M.Sc. Engg. Thesis

A USER ATTRIBUTE AWARE MULTI-FACTOR AUTHENTICATION FRAMEWORK FOR CLOUD BASED SYSTEMS

By
Md. Mijanur Rahman Howlader
ID# 0413052007P

Submitted to
Department of Computer Science and Engineering
(In partial fulfilment of the requirements for the degree of
Master of Science in Computer Science and Engineering)

Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology (BUET)
Dhaka-1205

The thesis titled "A USER ATTRIBUTE AWARE MULTI-FACTOR AUTHENTICATION FRAMEWORK FOR CLOUD BASED SYSTEMS", submitted by Md. Mijanur Rahman Howlader, Roll No. 0413052007P, Session April 2013, to the Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, has been accepted as satisfactory in partial fulfillment of the requirements for the degree of Master of Science in Computer Science and Engineering and approved as to its style and contents. Examination held on date, September 17, 2018.

Board of Examiners

Candidate's Declaration

This is hereby declared that the work titled as "A USER ATTRIBUTE AWARE MULTI-FACTOR AUTHENTICATION FRAMEWORK FOR CLOUD BASED SYSTEMS ", is the result of research carried out by me under the supervision of Dr. Md. Shohrab Hossain, in the Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, Dhaka-1000. I also assure that this thesis or any snippet of it has not been submitted somewhere else for the award of any other degree or certificates.

Mijan

Md. Mijanur Rahman Howlader
Candidate

Acknowledgements

Alhamdulillah, all praise is to Allah, who is merciful and capable of everything.

Firstly, I would like to express my heartfelt gratitude to my honorable supervisor, Dr. Md. Shohrab Hossain for his constant support for this work. He guided me in every step of this work and taught me many things outside the box. His generosity and motivations are invaluable.

I am also very thankful to the honorable members of my thesis committee, Prof. Dr. A. K. M. Ashikur Rahman, Dr. A.B.M. Alim Al Islam, Head of the Department of CSE, and especially the external member Prof. Dr. Shabbir Ahmed for their encouragements, insightful comments, and valuable suggestions. I am also grateful to all other honorable teachers of the department for their comments and suggestions.

I would like to thank Mr. Ishtiaq Ashiq, Mr. Sheikh Shahriar Hassan, Ahsan Tarique, for their help, comments to this work. They helped me in many ways to get this work done. I wish all the best in their life. Last but not the least, I remain grateful to my beloved parents and other family members for their inspirations behind every success of my life.

Contents

List of Figures

List of Tables

Acronym List

MFA	Multi Factor Authentication
PAP	Profile Acquisition Program
TOTP	Time-based One Time Password
DoS Attack	Denial of Service Attack
SaaS	Software as a Service
PaaS	Platform as a Service
IaaS	Infrastructure as a Service
MITM Attack	Man In the Middle Attack

Abstract

Cloud computing has drawn a lot of attention in recent years. The facilities offered in the form of Software as a Service (Saas), Platform as a Service (PaaS)and Infrastructure as a Service (IaaS) are very promising. Current usage of cloud services are increasing day-by-day but it is not upto the level as it has been expected, mainly because of many security concerns that include client authentication, data integrity, data confidentiality, data privacy, different security attacks, lack of trusts, etc. Authentication issue is the primary pit from the security point of view that cloud system vendors must address. A lot of work has been done to build robust authentication techniques though it is still a matter of worry for the cloud vendors. As a remedy for current security flaws, we have proposed a multi-factor authentication framework for cloud based systems utilizing three factors. Our first factor is traditional UserID-password. Second factor is software/hardware token generated TOTP. Our third factor is Authentication Score, which is calculated from clients host, network, location and learned attributes. Cloud server collects those dynamic parameters from the client device using a light agent called Profile Acquisition Program (PAP). PAP runs and provides data only during the time of client login request. We have simulated our proposed multi-factor authentication framework in java (Desktop system) and android (Mobile system). Our proposed MFA framework for cloud is found to be resilient against many of the authentication attacks where existing solution fails.

Chapter 1

Introduction

1.1 Introduction

Cloud computing is an IT paradigm that provides on demand computing resources that covers everything from application softwares to data centers. Resources provided on the cloud can be scale up and down dynamically and easily to meet the clients demand. This computing technology is mostly over the internet on a pay-per-use basis. Popularity of this technology is mainly for its cost effectiveness. Because, simple IT farms do not have to have a costly data center for their own to host their servers and applications but rather go for a cost effective solution in the cloud. Moreover, large IT farms need not to keep their unused resources idle but rather utilize profitably by provisioning them to cloud services. Cloud based systems facilitates their clients in the form of SaaS (Software as a Service), PaaS (Platform as a Service) and IaaS (Infrastructure as a Service) [1] [2] [3] [4] [5].

Aside from bundles of facilities provided by the cloud, there are quite a lot security threats as well. There are concerns with client authentication, data integrity and confidentiality, different sorts of attacks, lack of trusts, etc. Being a big global business place, cloud systems has become a very appealing target to hackers and cyber criminals. Cloud system vendors have been trying their best to keep up their reputation and trustworthiness, which are key factors for cloud systems popularity and progress. The architectural features of cloud such as multi-tenancy and virtualization allow the users to achieve better operating costs and fast acquisition of services and resources. However, to achieve the full benefits of cloud, the service providers need to tackle the security concerns raised by the fast growing cloud consumers [6] [7] [8] [9] [10] [11].

1.2 Problem Statement

From security point of view in a public network, authentication is the foremost barrier that the attacker has to cross. So fortify that is the biggest challenge for the cloud vendors. There have been many authentication breaches and attacks over the past decade that raise tension on cloud systems and threatening its business. Things could get really out of control if attacker crosses the authentication barrier and causes harm to data of clients.

Many researchers over the globe have been trying to provide strong authentication techniques utilising different authentication factors, but still there are areas to address and attacks to overcome. Incorporating clients behavioral patterns and usual login characteristics directly in the authentication process, helps to repulse known attacks on authentication.

1.3 Existing Works

A great deal of research has been done to give robust authentication technique for internet based cloud systems. Vorugunti [12] has proposed an mobile authentication technique using big data features. Jeong et al. [13] have given an authentication using profiling. Naushahi [14] have proposed a profile based access control system in cloud computing environment. Shi et al. [15] have proposed an implicit authentication for the cloud. Each authentication approach has its own inherent weaknesses. One suits one purpose and but fails to fulfill other obligations. Existing works are discussed in more details in Chapter 2.

1.4 Motivations

From security point of view, authentication breach is the first and foremost point of security breach. If attackers get pass this first barrier then anything can happen, then the upward curve of cloud systems popularity and progress could observe a sharp downfall. We can think of, how critical it could get, if attackers get authenticated into the cloud system where cloud hosts thousands of clients and organizations data. This may lead to a catastrophe and may trigger huge social and financial distress. Let us take a look on some of the authentication breach and data hacking instances-

- **Yahoo!**

  Date: 2013-14

  Impact: 3 billion user accounts.

  The Internet service company Yahoo! reported two major authentication and data breaches of user account data to hackers during the second half of 2016. The first announced breach, reported in September 2016, had occurred sometime in late 2014, and affected over 500 million Yahoo! user accounts. A separate data breach, occurring earlier around August 2013, was reported in December 2016. Initially believed to have affected over 1 billion user accounts, Yahoo! later affirmed in October 2017 that all 3 billion of its user accounts were impacted. Both breaches are considered the largest discovered in the history of the Internet. Specific details of material taken include names, email addresses, telephone numbers, encrypted or unencrypted security questions and answers, dates of birth, and hashed passwords.

- **Security Attack on eBay**

  Date: May 2014

  Impact: 145 million users compromised.

  Details: The online auction giant reported a cyber-attack on authentication in May 2014, that it said exposed names, addresses, dates of birth and encrypted passwords of all of its 145 million users. The company said hackers got into the company network using the credentials of three corporate employees, and had complete inside access for 229 days, during which time they were able to make their way to the user database.

- **Security breach at PayPal**

  In 2014, there was a security breach at PayPal that left so many users passwords and personal information exposed. After that PayPal adopted two factor authentication.

- **Security Breach in RSA SecurID**

  The attack on RSA SecurID central server. On March 17, 2011, RSA announced that a cyber attack on its systems was successful and resulted in the compromise and disclosure of information “specifically related to RSA’s SecurID two-factor authentication products”.

Table 1.1: Client's Mobility Adaptation By Different Authentication Approaches

Multi-factor Authentication Methods					
Criteria	Password Based Only	Smart Card Based Only	Biometric Based Only	Security Token Based Only	Our Approach
Device Change	No	No	No	No	Yes
Location Change	No	No	No	No	Yes
Network Change	No	No	No	No	Yes
Time and Time Zone Change	No	No	No	No	Yes

In addition, reputation is a principal factor for cloud systems popularity and progress. If authentication breach happens even for once, then users would be very reluctant to migrate towards cloud based systems from local systems. Most of the state-of-the-art authentication methods suffer from various security vulnerabilities such as, replay, brute-force, phishing, spoofing, impersonating, denial of service (DoS), etc. We want to get rid of those security threats and provide a viable multi-factor authentication (MFA) solution for cloud based systems.

Table 1.1 shows a comparative study of authentication approaches and their adaptation with the change of client device, location, network, time and time zone. Authentication approaches which are based on password, smart card, biometric and security token only, cannot detect or adapt on clients mobility. Because, these approaches do not keep or process any data of the client except for password, security token, biometric samples, etc. In our proposed MFA for the cloud, we preserve and process client machine, location, network and learned behavioral patters, to adapt with clients mobility and authenticate only the legitimate clients.

Fig. 1.1 depicts a summarized comparison of 1FA, 2FA and our proposed MFA. We have compared them against performance matrices, such as simplicity, privacy, attack tolerance and robustness. When we consider simplicity, single factor authentication is the best one. There is no other second or third factor to worry about. 2FA and our proposed MFA have mid level simplicity. But we need to keep in mind that, client always considers to trade a bit of method complexity for robust security.

Privacy is a key concern but not always. There is a big tradeoff between extra info sharing with the cloud server and security. While using userid-password, we do not share anything and the privacy is good but there is a big tradeoff in security. As we share more information with the cloud server, the more correct decision the cloud server can take while authenticating the client.

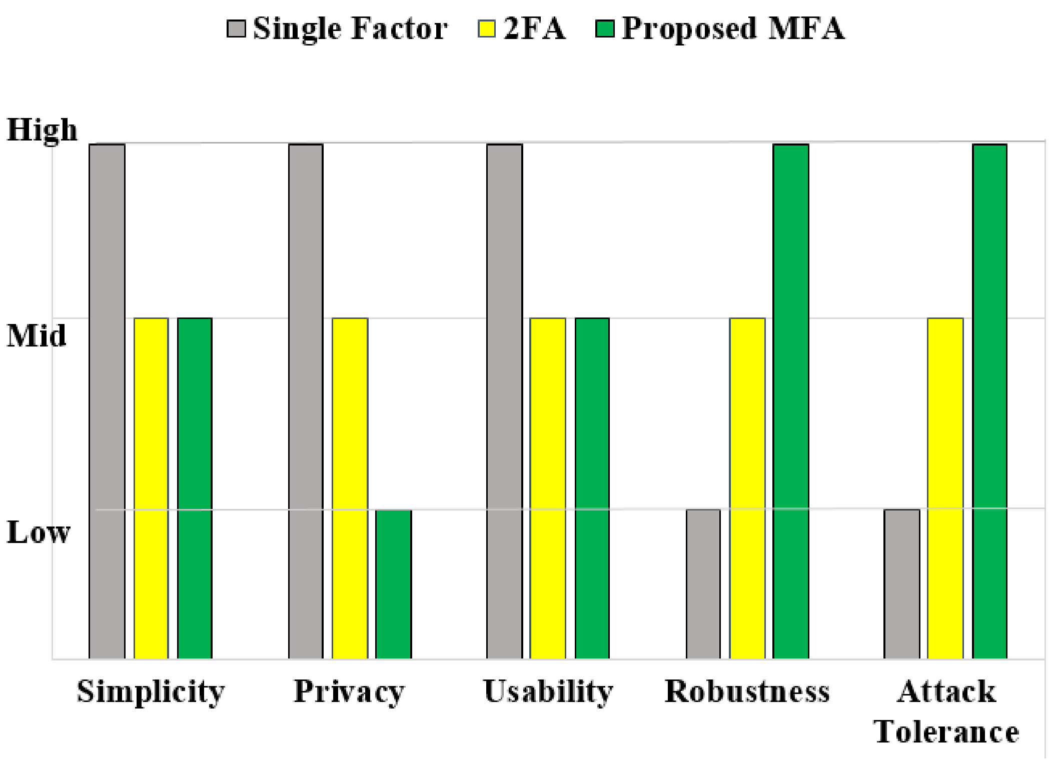


Figure 1.1: Comparison Among Single Factor Authentication, 2FA and Our Proposed MFA.

As we put more and more authentication factors together, authentication process becomes complex and lengthy where usability declines but on the positive side, security increases. We can add up five or ten factors together in the authentication process, but this will reduce the usability drastically. Clients are going to be reluctant to use it. So we have to keep a good balance between usability and security requirements from number of authentication factors.

Single factor authentication is more prone to security attacks. As we cautiously add more factors and parameters together, authentication becomes more securer and safer from security attacks. Adding authentication score to the authenticate clients makes it relatively easy to repulse many security attacks which was difficult to handle with existing authentication techniques.

1.5 Objectives

To overcome security flaws of current authentication techniques, we have proposed a MFA framework for cloud based systems. The main objectives of the research are summarized as follows:

(a) Propose a resilient multi-factor authentication (MFA) framework for cloud.

(b) Simulate the proposed multi-factor authentication framework through Java (desktop system) and Android (mobile system) simulation.

(c) Analyze the dynamic nature of clients profile behavior.

(d) Inspect alarming scenarios where existing authentication solution fails.

(e) Check the effectiveness of the proposed framework with our designed test cases.

(f) Analyze the response of our MFA framework against different attack and alarming scenarios.

(g) Compare our proposed MFA with existing authentication solutions.

1.6 Contributions

To overcome security pitfalls in cloud authentication, we have introduced authentication score in our MFA framework for cloud. This authentication score is calculated from set of client host, location, network and usual login attributes, which makes the authentication process more realistic and appropriate. By utilizing different authentication parameters with weight adjustments we made it really impossible for an attacker to get beyond the threshold value of the authentication score. Some alarming scenarios have been identified where authentication request from the client has to be rejected. At those scenarios, authentication request seems coming from the attacker even when the first two factors userID-password and TOTP from software/hardware token are correct. We have incorporated our authentication score with other two factors, userID-password and software/hardware token generated TOTP. We have analyzed the performance of our MFA framework against different types of security attacks and test cases to check its robustness and found satisfactory results. The results implied that the proposed MFA framework is feasible and secure for cloud computing authentication strategy. Our major contributions are summarized as follows-

- Introduced Authentication Score that incorporates clients host, location, network and learned attributes.
- Dynamic nature of clients profile behavior has been recorded and parameter weights are adjusted accordingly.
- Authentication score from dynamic and learned parameters can handle malicious attacks.

- Identified some alarming scenarios when authentication has to be denied for the client even if all factors are correct.
- Learned parameters can handle those alarming scenarios.
- Most of the existing authentication approaches blocks user but we are blocking suspicious machines.
- Incorporating RSA SecurID token with authentication score can repulse most of the known authentication attacks.
- Simulated our proposed framework in Java(Desktop System) and Android (Mobile System).
- Analyzed the response of our MFA framework against different attack scenarios and found satisfactory results.
- Compared our proposed MFA with existing authentication solutions.

Our proposed MFA framework rejects all attackers requests and only accepts valid clients' requests. Single factor compromise is not an issue for our MFA framework because, other factors still keep the system inaccessible to the attackers.

1.7 Organization of the Thesis

The remainder of this thesis is organized as follows: In Chapter 2, we have explored background studies about MFA (Multi-factor Authentication) along with its merits and demerits. Then, we have discussed some recent works on MFA. In Chapter 3, we have discussed our proposed MFA framework architecture in details. Chapter 4 is mainly about our implementations of MFA framework for both desktop and mobile. We also discussed our approach's learning and testing using a modeled dataset. In Chapter 5, we have explained our experiments and shown our results with comparison with existing authentication techniques. Finally, we concluded with some future prospects of MFA.

Chapter 2

Background and Existing Works

2.1 Background

In this chapter, we started with discussing about MFA with its benefits and challenges. Then we discussed the semantics of different attacks on authentication. After that, we have surveyed some related works on MFA.

2.1.1 Multi-factor Authentication (MFA)

Multi-factor authentication is a technique that utilizes more than one method from independent categories of authentication factors to verify user's identity. This provides additional layers of security when logging to the system or performing transactions online. Authentication factors can be grouped into four broad categories.

(i) Something the user knows (e.g., password, PIN, etc.)

(ii) Something the user possesses (e.g., cryptographic token (software token, hardware token), smart card, etc.)

(iii) Something the user is (e.g., biometric fingerprint, voiceprint, face, iris, hand geometry, retina, etc.)

(iv) Something the user has exhibited in the environment (e.g., user profile, host, network, location features, etc.)

While choosing multi-factor authentication method, one or more factors from at least two different groups have to be chosen. One can use password, different types of biometrics, different sorts of security tokens and user attributes. Here reducing the risk involves combining authentication factors in such a way so that only valid users can get his or her data. However, MFA shows the following two contradicting features when we add more factors to the authentication process.

- MFA reduces usability
- MFA enhances security

Along with the increase in number of authentication factor, authentication process becomes complex and lengthy where usability declines but security increases. We can add up to ten or twenty factors together in the authentication process. However, this will reduce the usability drastically. People will be reluctant to use it. Hence, there must be a balance between authentication complexity and usability.

Fig. 2.1 shows the authentication factors table. This table covers most of the authentication factors those are fully or partially in use today.

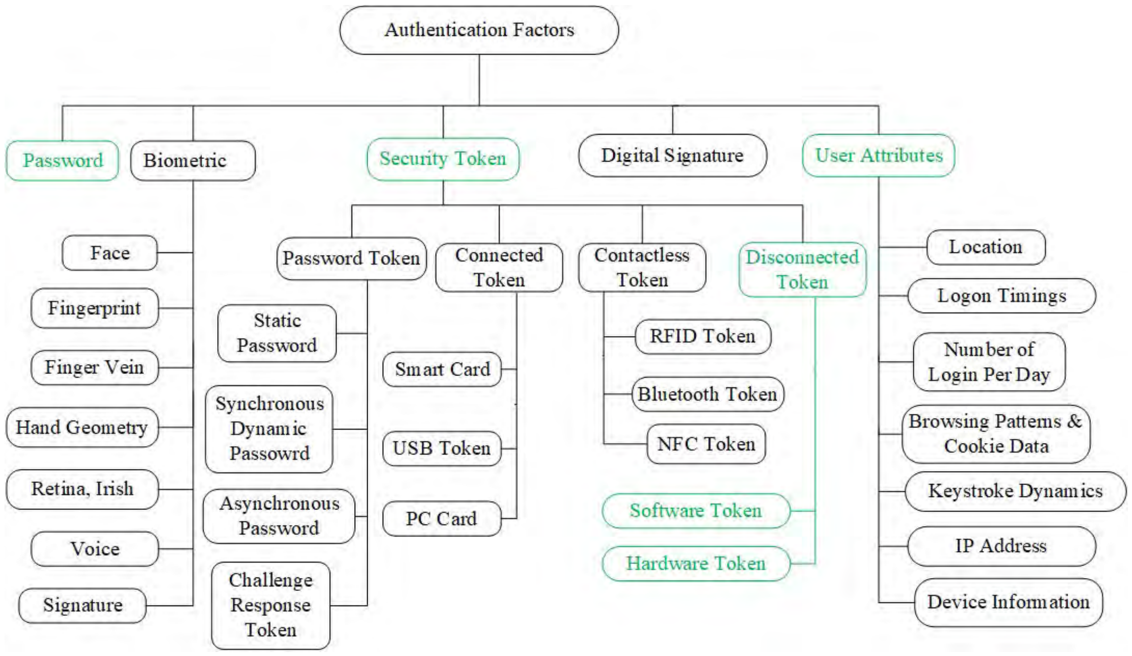


Figure 2.1: Different Types of Authentication Factors

2.1.2 Benefits of Multi-factor authentication

The benefits for MFA align very closely to the motivations for having multi-factor authentication. MFA comes with a bunch of facilities and security enhancements which are listed below-

(a) **Improved security**

The primary benefit of multi-factor authentication is to provide additional security by adding security protection in layers. The more layers/factors in place, the tougher it becomes for an intruder to gain access to the critical systems.

