to train the system. The proposed KD algorithm is applied only on FTP dataset for system training, and the cognitive module is tested for possible DDoS and MITM attacks by distance measurement. In real-world scenario, the initial system training with all cloud clients and protocols will consume time and resources, and therefore, it might be considered as overhead. The adaptive module is conceptualized and designed as a probability model, and therefore, the evaluation methodology uses *PRISM* to define the functionality of the system. However, the actual system functionality depends on the implementation of the module. A biased or faulty implementation may cause shutdown of a VM for a legitimate traffic. In addition, for the proposed authentication schemes, formal security analysis are performed using one protocol analyzer, which may not be sufficient to prove the effectiveness of the authentication schemes.

In order to address these challenges, we need to consider various validation techniques, data sets from heterogenous mobile clients, and implementation of the modules that support the proposed design.

6.4 Chapter Summary

- Urban population across the world is increasing every year.
- Cities across the world are transformed into smart cities with the help of ICT to provide customized on-demand critical services, and prevent wastage of critical

resources.

- It is important to address issues and challenges associated with a smart city development and operation.
- Mobile cloud computing can be used to transform a city into a smart city.
- Faster growth in technology and computation power makes most of the systems vulnerable to attacks.
- The thesis proposes a context-aware framework and a cloud of cloud model to secure critical services of a smart city.
- The framework ensures interoperability, layered security, and reconfigurability.
- The security framework has three modules. Cognitive module, adaptive module, and authentication module. These modules are deployed on different layers of the cloud of cloud model to provide a multistage security.
- Cluster based approach (KD algorithm) is proposed in the cognitive module for training the system, and pattern based incoming traffic filtration.
- In the KD algorithm a set of inter packet delay vectors $\{ipd_i, ipd_{(i+1)}\}$ are used instead of inter packet delays as data points, which generates clusters of patterns.
- The adaptive module is conceptualized as a probability model, which performs VM switching, and inner cloud or hyperviosr switching.
- The authentication module highlights three novel authentication schemes, which can perform mutual authentication of mobile client, and cloud server using message digest or hashed information, location vector, and timestamp.
- The proposed framework has benefits, and challenges. Benfits include loosly coupled modules to support reconfigurability, usage of inter packet delay vectors in clustering to improve quality of filtration, self-healing of cloud, and message digest based authentication.

- The major challenges are limited performance analysis, segmentation of only one set of traffic, and absence of real-world implementation.

Chapter 7

Conclusion

7.1 Conclusion

The advent of mobile cloud computing has widened the prospects of modern day computing. Numerous fields, including health and medical sciences, financial institutions, transportation, monitoring stations, etc., can benefit from mobile cloud computing. Many countries are trying to initiate smart city projects in order to provide efficient services of critical information infrastructure, which use mobile cloud computing as the underlying technology. Any sensitive and crucial information, if not well protected can cause data modification, data loss, and disruption of services. As security researchers, we believe it is our responsibility to ensure secure communication between a mobile client (cellphone, laptop, tablets, sensor nodes, RFID etc.) and a cloud server; particularly, in preventing unauthorized accesses and impersonation.

We introduce a reconfigurable *Context-Aware Framework* to ensure the security of authenticated sensitive data at the cloud server. This framework is deployed at the cloud premises, and therefore, no changes required at the client side. The framework provides an additional layer of security, which helps to safeguard the sensitive data and mobile cloud computing resources. It achieves traffic and client interface management, and traffic filtering via cognitive module, which is designed as an access control to the cloud resources. The cognitive module uses KD clustering algorithm to train the system, and squared Euclidean distance to validate the incoming traffic set. The quality of service and maximum uptime of cloud VMs are ensured by the effective inner cloud manangement of adaptive module. Furthermore, this module protects the cloud infrstructure from cross VM attacks. The adaptive module is designed with a probabilistic model and validated using PRISM. Finally, we present three authentication schemes that ensure mutual authentication between communication entities. These three proposed message digest based authentication schemes are evaluated using the protocol analyzer Scyther. Our experimental results indicate

that the proposed framework can withstand various attacks.

7.2 Future Work

The present system provides three phase security using authentication, traffic filtration, and hypervisor selection technique. However, these techniques are premature and requires large dataset generated from various mobile networks to train the cognitive module. The authentication scheme is validated using only formal security analysis, which could be a limitation in terms of effectiveness. Current system uses a hybrid approach in training of cognitive module, and implements two existing standard clustering algorithms. Any shortcomings of these algorithms might affect the performance of the cognitive module. The detailed design and development of adaptive module is pending, a variation of which could be implemented using Docker, a container management system.

The context-aware framework could be enhanced by incorporating intelligent agent, which will ensure maximum uptime of the system. In addition, it will manage the autonomous selection of various hypervisors, and fetch appropriate security algorithms for the authentication module. Furthermore, the knowledge-base can be enhanced by adding additional information, such as protocols, acceptable TTL values for a particular client, and possible payload sizes etc., which can be used as features in traffic filtration process. An additional rule-based techniques may be introduced in future to filter the incoming traffic. This extension can provide cloud vendors the flexibility of utilizing their existing rule-based techniques as an additional layer of security. The rule-based techniques can be deployed as $Phase3$ traffic filtration in the Cognitive module.

7.3 Chapter Summary

- Presents the conclusion and future work.
- Many countries are trying to initiate smart city projects.
- Mobile cloud computing is the underlying technology for smart city applications.
- Sensitive information is susceptible to attacks; therefore, security is crucial.

- Proposed context-aware framework can perform traffic filtration, VM selection, and mutual authentication.
- As future work, large heterogeneous dataset could be considered for validation of the system.
- Testing of authentication module is limited to formal analysis, which could be extended.
- In addition to VMs, adaptive module could be implemented and tested as a container management system.

Appendix A

Verification of Elbow Method

KMeans Algorithm to compute within cluster sum of squares with different k values

Chosen Sample size: 5861 data points

Elbow method detects optimum value for k: 6 (Fig. A.1)

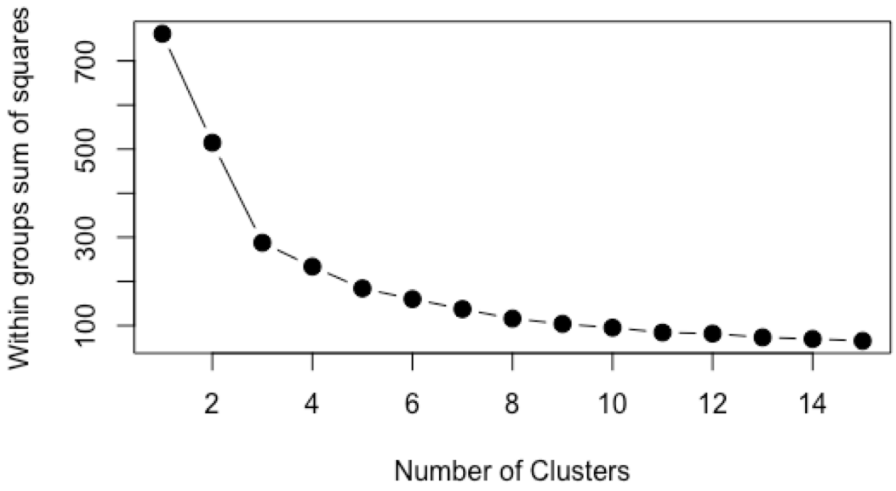

Figure A.1: Elbow method analysis to find the optimum value of k for K-Means

The following experiment verifies the accuracy of Elbow method

when k = 2

results$size [1] 1623 4238

Within cluster sum of squares by cluster: [1] 365.4589 149.0424 (between_SS / total_SS = 32.4 %)

Cluster means:

curr_ipd next_ipd

1 0.5058472 0.5168638

2 0.1891340 0.1848655

when k = 3

results$size [1] 947 3966 948

Within cluster sum of squares by cluster: [1] 88.26257 105.43825 93.99110 (between_SS / total_SS = 62.2 %)