(b) **Safeguard against security attacks**

Adding multiple layers of authentication steps and combining different factors provides safeguard against most of the security threats.

(c) **Increased flexibility and productivity**

Being able to remove the burden of only passwords by using combinations of alternative factors has the potential to increase productivity and bring a better usability experience due to the increased flexibility of factor types. In the right environment and situation, there could even be an opportunity for a potential reduction in operational costs.

(d) **Achieve compliance**

Another benefit of multi-factor authentication is being able to achieve the necessary compliance requirements of security specific to any organization which in turn mitigate audit findings and avoiding potential fines.

2.1.3 Challenges of Multi-factor authentication

While there are many benefits of MFA, as with any technology, there are challenges as well.

(a) **Usability**

Along with with increase in authentication factors, authentication process becomes complex and slower where usability declines. But perhaps the bigger usability challenge is that different applications and systems often require different types of MFA.

(b) **Cost**

This one could be considered as the number one challenge for multi-factor authentication. As with the new technology like MFA, there is always an initial cost increase. Most of the cost incurred by MFA brought by additional support, maintenance and training, new service deployment, application development, software and hardware tokens, etc.

(c) **Varying risks**

While choosing authentication technique we need consider that one authentication method may be stronger than others but each method has its own inherent security risks. It is very essential to understand these risks and the exact level of security requirements for a certain type of organization.

(d) **Backup options**

It is very critical to have backup plan in place for multi-factor solution. For an instance, what if a client's security token or mobile phone is lost or stolen? Are there any other ways for clients to gain emergency access into the system?

(e) **Complexity**

With the inclusion of different authentication factors, for some of them there may be need of additional drivers, complex and arduous deployments, maintenance and support. There may be a need to compatibility checking as the environment reforms.

(f) **Finding the right MFA solution**

By understanding what is motivating managements of an organization to implement multi-factor authentication, they will be better positioned to weigh the benefits and challenges of potential MFA options and to select a solution that best fits for their organization's needs and requirements. We need to remember, the right solution should help organization improve security, meet compliance requirements, and even improve the productivity of users, while minimizing challenges, such as technical gaps, usability issues, and inherent complexities.

2.1.4 Security Token

Security tokens are physical device or software, used to get access into an electronically restricted source. It acts like an electronic key that provides an additional level of security.
There are many varieties of tokens some of which store passwords or cryptographic keys like digital signature or biometrics like palm print, finger print, etc. or simply a software with a specific token generation algorithm that runs in clients mobile or desktop. Design of security token differs from hardware device with firm tamper resistant packaging to device with small keypads to allow users to enter a PIN with display capability to show generated key number.

Software token offers many conveniences over hardware tokens. For an instance, there is no need to purchase additional hardware because the digital certificate or software token application can be installed in the mobile or desktop. To provide a hardware token for each customer will incur a bulk amount of money. Software tokens are easy to deploy and easy to manage and they do not have a battery that will run out. In fact, most of the smartphone users keep their phone with them all the time so that, there is no chance of forgetting their security token at home.

Software tokens are a bit less secure than hardware tokens because, the digital certificate or TOTP application resides within the mobile or desktop. There are a lot of instances where mobile phone or desktop are being hacked to extract information and the digital certificate can be exported. Even those applications can be tapped remotely to access the data. With the advancement of technology, there is a constant motivation for the attackers to exploit them [16].

In contrary, hardware tokens do not have the vulnerabilities because of tamper resistant built and stored in a dedicated hardware and a duplicate copy of it cannot be made. As a result, they are much more secure choice for MFA.

2.2 Attacks on Authentication

There have been many sort of attacks to make cloud authentication system vulnerable. Researchers doing their very best for the enhancements of authentication strategies and filling exposed holes. On the other hand, attackers try to find shortfalls of current authentication methods to get their way into the cloud system to do malicious operations usually to snatch money. Some of the possible attacks which directly or indirectly troubles a valid authentication are as follows-

2.2.1 Password Discovery Attack

Attacker may choose to discover the password of the client in many ways. Some of them are as follows-

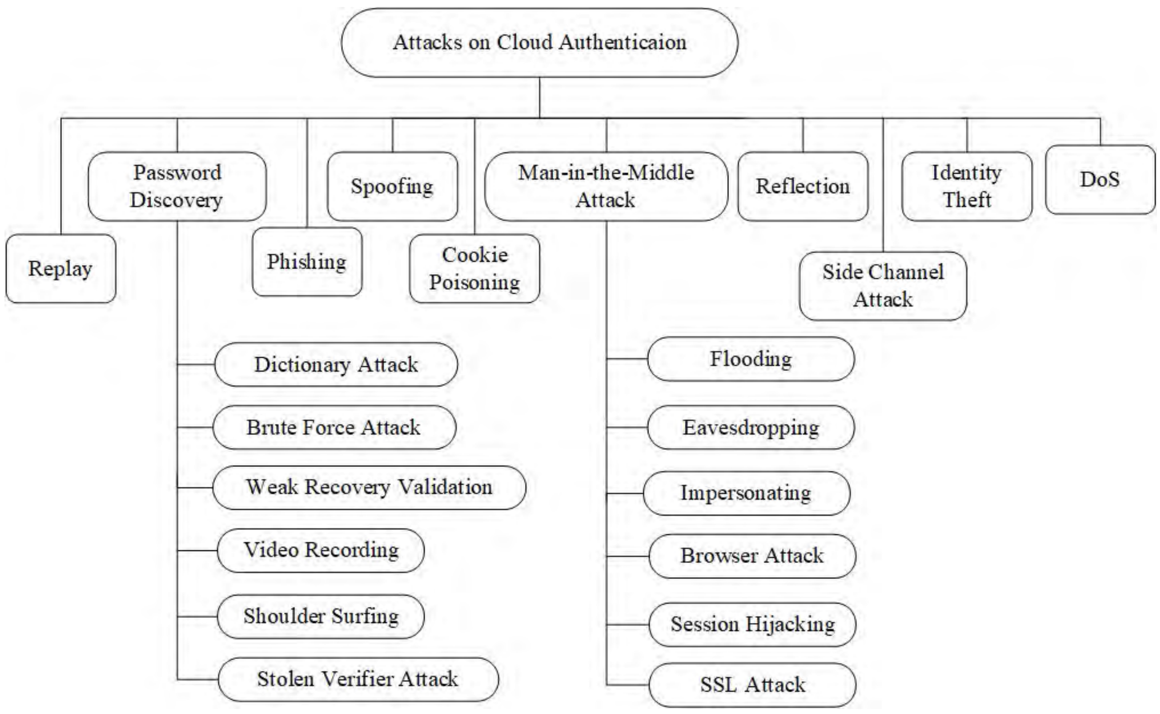


Figure 2.2: Attacks on cloud authentication.

2.2.1.1 Dictionary Attack

These are programs with built in dictionaries. They would use all dictionary words to attempt and find the correct password, in the hope that a user would have used a standard dictionary word. We can prevent this sort of attack by using random characters for a password which does not form a dictionary word. Choosing words from regional languages are still vulnerable because attacker may even has those dictionaries on their hand [17].

2.2.1.2 Brute Force Attack

Allows an attacker to guess a person's user name, password, credit card number, or cryptographic key by using an automated process of trial and error. This process involves checking all possible options until the correct match is found. This is an exhaustive effort which does not use any intellectual strategies [18].

2.2.1.3 Weak Password Recovery Validation

Allows an attacker to access a web site that provides them with the opportunity to illegally obtain, change, or recover another user's password. We call a password recovery system weak when the information needed to examine a user identity can be compromised through brute force attack or easily guessed secret questions.

2.2.1.4 Video Recording Attack

This sort of attack is launched at public places where the attacker may capture a video with a camera of login credentials like username, password, credit card details while the users enter it.

2.2.1.5 Shoulder Surfing

Attacks using social engineering such as monitoring the keyboard entry by the user or collecting his personal information to verify whether it is used as a password or forms part of the password. Countermeasure: Shoulder surfing attack can be avoided by providing wrong information for security questions and by providing passwords with spelling mistakes. While entering a PIN at an ATM or typing password on a keyboard, it can be covered so as to prevent this attack. When the transaction is over at an ATM counter, all the 10 keys from 0 to 9 can be pressed in order to confuse the attacker. Similarly, while entering your password on a keyboard, type some unwanted characters in between and use backspace to delete and then proceed with actual characters. This may also mislead the attacker [19].

2.2.1.6 Stolen Verifier Attack

The attacker performs this attack by accessing the password table stored at the verifier. Then he launches an offline guessing attack by running a script which performs hash on each entry of the dictionary and compares the generated message digest with the stored digest of the verifier, until a match is found. This attack can have a disastrous effect in a cloud environment hosting data belonging to multiple customers. In most of the cases, malicious insiders who are responsible for the operation of the cloud may get involved in this sort of attack [17].

2.2.2 Man-in-the-Middle Attack

Here the attacker secretly intercepts and possibly alters the communication between two parties(Cloud server and client) who believe they are directly communicating with each other. There are many types of man-in-the-middle attack. Some of them are as follows-

2.2.2.1 Flooding Attack

In this attack a flood of requests from the attacker overwhelms the the cloud server. Before providing the requested service, cloud server checks for the authenticity of the requested jobs and the process consumes CPU utilization, memory etc. Processing of these bogus requests, make legitimate service requests to starve, and as a result the server will offload its jobs to another server, which will also eventually arrive at the same situation. The adversary is thus successful in engaging the whole cloud system, by attacking one server and propagating the attack further by flooding the entire system. Flooding attack can be handled by organizing all the servers providing a specific cloud service as a fleet and these servers communicating among themselves regarding the incoming requests by message passing. Again a hypervisor can be used to schedule the requests among the fleets, determine the authenticity of the requests and prevent the fleets from being overloaded with bogus requests from an adversary. This attack can be controlled by data transfer throttling, fool proof authentication mechanisms and mechanisms that filter out bogus requests [17].

2.2.2.2 Eavesdropping Attack

Eavesdropping involves the act of listening to the communication channel established between two authorized users. In a cloud environment, a traffic eavesdropper passively intercepts the data transferred within a cloud by loading a bit of code on a cloud server or listens to data moving from a cloud consumer to provider and makes an unauthorized copy of the message.The attacker can use the illegally gathered information to get valid credentials of an authorized user which can be user to launch impersonating attack. Eavesdropping attack, in a cloud environment which results in information disclosure can be minimized by enforcing proper authorization procedures and by transmitting the data over a secure connection such as HTTPS. Encrypting the transmitted data and attaching a signature to the same can help the destination to ensure the integrity and authenticity of data. Adopting privacy-enhancing protocols which minimize the requirement of transmitting identity credentials from the cloud service user to the verifier will discourage the illegal activity of eavesdroppers. Authentication Protocols that protect secrets, ensures user anonymity and Password Authenticated Key exchange (PAKE) protocols are much preferred in a multi-tenant cloud environment [17].

2.2.2.3 Impersonating Attack

Here the adversary pretends to be a valid server or user and lures a valid entity to reveal the authenticating credentials which in turn is used to gain unauthorized access to the resources. Verifier Impersonation attack, phishing attack etc. can be categorized as impersonation attacks. Most of the times, in Phishing attacks the users are made to believe that they are communicating with valid server by creating a web page that look similar to the valid server page. In verifier impersonation attack, the attacker pretends to be the verifier and lure the customer to share the authentication keys or data, which may then be used to authenticate falsely to the verifier. . In November 2007, an employee of SaaS vendor, SalesForce was victimized by a phishing attack which resulted in the exposure of the SalesForce account information of some customers. In a cloud environment, this can be mitigated by using two-factor and multi factor authentication mechanisms that rely on personally identifiable information (PII) in addition to passwords. Also privacy enhancing protocols that protect secrets and avoid storage of secrets can help to keep impersonation attacks under control [17].

2.2.2.4 Browser Attack

This attack which results in data stealing is committed by sabotaging the signature and encryption during the translation of SOAP messages in between the web browser and web server, causing the browser to consider the adversary as a legitimate user and process all requests, communicating with web server. For authenticating the clients, current web browsers rely upon SSL/TLS as they are not able to apply WS-Security. Nevertheless, SSL/TLS only supports point-to-point communications and this makes the authentication process insecure. Also SSL/TLS has been broken by MarlinSpike using “Null-Prefix Attack” and attackers are able to perform this technique in order to request services from cloud systems without a valid authentication. The potential counter measure for this is that the vendors that create web browsers apply WS-Security concept, which works at message level, within their web browsers. WS-Security permits web browsers to use XML encryption to provide end-to-end encryption in SOAP messages which prevents sniffing of messages [17].

2.2.2.5 Session Hijacking

Session hijacking is possible, if the Session ID issued to the authenticated users is not protected properly, which in turn can be used for spoofing identity. Session side-jacking uses packet sniffing tools to capture a login sequence and thus gain access to the users session key Encrypting the communication channel can prevent this type of Session hijacking attack. These attacks exploiting the loopholes such as insecure communication protocols and unencrypted data can be thwarted by using a secure communication protocol such as HTTPS, by encrypting the files that store user or administrative login credentials etc. A strong authentication mechanism that rules out the possibility of unauthorized authentication and mechanisms that protect secrets such as session keys or avoid the storage of secrets is required in a cloud environment to prevent such attacks. The side-jacking attack can be mitigated by avoiding the transfer of session keys across the communication channel. A key exchange mechanism, that involves the calculation of session key separately by the client and server, resulting in the same key value, can also be adopted [17].

2.2.2.6 SSL Attack

SSL is a method of encryption used by various network communication protocols. Conceptually, SSL runs above TCP/IP, providing security to users communicating over other protocols by encrypting communications and authenticating communicating parties. SSL DDoS attacks and SSL DoS attacks target the SSL handshake mechanism, send garbage data to the SSL server, or abuse functions related to the SSL encryption key negotiation process. SSL attacks in the form of a DoS attack can also be launched over SSL-encrypted traffic, making it extremely difficult to identify.
A single standard home PC can take down an entire SSL-encrypted web application, and several computers can take down a complete farm of large, secured online services. SSL attacks are popular because each SSL session handshake consumes 15 times more resources from the server side than from the client side. Such attacks are ”asymmetric” because it takes significantly more server resources to deal with the attack than it does to launch it.

2.2.3 Spoofing Attack

Spoofing is an act of utilizing faked email header or IP address to deceive the recipients. Email spoofing is usually used in conjunction with phishing. An attacker may spoof their email address as a legitimate address and send and email with a link to the user and asks them to click on it. If the users is deceived by it and clicks, a malware or virus is installed in his/her computer automatically which may damage the operating system and critical applications while it propagates through the network, leaving the clients and vendors at risk [17] [19].

2.2.4 Replay Attack

It is also known as playback on which a valid authentication information from a client is intercepted by an adversary and re-transmits delayed replay messages to the server to gain access into the cloud system as a honest valid client. It is an attack on a security protocol using replay of messages from a different context into the intended (or original and expected) context, thereby fooling the honest participant(s) into thinking they have successfully completed the protocol run [18].

2.2.5 Cookie Poisoning

The cookie may store information such as a session identifier, user id, credit card number, pricing information, user preferences, and more. It is a known technique for achieving impersonation by which identity related information stored in the session cookies are being manipulated. Thus an attacker impersonate a valid client and gain access into the cloud system to perform actions on behalf of a victim. This attack which involves tampering with data can be handled by attaching the hash values of the data stored in the cookies and recalculating the same at the destination. Use of Message authentication codes, tamper resistant protocols and digital signatures can also aid in the detection and prevention of modifying the cookies [17] [19].

2.2.6 Proofing/Identity Theft attack

Anywhere personal information resides, is a potential target of identity theft attacks. In is attack, attacker deliberately uses the theft credentials of a valid clients to gain access to a cloud system [20].

2.2.7 Side Channel Attack

An attacker could aim to breach cloud systems by placing a malicious virtual machine in close proximity to a target cloud server and try to gain technical knowledge of the internal operation of the cloud systems [21].

2.2.8 Phising Attack

It is a type of attack to trick someone into giving up valuable credentials. Here the attacker disguises as a trustworthy entity in an electronic communication. It may come through as a fake website link, email, call, text message all to get you to expose your account details like username, password, credit card number, social security number, birthdate, bank account number, pin etc. A phishing scam usually provides a link as depicted in Figure-5 to a bogus website where the end-user is required to enter sensitive account information and ended up loosing valuable data and money. Attackers may use their skills to make look-alike sites using HTML design and Web programming, so the untrained eye can be easily fooled.

Client is need to be aware of any e-mail or website employs tactics to create pressure to provide the information quickly. Also be aware of the misspellings and misuse of language in e-mails and websites which often indicative of fraud. If legitimacy of the email is not convincing, we can contact the company by phone, using a trusted number, to make sure that the e-mail is authentic. we need aware of using hyperlinks included in e-mails, as they can display one URL and may actually linking to another one. Also we need to be careful of the @ symbol in the URL. This @ symbol is used to specify specific user for the site for example, example http://user@domain. Another thing we need to be aware of to get rid of phishing attack is unregistered sites. For example http://www.facebooks.com. See the "s"at the end of facebook and it is definitely not facebook.com [17] [19] [20].

2.2.9 Reflection Attack

Reflection attacks uses the same protocol in both directions. It is performed on mutual authentication schemes wherein the attacker tricks the target into revealing the secret to its own challenge. This attack normally done by creating parallel session by an unauthorized user to establish a valid session with the server. The attacker impersonates a valid user and requests a login session to the server. The server, as part of authenticating the requester, sends him a challenge and requests the attacker to send back his secret response. Since he is not a legitimate user, the attacker will not know this secret. From the servers perspective, it was the victim who sent the original request. All the data from those clients piles up, congesting the target's Internet connectivity. With the maximized bandwidth, normal traffic cannot be serviced and clients cannot connect which may lead to denial of service attack(DoS) attack.

In a cloud scenario, keeping track of the sessions and the secrets used for each session as well as limiting the number of established sessions can help to minimize reflection attacks. Again ensuring that the communication messages exchanged between the user and the cloud server during the authentication process are not symmetrical in nature can help mitigate reflection and parallel session attacks [17].

2.2.10 Denial of Service Attack

In multi-tenant shared cloud systems a victim may witness deliberate failed logins and unable to access his account. [22]. The main purpose of attacker is to overload the target cloud server with fake authenticatoon/service requests to make it difficult or impossible to respond to legitimate requests. Unable to tackle all the requests on its own the overwhelmed server delegates the work load to other similar service instances which ultimately leads to flooding attacks. This attack on availability can be controlled to a certain extent by data transfer throttling which deliberately regulates the amount of data transferred per unit time among the communicating entities and by limiting the allocation of network bandwidth [17].

2.3 Related Works on MFA

Numerous works have been done on constructing usable and robust authentication techniques. Providing strong authentication for the cloud based systems has given it new challenges to accomplish. Single factor authentication has been facing a dilemma of attacks and drawbacks as the organizations are tending to move towards MFA. Let us mention a few of those-

2.3.1 Single-Sing-On (SSO) Authentication

Powell et al. [23] proposed an SSO authentication technique for heterogeneous clouds without changing the design of existing systems using a proxy certificate repository and shibboleth authentication technology. The problems with SSO is like one password to rule them all means one password to compromise them all. Also SSO is a single point of failure and using a third party to service to deal with total authentication might not be good idea for financial organizations. Moreover, SSO providers present a very appealing target to hackers and cyber criminals and any data loss they experience could prove disastrous for their users.