Cluster means:

curr_ipd next_ipd

1 0.7306230 0.2151954

2 0.1789208 0.1818129

3 0.2331646 0.7357278

when k = 4

results$size [1] 3161 419 1397 884

Within cluster sum of squares by cluster: [1] 62.25472 45.46696 40.36569 86.15371 (between_SS / total_SS = 69.2 %)

Cluster means:

curr_ipd next_ipd

1 0.1310566 0.1860930

2 0.9392601 0.2591408

3 0.4372226 0.1857838

4 0.2306787 0.7533597

when k = 5

results$size [1] 1471 155 874 684 2677

Within cluster sum of squares by cluster: [1] 29.11794 22.99558 81.02530 26.31554 47.41268 (between_SS / total_SS = 72.8 %)

Cluster means:

curr_ipd next_ipd

1 0.1761047 0.3335690

2 0.3447742 1.1877419

3 0.7517506 0.2186384

4 0.2167544 0.6742836

5 0.1885543 0.1102914

when k = 6

results$size [1] 167 2339 308 1280 1076 691

Within cluster sum of squares by cluster: [1] 23.53367 25.64493 33.01565 18.21336 30.61057 27.90181 (between_SS / total_SS = 79.1 %)

Cluster means:

curr_ipd next_ipd

1 0.3174850 1.1716766

2 0.1524198 0.1098247

3 1.0161039 0.2592533

4 0.1493906 0.3379531

5 0.5102231 0.1840985

6 0.2313025 0.6646310

when k = 7

results$size [1] 192 1160 152 1201 1829 664 663

Within cluster sum of squares by cluster: [1] 21.83618 16.21345 22.41995 15.59809 12.09907 24.07011 25.16311 (between_SS / total_SS = 82.0 %)

Cluster means:

curr_ipd next_ipd

1 1.1332812 0.2494792

2 0.1453534 0.3542759

3 0.3451974 1.1909868

4 0.3386511 0.1440216

5 0.1103116 0.1142701

6 0.6482681 0.2249398

7 0.2186124 0.6804072

when k = 8

results$size [1] 153 1825 126 553 318 1141 547 1198

Within cluster sum of squares by cluster: [1] 15.00394 12.03922 17.73672 12.15519 16.08051 14.48566 12.75093 15.37804 (between_SS / total_SS = 84.8 %)

Cluster means:

curr_ipd next_ipd

1 1.1864052 0.2069935

2 0.1105973 0.1134740

3 0.3076191 1.2434127

4 0.6657324 0.1594937

5 0.5616038 0.5795283

6 0.1420158 0.3500438

7 0.1546252 0.6942048

8 0.3397830 0.1463105

when k = 9

results$size [1] 553 1068 77 1189 231 678 147 1559 359

Within cluster sum of squares by cluster: [1] 12.179456 8.493654 12.438924 15.601778 11.312801 13.084489 14.284332 8.453707 [9] 9.586613 (between_SS / total_SS = 86.2 %)

Cluster means:

curr_ipd next_ipd

1 0.6662387 0.15924050

2 0.1165637 0.28191948

3 0.3780520 1.36896103

4 0.3425484 0.15000000

5 0.6367532 0.60731602

6 0.2005162 0.49781711

7 1.1967347 0.20006803

8 0.1140346 0.09534958

9 0.1573259 0.81768802

when k = 10

results$size [1] 453 139 1073 181 1032 233 538 550 1607 55

Within cluster sum of squares by cluster: [1] 6.783522 13.459143 10.254965 8.956398 8.712893 7.179676 8.410311 11.889677 [9] 8.682902 9.560872 (between_SS / total_SS = 87.7 %)

Cluster means:

curr_ipd next_ipd

1 0.3818102 0.3686755

2 1.2104316 0.1930935

3 0.3221994 0.1175303

4 0.6972376 0.6311050

5 0.1200485 0.3095543

6 0.1821030 0.9252790

7 0.1541078 0.5938104

8 0.6677455 0.1532545

9 0.1060112 0.1053578

10 0.4101818 1.4549091

We can observe that for different values of "k" the "within cluster sum of squares" changes. KMeans maximizes the between group dispersion and minimizes the sum of squares by certain percentage. The results show that when k = 6, sum of square is 79.1 %. This value changes by smaller percentage, if k is increased.

Appendix B

Selection of Optimum k for KNNdistplot

Value of obtained *eps* for different values of k

In this experiment, for all the clusters' (5 clusters generated using K-Means) data points, we have used two seperate values of k. Let us consider, each cluster has m_i data points, where i represents the cluster number. In one plot, we have used $k = minPts$, and in other $k = \sqrt{m_i}$ For these two seperate values of k, we use KNNdistplot to generate the knee for optimum *eps* detection

For Cluster 1

Case 1A:

$m = 2218$
$k = minPts = 4$
Obtained $eps = 0.0015$

Case 1B:

$m = 2218$
$k = minPts = 47$
Obtained $eps = 0.005$

Fig. B.1 shows the knee plots for $k = 4$, and $k = 47$.

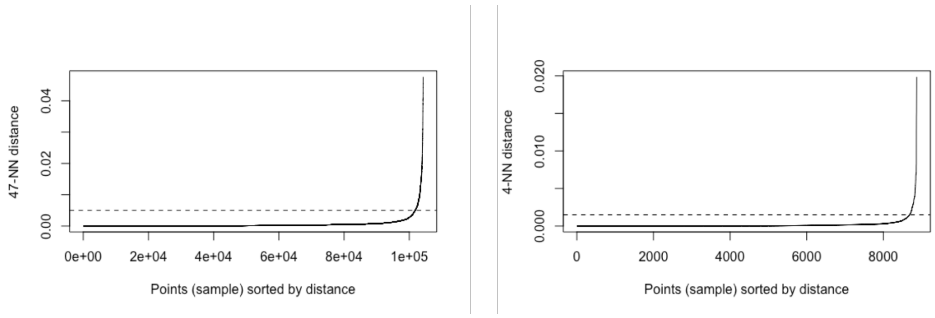

Figure B.1: knee plot for Cluster 1 data points when $k = 4$, and $k = 47$

For Cluster 2

Case 2A:

$m = 53352$
$k = minPts = 4$
Obtained $eps = 0.00005$

Case 2B:

$m = 53352$
$k = minPts = 231$
Obtained $eps = 0.0003$

Fig. B.2 shows the knee plots for $k = 4$, and $k = 231$.

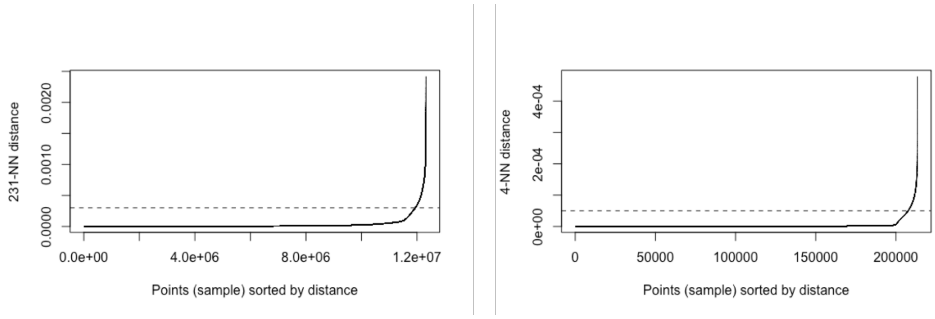

Figure B.2: knee plot for Cluster 2 data points when $k = 4$, and $k = 231$

For Cluster 3

Case 3A:

$m = 82$

$k = minPts = 4$

Obtained $eps = 0.015$

Case 3B:

$m = 82$

$k = minPts = 9$

Obtained $eps = 0.02$

Fig. B.3 shows the knee plots for $k = 4$, and $k = 9$.