Alves et al. [24] proposed a MFA framework for cloud utilizing traditional userID/password and OTP second factor and OpenID protocol for identity management which provides single-sign on access to the cloud. Here, OpenID allows users to log into multiple unrelated sites without then need of having separate identity for each.

2.3.2 MFA with Biometrics

Ziyad et al. [25] proposed a multi-factor biometric authentication that adopted palm vein and fingerprint. They stored palm vein data in multi-component smart card and the fingerprint data in the central database of the cloud server. They enhanced the security by performing the biometric matching process in the smart card with Match-on-Card technology and data never leave the smart card. But the key problem that arises here is in the case of stolen and fabricated smart card.

Yassin et al. [26] proposed a two-factor authentication scheme based on Schnorr digital signature and feature extraction from fingerprint. Their second factor relies on signed users password with fingerprint's features. But this may cause problem with different sample of user fingerprint while generating the hash of it and initial registration process may fail while there is no secure channel between user and the cloud server.

Yassin et al. [27] proposed two factor cloud authentication scheme where onetime password's anonymity is used as a first factor and partial image encryption based on edge detection is used as a second factor. They used Canny's edge detection process for fast partial image encryption with symmetric encryption is done as a second factor. Here, image edge pixels are encrypted using the stream cipher as it holds most of the images data. However, problem may arise with the second factor when image contrast and quality is low.

Nagaraju et al. [28] has given a multi-factor biometric authentication scheme that integrates the bio-metric fingerprint with user-id, password and One-Time Password (OTP). But this state of the art userID-password, biometric fingerprint and OTP all suffers from various attack described earlier in this section.

Raja et al. [29] proposed a multi-modal biometric authentication system using face, periocular and iris characteristics, combining three factors: password, smart card and biometrics. Although, their proposed authentication system is quite good but the usability is low. This is because to authenticate a valid user and smart card, a biometric input scan point and a password input screen are required which makes it quite cumbersome.

2.3.3 MFA with Security Token

Security tokens are physical device or software, used to get access into an electronically restricted source. It acts like an electronic key that provides an additional level of security.There are many varieties of tokens some of which store passwords or cryptographic keys like digital signature or biometrics like palm print, finger print, etc. or simply a software with a specific token generation algorithm that runs in clients mobile or desktop. Design of security token differs from hardware device with firm tamper resistant packaging to device with small keypads to allow users to enter a PIN with display capability to show generated key number.

Software token offers many conveniences over hardware tokens. For an instance, there is no need to purchase additional hardware because the digital certificate or software token application can be installed in the mobile or desktop. To provide a hardware token for each customer that will incur a bulk of money. Software token are easy to deploy and easy to manage and they do not have a battery that will run out. In fact, most of the smartphone users keep their phone with them all the time so that, there is no chance of forgetting their security token at home.
Software tokens are a bit less secure than hardware token because, the digital certificate or TOTP application resides within the mobile or desktop. There are a lot of instances where mobile phone or desktop are being hacked to extract information and the digital certificate can be exported. Even those applications can be tapped remotely to access the data. With the advancement of technology, there is a constant motivation for the attackers to exploit them [16].

In contrary, hardware tokens do not have the vulnerabilities because of tamper resistant built and stored in a dedicated hardware and a duplicate copy of it cannot be made. As a result, they are much more secure choice for MFA.

2.3.3.1 Software/Hardware Token Based

Aloul et al. proposed a two factor authentication scheme that involves use of a mobile phone as a software token for OTP generation. They have implemented a SMS-based procedure for retrieving password and as a possible mean of synchronization. Their OTP is generated by factors that are unique to both the user and the mobile device itself (IMEI number, IMSI number, username, PIN, hour, minute, date etc.).

Attab et al. [30] proposed an authentication technique for cloud computing using USB token with a combination of hash function and Diffie-Hellman key exchange.

2.3.3.2 Smart Card Based

Candan et al. [31] proposed a two factor authentication technique that incorporates smart card along with a password. Smart card communicates with the server and upon verifying secret keys the client can authenticated itself with server. Their scheme seems withstand most common security breaches as well as compromised smart card scenarios and offline dictionary attacks on the passwords.

Bae et al. [32] proposed a smart card based authentication protocol for multi-server based IoT environment. They performed authentication for each entity where users have to go through the authentication process using a smart card transmitted from an authentication server, and to login to a server connected to the IoT.

Huszti et al. [33] proposed a two factor authentication protocol that utilizes a static password and an OTP generated by smart card. They have used Merkle tree to verify the OTP.

2.3.4 Miscellaneous Authentication Techniques

2.3.4.1 TrustCube

Chow et al. [34] proposed this authentication technique which has the following features-

- Manages the authentication infrastructure by policy based implicit authentication.
- Translates user behavior into scores.
- Addresses authentication on both mobile and non-mobile devices.
- Offload's computation from clients and offers dynamic provisioning of compute resources.
- Participants: client device, data collector, authentication engine, and authentication consumer.

Authentication process of their approach:

- The Client Device periodically forwards context and activity data based on policies to a Data Collector.

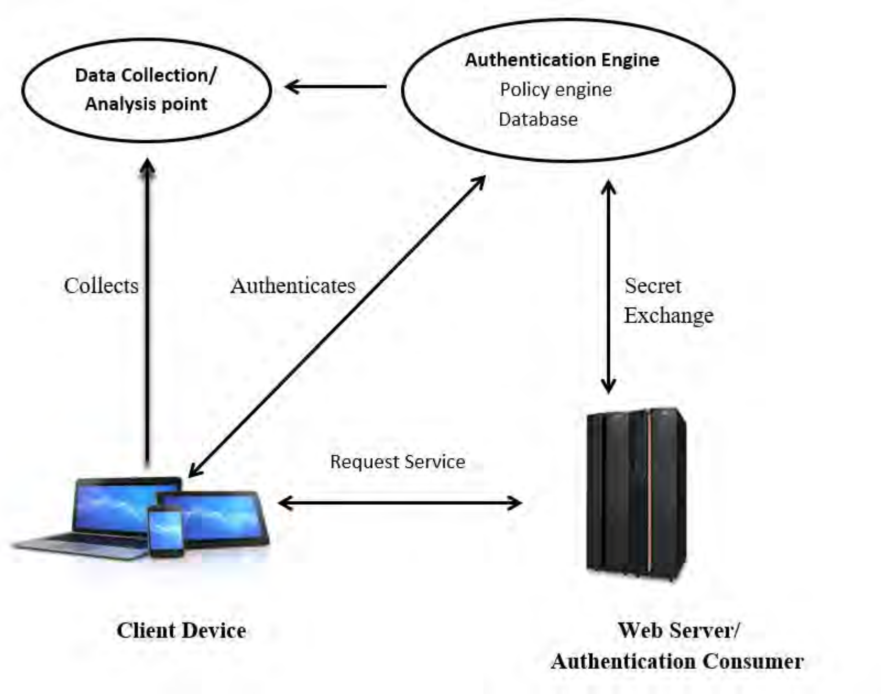


Figure 2.3: TrustCube Authentication Architecture

- The Client Device requests a service from the Authentication Consumer.
- For each request, the authentication consumer will register a policy with the authentication engine. Here, the policy includes at least three parts: the access request, the information to be collected from client devices or data collector for this access request, and a rule to generate the authentication result.
- The authentication engine obtains data from data collectors, and may request data directly from client devices.
- The Authentication Engine exchanges a secret with Authentication Consumer during authentication (in order to later verify authentication results).
- The authentication engine then applies the authentication policy and determines the authentication result and sends this back to the authentication consumer. Based on that result, the authentication consumer will either provide the service or reject the request of the of the client device.

2.3.4.2 MACA

Liu et al. [35] proposed this privacy-preserving multi-factor authentication system utilizing the features of big data. Here, the authentication factors are a password and a hybrid profile of user behavior.

This MFA technique considers both user privacy and usability combining big data features. They adopted fuzzy hashing and fully homomorphic encryption (FHE) to protect users sensitive profiles and to handle the varying nature of the user profiles.

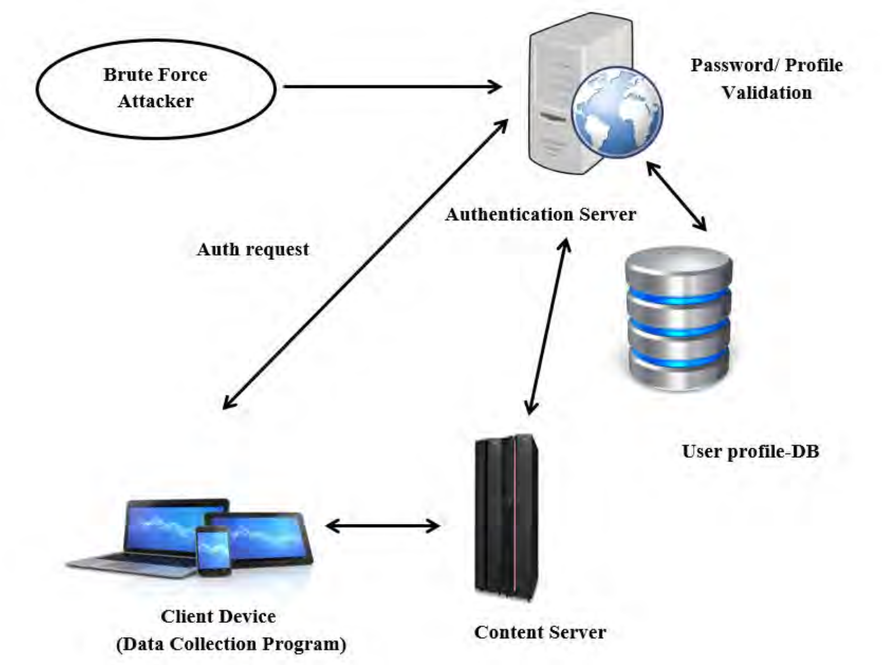


Figure 2.4: MACA: A privacy-preserving multi-factor cloud authentication system utilizing big data

Assumptions of their approach:

- Perfect knowledge assumption: They assumed that the adversary has perfect knowledge of the multi-factor authentication system.
- First-Factor knowledge assumption: They assumed that the adversary knows the victim's first authentication factor, which is the user password.

They tested their system against the following types of adversaries-

- Brute-force attacker: Knows the first factor and exhaustively searches for correct user profile.
- Honest-but-curious server attacker: the server in the cloud that processes the authentication requests tries to derive the underlying sensitive information from the users cryptographic profile.

The architecture of their developed system has four primary components

- An open-source program that runs on the users local host.
- A user profile database (UPDB) that stores the users information in a privacy-preserving fashion.
- An authentication server (AS) that processes and validates users login request in the cloud environment and a content server.

Some extension of this work could be adding more features based of different scenarios and including a weighting scheme on features.

2.3.5 MFA with User Profile Attributes

Uluagac et al. [36] proposed MFA technique utilizing password and hybrid profile of user behavior as second factor with host- and network-based features for continuous re-authentication.

Ibrahim et al. [37] proposed and attribute based authentication for the cloud on which a cloud user, in a one move non-interactive way with one-group-element private key and a set of unique non-secret attributes tokens, to prove his/her identity to the cloud server.

Xuejiao Liu et al. proposed an attribute base access control and authentication for cloud computing by extending ciphertext-policy attribute-based encryption (CP-ABE) with a hierarchical structure of multi-authorities and exploiting attribute-based signature (ABS). Although ABE concept is very powerful and a promising mechanism, ABE systems suffer mainly from two drawbacks: non-efficiency and non-existence of attribute revocation mechanism. Other main challenges are key coordination, escrow and revocation.

2.3.6 Problems with Existing Authentication Factors: Summary

Let us give a summary-of problems of existing authentication techniques. It is obvious and a common thinking of computer world that Password is no longer secure. Its trend is over. There are quite a few problems with biometric features, such as fingerprint, faceprint, iris/retinal scan. We just cannot get a new pattern. Once a biometric pattern is compromised, is compromised forever. Biometrics are easier to hack than passwords because, Security cameras are crawling everywhere and its difficult to imagine a world where everyone wears gloves and masks to hide biometric traits. Security token like smart card can be lost or stolen and duplicate copy of it can be made.

Revealing users characteristics like profile, host and network features raises privacy concern and the cloud server has to find a way of keeping them secret.

Biometric features are substantially unique but suffers many security threats. One of the threat is spoofing where users biometric templates can be misused. Fake biometrics can be used by the impostors during enrollment phase of authentication. In addition, genuine template can be replaced by imposters template to have unauthorised access. Spoofing attack presumes replay of raw biometric data or features extracted from raw biometric data in order to fool the authentication server into believing the attacker as real user. Users biometrics features may have extra noise and thus match with incorrect user template, causing false detection. In biometric features interclass similarity or inter-class variability in the feature set may induce hight false detection rate. Data collection unit for biometric may fail to acquire biometric traits of user due to the limits of proper capturing technology or adverse environmental conditions [38]. Let us summarize the problems of existing authentication factors-

(a) **UserID/password (Known Factor)**

- Breakable
- Vulnerable to security attacks.

(b) **Biometrics (Is Factor)**

- One sample for lifetime.
- Once compromised, its compromised forever.
- Security cameras are crawling everywhere and biometric traits are difficult to hide.
- In some sense password is more secure than biometrics.
- We can change the password but not the biometrics!

(c) **Security Token (Possession Factor)**

- Can be lost, stolen or broken
- For most of them duplicate copy of it can for made.

(d) **Users Characteristics (e.g., profile, host, network, etc.)**

- Privacy Concerns

Chapter 3

Proposed Scheme

We have proposed a user attribute aware multi-factor authentication framework especially for cloud based systems. At first, we give an overview of our proposed MFA authentication framework which consists of following factors-

- Factor-1: UserID with a password.
- Factor-2: Security token (Software/Hardware) generated TOTP.
- Factor-3: Authentication score: Calculated from users host, network, location, and learned attributes.

The purpose of our work is to build a authentication framework that will reject attackers login request and would accept only valid clients' requests. Single factor in the hand of attacker, is not enough to hack into the system because other factors would still make the system inaccessible. Let us give a summary of all the factors we have used in our MFA.

3.1 UserID-Password (Factor-1)

The first factor of our MFA authentication framework is userID with a password. There is a basic signup screen where the client register itself with the cloud server by providing the basic details like name, userID, password, city, country, secret questions etc. Here, the user must follow the given password policy while choosing the password. This policy is imposed by the cloud server and the client is bound to obey it for its safety.

Password Policy

- Password must contain at least 8 characters.
- Password must not contain the user's name or userID or any full dictionary words.
- Password need to have at least one uppercase letter and one digit (0-9).
- At least one non-alphanumeric characters (special characters): #$%&˜( ){}_+-*/=‘.
- Password has to be renewed after each 30 days.
- At most five attempts at a time are allowed to enter correct UserID and password.
- A user can not reuse his last five passwords.

If a user fails to enter the correct userID-password within five attempts then an hour block window is imposed for the same machine IP-MAC address. During this block window, cloud server discards every authentication request from that potential client machine. This penalty is compounded, for the next rounds of five attempts failures and it will result in two hour, six hour block window. If such type of failure happens three consecutive rounds, then cloud server adds that machine information into its blocked machine list. The above complexity requirements are enforced when passwords are required to be created or changed.

3.2 Security Token (Factor-2)

For security token, we have chosen RSA SecurID. The RSA SecurID authentication mechanism consists of a token, which is either hardware (e.g., a key fob) or software (a soft token). This token is assigned to a computer user and which generates an authentication code at fixed intervals (usually 60 seconds) using a built-in clock and the card's factory-encoded almost random key (known as the seed). The seed is different for each token, and is loaded into the corresponding RSA SecurID server (RSA Authentication Manager, formerly ACE/Server) as the tokens are purchased. On-demand tokens are also available, which provide a token code via email or SMS delivery, eliminating the need to provision a token to the user [39].

The token hardware is designed to be tamper-resistant to deter reverse engineering. When software implementations of the same algorithm (”software tokens”) appeared on the market, public code had been developed by the security community allowing a user to emulate RSA SecurID in software, but only if they have access to a current RSA SecurID code, and the original 64-bit RSA SecurID seed file introduced to the server. Later, the 128-bit RSA SecurID algorithm was published as part of an open source library. In the RSA SecurID authentication scheme, the seed record is the secret key used to generate Time based one-time passwords (TOTP). Newer versions also feature a USB connector, which allows the token to be used as a smart card-like device for securely storing certificates. Some popular forms of security tokens are shown in Fig. 3.1 [40].

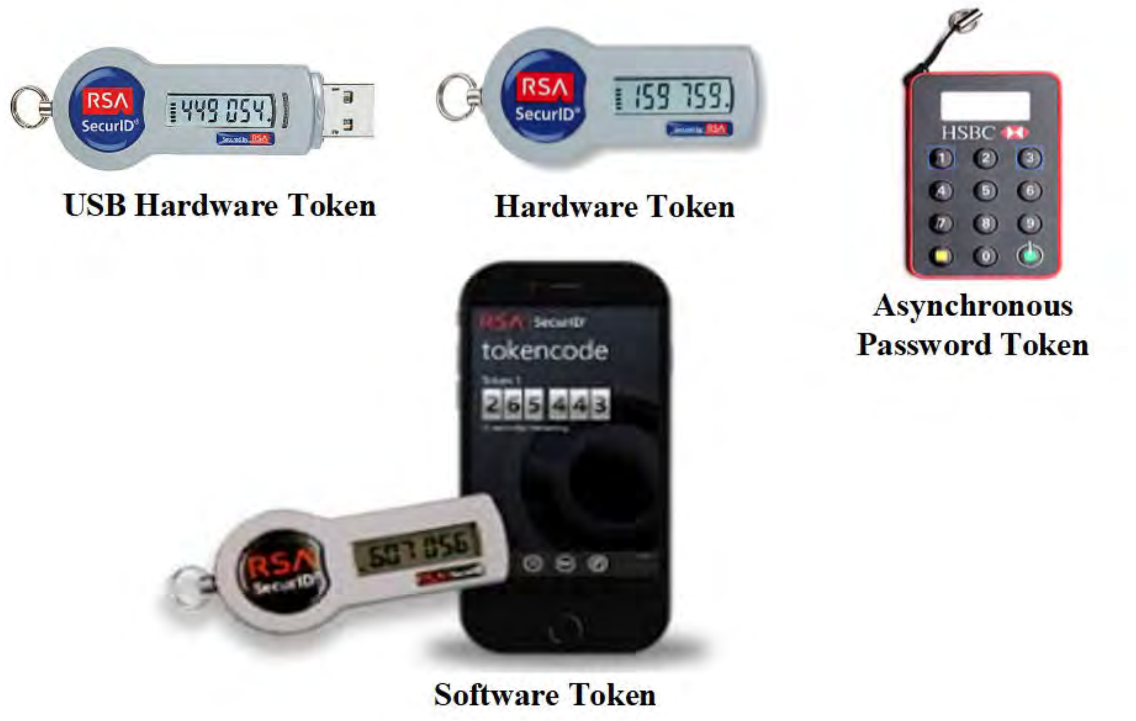


Figure 3.1: Hardware and Software Tokens

A user authenticating to a network resource(Such as, a dial-in server or a firewall) needs to enter both a Personal Identification Number(PIN) and the number being displayed at that moment on their RSA SecurID token. Though increasingly rare, some systems using RSA SecurID disregard PIN implementation altogether, and rely on password/RSA SecurID code combinations. The server, which also has a real-time clock and a database of associated seed records, authenticates a user by computing the number the token is supposed to be showing at that moment and checking this against what the user entered [16].

3.3 User Attribute Aware Authentication Score (Factor-3)

Authentication score is a cumulative weighted summation of authentication parameters. Each parameter weight is set depending on its importance during the authentication. The cumulative summation of the authentication parameters is calculated out of a hundred. While assigning initial weight values for different groups of parameters, we need to make some smart guesses. These weights and even the threshold of authentication score are not concrete but rather adjustable according to the security needs. Authentication score calculation parameters are divided into four groups, such as a) machine bound parameters, b) network bound parameters, c) location bound parameters, and d) learned parameters.

(a) **Machine Bound Parameters**

(i) Browser cookie

(ii) Private IP address

(iii) MAC address

(iv) Hostname

(v) Operating system name

(vi) Operating system version

(vii) Operating system type (32/64 bit)

(viii) Operating system username

$$HostTotal = \sum_{i=1}^{h} HostParams \tag{3.1}$$

For a new authentication request from the client, these host based parameters, such as browser cookie (10), private IP address (20), MAC address (20), hostname (10), operating system name (10), operating system version (10), operating system type (10), operating system username (10), make a summation of weights upto 100 depending on the parameter match with previous successful logins from trusted machines.

(b) **Network Bound Parameters**

(i) Public IP address

(ii) DNS server address

(iii) Gateway address

$$NetTotal = \sum_{i=1}^{n} NetworkParams \tag{3.2}$$

These network based parameters, such as public IP address (40), DNS server address (30), gateway address (30), make a summation of weights upto 100, depending on the match with previous successful logins from trusted machines.

(c) **Location Bound Parameters**

(i) Current time

(ii) Time zone

(iii) Latitude

(iv) Longitude

(v) Country

(vi) City

$$LocTotal = \sum_{i=1}^{l} LocationParams \tag{3.3}$$

These location based parameters, such as current time(30), time zone (20), latitude (5), longitude (5), country (30), city (10) make a summation of weights upto 100, depending on the match with previously successful login requests.

(d) **Learned Parameters**

(i) Average Login Attempts

(ii) Average Login Failure Rate

(iii) Average Successful Login Rate

(iv) Login Time Range

$$LearnTotal = \sum_{i=1}^{s} LearnedParams \tag{3.4}$$

Learned parameters, such as average login attempts (30), average login failure rate (30), average successful login rate (30), login time range (10), make a summation of weights upto 100, when all of them are within the upper bound of previously successful login requests. Here today's value of those four learned parameters are compared with last three (approximately) months averages. And between those two values, maximum of five percent deviation is allowed.

Equation 3.1, 3.2, 3.3 and 3.4 show the calculation process of HostSum, NetSum, LocSum and LearnedSum, which are the summation of host , network, location and learned parameters weights respectively. For a new client authentication request, cloud server matches each parameter value of the client with previously successful login values. Change of a dynamic parameter value in a new authentication request, sets its weight to zero for this request. We have rationally set the weights of these host/machine bound parameters to reflect its importance in the authentication process. These learned parameters cannot be used with full potential for new cloud users. A new cloud user has only one authentication history after the signup with the client.

$$AuthScore = \sum W_{\text{h}} * HostTotal + W_{\text{n}} * NetTotal + W_{\text{l}} * LocTotal + W_{\text{h}} * LearnTotal \quad (3.5)$$

Equation 3.5 is used to calculate the cumulative authentication score. It consists of mainly two parts: dynamic and learned parameters. Dynamic parameters consists of host, network and location bound parameters that make fifty percent of the authentication score. Learned parameters make the other fifty percent of the authentication score. Learned parameters carry half of the total weight because they remain useful disregarding the change of host, location and network of the client.

3.4 Authentication Process

(a) **Signup**

At first, user registers with the cloud server from mobile device/desktop with basic details such as name, email, userID, password, city, country, secret question, etc. While providing password the user has to follow the password policy imposed by the cloud server. By submitting the input form with correct data user completes the basic signup process.

(b) **Deployment: Software Token and PAP installation**

User gets the PAP agent and software token from cloud server download page. Then the user installs the PAP and software token. Steps and guidelines of installing these softwares

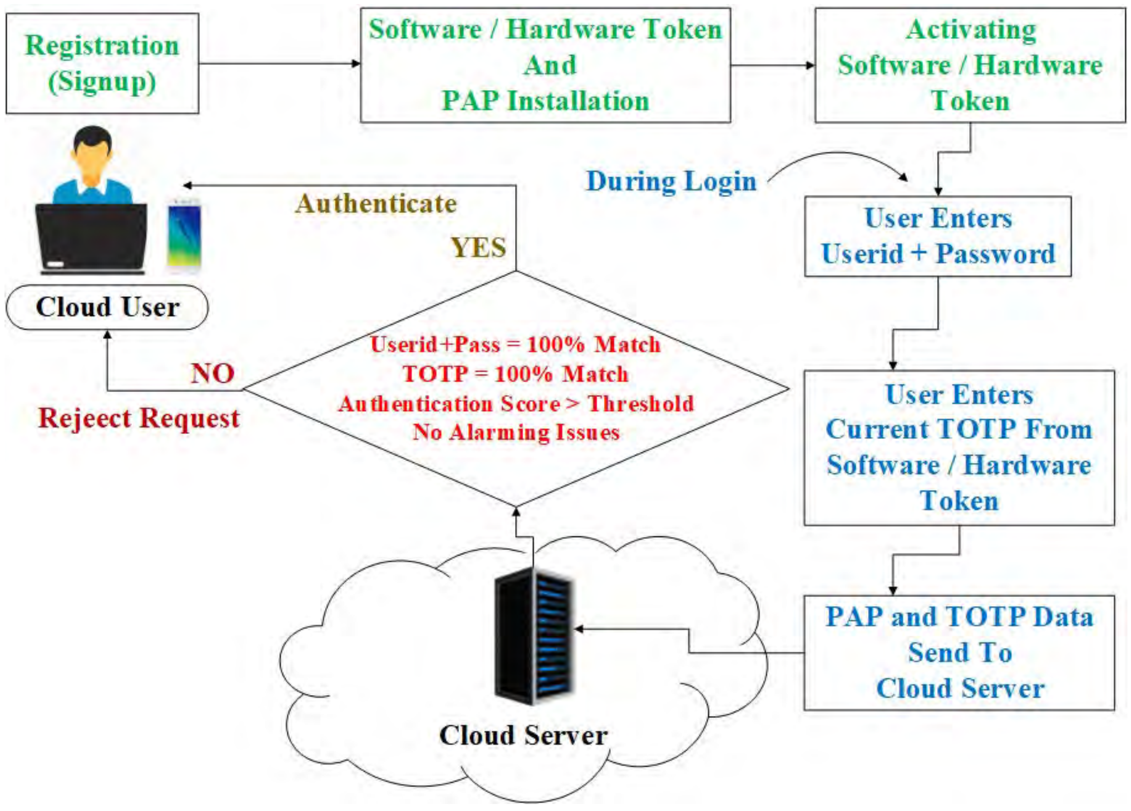

Registration (Signup)
Software / Hardware Token And PAP Installation
Activating Software / Hardware Token
During Login
User Enters Userid + Password
Authenticate
YES
Cloud User
NO
Rejeect Request
Userid+Pass = 100% Match
TOTP = 100% Match
Authentication Score > Threshold
No Alarming Issues
User Enters Current TOTP From Software / Hardware Token
PAP and TOTP Data Send To Cloud Server
Cloud Server

Figure 3.2: Authentication Flow of our Proposed MFA framework.

are provided in a manual. Software token requires activation. Client has to import a given software token file into the software. Once software/hardware token is activated then it is ready to generate TOTP.

(c) **Data collection**

This step facilitates cloud server with necessary data which are required to authenticate a client. This includes collection of software/hadrware token generated TOTP and a list of authentication score calculation parameters collected by the PAP agent. We have used java codes to collect those parameters from clients device (mobile/desktop) and send them to the cloud server. This very light profile acquisition agent does not run on the background all the time, thus does not consume device resources such as CPU, memory etc. This agent is only runs during the time of client authentication request and quits itself automatically just after that.

(d) **Authentication and Rejection**

Firstly, cloud sever first checks for a correct userID-password. A full match is required for it. Secondly, cloud server checks for a correct TOTP from the client. A full match is required for that too. Finally, cloud server calculates an authentication score from the parameters provided by the PAP agent and learned parameters of that client from its knowledge base. Data provided by the PAP agent are mostly machine , network , and location bound parameters. These parameters are only useful when the client tries to login from the same machine, location and the same network. If the calculated authentication score of that client is greater than a predefined threshold value (e.g., 75 out of 100), and there is no such alarming scenarios (Discussed in section 5.2) then the user is authenticated as it shown in the Fig. 3.2. Otherwise, the user is rejected and the failed attempt count of the user is increased by one. Here, we increase the failed attempt count by one only when the userid-password and TOTP are correct. Otherwise, a malicious attacker who knows one of the authentication factor, may make the user and the device blocked by too many failed attempts.

Chapter 4

Implementations

While implementing our proposed multi-factor authentication framework for cloud based systems, we need to consider three parts. First part is to implement a strong password policy which will make the initial userID-password very tough to break for the attackers. The second part is to use a strong security token algorithm that will generate a Time-Based One-Time Password (TOTP) every minute at the client side. This security token works offline with the cloud server. The third part is the installation of PAP agent at the client side, that provides client's current host, location and network data on demand to the cloud server during the authentication process. In this chapter, we describe the implementation details and use of each of the three factors of our MFA framework. We then show, how each factor is combined to give a strong MFA strategy for cloud based systems.

4.1 Software/Hadrware Token

We have chosen RSA SecurID as the second factor for our proposed multi-factor authentication framework. RSA SecurID is a commercial product of EMC corporation. This product has been used all over the world as part of 2FA, mostly by financial organizations, such as banks that require a strong authentication system to secure their data and users. This RSA SecurID product can be used by taking a bulk license. Both software and hardware version of this is available in the market. For our testing purposes, we have chosen software token version of RSA SecurID because it is easier to understand and there is no need to buy hardware token. We can use this too for testing our MFA authentication mechanism by signing up for a thirty day trial at here https://register.testdrive.tryrsasecurid.com. We can make use of RSA securID software token very easily by installing the software token application and importing a software token into it. Once the token is imported into the application, it is ready for use and a TOTP will be generated and shown in the display screen of the token. The validity of each TOTP is 60 seconds. However, this can be customized to 30 seconds.

4.2 PAP Implementation

We have developed a Profile Acquisition Program (PAP) agent with Java programming language. This PAP agent will make use of the machine bound parameters (e.g., Browser Cookie, IP address, MAC address, OS name, OS version, OS type (32/64 bit), Hostname, OS username, etc.), network based parameters (e.g., Public IP address, DNS server address, Gateway address, etc.), location bound parameters (e.g., Current time, Time zone, Latitude, Longitude, Country, City, Usual login time range, etc.). Once PAP collects these data from client site, these are sent to the cloud server for further processing with some other learned parameters in order to calculate the authentication score. This application will be installed in the client machine (desktop/mobile) and never runs on the background all the time. During the authentication request from the client, this application collects the list of authentication parameters and put them in single a file, encrypt the file (AES-128) and send to the cloud server through an open socket. Upon sending, the file will be deleted from the client machine. Even the server deletes the file after updating the knowledge base of the client.

4.3 Data Collection

In this step, cloud sever receives necessary credentials and data which are required to authenticate a valid client. This includes receiving and matching a valid userID/password, software/hadrware token generated TOTP and a list of authentication score calculation parameters collected by PAP agent installed at the client device. We have used java code to collect those parameters from clients device (mobile/desktop) and to send to the cloud server. One entity is cloud server class which runs and listens for client request in a socket. Other one is the client class which calls a PAP class in the same program to get the values of the list of authentication parameters. Upon getting PAP data, the client opens a socket with the server and transfers those credentials and data to the cloud server through server socket. Cloud server then processes those PAP data for machine, location, network bound and learned parameters to calculate the authentication score.

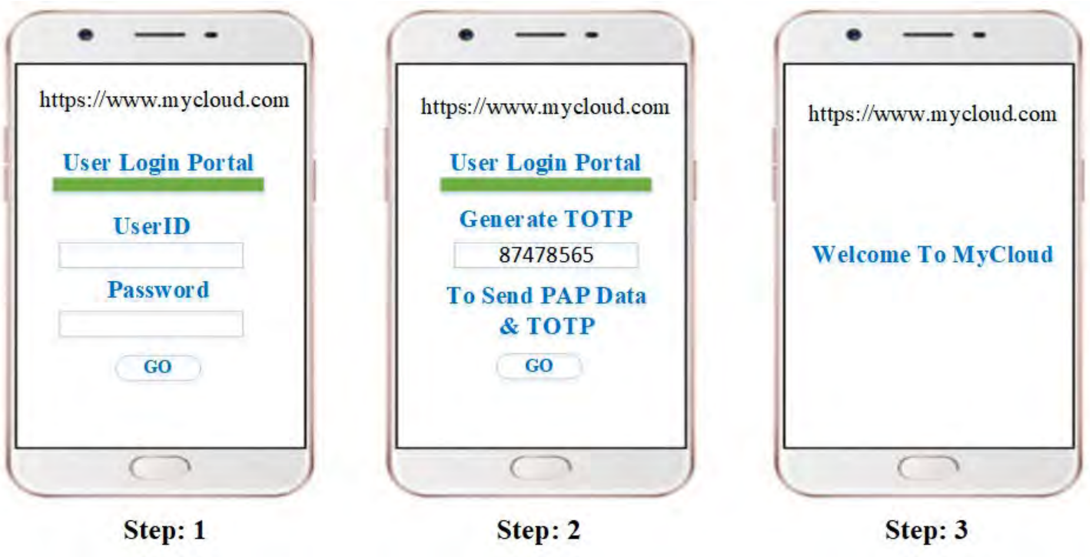


Figure 4.1: Data Collection Process from Smartphone

Fig. 4.1 depicts the login and data collection screen for the cloud. While the client clicks on the ”Generate TOTP” button, an eight digit TOTP value is generated and shown on the screen. At that same time, this data collection agent runs and collects those authentication parameters values. When the client presses the ”Go” button, both software token generated TOTP value and data collection agent collected authentication parameters values are sent to the cloud. There is no plain text data exchange between the cloud server and the client. We have made use of AES-128 encryption scheme to secure the data transmission between the cloud sever and client. If the TOTP provided by the client matches the TOTP on cloud server side and the authentication score exceeds the predefined threshold value, then the client is successfully logged in to the system.

4.4 Knowledge Base

With successful and unsuccessful authentication of the client, a knowledge base has been developed at the cloud server. This knowledge base keeps track of dynamic and learned parameters, blocked and trusted machines, of each of its clients. A machine might get blocked, when authentication is failed in alarming scenarios. We keep record of it in a separate database table keeping userid as the PK. Further authentication request and signup are disallowed from blocked machines.

With successful login, a new machine can become trusted and we keep records of it in a separate database table. If a trusted machine fails in learned parameter check, it becomes untrusted machine again. Client remove trusted and blocked machines from his cloud setting page in the cloud server.

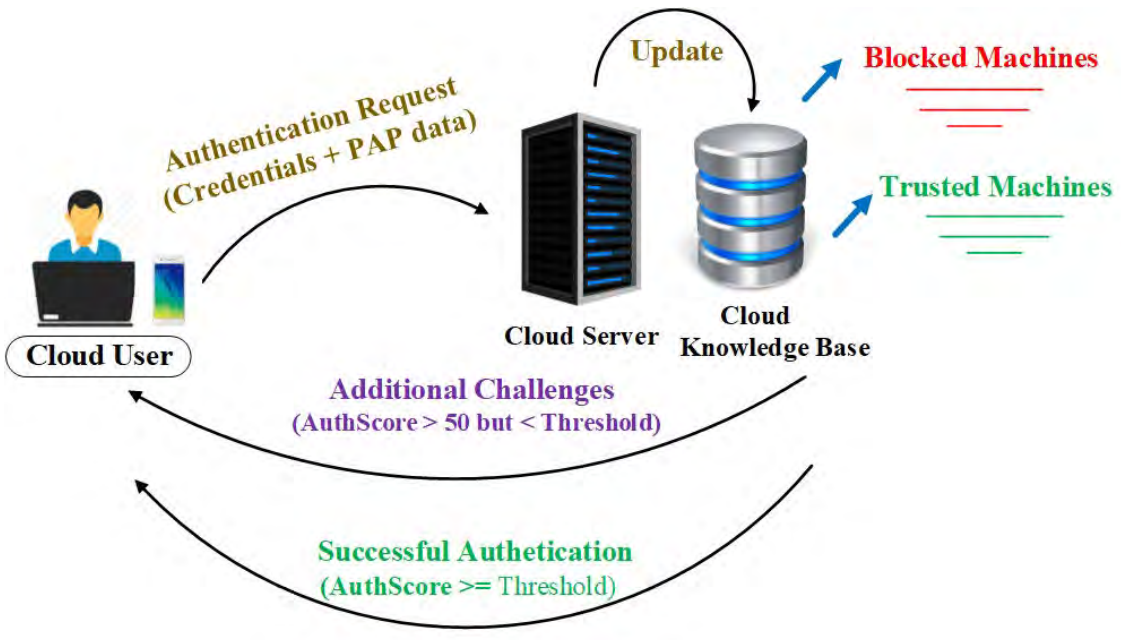


Figure 4.2: Knowledge Base for Blocking and Trusted Machines.

4.5 Variation of Authentication Score

To calculate the score, we have to have a clear understanding of authentication parameters which are used to calculate the authentication score. As we said earlier, we have categorised the list of authentication parameters into four categories. We have already shown the score calculation process with equations in Chapter 3. Let us make a few tweaks in the weights of authentication parameter set to get a better authentication result.

In our simulation, the weights of those four types of parameters are used as follows-

Weight of host based parameters (W_h) = 0.0-0.30 (0-30%)

Weight of network based parameters (W_n) = 0.0-0.10 (0-10%)

Weight of location based parameters (W_l) = 0.0-0.10 (0-10%)

Weight of Learned parameters (W_s) = 0.0-0.50 (0-50%)

We have made a dataset by varying clients machine, network and location parameters. Some portion of this dataset is shown in the Appendix. Learned parameters are updated with each authentication requests dynamic parameter set from the client. Fig. 4.3 shows decision making process of authentication using adapted weights for parameter set. This weighted summation has to beyond a predefined threshold value T, and only then the user will be authenticated (Y=1). Y=0 indicates authentication request is unsuccessful. And from that unsuccessful list, additional challenges are imposed on the client if the authentication score from new request is beyond fifty but less than the threshold value. Below fifty (Failed in learned parameter check) authentication score are discarded and failed attempt count will be increased by one for that client.