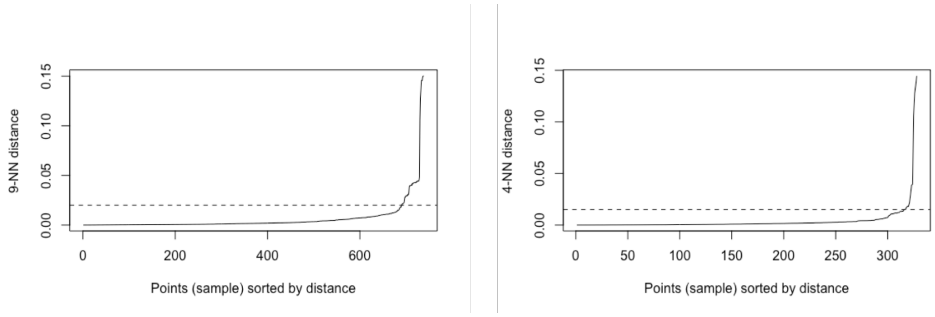

Figure B.3: knee plot for Cluster 3 data points when $k = 4$, and $k = 9$

For Cluster 4

Case 4A:

$m = 2266$

$k = minPts = 4$

Obtained $eps = 0.0015$

Case 4B:

$m = 2266$

$k = minPts = 48$

Obtained $eps = 0.0015$

Fig. B.4 shows the knee plots for $k = 4$, and $k = 48$.

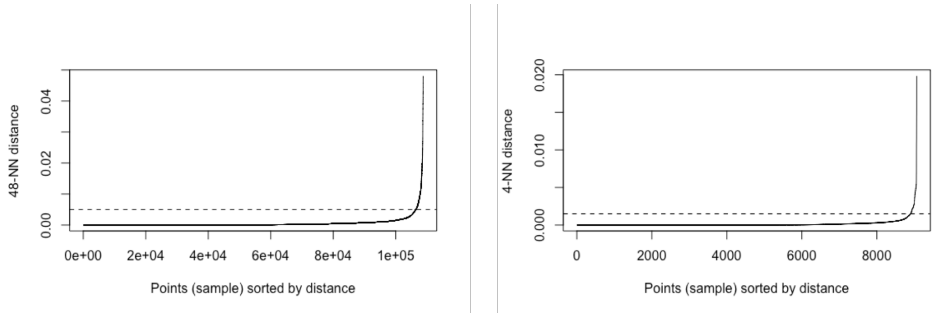

Figure B.4: knee plot for Cluster 4 data points when $k = 4$, and $k = 48$

For Cluster 5

Case 5A:

$m = 83$

$k = minPts = 4$

Obtained $eps = 0.015$

Case 5B:

$m = 83$

$k = minPts = 9$

Obtained $eps = 0.04$

Fig. B.5 shows the knee plots for $k = 4$, and $k = 9$.

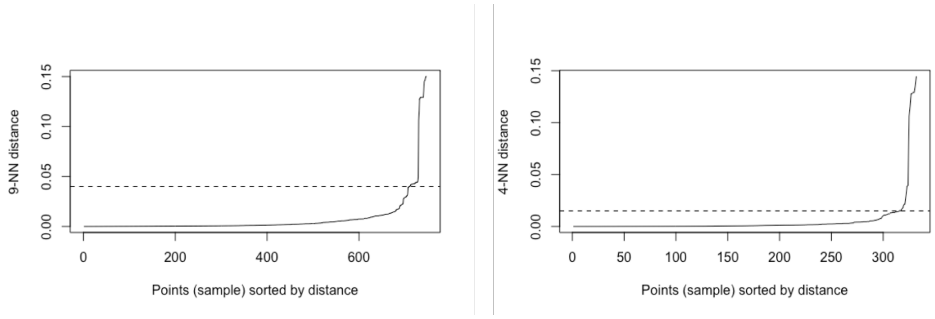

Figure B.5: knee plot for Cluster 5 data points when $k = 4$, and $k = 9$

Reason for selecting smaller k values

Tab B.1 shows that DBSCAN detects more number of outliers, when smaller k is chosen. In our research we have chosen $k = minPts$ because we want to train the system to detect more outliers in order to prevent possible attacks.

Table B.1: Outlier detection by DBSCAN for two seperate *eps* values

	C1	C2	C3	C4	C5
TotalPoints (m)	2218	53352	82	2266	83
Outliers(k=4)	42	1439	2	37	3
Outliers(k=sqrt(m))	11	0	2	5	2

Appendix C

Identifying Knee and DBSCAN plots

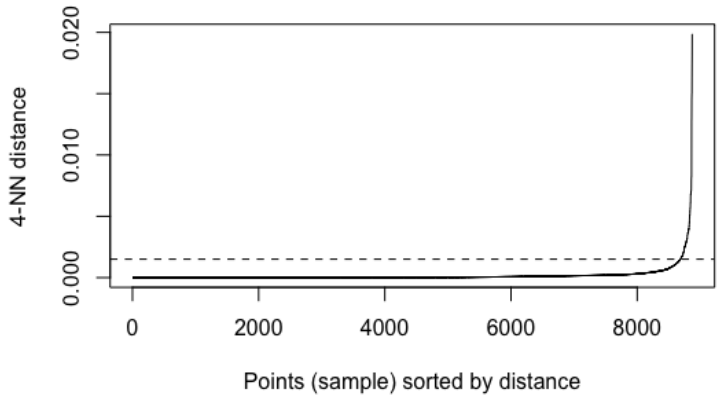

Figure C.1: KNNdist plot applied on cluster 1 to find the optimum *eps* value for DBSCAN

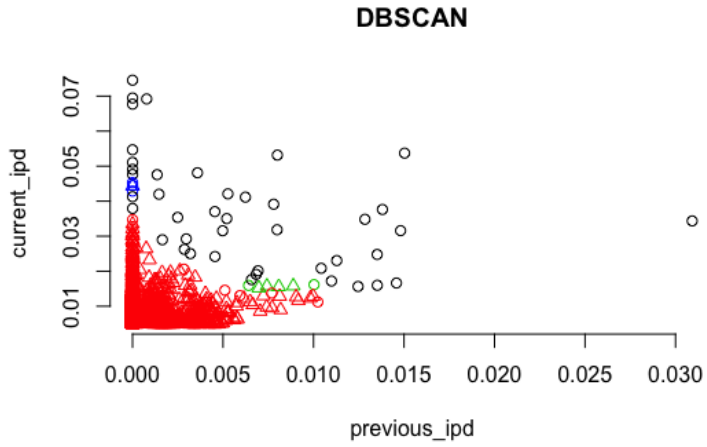

Figure C.2: DBSCAN Density-based clustering on Cluster 1 data set

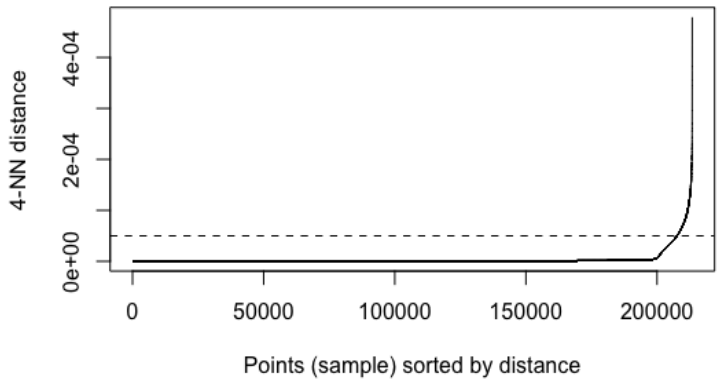

Figure C.3: KNNdist plot applied on cluster 2 to find the optimum *eps* value for DBSCAN