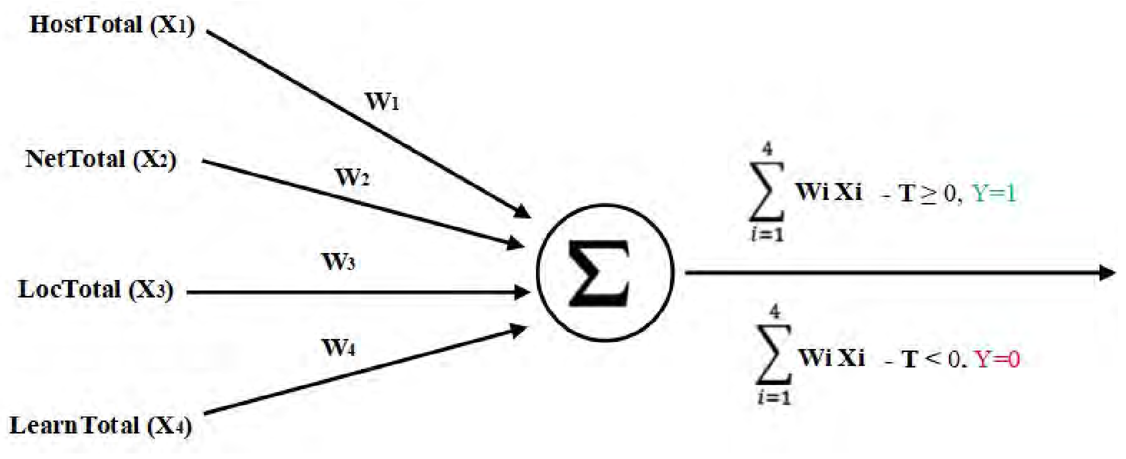

HostTotal (X1)
NetTotal (X2)
LocTotal (X3)
LearnTotal (X4)
W1
W2
W3
W4
Σ
$\sum_{i=1}^{4}$ Wi Xi - T ≥ 0, Y=1
$\sum_{i=1}^{4}$ Wi Xi - T < 0, Y=0

Figure 4.3: Authentication Decision Making Using Perceptron Learning Algorithm.

$$W_{\mathrm{i}} = W_{\mathrm{i}} + \Delta W_{\mathrm{i}} \tag{4.1}$$

$$\Delta W_{\mathrm{i}} = 0.01(Bias) \tag{4.2}$$

Equation 4.1 and 4.2 weight adaptation with bias of 0.01. This bias is one percent of the total weight. Table 4.1 shows adapted weights (W1, W2, W3, W4) of authentication parameter set after processing the dataset with single layer perceptron learning algorithm. For host, network and location parameters “U”and “M”denotes unmatched and matched parameters respectively. We call it matched host or location or network, if at least eighty percent of the current authentication parameter set in PAP provided data for each category matches with any of previous successful login data. If the matched percentage is less than eighty percent in each authentication parameter category then we name it unmatched. Here, we discard the percentage of unmatched parameters in weighted measurements. A parameter set either of host, network, or location will get zero weight

Table 4.1: Weight adaptation of Each Parameter Set Using Perceptron Learning Algorithm.

Authentication Parameters				Learned Weight				Output
Host	Network	Location	Learned	W1	W2	W3	W4	Y
U	U	U	M	0.00	0.00	0.00	0.45	0
U	U	M	M	0.00	0.00	0.05	0.45	0
U	M	U	M	0.00	0.05	0.00	0.45	0
U	M	M	M	0.00	0.04	0.05	0.50	0
M	U	U	M	0.25	0.00	0.00	0.45	0
M	U	M	M	0.30	0.00	0.10	0.50	1
M	M	U	M	0.27	0.08	0.00	0.45	1
M	M	M	M	0.25	0.09	0.10	0.50	1

if its parameters total (HostTotal, NetTotal, LocTotal) falls below eighty percent.

Here, we are considering only matched learned parameters, because for first three learned parameters (Average Login Attempts, Average Login Failure Rate, Average Successful Login Rate) mismatch there will be an instant authentication rejection (Tentative client machine enters into blocked machines list). If there is a mismatch in last learned parameter (Login Time Range), additional test will be imposed.

For Y=1 output scenarios in table 4.1 client will be authenticated, but if Y=0 then additional challenges will be imposed for the client. If the client tries to login from a different device then location and network bound parameters can get upto half of their highest weight i.e 0.05 out of 0.01. 100% weight from the network and location bound parameters are applicable only when the current client device is already in the trusted machines list. The reason is, if the network and host portion starts with the maximum weight then an attacker may get the benefit in authentication score, residing in the same network and location as the client does. We have also tried varying the fifty-fifty proportions of these dynamic and learned parameters. Putting a weight more than fifty on dynamic parameters side, makes the client machine, location and network dependent. And putting a weight more than fifty percent on the learned parameters, makes the importance of clients mobility adaptation on the authentication of the client very trivial.

4.5.1 Machine Bound Parameters

We have used eight machine bound parameters such as, browser cookie data, private ip address, MAC address, OS name, OS version, OS type (32/64 bit), OS username, hostname, etc. These parameters are clearly bound to a specific machine and they have a little worth during authentication when a valid client tries to login to the cloud system from a different machine. But a perfect match of these parameters in a new authentication request, will make it easy for the cloud server to authenticate the client. Here, operating system name, operating system version, operating system type (32/64 bit), may remain the same for different clients because of same OS and the same version installation. Even the hostname of the device can be similar for two different clients. However, these eight machine parameters helps to identify trustworthy machines for a client.

4.5.1.1 Change of Client Host Machine

This change may occur sometimes when a client accesses the cloud service from a shared device or from a newly purchased device or from a device of his friends or family. But the key idea is that, we do not want to keep our MFA framework machine bound. These parameters are only useful to validate a client when the client logins from a machine from which the client had a successful login history. While login from a new machine, the client has to go through additional checks to prove its validity. Once a machine is validated after a successful login it will remain in the trustworthy machine list of that specific client. Let us consider, we have given a score of thirty to all the host-based parameters. This weight is divided among all the host based parameters.

4.5.2 Network Bound parameters

Parameters such as public IP address, DNS server address and gateway address are clearly network bound. Client can change his/her usual login machine within the same network. For example, a valid client may try to login to the cloud server from a WiFi router using his laptop, desktop and smartphone. Here, machine bound parameters will change but network bound parameters will remain the same.

4.5.2.1 Change of Client Network

This is another dynamic parameter. Client may try to login from different networks with different configurations where Public IP address, DNS server address, and the gateway address vary. If the client logins from the same network of any previously successful login then this parameter can be used with full potential to make a good authentication score. In contrary, a client tries to login from a unfamiliar network may lead the weighted sum to zero from network parameters. We call it a new network if there is less than eighty percent match with a previous successful login.

4.5.3 Location Bound Parameters

We have used six location bound parameters and they are a) current time, b) time zone, c) latitude, d) longitude, e) country and f) city. These location bound parameters are only helpful when the client tries to login from the same location like his workplace, house, etc. These parameters may change partially, for example, when when the client tries to login from the same country with different time zone, from a different city within the same country.

4.5.3.1 Change of Client Location

Client location is the most frequently changing parameter while accessing cloud service by the client. When the client accesses cloud service frequently from some certain locations these parameters helps to make a good authentication score. This whole value may drop near to zero if the current location of the client is absolutely new. We call it a new location if there is less than eighty percent match with a previous successful login.

4.5.3.2 Change of Client Host Machine, Location and Network

This could be the situation of most sensitiveness where current host machine, location and network all are new. Among the authentication score parameters, machine bound, location bound and network bound parameters consists of fifty percent of the total authentication score. Other fifty percent comes from the learned parameters. Even though learned parameters make up a weighted sum of fifty for the client but the client cannot login since then authentication score will be less than the threshold value. Additional challenges are asked here and the client has to answer them correctly.

4.5.4 Learned Parameters

There are few learned parameters, which play a pivotal role during the authentication process. These parameters do not change while a valid client tries to login from a different machine. But the cloud sever can detect suspicious request by checking the deviation of parameters value in the new request. These parameters exhibit clients habitual login characteristics. Usually, a valid client is supposed to incline to it. If we consider the cons, then these learned parameters will not be useful for a new client with no login history. But as the time progresses, the usefulness of these parameters also rises.

a) Average Login Attempts

Average login attempts is a crucial parameter. If we consider number of login attempts for a valid user, it could be five to twenty (more or less). But an attacker/robot may try to impersonate a valid client by trying millions of combinations of credentials within a minute or so to get a match. But in our MFA approach, we have put a limit of maximum five wrong attempts for first two credentials at a time. After that there will be a cut off period of one hour when the client will not get any chance to login from the same machine. These fake login attempts will not hamper valid clients and does not have any impact on the authentication score. Because a valid clients may try to login from a different machine and may not be aware of the fact that many fake login attempts has been made for forge their credentials. The idea is to prevent the attackers, not the legitimate clients.

b) Average Login Failure Rate

One of the learned parameters is average Login fail rate. This fail rate will be resulted by matched userID/password but a wrong TOTP or a low authentication score. To be as a failed attempt at least userID-password has to be correct. For a valid cloud client, this would be very less because a valid client will not enter a wrong userID-password and wrong TOTP so many times. A valid user is likely to enter wrong credentials only for a few times that should be within a certain tolerance level. On the other hand, an attacker may try to authenticate itself to the cloud system using numerous userID-password and TOTP. So the failure rate for the attackers machine will be very high.

c) Average Successful Login Rate

Average successful login rate will be mostly equal with the number of login attempts for a valid client. Because, if it is a valid client then none of login request is going to be rejected since there is no such deviation in login parameters. However, this parameter is zero for an attacker because none of the request from the attacker is going to be accepted. These learned parameters have been calculated at the server side by keeping clients records for at least a month or even a year.

d) Login Time Range

Among the learned parameters, login time range may vary significantly with the change of time zone. For an instance, Dhaka is UTC+06:00 and New York is UTC-05:00. So while its noon at Dhaka, its midnight at New York. So a cloud user who usually starts using the cloud service at 9 am in the morning and do not sign in after midnight shall have to adjust with the time zone. So with the time zone change, usual login time range should also change accordingly.

4.6 Solution: Authentication Score Falls Below the Threshold

When the cumulative authentication score is below the threshold for a client, we have to find a way for the client to get pass this barrier with minimal complexity and at the same time, make it impossible for the attackers to break in. Clients are asked to overcome a couple of challenges within a minute-

1. Answer a secret question.
2. Enter alpha-numeric code sent in the mobile message.

Two secret questions answer have been set by each client during the signup process with the cloud vendor where the answer has to be more than a single word. Answering the secret question and acknowledging with correct alpha-numeric code fills in the lagging part of authentication score of the client and allow the client to access its cloud resources. The machine from which this authentication success has happened enters in to the trustworthy machines list of this client. However, these secret questions and alpha-numeric code does not rescue the client from first three learned parameters variation above the tolerance level of five percent.

For a newly signed user, additional challenges are imposed on the first login because of the absence of trusted host or learned parameters at the cloud server side. If the client can login successfully then this machine will be added in the trusted host list. Cloud server will utilize this minimal learned data of the client.

4.7 Adapting Machine Learning

Apart from the rule based approach shown in Section 5, we have utilized machine learning to learn the weights and threshold from the dataset itself. This learning methodology would allow the cloud system to automate the process of authenticating the client with varying authentication parameters.

4.7.1 Training and Testing

In the linear regression model, dependent variable y is considered continuous, whereas in logistic regression it is categorical, i.e., discrete. In application, the former (Linear) is used in continuous regression settings while the latter is used for binary classification or multi-class classification (where it is called multinomial logistic regression). Instead of continuous Y in liner regression, logistic one is regressing for the probability of a categorical outcome. In simplest form, this means that we're considering just one outcome variable and two states of that variable, either 0 or 1.

Since our dataset shown in appendix has discrete host, network, location and learned parameters due to regularly changed user parameters provided by PAP. Due to this dynamic nature of users everchanging parameters, logistic regression would be more useful to classify the dataset output into 0 or 1. Our dataset has a probable y, depending on the outcome of each parameter set (host, network, location and learned). Our dataset has 1000 datapoints for only one user, which we have got by varying different user parameters. Logistic regression (classifier) has been used in the training and testing-

$$\begin{aligned}&LogisticRegression(C = 1.0, class_weight = None, dual = False, fit_intercept = True,\\&intercept_scaling = 1, max_iter = 100, multi_class = ovr, n_jobs = 1, penalty = l2,\\&random_state = None, solver = liblinear, tol = 0.0001, verbose = 0, warm_start = False)\end{aligned} \tag{4.3}$$

4.7.2 K-fold Cross Validation

That k-fold cross validation is a procedure used to estimate the skill of the model on new data. Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample (In our case only 1000). K-fold cross validation (K=5) were performed to determine the weight vectors and the threshold. For Each K, cross validation weights are learned from the

Table 4.2: K-Fold Cross Validation Result

K=1	W1	W2	W3	W4
Cross Validation Weight	0. 03668886	0.01866903	0.02600933	0.02809699
Cross Validation Threshold	6.65478239			
Cross Validation F1-Score	0.7619047619047619			
K=2	W1	W2	W3	W4
Cross Validation Weight	0.03710081	0.01884012	0.02571508	0.02655423
Cross Validation Threshold	6.56047825			
Cross Validation F1-Score	0.779874213836478			
K=3	W1	W2	W3	W4
Cross Validation Weight	0. 0.0369362	0.02050989	0.02366929	0.02861264
Cross Validation Threshold	6.57424976			
Cross Validation F1-Score	0.8592592592592593			
K=4	W1	W2	W3	W4
Cross Validation Weight	0.032209756	0.01893076	0.02918359	0.02569319
Cross Validation Threshold	6.25509597			
Cross Validation F1-Score	0.7333333333333333			
K=5	W1	W2	W3	W4
Cross Validation Weight	0.03220928	0.01784176	0.02686732	0.03664717
Cross Validation Threshold	6.99058635			
Cross Validation F1-Score	0.6804123711340206			

dataset (Shown in the appendix). Table 4.2 shows the results from the K-fold cross validation.

We have used F1-score as performance metric. In statistical analysis of binary classification, the F1 score (also F-score or F-measure) is a measure of a test's accuracy. It considers both the precision p and the recall r of the test to compute the score: p is the number of correct positive results divided by the number of all positive results returned by the classifier, and r is the number of correct positive results divided by the number of all relevant samples (all samples that should have been identified as positive). The F1 score is the harmonic average of the precision and recall, where an F1 score reaches its best value at 1 (perfect precision and recall) and worst at 0. In our testing, K=3 has the best score of 0.8592.

Chapter 5

Experiments and Results

In this chapter, we present our results and analysis. There are quite a few things to test our proposed MFA system robustness. The first thing is to justify, how our proposed MFA framework responds with different sorts of security attacks.

5.1 Performance against attack scenarios

There are several kinds of authentication attacks on which we can check our proposed MFA frameworks performance and evaluate whether this system can overcome those attacks. Let us start with credential discovery attacks, such as brute force attack, dictionary attack, video recording attack, etc.

5.1.1 Credential Discovery Attack (e.g., Brute Force, Dictionary, etc.)

In our proposed MFA framework, first factor is userID-password, and we all know naive passwords are prone to credential discovery attacks, such as brute force attack, dictionary attack, video recording attack, etc. But we can restrict these sort attacks by imposing a strong password policy (Discussed in section 4.1), which makes it really hard to crack for brute force attackers. However, our second factor hardware/software token generated TOTP (Time-based One-Time Password), which is valid for a minute. Each TOTP is completely unrelated to one another. This TOTP cannot be generated correctly by attackers using discovery attacks (e.g., brute force attacks) until the attacker is aware of the seed of that specific token and the underlying algorithm. Our third factor is authentication score, calculated at server side from PAP data sent by the client during authentication and it is not prone to these discovery attacks. Client's host, location and network bound parameters, are not discoverable until the client machine is compromised. Moreover, heavyweight assigned for learned parameters will not allow the attackers to get a score beyond the authentication threshold. Furthermore, the authentication score calculation is done at the sever side and the score calculation formula is only known to the cloud server. Dictionary based and video recording attacks are only useful against password but they are useless to TOTP and authentication score. So it can be clearly concluded that, credential discovery attacks are not successful in our proposed system.

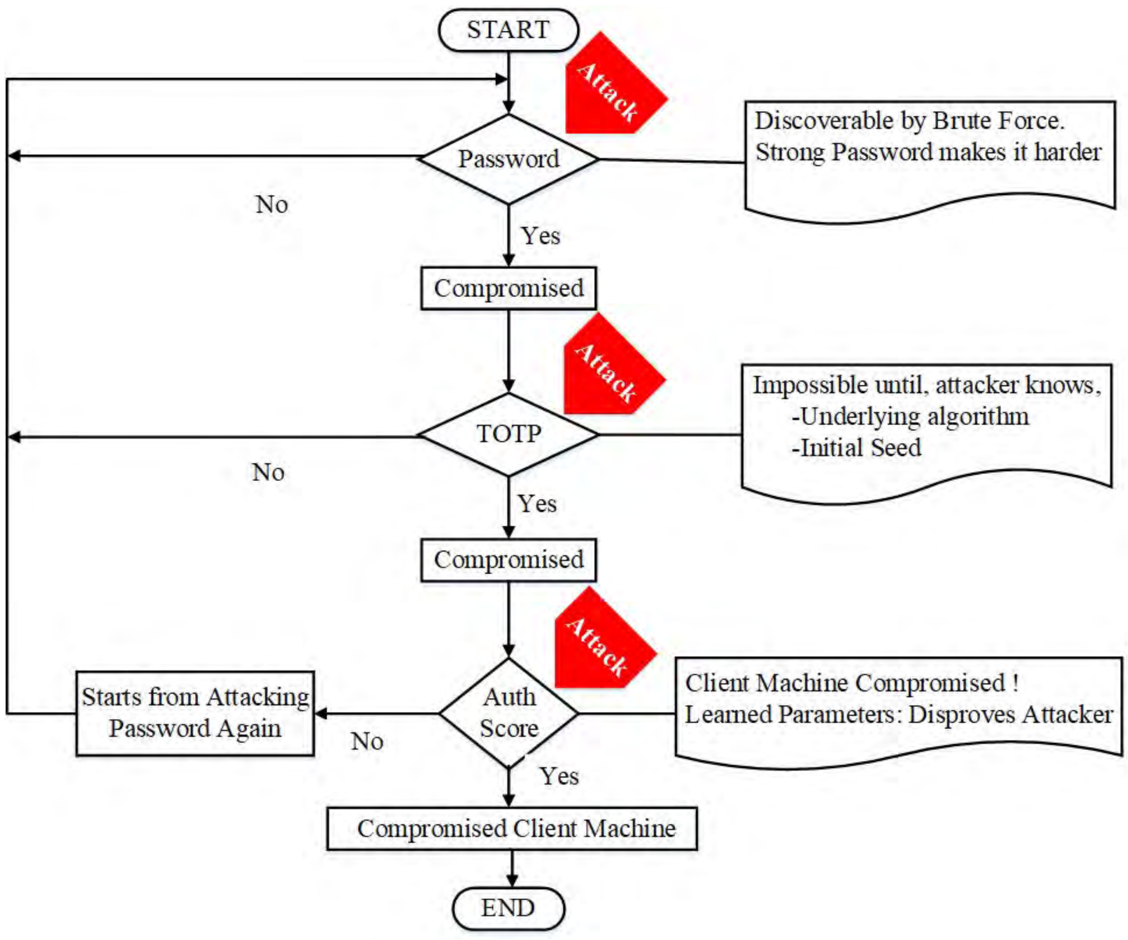

START
Attack
Password
Discoverable by Brute Force.
Strong Password makes it harder
No
Yes
Compromised
Attack
TOTP
Impossible until, attacker knows,
-Underlying algorithm
-Initial Seed
No
Yes
Compromised
Attack
Auth Score
Client Machine Compromised !
Learned Parameters: Disproves Attacker
Starts from Attacking Password Again
No
Yes
Compromised Client Machine
END

Figure 5.1: Brute Force Attack: Response of our Proposed MFA.

We explain further in the following discussion, that why our proposed cloud authentication system will be able to protect against credential discovery attacks.