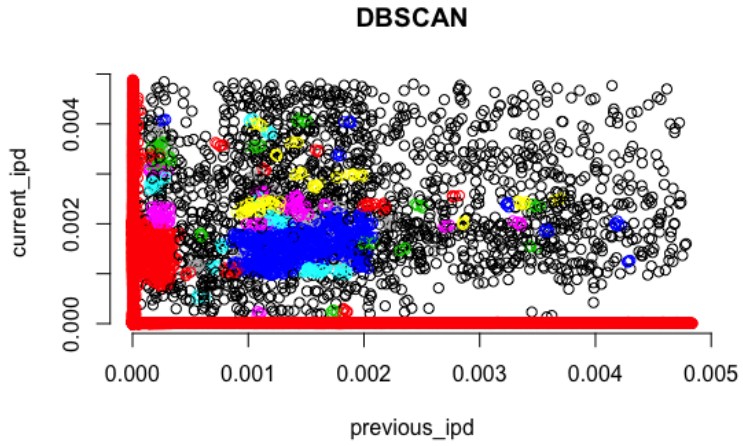

Figure C.4: DBSCAN Density-based clustering on Cluster 2 data set

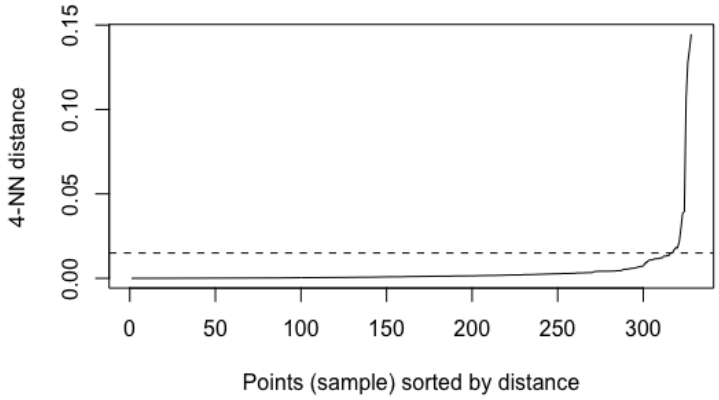

Figure C.5: KNNdist plot applied on cluster 3 to find the optimum *eps* value for DBSCAN

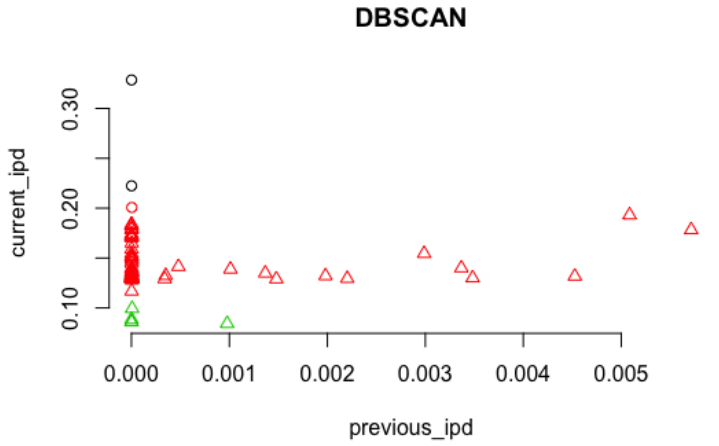

Figure C.6: DBSCAN Density-based clustering on Cluster 3 data set

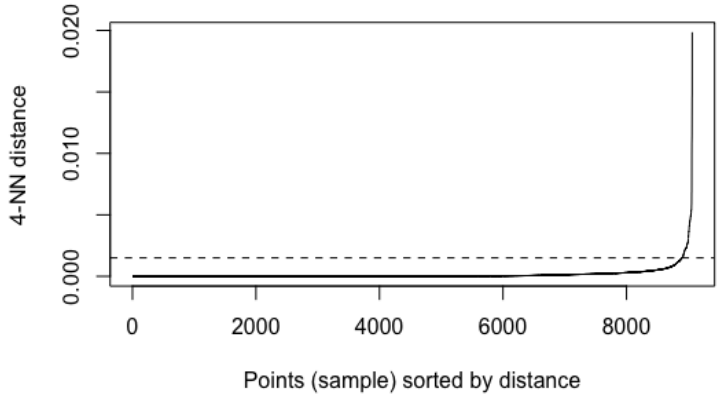

Figure C.7: KNNdist plot applied on cluster 4 to find the optimum *eps* value for DBSCAN

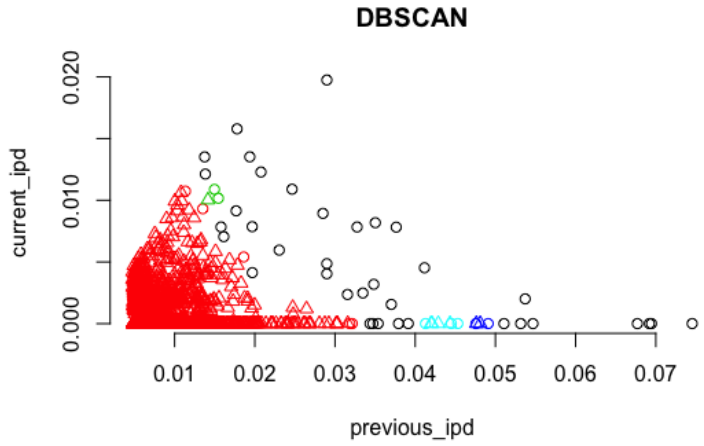

Figure C.8: DBSCAN Density-based clustering on Cluster 4 data set

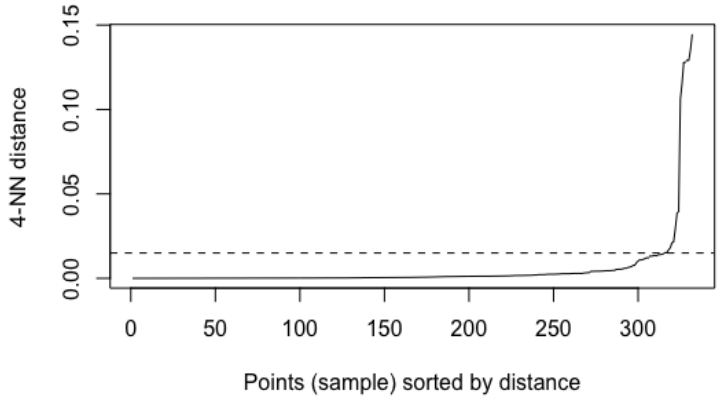

Figure C.9: KNNdist plot applied on cluster 5 to find the optimum *eps* value for DBSCAN

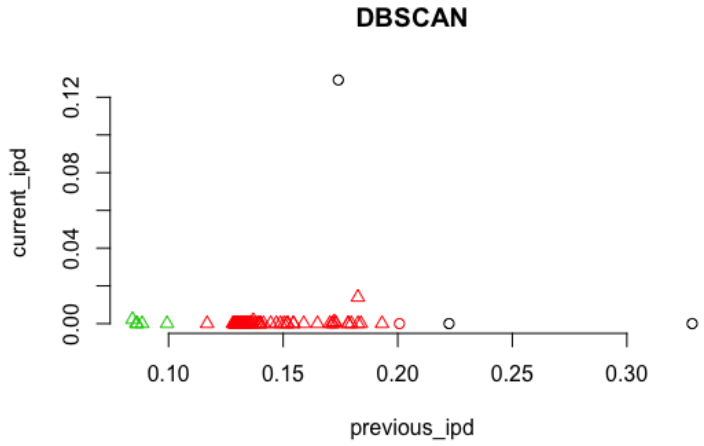

Figure C.10: DBSCAN Density-based clustering on Cluster 5 data set

Appendix D

Publications

D.1 Published

[$P1$] S. Dey, Q. Ye and S. Sampalli, AMLT: A Mutual Authentication Scheme for Mobile Cloud Computing, *In proceedings of 2018 IEEE Confs on Internet of Things, Green Computing and Communications, Cyber, Physical and Social Computing, Smart Data, Blockchain, Computer and Information Technology, Congress on Cybermatics*, Canada, pages 700-705, 2018 [42]