(i) Password

- Discoverable by brute force attack

- Strong password policy enforcement makes it harder for attackers
- If compromised, it can be misused until the password is changed

(ii) TOTP : Not calculable by brute force, until-

- The underlying algorithm is compromised and
- Token seed is stolen by the attacker

(iii) Authentication Score Parameters

- Dynamic Parameters are known to attackers: Compromised client machine
- Learned Parameters : Disproves the attacker

5.1.2 Spoofing Attack

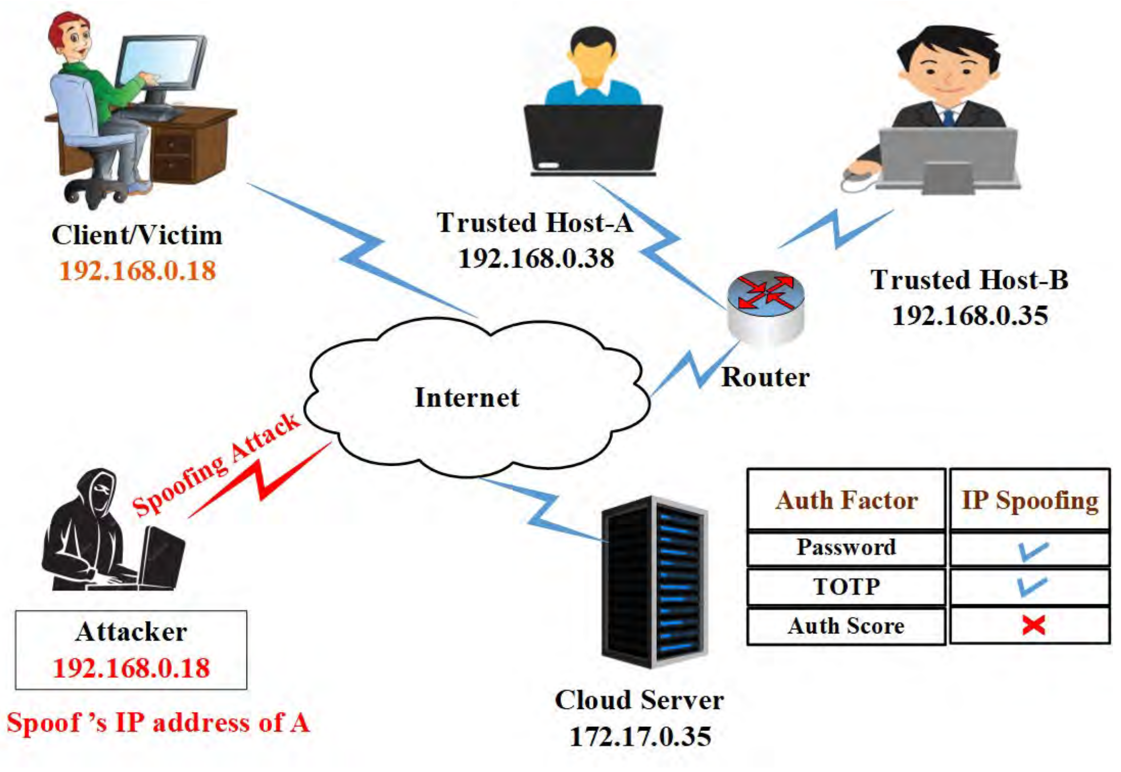
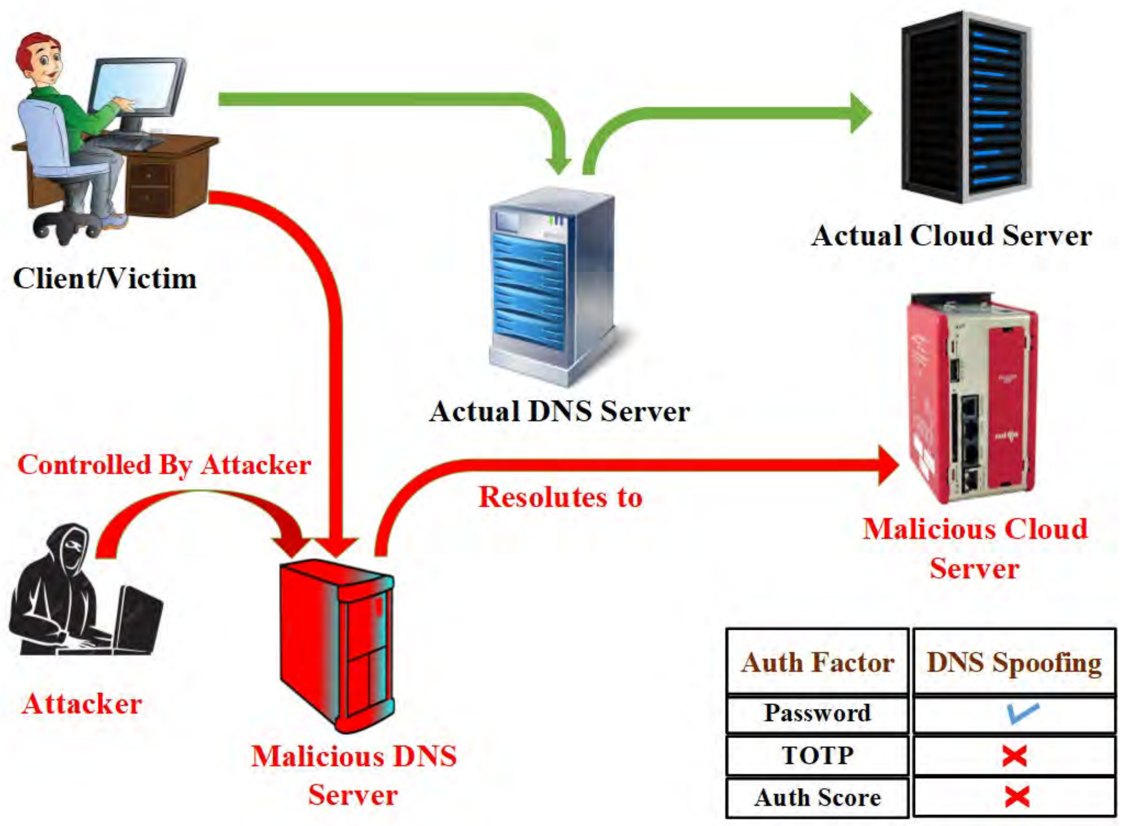


Figure 5.2: IP Spoofing Attack: Response of our Proposed MFA.

Spoofing refers to the act of fooling, presenting a false information in a credible way. It is a kind of impersonating attack, where an attacker tries to introduce itself as valid cloud server to the clients. Thus, this attack may trick the client into doing something that they might not ordinarily do. For example, during email address verification of client accounts, an attacker may send an email to the client with a hyperlink (in the email body) to verify the account which actually installs a malware in the client device that eventually steals sensitive data from the client machine and cause harm to the client in many ways.

5.1.2.1 IP Spoofing

One of the most common form of this attack is IP spoofing. In this attack, the attacker modifies the IP packet header with a forged (spoofed) source IP address. After that, it seems the transmission are originating from a trusted client machine. Only spoofing the IP might not get the attacker going, because of other authentication score parameters. And to know all other host, location and network based parameters, the client machine has to be compromised.



Auth Factor	DNS Spoofing
Password	✓
TOTP	✗
Auth Score	✗

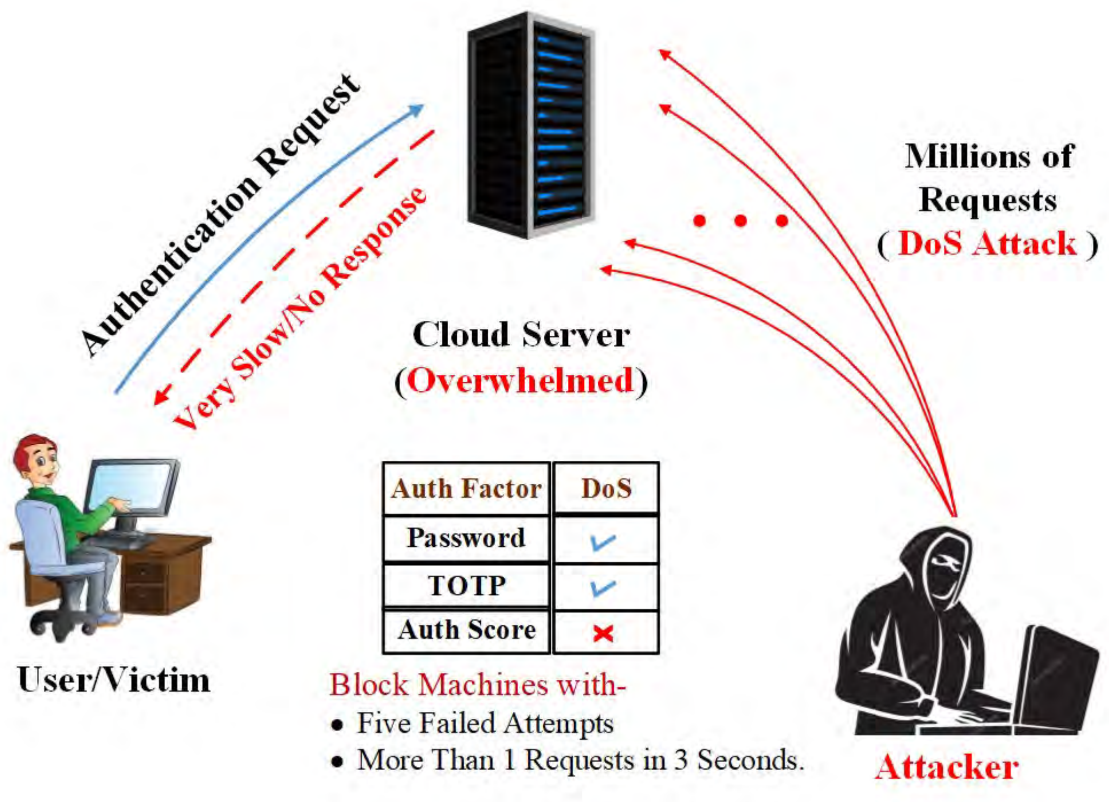
Figure 5.3: DNS Spoofing Attack: Response of our proposed MFA.

5.1.2.2 DNS Spoofing

As we all know, a DNS server translates human-readable names like ”mycloud.com”, into corresponding IP addresses. When DNS responses are forged, the victim will unknowingly connect to a different cloud server than the one it intended to reach. Here the attacker performs man-in-the-middle (MITM) attack by impersonating the real DNS server. Fig 5.3 shows, how the clients fall victim of malicious DNS server. This attack can be used to transparently capture users login credentials by fake cloud servers. But capturing encrypted PAP data and TOTP from the client does not help the attacker to impersonate as the correct cloud user to the cloud server. Because, TOTP becomes invalid in the vary next minute and encrypted PAP data in the hand of an attacker, clearly of no use. But within one minute of time, until the TOTP gets invalid, the captured TOTP can be reused for malicious purposes if its not securely encrypted.

5.1.3 Denial of Service (DoS) Attack

Using updated antivirus program and strong firewall policy might not be enough to prevent DoS attacks. There are few ways we can prevent the cloud sever from being overwhelmed by fake authentication requests. Ever though, these fake requests are not successful but they can put a huge bulk of load to the cloud server to consume its CPU, memory and network resources which will eventually make the cloud server unable to process valid clients requests.



Auth Factor	DoS
Password	✓
TOTP	✓
Auth Score	✗

Block Machines with-
- Five Failed Attempts
- More Than 1 Requests in 3 Seconds.

Figure 5.4: DoS Attack: Response of our Proposed MFA.

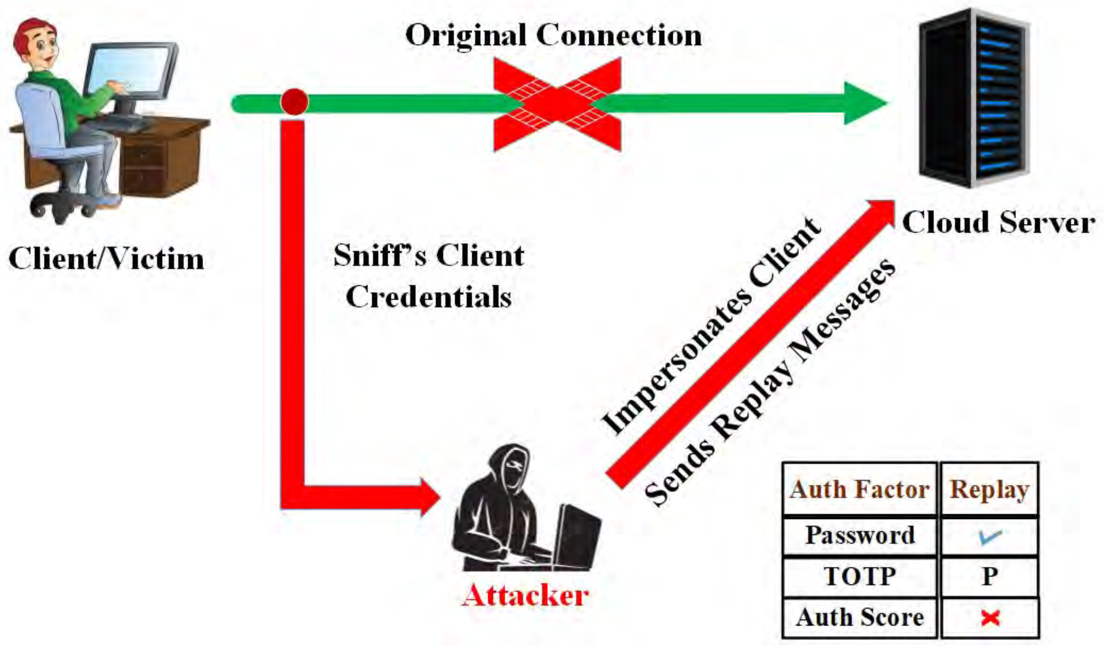
IP spoofing can used to overload the cloud server with traffic in two ways. One way is to flood the cloud server with authentication request packets from multiple spoofed client addresses. This method will overwhelm the cloud server with more data than it can handle. The other method is to spoof the cloud servers IP address and send packets from that address to many different clients on the network like a broadcast. The huge tide of response packets from different clients will overwhelm the cloud server.

To solve this issue, one approach is to block those machines which are unable to get authenticated in five attempts. Our blocking approach is one hour at first round. Then it increases in the form of two hour, six hour in the subsequent rounds. when this sort of failure happens in three subsequent rounds then it will enter into a complete blocked machine list and no further request will be processed from that machine. In that case, one may think why we are not blocking the client itself. Because, attackers may intend to hamper valid clients activity by sending too many incorrect requests to the cloud server, hoping this will block valid clients and valid clients are totally unaware of this activity.

Another approach is to check the number of login attempts. We block those machines which send more than one request in three seconds. An attacker robot or program may make hundreds of login attempts in a minute but a valid human client can never do this. To put username with a password and TOTP, at least three second is needed for a client with a very fast typing speed. But if there is one attempt at every second or even faster, then it sounds suspicious and it is suspicious. This makes us think that there's no human doing this. We shall block those machine which originates more login requests than a human can do.

5.1.4 Replay Attack

Replay attack is a kind of Man-in-the-Middle attack, where the attacker is positioned in between the client and the server. The attacker intercepts communicating data for alteration and replaying with the intention of doing the same work that a valid client is supposed to do. This makes userID-password prone to replay attack. But our second factor software/hardware token generated TOTP is not prone to this attack. Because, even if the attacker intercept the current communication of client with the sever, but replaying it later will result in a mismatch of TOTP and this interception of credentials becomes fruitless.



Auth Factor	Replay
Password	✓
TOTP	P
Auth Score	✗

Figure 5.5: Replay Attack: Response of our Proposed MFA.

Replay attack is only possible, if this can done within its one minute validity. In the Fig. 5.5, 'P' indicates partially possible within one minute of TOTP active time. We can change that TOTP validity time to thirty seconds to enhance the authentication process strength. Authentication score is not also prone to replay attack because, current timestamp of the client is included as part of the PAP data. Replaying it later will result in a rejection of authentication request because of timestamp mismatch. Encrypted authentication score parameters are not also prone to replay attack.

5.1.5 Phishing Attack

Unlike spoofing, phishing is a kind of attack where the attacker tries to trick the client into giving up sensitive information. It may come through as a fake website link, email, call, text message, all to get the client to expose its account details like username, password, credit card number, social security number, birthdate, bank account number, pin etc.

In most of the cases, phishing attack can be prevented by conducting training sessions with mock phishing scenarios. Clients can gain an understanding of phishing attacks via brief documents with pictorial explanations. It would be impossible for the cloud vendor to provide security to clients, if a valid client gives all of his/her login credentials on phished website link or on phished web link given in emails or tell credential related information to fake calls seems

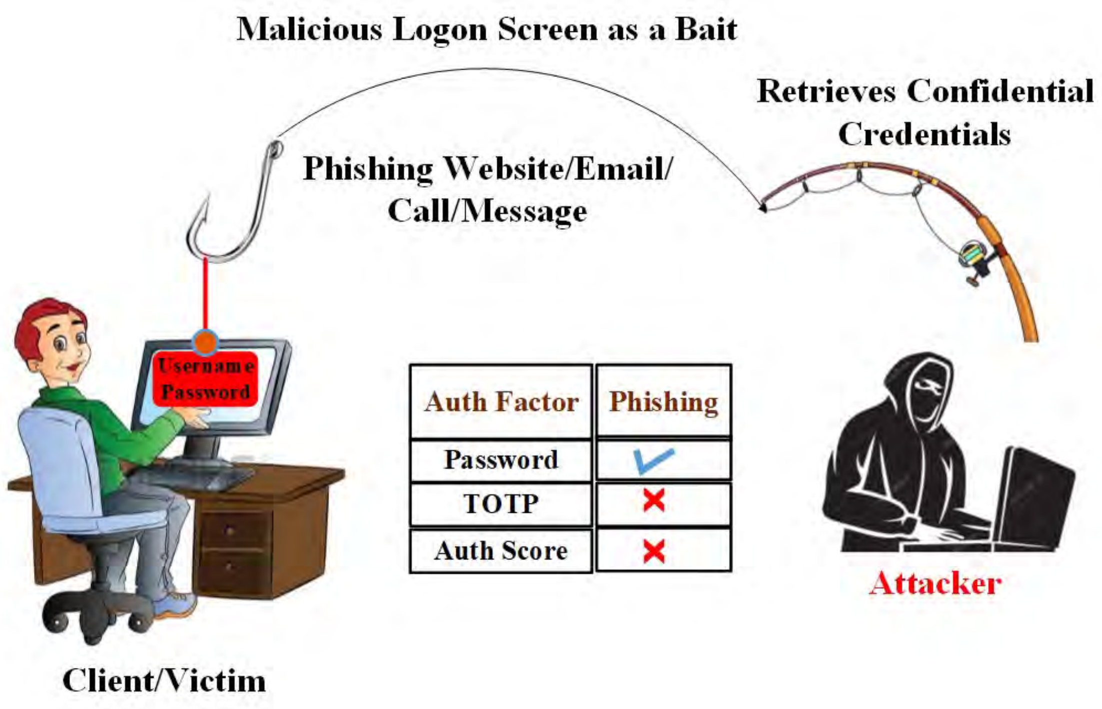

Malicious Logon Screen as a Bait
Retrieves Confidential Credentials
Phishing Website/Email/ Call/Message
Username
Password
Auth Factor | Phishing
Password | ✓
TOTP | ✗
Auth Score | ✗
Attacker
Client/Victim

Figure 5.6: Phishing Attack: Response of our Proposed MFA.

coming from cloud vendors or reply with credentials on fake messages asking for credentials. Our second authentication factor TOTP will stop the attackers for future login but within 1 minute of TOTP validity time the client is under risk. And encrypted authentication score parameters in the hand of attacker is of no use. Moreover, spam filters can be enabled to recognize and prevent emails from suspicious sources from ever reaching the inbox of clients. Up to date system and security patches, firewall, anti-virus can help with the cause.

In Table 5.1, we have shown a comparison of our proposed MFA with others approaches. Liu et al. [35] proposed MACA, a MFA solution for cloud, using password and hybrid user profile. Their user profiling consists of host-based characteristics and network flow-based features. They did not consider clients location features, so their scheme cannot adapt with clients location change. Their scheme is not safe from replay attack. Because, if the clients password and hybrid profile fall in the hand of an attacker, then it can replayed. Moreover, no methodology offered in their approach that will guard them against DoS attack. If the attacker overwhelm the cloud server with a flood of authentication requests, then there is no way they can revoke those attacking requests without total multi-factor checks.