[$P2$] A. Odebode, Q. Ye, S. Sampalli, and S. Dey, KD1: A Sampling-based Clustering Scheme for Large Data Sets, *In proceedings of 2018 IEEE Confs on Internet of Things, Green Computing and Communications, Cyber, Physical and Social Computing, Smart Data, Blockchain, Computer and Information Technology, Congress on Cybermatics*, Canada, pages 822-827, 2018 [90]

[$P3$] S. Dey, S. Sampalli, and Q. Ye, Security and privacy issues in mobile cloud computing, *International Journal of Business and Cyber Security*, vol. 1, no. 1, pp 3143, July 2016 [41]

[$P4$] S. Dey, S. Sampalli, and Q. Ye, MDA: message digest-based authentication for mobile cloud computing, Journal of Cloud Computing, vol. 5, no. 1, pp. 18, 2016 [40]

[$P5$] S. Dey, S. Sampalli, and Q. Ye. A context-adaptive security framework for mobile cloud computing, *In 2015 11th International Conference on Mobile Ad-hoc and Sensor Networks (MSN)*, China, pages 8995, Dec 2015 [39]

[$P6$] S. Dey, S. Sampalli, and Q. Ye, A light-weight authentication scheme based on message digest and location for mobile cloud computing, *In Performance Computing and Communications Conference (IPCCC), 2014 IEEE International*, Austin, TX, USA, pages 12, Dec 2014 [38]

[*P7*] S. Dey, S. Sampalli, and Q. Ye, Message digest as authentication entity for mobile cloud computing, *Performance Computing and Communications Conference (IPCCC), 2013 IEEE 32nd International*, California, USA, pages 16, 2013 [37]

Appendix E

Copyright Permissions

Message digest as authentication entity for mobile cloud computing [37]

A light-weight authentication scheme based on message digest and location for mobile cloud computing [38]

A Context-Adaptive Security Framework for Mobile Cloud Computing [39]

MDA: message digest-based authentication for mobile cloud computing [40]

Security and Privacy Issues in Mobile Cloud Computing [41]

Jan 10, 2018

International Journal of Business and Cyber Security (IJBCS)
United Kingdom

I am preparing my Ph.D. thesis for submission to the Faculty of Graduate Studies at Dalhousie University, Halifax, Nova Scotia, Canada. I am seeking your permission to include a manuscript version of the following paper(s) as a chapter in the thesis:

Security and privacy issues in mobile cloud computing, Saurabh Dey, Srinivas Sampalli and Qiang Ye, International Journal of Business & Cyber Security, volume 1, pp 31-43., July 2016

Canadian graduate theses are reproduced by the Library and Archives of Canada (formerly National Library of Canada) through a non-exclusive, world-wide license to reproduce, loan, distribute, or sell theses. I am also seeking your permission for the material described above to be reproduced and distributed by the LAC(NLC). Further details about the LAC(NLC) thesis program are available on the LAC(NLC) website (www.nlc-bnc.ca).

Full publication details and a copy of this permission letter will be included in the thesis.

Yours sincerely,

Saurabh Dey

Permission is granted for:

a) the inclusion of the material described above in your thesis.

b) for the material described above to be included in the copy of your thesis that is sent to the Library and Archives of Canada (formerly National Library of Canada) for reproduction and distribution.

Name: Dr. P. R. Datta Title: Executive Chair

Signature: [redacted] Date: 18/01/2018

Bibliography

[1] EUREKA Innovation Event 2016. Smart Citites - Sustainable and Attractive Communities. http://www.vinnova.se/sv/Aktuellt--publicerat/Kalendarium/2016/160426-EUREKA-Innovation-Event-2016/, April 2016. [Online; accessed July-2016].

[2] IDC 2016. Worldwide Cloud IT Infrastructure Spend Grew 21.9% to $29.0 Billion in 2015. http://www.idc.com/getdoc.jsp?containerId=prUS41176716, 2016. [Online; accessed May-2016].

[3] ABB. Smart City Power Distribution. https://library.e.abb.com/public/26cc45d6bdccd48dc1257e06004d4cfc/Smart_City_Power_Distr_broch_758009_LRENa_without_pocket_for_web.pdf, 2015. [Online; accessed July-2016].

[4] AGT and Cisco. AGT and Cisco Traffic Incident Management Solution : Improving Traffic Safety and Efficiency, 2014.

[5] Z. Ahmad, K. E. Mayes, S. Dong, and K. Markantonakis. Considerations for mobile authentication in the Cloud. *Information Security Technical Report*, 16(3-4):123–130, August 2011.

[6] A. A. Aldossary and A. M. Zeki. Web User' Knowledge and Their Behavior towards Security Threats and Vulnerabilities. In *2015 4th International Conference on Advanced Computer Science Applications and Technologies (ACSAT)*, pages 256–260, Dec 2015.

[7] M. Alizadeh and W. H. Hassan. Challenges and opportunities of Mobile Cloud Computing. *2013 9th International Wireless Communications and Mobile Computing Conference (IWCMC)*, pages 660–666, July 2013.

[8] M. Almi'ani, A. A. Ghazleh, A. Al-Rahayfeh, and A. Razaque. Intelligent intrusion detection system using clustered self organized map. In *2018 Fifth International Conference on Software Defined Systems (SDS)*, pages 138–144, April 2018.

[9] S. M. Alqahtani and R. John. A comparative analysis of different classification techniques for cloud intrusion detection systems' alerts and fuzzy classifiers. In *2017 Computing Conference*, pages 406–415, July 2017.

[10] M. Alrokayan and R. Buyya. A Web Portal for Management of Aneka-based Multicloud Environments. In *Proceedings of the Eleventh Australasian Symposium on Parallel and Distributed Computing - Volume 140*, AusPDC '13, pages 49–56, Darlinghurst, Australia, Australia, 2013. Australian Computer Society, Inc.

[11] M. M. Alshammari, A. A. Alwan, A. Nordin, and I. F. Al-Shaikhli. Disaster recovery in single-cloud and multi-cloud environments: Issues and challenges. In *2017 4th IEEE International Conference on Engineering Technologies and Applied Sciences (ICETAS)*, pages 1–7, Nov 2017.

[12] S. Avramenko, S. Esposito, M. Violante, M. Sozzi, M. Traversone, M. Binello, and M. Terrone. An Hybrid Architecture for consolidating mixed criticality applications on multicore systems. In *2015 IEEE 21st International On-Line Testing Symposium (IOLTS)*, pages 26–29, July 2015.

[13] L. Badia, D. Munaretto, A. Testolin, A. Zanella, M. Zorzi, and M. Zorzi. Cognition-based networks: Applying cognitive science to multimedia wireless networking. *Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014*, pages 1–6, jun 2014.

[14] The World Bank. Internet users (per 100 people) . `http://data.worldbank.org/indicator/IT.NET.USER.P2`, 2014. [Online; accessed July-2016].

[15] M.A. Barlow. *The Pythagorean Theorem*. WestBow Press, 2016.

[16] R. Barona and E. A. M. Anita. A survey on data breach challenges in cloud computing security: Issues and threats. In *2017 International Conference on Circuit ,Power and Computing Technologies (ICCPCT)*, pages 1–8, April 2017.

[17] A. Behl. Emerging security challenges in cloud computing: An insight to cloud security challenges and their mitigation. *2011 World Congress on Information and Communication Technologies*, pages 217–222, December 2011.

[18] D. P. Benjamin. A cognitive approach to intrusion detection. In *2007 IEEE Symposium on Computational Intelligence in Security and Defense Applications*, pages 161–168, April 2007.

[19] D. Bhamare, T. Salman, M. Samaka, A. Erbad, and R. Jain. Feasibility of Supervised Machine Learning for Cloud Security. In *2016 International Conference on Information Science and Security (ICISS)*, pages 1–5, Dec 2016.

[20] F. V. Breugel. Deciding Probabilistic Bisimilarity Distance One for Labelled Markov Chains. `https://simons.berkeley.edu/sites/default/files/docs/8986/talk-2.pdf`. [Online; accessed May-2018].

[21] British Standards Institution. Smart City Framework Guide to establishing strategies for smart cities and communities. 2014.

[22] A. Caragliu, C. Del Bo, and P. Nijkamp. Smart cities in Europe. Serie Research Memoranda 0048, VU University Amsterdam, Faculty of Economics, Business Administration and Econometrics, 2009.

[23] S. Carpendale. Evaluating Information Visualizations. In Andreas Kerren, JohnT. Stasko, Jean-Daniel Fekete, and Chris North, editors, *Information Visualization*, volume 4950 of *Lecture Notes in Computer Science*, pages 19–45. Springer Berlin Heidelberg, 2008.

[24] S.S. Chapade, K.U. Pandey, and D.S. Bhade. Securing Cloud Servers Against Flooding Based DDOS Attacks. *2013 International Conference on Communication Systems and Network Technologies*, pages 524–528, April 2013.

[25] F. Chen and R. Li. Sink Node Placement Strategies for Wireless Sensor Networks. *Wirel. Pers. Commun.*, 68(2):303–319, January 2013.

[26] L. Chen, H. Tang, and J. Wang. Analysis of VANET Security Based on Routing Protocol Information. *Fourth International Conference on Intelligent Control and Information Processing (ICICIP)*, pages 134–138, Jun 2013.

[27] Y. Chen, L. Li, and Z. Chen. An Approach to Verifying Data Integrity for Cloud Storage. In *2017 13th International Conference on Computational Intelligence and Security (CIS)*, pages 582–585, Dec 2017.

[28] Z. Chen, J. Guo, and Q. Liu. Dbscan algorithm clustering for massive ais data based on the hadoop platform. In *2017 International Conference on Industrial Informatics - Computing Technology, Intelligent Technology, Industrial Information Integration (ICIICII)*, pages 25–28, Dec 2017.

[29] F. Chi, X. Wang, and W. Cai. Ad-hoc Cloudlet Based Cooperative Cloud Gaming. 7161(c):1–14, 2015.

[30] R. Chow, M. Jakobsson, R. Mauoke, J Molina, Y Niu, E. Shi, and Z. Song. Authentication in the Clouds : A Framework and its Application to Mobile Users. *ACM Cloud Computing Security Workshop (CCSW)*, pages 1–6, Oct 2010.

[31] P. Comerford, J. N. Davies, and V. Grout. Reducing Packet Delay through Filter Merging. In *2016 IEEE/ACM 9th International Conference on Utility and Cloud Computing (UCC)*, pages 358–363, Dec 2016.

[32] KPMG International Cooperative. Journey to the Cloud. `http://www.gartner.com/newsroom/id/3616417`, February 2017. [Online; accessed July-2017].

[33] A. Corici, A. Elmangoush, R. Steinke, T. Magedanz, J. Mwangama, and N. Ventura. Utilizing M2M technologies for building reliable smart cities. *2014 6th International Conference on New Technologies, Mobility and Security - Proceedings of NTMS 2014 Conference and Workshops*, pages 0–4, 2014.

[34] C. Cremers. The Scyther Tool: Verification, Falsification, and Analysis of Security Protocols. *Proceedings of the 20th International Conference on Computer Aided Verification, Princeton, USA,*, 2008.

[35] C.J.F. Cremers. The Scyther Tool: Verification, Falsification, and Analysis of Security Protocols. In Aarti Gupta and Sharad Malik, editors, *Computer Aided Verification*, volume 5123 of *Lecture Notes in Computer Science*, pages 414–418. Springer Berlin Heidelberg, 2008.

[36] C. Dall, S. W. Li, J. T. Lim, J. Nieh, and G. Koloventzos. Arm Virtualization: Performance and Architectural Implications. In *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*, pages 304–316, June 2016.

[37] S. Dey, S. Sampalli, and Q. Ye. Message Digest as Authentication Entity for Mobile Cloud Computing. *Performance Computing and Communications Conference (IPCCC), 2013 IEEE 32nd International*, pages 1–6, 2013.

[38] S. Dey, S. Sampalli, and Q. Ye. A light-weight Authentication Scheme Based on Message Digest and Location for Mobile Cloud Computing. In *33rd International Performance Computing and Communications Conference (IPCCC), 2014 IEEE International*, pages 1–2, Dec 2014.

[39] S. Dey, S. Sampalli, and Q. Ye. A Context-Adaptive Security Framework for Mobile Cloud Computing. In *2015 11th International Conference on Mobile Ad-hoc and Sensor Networks (MSN)*, pages 89–95, Dec 2015.

[40] S. Dey, S. Sampalli, and Q. Ye. MDA: Message Digest-based Authentication for Mobile Cloud Computing. *Journal of Cloud Computing*, 5(1):18, November 2016.

[41] S. Dey, S. Sampalli, and Q. Ye. Security and Privacy Issues in Mobile Cloud Computing. *International Journal of Business and Cyber Security*, 1(1):31–43, July 2016.

[42] S. Dey, Q. Ye, and S. Sampalli. AMLT: A Mutual Authentication Scheme for Mobile Cloud Computing. *2018 IEEE Confs on Internet of Things, Green Computing and Communications, Cyber, Physical and Social Computing, Smart Data, Blockchain, Computer and Information Technology, Congress on Cybermatics*, pages 700–705, 2018.

[43] R. I. Dinita, G. Wilson, A. Winckles, M. Cirstea, and T. Rowsell. A novel autonomous management distributed system for cloud computing environments. In *IECON 2013 - 39th Annual Conference of the IEEE Industrial Electronics Society*, pages 5620–5625, Nov 2013.

[44] A. Eaves and M. Stockman. Desktop as a service proof of concept. *Proceedings of the 13th annual conference on Information technology education - SIGITE '12*, page 85, 2012.

[45] IDG Enterprise. IDG Enterprise Cloud Computing Survey Insight into the Advancement of Enterprise Cloud Adoption. `http://www.idgenterprise.com/resource/research/2015-cloud-computing-study`, 2015. [Online; accessed May-2016].

[46] M. Ester, H. P. Kriegel, and X. Xu J. Sander. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD-96 Proceedings*, pages 226–231, 1996.

[47] E. D. Frauenstein and S. V. Flowerday. Social network phishing: Becoming habituated to clicks and ignorant to threats? In *2016 Information Security for South Africa (ISSA)*, pages 98–105, Aug 2016.

[48] H. Garcke, T. Preusser, M. Rumpf, A. Telea, U. Weikard, and J. Van Wijk. A continuous clustering method for vector fields. In *Proceedings Visualization 2000. VIS 2000 (Cat. No.00CH37145)*, pages 351–358, Oct 2000.

[49] Gartner. Gartner Says Worldwide Public Cloud Services Market to Grow 18 Percent in 2017. `http://www.gartner.com/newsroom/id/3616417`, February 2017. [Online; accessed July-2017].

[50] D. Gautam and V. Tokekar. An approach to analyze the impact of DDOS attack on mobile cloud computing. In *2017 International Conference on Information, Communication, Instrumentation and Control (ICICIC)*, pages 1–6, Aug 2017.

[51] BSI Group. The Role of Standards in Smart Cities . `http://www.bsigroup.com/LocalFiles/en-GB/smart-cities/resources/The-Role-of-Standards-in-Smart-Cities-Issue-2-August-2014.pdf`, 2014. [Online; accessed May-2016].

[52] A. Guy and Corporation Intel. Smart Citites USA. `http://smartamerica.org/teams/smart-cities-usa/`, June 2014. [Online; accessed August-2016].

[53] J. B. D. Joshi H. Takabi and G. Ahn. Security and Privacy Challenges in Cloud Computing Environments. *Security & Privacy, IEEE*, 8:24–31, December 2010.

[54] M. Hahsler. Density Based Clustering of Applications with Noise (DBSCAN) and Related Algorithms. pages 20–21, 2018.

[55] A. P. HIMSS. A HIMSS ASIA PACIFIC EXCLUSIVE ARTICLE, May 2016.