Chow et al. [34] proposed a trustcube framework for cloud authentication, which provides implicit authentication of the clients to the cloud. Only implicit behavior has never been enough to

Table 5.1: Comparison of Our Proposed MFA with Others Approaches Through Resistivity Against Different Attack Scenarios

Multi-factor Authentication Approaches Resistance						
Attacks Scenarios	Liu et al.	Chow et al.	Raja et al.	Yassin et al.	Candan et al.	Our Approach
Credential Discovery	No	No	No	No	No	No
Spoofing	No	No	No	No	No	No
Replay	Yes	Yes	No	No	No	No
DoS	Yes	Yes	Yes	Yes	Yes	No
Phishing	No	No	Yes	No	Yes	No

prevent all the attacks, like replay. Attackers can replay captured implicit behavioral data whenever they want, to authenticate themselves to the cloud. Moreover, they did not tell what they are going to do for clients who just signed up to cloud system now and have no implicit features saved in the cloud. This scheme will also suffer from DoS attack because there is full cycle of implicit parameters check for the cloud server for each client request and there is no cautionary measures to tackle a lot of requests at once. Phising and spoofing attacks are not successful with implicit authentication.

Raja et al. [29] proposed a MFA method that used uses face, periocular and iris biometric characteristics for authentication. Since their MFA factors are biometric, they have no other rescue for the client if the samples are compromised. Their scheme will also suffer from DoS attack, since they provided nothing to restrict DoS attackers trying to overwhelm the cloud sever with random biometric samples. If the clint gives his biometric samples in a phished website, then there is no rescue for the client.

Yassin et al. [26] proposed an MFA solution for the cloud using Schnorr digital signature and feature extracted from fingerprint. If the biometric fingerprint falls in the hand of an attacker, then only digital signature is not enough to provide strong authentication [41]. Their scheme also suffers from DoS attack, since they offered no methodology to handle a lot of requests at once. But in our scheme we do not process authentication requests from clients who made five consecutive failed attempts or sends more than three requests in one second. Rather we put these suspicious request producing machines in the blocked list of the client.

Candan et al. [31] has proposed a multi-factor authentication using password and smart card. With stolen or lost smart card, password can discovered. If the clint enter his password and smart card credentials in a phished website, then there is no rescue for the client. Their scheme will also suffer from Dos attack, while handling a huge bulk of authentication requests at once, as their approach has no remedy plan.

5.2 Sensitivity Analysis : Alarming Scenarios

Although we have categorized our authentication factors into groups and mostly they perform together in groups but some factors of each group have more importance and their values are more critical during the authentication process. Some parameters such as, hostname, operating system name, operating system version, deducts their parameter weight off when there is no match of them in the client authentication request. But mismatch of some parameters in some alarming scenarios, might discard the whole authentication request, even if other other credentials have a perfect match.
As we stated earlier, we have put more weights on the learned parameters (50%) and sudden variation (more than 5%) in the learned parameter will result in authentication failure, even if other two factors are correct. To measure the variation for a client request, we compare today's value of each learned parameter with last three months average. Let us discuss some of those alarming scenarios in the following paragraphs.

1. **Timestamp Variation**

   We are allowing a maximum variation of two minute in timestamp, between client and server clocks. A request from an older timestamp will not be processed further and discarded, even if every other credential matches. Because, requests from an older timestamps means that the cloud server has fallen victim of replay attack. The machine requesting authentication from the cloud server gets blocked for timestamp mismatch.

2. **Unrealistic Location Change**

   For an instance, a client has been authenticated to the cloud server at time T from Dhaka, and within ten minutes of time, the same client requesting for authentication to the cloud server from London, even with correct credentials. This authentication request is supposed to be rejected by the server and requires closer analysis. A client cannot just travel from Dhaka to London within ten minutes. The flight distance from Dhaka to London is about 8000km and it takes about 1l hour to reach there. This authentication request seems very suspicious and keeping this location parameters and their sudden change under close monitoring can help the cloud to revoke attacks.

3. **Change of Usual Login Time Range**

   Sudden change of usual login time range is required to be taken under strict monitoring. For example, a client's usual login window is 8.00 am to 12.00am, which has been learned throughout past three months. Suddenly there is an authentication request to the cloud server at 4.00 am, late night. Even with all the correct credentials like userID-password, correct TOTP, additional tests has to be given to client to prove its identity. This may sound hazardous but it may save reputations and money of people and organizations on certain situations.

4. **Sudden Abrupt Rise of Number of Login Attempts Per Day**

   A valid client signs in to the cloud, for example, upto 1-30 times a day and at the weekend there might be no login request. But when it is the case of an attacker robot or agent, they may try to forge a valid clients multi-factor credentials by attempting so many times until they get through. The frequency of authentication attempt is usually very high for an attacker, such as thousands of requests in a minute. This is clearly not from a human client. This client machine has been put into the blocked machines list at once. Average number of login attempts per day is closely related to average login failure rate per day. For an attacker, most of the attempts results in failure. In the opposing site, most of the attempts for a valid client usually result in successful authentication.

5. **Sudden Abrupt Rise of Average Login Failure Rate Per day**

   Login failure rate is also very minimum for a valid client. As we know, a valid client in bad cases might have approximately twenty login failure per day at most to put the correct credentials during the sign in but more than that is clearly suspicious. For and attacker, this parameters value is very large. The machine which fails to authenticate within the client's average learned failure rate, is required to be taken into the blocked machines list. Here we are not blocking the client. Because a valid client may not be aware, that some attacker is trying to forge his credentials to access his data in the cloud system. Our plan is to immobilize the attacker but not the valid clients. Valid clients can see those blocked machines details in their cloud settings page to get aware of the situation.

5.3 Viability Against Test Cases

We have made a dataset which consists of 1000 test case entries for a valid client to feed into the cloud server program. Each entry of the dataset contains dynamic data with different variations of host, location and network parameters. A sample of the dataset is shown in the appendix. We wanted to see which request crosses the authentication score and which does not. Here, learned parameters for the client are updated after processing each dataset entries. We have drawn a pie chart from the result shown in Fig. 5.7. We used seventy five as the authentication threshold for this experiment.

In Fig. 5.7, legitimate % actually denotes the percentage of accepted client requests, which

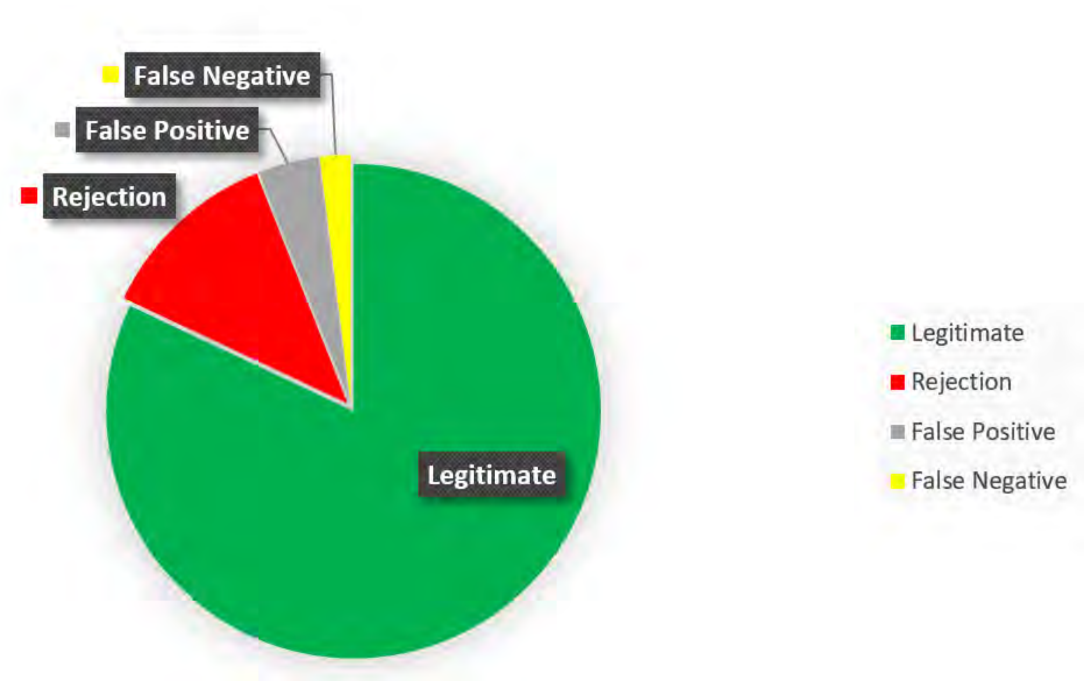


Figure 5.7: Spanning of Client Authentication Response by Cloud Server on Test Cases.

is about 80% of the pie. Rejection % denotes the percentage of rejected client requests, which is about 15% of the pie. Some requests get instant rejection for some alarming scenarios. For example, if the request comes with a old timestamp, it is rejected instantly. We checked abnormal and abrupt host, location, network parameters change in dataset entries. For example, a request coming from one location now and in the very next minute request from the same user coming from a far location. These sort ambiguous change of parameters will result in authentication rejection even if the first two factors are correct. There is also a bit of false negative (FN%) which is about 4% of the pie. This FN% comes into play when valid user is rejected. There is also a bit for false Positive (FP%) which is about 1% of the pie. This FP % happens when a suspicious request is accepted.

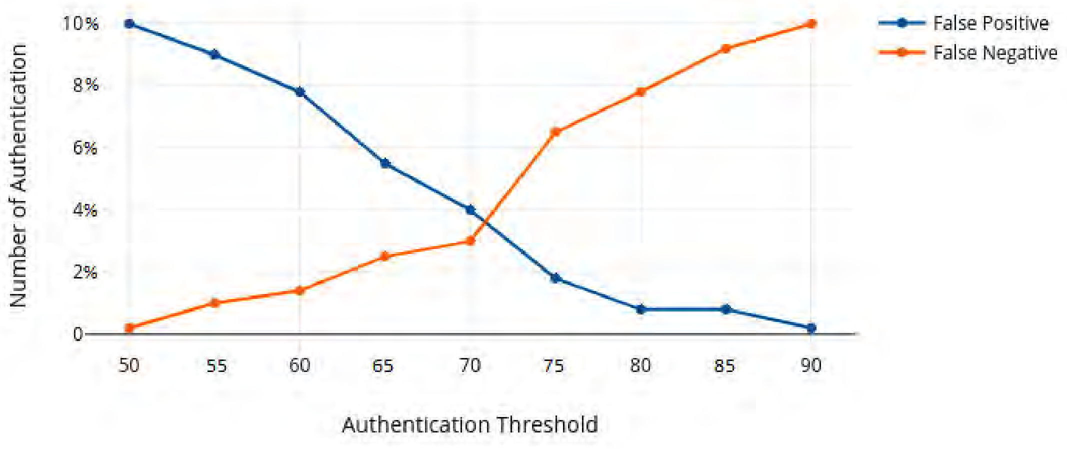


Figure 5.8: False Positive vs. False Negative, with the Change of Threshold.

5.4 Varying Threshold Level

A very high authentication threshold makes the authentication tougher and increases the False Negative proportions. Because, if the host, location and network bound parameters, all changes simultaneously, then even a valid user will find it difficult to make a score beyond the authentication threshold. More False Negative creates client dissatisfaction. On the other hand, FP% is produced when an invalid request crosses the authentication barrier. This FP portion will increase significantly when the authentication threshold is very low. More False positive is a big worry for the cloud vendors.

Fig. 5.8 shows that the authentication threshold is varied from 50 to 90. After processing the dataset, we can see, FP curve falls sharply when the authentication threshold is below the level of 70. Because, a lower authentication threshold gives some flexibility to the client on some parameters. But FN curve rises sharply when the authentication threshold is beyond the level of 70. Because, a system with higher authentication threshold allows very little deviation on authentication parameters. From the intersection of FP and FN curves, we can conclude that, to keep a good balance between FP and FP, more than seventy threshold is necessary.

5.5 Result Summary

Let us summarise our work-

- We have introduced authentication score that incorporates clients host, location, network and learned attributes.
- Dynamic nature of clients profile behavior has been recorded and parameters weights are adjusted accordingly.
- Cloud server knowledge-base keep on updating some learned parameters that helps to repulse some unusual attacks.
- Identified some alarming scenarios when authentication has to be denied for the client even if all MFA factors are correct.
- Simulated our proposed framework in Java (Desktop System) and Android (Mobile System).
- Analyzed the response of our MFA framework against different attack scenarios and found satisfactory results.
- Compared the performance of our Proposed MFA with existing authentication solutions.

5.6 Limitations of Our Approach

(i) **Security Questions**

In most cases, secret questions opens a security gate for the attackers because they are easier to hack than passwords. These questions are very common, such as “What is your parents hometown?”, “What is you pet name?”, “What is the first book you read?”, etc. People who are in close connection to the user or connected to user in social networking sites can make a good guess to have a match. So the thing that we can do here is to make the secret questions tough. This can be done by choosing some rare, multiple part questions that the user might not share in public and for the attacker it will be very difficult to get a match by brute force attack. Some questions of such kind could be, “Name a worst movie with best actor?“.

Answer of this question could be, “ Leonardo DiCaprio - Total Eclipse”. Another question of such type could be, “Name your favourite touring place-probable visiting year-probable month?”. Answer of this questions could be, “Moulovibazar-2018-December”.

(ii) **Stolen/Lost Client Device**

If the client device is lost or stolen by the attacker, then the attacker has a big chance to login to the cloud system with the identity of that valid client. If the attacker can hack into the stolen or lost client device, then he can exploit the software token TOTP and get benefited in authentication score because, he is login to cloud system from a trustworthy machine. This problem with software token can be avoided by using physical hardware token.

(iii) **Compromised Host**

If the client host device is compromised then it will be difficult to stop the attacker to login to the cloud system using authentication score, when the password and software/hardware token generated TOTP are already compromised. Because, attacker can make full use of software token(TOTP) installed in the device and get a high authentication score, since he is login from a trusted client device.

(iv) **One Minute TOTP Validity**

Although, TOTP are not passed through the network in plaintext format. However, if the encryption can be cracked then TOTP can be replayed within its one minute of valid period.

(v) **Attacker in Close Proximity**

Even if the client machine is not compromised, an attacker in close proximity of the client can get some authentication score benefit, since he is trying to login from the same network and location like the client does.

Chapter 6

Conclusion and Future Prospects

6.1 Conclusion

With the increasing popularity of cloud computing different security attacks tend to make its future at stake. Since, authentication barrier is the foremost target point of attackers, that is why we wanted to fortify it for cloud services. Current authentication strategies for cloud suffers from different security attacks. To overcome those attacks we have proposed a multi-factor authentication framework. Our authentication framework consists three factors. First one is the improved version of traditional userID-password with a strong password policy. We have incorporated RSA securID software/hardware token generated TOTP as the second factor which will safeguard the cloud systems from many authentication attacks. Finally, integrating clients host, network, location and learned parameters to calculate an authentication score and applying it directly to the the authentication process itself makes our MFA strategy more realistic and robust. These approach will handle some alarming cases of authentication attacks that existing authentication systems can not handle. We have simulated our proposed MFA in java and android and tested with test cases with valid and suspicious authentication parameter combinations. We have found our MFA can successfully repulse authentication requests that seems fraudulent. Incorporating our MFA framework in cloud systems will fight security threats on authentication and will offer great reliability to cloud vendors and clients.

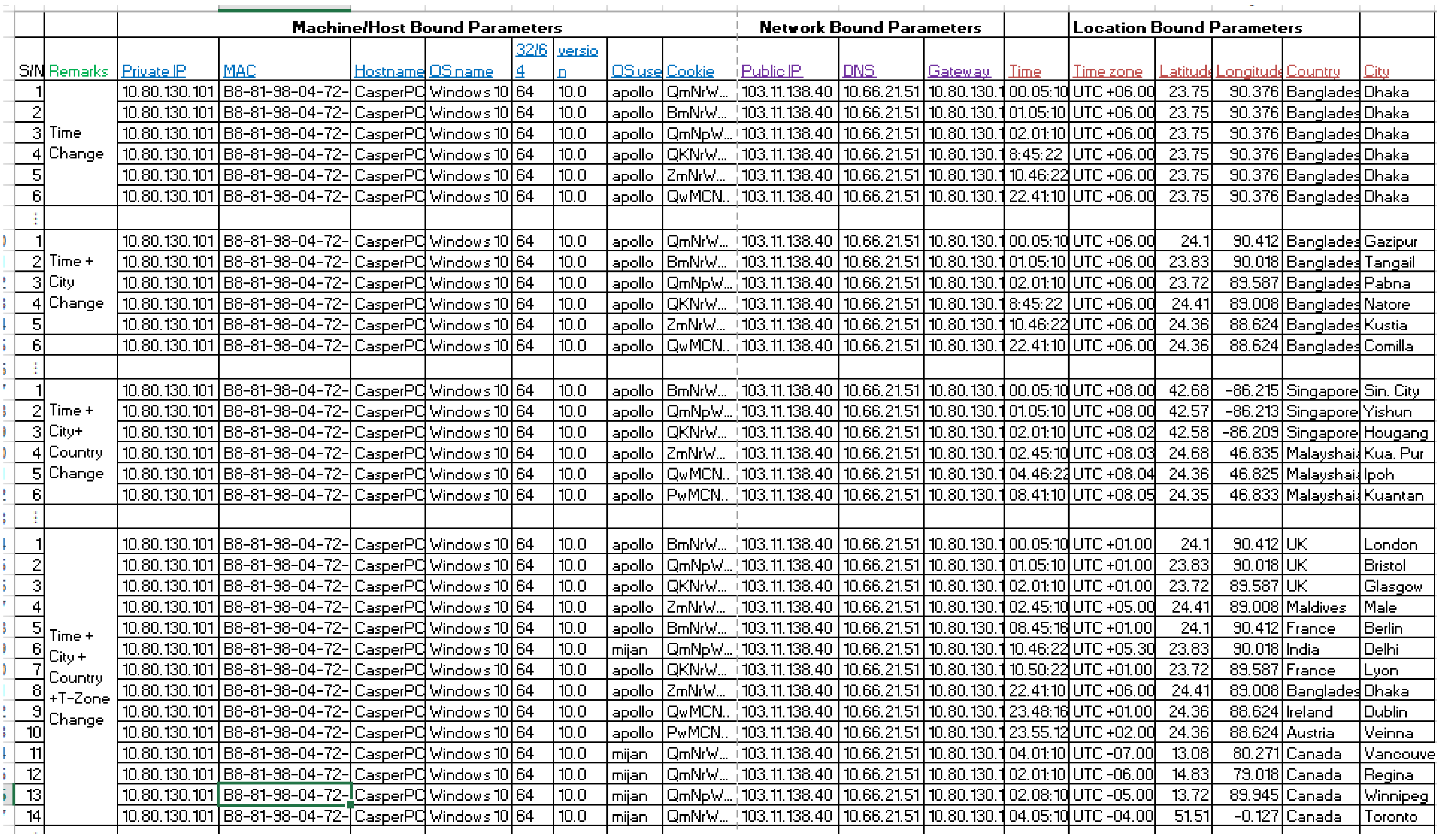
6.2 Future Prospects

At the beginning of authentication era, even the passwords were thought to be enough to provide authentication. Time has evolved so quickly that, even multi-factor authentication techniques suffer from different forms of security attacks today. Incorporating users host, network, location, usual login characteristics directly during authentication process helps to prevent most of the known authentication attacks. In future, we will study more authentication parameters and fine tune their weights. We will study, which parameters and how many of them are really necessary to provide strong authentication while keeping privacy of the client as well. We will also work to give mutual authentication between client and cloud server, utilizing both client and cloud server parameters.