[56] G Hollestelle, S. Mauw, and CJF. Cremers. Systematic Analysis of Attacks on Security Protocols. Master's thesis, Technical University of Eindhover, Department of Mathematics and Computer Science, November 2005.

[57] R. Hope, T. Foster, A. Money, M. Rouse, N. Money, and M. Thomas. Smart Water Systems. *Oxford University*, April 2011.

[58] D. Huang and Z. Zhou. Secure data processing framework for mobile cloud computing. *2011 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 614–618, April 2011.

[59] A. Iqbal, C. Pattinson, and A. L. Kor. Performance monitoring of Virtual Machines (VMs) of type i and ii hypervisors with SNMPv3. In *2015 World Congress on Sustainable Technologies (WCST)*, pages 98–99, Dec 2015.

[60] L. R. Izquierdo, S. S. Izquierdo, J. M. Galán, and J. I. Santos. Techniques to Understand Computer Simulations: Markov Chain Analysis. *Journal of Artificial Societies and Social Simulation*, 12(1):6, 2009.

[61] D. Jana. Efficient Management of Security and Privacy Issues in Mobile Cloud Environment. *Annual IEEE Indian Conference (Indicon)*, 2013.

[62] J. G. Jetcheva, Y.-C. Hu, S. PalChaudhuri, A. K. Saha, and D. B. Johnson. CRAWDAD dataset rice/ad_hoc_city (v. 2003-09-11). Downloaded from `https://crawdad.org/rice/ad_hoc_city/20030911/bus_mobility`, September 2003. traceset: bus_mobility.

[63] T. J. Jeyaprabha, G. Sumathi, and P. Nivedha. Smart and secure data storage using Encrypt-interleaving. In *2017 Innovations in Power and Advanced Computing Technologies (i-PACT)*, pages 1–6, April 2017.

[64] C. Jiang, D. Ou, Y. Wang, X. You, J. Zhang, J. Wan, B. Luo, and W. Shi. Energy efficiency comparison of hypervisors. In *2016 Seventh International Green and Sustainable Computing Conference (IGSC)*, pages 1–8, Nov 2016.

[65] S. Julian and S. Shuey, M.and Cook. Containers in Research: Initial Experiences with Lightweight Infrastructure. In *Proceedings of the XSEDE16 Conference on Diversity, Big Data, and Science at Scale*, XSEDE16, pages 25:1–25:6, New York, NY, USA, 2016. ACM.

[66] M. F. Kabir and S. Hartmann. Cyber security challenges: An efficient intrusion detection system design. In *2018 International Young Engineers Forum (YEF-ECE)*, pages 19–24, May 2018.

[67] D. Dattatray Kankhare and A. A. Manjrekar. A cloud based system to sense security vulnerabilities of web application in open-source private cloud IAAS. In *2016 International Conference on Electrical, Electronics, Communication, Computer and Optimization Techniques (ICEECCOT)*, pages 252–255, Dec 2016.

[68] M. R. M. Kassim and A. N. Harun. Wireless sensor networks and cloud computing integrated architecture for agricultural environment applications. In *2017 Eleventh International Conference on Sensing Technology (ICST)*, pages 1–5, Dec 2017.

[69] T. Kim, Y. Choi, S. Han, J. Y. Chung, J. Hyun, J. Li, and J. W. K. Hong. Monitoring and detecting abnormal behavior in mobile cloud infrastructure. In *2012 IEEE Network Operations and Management Symposium*, pages 1303–1310, April 2012.

[70] D. Knuth and A. Yao. *Algorithms and Complexity: New Directions and Recent Results*, chapter The complexity of nonuniform random number generation. Academic Press, 1976.

[71] N. Kumar, V. Katta, H. Mishra, and H. Garg. Detection of Data Leakage in Cloud Computing Environment. In *2014 International Conference on Computational Intelligence and Communication Networks*, pages 803–807, Nov 2014.

[72] M. Kwiatkowska, G. Norman, and D. Parker. PRISM 4.0: Verification of probabilistic real-time systems. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6806 LNCS:585–591, 2011.

[73] C. C. Lee, Y. M. Lai, C. T. Chen, and S. D. Chen. Advanced secure anonymous authentication scheme for roaming service in global mobility networks. In *Wireless Personal Communications*, volume 94, page 12811296, June 2017.

[74] H. Lee, K. Lee, S. Hong, K. Song, T. Roh, J. Bae, and H.-J. Yoo. A 5.5mW IEEE-802.15.6 wireless body-area-network standard transceiver for multichannel electro-acupuncture application. In *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2013 IEEE International*, pages 452–453, Feb 2013.

[75] X. Li, K. Jiang, H. Wang, X. Zhu, R. Shi, and H. Shi. A novel k-means classification method with genetic algorithm. In *2017 International Conference on Progress in Informatics and Computing (PIC)*, pages 40–44, Dec 2017.

[76] Y. Li and L. Guo. An efficient network anomaly detection scheme based on tcm-knn algorithm and data reduction mechanism. In *2007 IEEE SMC Information Assurance and Security Workshop*, pages 221–227, June 2007.

[77] L. Liberti and C. Lavor. *Euclidean Distance Geometry: An Introduction*. Springer Undergraduate Texts in Mathematics and Technology. Springer International Publishing, 2017.

[78] G. Lowe. A hierarchy of authentication specifications. *Proceedings 10th Computer Security Foundations Workshop*, pages 31–43, June 1997.

[79] R. Madhusudhan and K. S. Suvidha. An enhanced secure authentication scheme with user anonymity in mobile cloud computing. In *2017 International Conference on Public Key Infrastructure and its Applications (PKIA)*, pages 17–22, Nov 2017.

[80] A. M. Marques and Fyodor. Zenmap Reference Guide (Man Page). `https://nmap.org/zenmap/man.html`. [Online; accessed May-2018].

[81] D. Mathew and B. A. Jose. Performance analysis of virtualized embedded computing systems. In *2017 7th International Symposium on Embedded Computing and System Design (ISED)*, pages 1–5, Dec 2017.

[82] E. Mathisen. Security Challenges and Solutions in Cloud Computing. *5th IEEE International Conference on Digital Ecosystems and Technologies (IEEE DEST 2011)*, 5:208–212, June 2011.

[83] D. McGaughey, T. Semeniuk, R. Smith, and S. Knight. A systematic approach of feature selection for encrypted network traffic classification. In *2018 Annual IEEE International Systems Conference (SysCon)*, pages 1–8, April 2018.

[84] G. Metta, L. Natale, S. Pathak, L. Pulina, and A. Tacchella. Safe and Effective Learning: A Case Study. *IEEE International Conferernce on Robotics and Automation*, pages 4809–4814, May 2010.

[85] D. Moloja and N. Mpekoa. Towards a cloud intrusion detection and prevention system for m-voting in south africa. In *2017 International Conference on Information Society (i-Society)*, pages 34–39, July 2017.

[86] K. K. Nguyen, D. T. Hoang, D. Niyato, P. Wang, D. Nguyen, and E. Dutkiewicz. Cyberattack detection in mobile cloud computing: A deep learning approach. In *2018 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–6, April 2018.

[87] J. Nithisha and J. Vinisha. An intelligent packet forwarding in router using self learning classifier. In *2017 Third International Conference on Science Technology Engineering Management (ICONSTEM)*, pages 324–327, March 2017.

[88] T.H. Noh. End-to-end self-healing SDH/ATM networks. In *Global Telecommunications Conference, 1996. GLOBECOM '96. 'Communications: The Key to Global Prosperity*, volume 3, pages 1877–1881 vol.3, Nov 1996.

[89] J.R. Norris. *Markov Chains*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 1998.

[90] A. A. Odebode, Q. Ye, S. Sampalli, and S. Dey. KD1: A Sampling-based Clustering Scheme for Large Data Sets. *2018 IEEE Confs on Internet of Things, Green Computing and Communications, Cyber, Physical and Social Computing, Smart Data, Blockchain, Computer and Information Technology, Congress on Cybermatics*, pages 822–827, 2018.

[91] J. Opara-Martins, R. Sahandi, and F. Tian. Critical review of vendor lock-in and its impact on adoption of cloud computing. In *International Conference on Information Society (i-Society 2014)*, pages 92–97, Nov 2014.

[92] S. Pan, T. Morris, and U. Adhikari. Developing a hybrid intrusion detection system using data mining for power systems. *IEEE Transactions on Smart Grid*, 6(6):3104–3113, Nov 2015.

[93] T. Phillips, T. Karygiannis, and R. Kuhn. Security standards for the RFID market. *Security Privacy, IEEE*, 3(6):85–89, Nov 2005.

[94] K. Popović and Ž. Hocenski. Cloud computing security issues and challenges. In *The 33rd International Convention MIPRO*, pages 344–349, May 2010.

[95] H. Qi and A. Gani. Research on Mobile Cloud Computing : Review , Trend and Perspectives. pages 195–202, 2012.

[96] H. Rajaguru and S.K. Prabhakar. *KNN Classifier and K-Means Clustering for Robust Classification of Epilepsy from EEG Signals. A Detailed Analysis.* Anchor Academic Publishing, 2017.

[97] I. A. Rassan and H. AlShaher. Securing Mobile Cloud Computing Using Biometric Authentication (SMCBA). In *2014 International Conference on Computational Science and Computational Intelligence*, volume 1, pages 157–161, March 2014.

[98] A. G. Revar and M. D. Bhavsar. Securing user authentication using single sign-on in Cloud Computing. *2011 Nirma University International Conference on Engineering*, pages 1–4, December 2011.

[99] M. Ring, A. Dallmann, D. Landes, and A. Hotho. Ip2vec: Learning similarities between ip addresses. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 657–666, Nov 2017.

[100] T. Ristenpart, E. Tromer, H. Shacham, and S. Savage. Hey, you, get off of my cloud: Exploring information leakage in third-party compute clouds. In *Proceedings of the 16th ACM Conference on Computer and Communications Security*, CCS '09, pages 199–212, New York, NY, USA, 2009. ACM.

[101] S. Saha and A. Majumdar. Data centre temperature monitoring with ESP8266 based Wireless Sensor Network and cloud based dashboard with real time alert system. In *2017 Devices for Integrated Circuit (DevIC)*, pages 307–310, March 2017.

[102] R. V. Sampangi, S. Dey, S. R. Urs, and S. Sampalli. a Security Suite for Wireless Body Area Networks. *International Journal of Network Security & Its Applications (IJNSA)*, 4(1):97–116, 2012.

[103] B. Satzger, W. Hummer, C. Inzinger, P. Leitner, and S. Dustdar. Winds of Change: From Vendor Lock-In to the Meta Cloud. *IEEE Internet Computing*, 17(1):69–73, Jan 2013.

[104] A. A. Shaikh. Attacks on cloud computing and its countermeasures. In *2016 International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES)*, pages 748–752, Oct 2016.

[105] F. B. Shaikh and S. Haider. Security threats in cloud computing. In *2011 International Conference for Internet Technology and Secured Transactions*, pages 214–219, Dec 2011.

[106] V. Shakya and R. R. S. Makwana. Feature selection based intrusion detection system using the combination of DBSCAN, K-Mean++ and SMO algorithms. In *2017 International Conference on Trends in Electronics and Informatics (ICEI)*, pages 928–932, May 2017.

[107] A. C. Siang and R. K. Rao. Theories of learning: a computer game perspective. *Multimedia Software Engineering, 2003. Proceedings. Fifth International Symposium on*, pages 239–245, 2003.

[108] K. Sigman. Discrete-time markov chains. `http://www.columbia.edu/~ks20/stochastic-I/stochastic-I-MCI.pdf`, 2009. [Online; accessed May-2018].

[109] Improved Outcomes Software. Manhattan. `http://www.improvedoutcomes.com/docs/WebSiteDocs/Clustering/Clustering_Parameters/Manhattan_Distance_Metric.htm`. [Online; accessed May-2018].

[110] Statista.com. Smartphone users worldwide 2014-2019. `http://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/`, 2016. [Online; accessed Jun-2016].

[111] T. Stuart, Y. Andy, N. Kumar, and J. Macaulay. The Mobile Cloud: When Two Explosive Markets Collide (Overview). `http://www.cisco.com/c/dam/en_us/about/ac79/docs/sp/Mobile-Cloud-Overview-POV.pdf`, 2011. [Online; accessed March-2016].

[112] A. M. Talib, R. Atan, R. Abdullah, and M. Azrifah. CloudZone: Towards an integrity layer of cloud data storage based on multi agent system architecture. *2011 IEEE Conference on Open Systems*, pages 127–132, September 2011.

[113] A. Thongthua and S. Ngamsuriyaroj. Assessment of Hypervisor Vulnerabilities. In *2016 International Conference on Cloud Computing Research and Innovations (ICCCRI)*, pages 71–77, May 2016.

[114] D. C. Trancă, D. Rosner, R. Curatu, A. Surpăteanu, M. Mocanu, Ş. Pardău, and A. V. Pălăcean. Industrial WSN node extension and measurement systems for air, water and environmental monitoring: IoT enabled environment monitoring using NI WSN nodes. In *2017 16th RoEduNet Conference: Networking in Education and Research (RoEduNet)*, pages 1–6, Sept 2017.

[115] S. Treetippayaruk and T. Senivongse. Security vulnerability assessment for software version upgrade. In *2017 18th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, pages 283–289, June 2017.

[116] W. N. Venables, D. M. Smith, and the R Core Team. An Introduction to R. pages 1–99, 2018.

[117] M. Verma and I. Schoen. V2C : A Secure Vehicle to Cloud Framework for Virtualized and On-Demand Service Provisioning. pages 148–154.

[118] L. Vrbský, M. S. da Silva, D. L. Cardoso, and C. R. L. Francês. Clustering techniques for data network planning in smart grids. In *2017 IEEE 14th International Conference on Networking, Sensing and Control (ICNSC)*, pages 7–12, May 2017.

[119] Y. Wang. Cognitive computing and World Wide Wisdom (WWW+). *Cognitive Informatics (ICCI), 2010 9th IEEE International Conference on*, II(Iicicc), 2010.

[120] D. Wei, J. Hoon, and C. Euiin. User Authentication using Profiling in Mobile Cloud Computing. *AASRI Conference on Power and Energy Systems*, 2:262–267, 2012.

[121] T.-H. Wu. A Passive Protected Self-Healing Mesh Network Architecture and Applications. *IEEE/ACM Transactions on Networking*, 2(1), Feb 1994.

[122] Z. Xiaofeng and H. Xiaohong. Research on intrusion detection based on improved combination of k-means and multi-level svm. In *2017 IEEE 17th International Conference on Communication Technology (ICCT)*, pages 2042–2045, Oct 2017.

[123] K. Xing, C. Hu, J. Yu, X. Cheng, and F. Zhang. Mutual privacy preserving k -means clustering in social participatory sensing. *IEEE Transactions on Industrial Informatics*, 13(4):2066–2076, Aug 2017.

[124] Z Yandong and Z Yongsheng. Cloud computing and cloud security challenges. *Information Technology in Medicine*, pages 1084–1088, 2012.

[125] K.-Y. Yoo. A lightweight multi-user authentication scheme based on cellular automata in cloud environment. *2012 IEEE 1st International Conference on Cloud Networking (CLOUDNET)*, 1(1):176–178, November 2012.

[126] Q. Zhang, L. Cheng, and R. Boutaba. Cloud computing: state-of-the-art and research challenges. *Journal of Internet Services and Applications*, 1(1):7–18, April 2010.

[127] Z. Zou, Y. Bao, F. Deng, and H. Li. An Approach of Reliable Data Transmission With Random Redundancy for Wireless Sensors in Structural Health Monitoring. *IEEE Sensors Journal*, 15(2):809–818, Feb 2015.