Appendix A

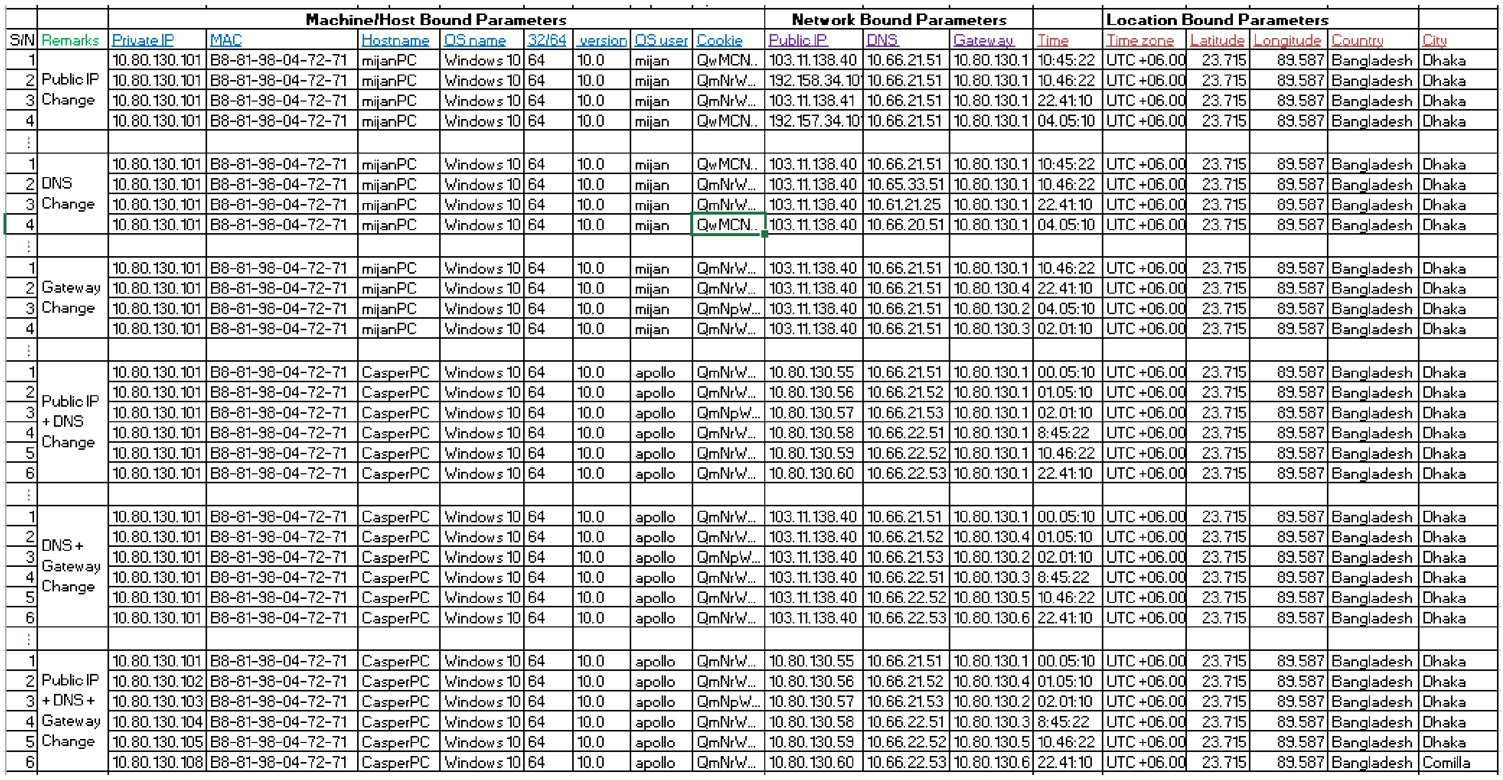
Appendix

A.1 Data Sets



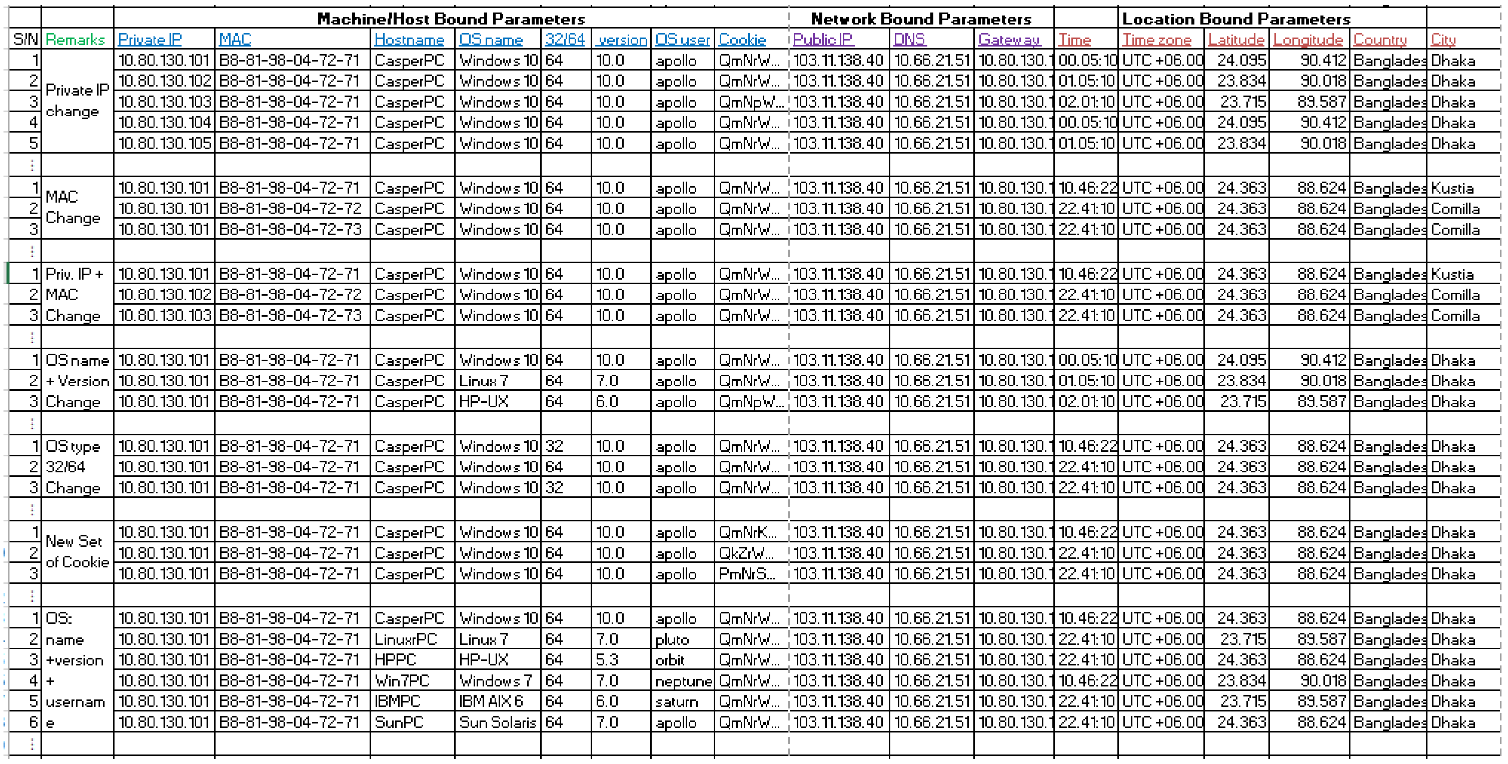
		Machine/Host Bound Parameters								Network Bound Parameters				Location Bound Parameters				
S/N	Remarks	Private IP	MAC	Hostname	OS name	32/64	version	OS use	Cookie	Public IP	DNS	Gateway	Time	Time zone	Latitude	Longitude	Country	City
1		10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	00.05:10	UTC +06.00	23.75	90.376	Banglades	Dhaka
2		10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	BmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	01.05:10	UTC +06.00	23.75	90.376	Banglades	Dhaka
3	Time	10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	QmNpW...	103.11.138.40	10.66.21.51	10.80.130.1	02.01:10	UTC +06.00	23.75	90.376	Banglades	Dhaka
4	Change	10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	QKNrW...	103.11.138.40	10.66.21.51	10.80.130.1	8:45:22	UTC +06.00	23.75	90.376	Banglades	Dhaka
5		10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	ZmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	10.46:22	UTC +06.00	23.75	90.376	Banglades	Dhaka
6		10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	QwMCN..	103.11.138.40	10.66.21.51	10.80.130.1	22.41:10	UTC +06.00	23.75	90.376	Banglades	Dhaka
⋮																		
1		10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	00.05:10	UTC +06.00	24.1	90.412	Banglades	Gazipur
2	Time +	10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	BmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	01.05:10	UTC +06.00	23.83	90.018	Banglades	Tangail
3	City	10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	QmNpW...	103.11.138.40	10.66.21.51	10.80.130.1	02.01:10	UTC +06.00	23.72	89.587	Banglades	Pabna
4	Change	10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	QKNrW...	103.11.138.40	10.66.21.51	10.80.130.1	8:45:22	UTC +06.00	24.41	89.008	Banglades	Natore
5		10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	ZmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	10.46:22	UTC +06.00	24.36	88.624	Banglades	Kustia
6		10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	QwMCN..	103.11.138.40	10.66.21.51	10.80.130.1	22.41:10	UTC +06.00	24.36	88.624	Banglades	Comilla
⋮																		
1		10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	BmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	00.05:10	UTC +08.00	42.68	-86.215	Singapore	Sin. City
2	Time +	10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	QmNpW...	103.11.138.40	10.66.21.51	10.80.130.1	01.05:10	UTC +08.00	42.57	-86.213	Singapore	Yishun
3	City+	10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	QKNrW...	103.11.138.40	10.66.21.51	10.80.130.1	02.01:10	UTC +08.02	42.58	-86.209	Singapore	Hougang
4	Country	10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	ZmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	02.45:10	UTC +08.03	24.68	46.835	Malayshai	Kua. Pur
5	Change	10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	QwMCN..	103.11.138.40	10.66.21.51	10.80.130.1	04.46:22	UTC +08.04	24.36	46.825	Malayshai	Ipoh
6		10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	PwMCN..	103.11.138.40	10.66.21.51	10.80.130.1	08.41:10	UTC +08.05	24.35	46.833	Malayshai	Kuantan
⋮																		
1		10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	BmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	00.05:10	UTC +01.00	24.1	90.412	UK	London
2		10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	QmNpW...	103.11.138.40	10.66.21.51	10.80.130.1	01.05:10	UTC +01.00	23.83	90.018	UK	Bristol
3		10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	QKNrW...	103.11.138.40	10.66.21.51	10.80.130.1	02.01:10	UTC +01.00	23.72	89.587	UK	Glasgow
4		10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	ZmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	02.45:10	UTC +05.00	24.41	89.008	Maldives	Male
5	Time +	10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	BmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	08.45:16	UTC +01.00	24.1	90.412	France	Berlin
6	City +	10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	mijan	QmNpW...	103.11.138.40	10.66.21.51	10.80.130.1	10.46:22	UTC +05.30	23.83	90.018	India	Delhi
7	Country	10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	QKNrW...	103.11.138.40	10.66.21.51	10.80.130.1	10.50:22	UTC +01.00	23.72	89.587	France	Lyon
8	+T-Zone	10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	ZmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	22.41:10	UTC +06.00	24.41	89.008	Banglades	Dhaka
9	Change	10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	QwMCN..	103.11.138.40	10.66.21.51	10.80.130.1	23.48:16	UTC +01.00	24.36	88.624	Ireland	Dublin
10		10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	apollo	PwMCN..	103.11.138.40	10.66.21.51	10.80.130.1	23.55.12	UTC +02.00	24.36	88.624	Austria	Veinna
11		10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	mijan	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	04.01:10	UTC -07.00	13.08	80.271	Canada	Vancouve
12		10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	mijan	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	02.01:10	UTC -06.00	14.83	79.018	Canada	Regina
13		10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	mijan	QmNpW...	103.11.138.40	10.66.21.51	10.80.130.1	02.08:10	UTC -05.00	13.72	89.945	Canada	Winnipeg
14		10.80.130.101	B8-81-98-04-72-	CasperPC	Windows 10	64	10.0	mijan	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	04.05:10	UTC -04.00	51.51	-0.127	Canada	Toronto

Figure A.1: Dataset Part-1



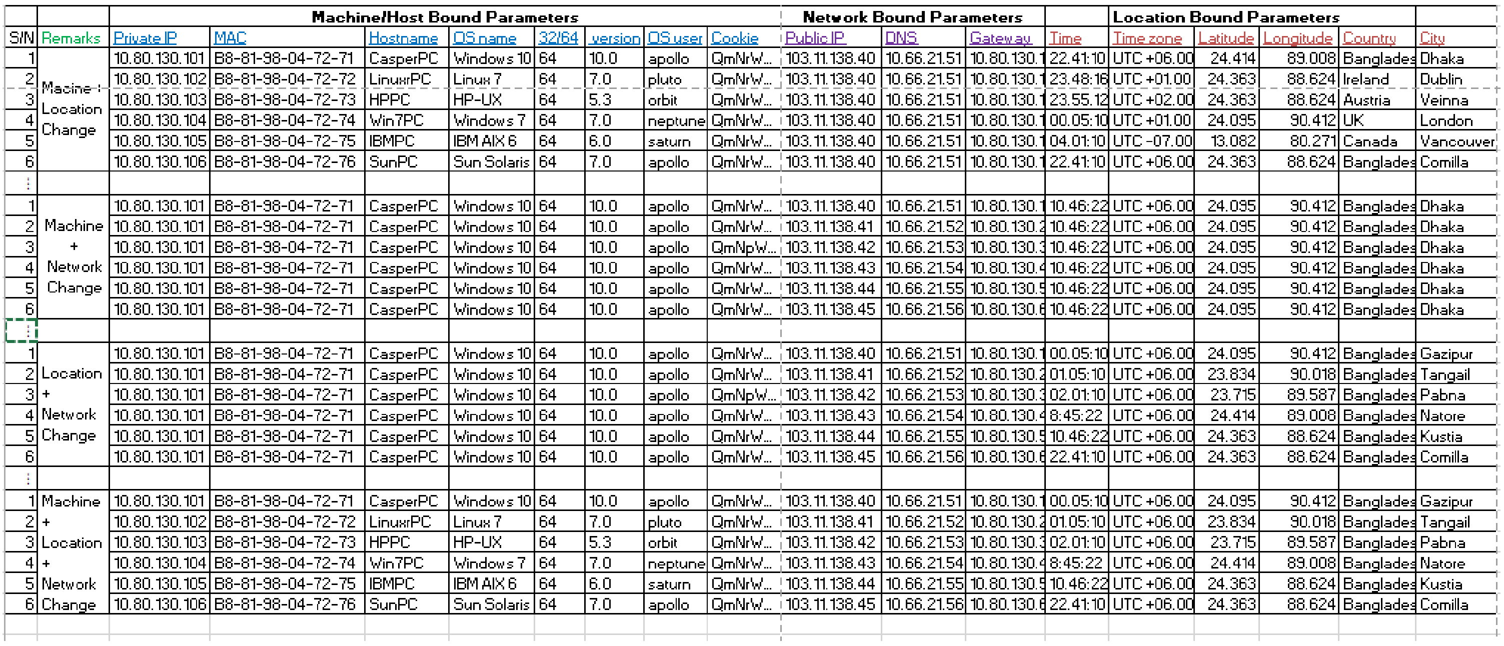
		Machine/Host Bound Parameters								Network Bound Parameters				Location Bound Parameters				
S/N	Remarks	Private IP	MAC	Hostname	OS name	32/64	version	OS user	Cookie	Public IP	DNS	Gateway	Time	Time zone	Latitude	Longitude	Country	City
1		10.80.130.101	B8-81-98-04-72-71	mijanPC	Windows 10	64	10.0	mijan	QwMCN..	103.11.138.40	10.66.21.51	10.80.130.1	10:45:22	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
2	Public IP	10.80.130.101	B8-81-98-04-72-71	mijanPC	Windows 10	64	10.0	mijan	QmNrW...	192.158.34.10	10.66.21.51	10.80.130.1	10.46:22	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
3	Change	10.80.130.101	B8-81-98-04-72-71	mijanPC	Windows 10	64	10.0	mijan	QmNrW...	103.11.138.41	10.66.21.51	10.80.130.1	22.41:10	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
4		10.80.130.101	B8-81-98-04-72-71	mijanPC	Windows 10	64	10.0	mijan	QwMCN..	192.157.34.10	10.66.21.51	10.80.130.1	04.05:10	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
⋮																		
1		10.80.130.101	B8-81-98-04-72-71	mijanPC	Windows 10	64	10.0	mijan	QwMCN..	103.11.138.40	10.66.21.51	10.80.130.1	10:45:22	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
2	DNS	10.80.130.101	B8-81-98-04-72-71	mijanPC	Windows 10	64	10.0	mijan	QmNrW...	103.11.138.40	10.65.33.51	10.80.130.1	10.46:22	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
3	Change	10.80.130.101	B8-81-98-04-72-71	mijanPC	Windows 10	64	10.0	mijan	QmNrW...	103.11.138.40	10.61.21.25	10.80.130.1	22.41:10	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
4		10.80.130.101	B8-81-98-04-72-71	mijanPC	Windows 10	64	10.0	mijan	QwMCN..	103.11.138.40	10.66.20.51	10.80.130.1	04.05:10	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
⋮																		
1		10.80.130.101	B8-81-98-04-72-71	mijanPC	Windows 10	64	10.0	mijan	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	10.46:22	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
2	Gateway	10.80.130.101	B8-81-98-04-72-71	mijanPC	Windows 10	64	10.0	mijan	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.4	22.41:10	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
3	Change	10.80.130.101	B8-81-98-04-72-71	mijanPC	Windows 10	64	10.0	mijan	QmNpW...	103.11.138.40	10.66.21.51	10.80.130.2	04.05:10	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
4		10.80.130.101	B8-81-98-04-72-71	mijanPC	Windows 10	64	10.0	mijan	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.3	02.01:10	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
⋮																		
1		10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	10.80.130.55	10.66.21.51	10.80.130.1	00.05:10	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
2	Public IP	10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	10.80.130.56	10.66.21.52	10.80.130.1	01.05:10	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
3	+ DNS	10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNpW...	10.80.130.57	10.66.21.53	10.80.130.1	02.01:10	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
4	Change	10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	10.80.130.58	10.66.22.51	10.80.130.1	8:45:22	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
5		10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	10.80.130.59	10.66.22.52	10.80.130.1	10.46:22	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
6		10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	10.80.130.60	10.66.22.53	10.80.130.1	22.41:10	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
⋮																		
1		10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	00.05:10	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
2	DNS +	10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.40	10.66.21.52	10.80.130.4	01.05:10	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
3	Gateway	10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNpW...	103.11.138.40	10.66.21.53	10.80.130.2	02.01:10	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
4	Change	10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.40	10.66.22.51	10.80.130.3	8:45:22	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
5		10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.40	10.66.22.52	10.80.130.5	10.46:22	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
6		10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.40	10.66.22.53	10.80.130.6	22.41:10	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
⋮																		
1		10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	10.80.130.55	10.66.21.51	10.80.130.1	00.05:10	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
2	Public IP	10.80.130.102	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	10.80.130.56	10.66.21.52	10.80.130.4	01.05:10	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
3	+ DNS +	10.80.130.103	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNpW...	10.80.130.57	10.66.21.53	10.80.130.2	02.01:10	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
4	Gateway	10.80.130.104	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	10.80.130.58	10.66.22.51	10.80.130.3	8:45:22	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
5	Change	10.80.130.105	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	10.80.130.59	10.66.22.52	10.80.130.5	10.46:22	UTC +06.00	23.715	89.587	Bangladesh	Dhaka
6		10.80.130.108	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	10.80.130.60	10.66.22.53	10.80.130.6	22.41:10	UTC +06.00	23.715	89.587	Bangladesh	Comilla

Figure A.2: Dataset Part-2



		Machine/Host Bound Parameters								Network Bound Parameters				Location Bound Parameters				
S/N	Remarks	Private IP	MAC	Hostname	OS name	32/64	version	OS user	Cookie	Public IP	DNS	Gateway	Time	Time zone	Latitude	Longitude	Country	City
1		10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	00.05:10	UTC +06.00	24.095	90.412	Banglades	Dhaka
2	Private IP	10.80.130.102	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	01.05:10	UTC +06.00	23.834	90.018	Banglades	Dhaka
3	change	10.80.130.103	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNpW...	103.11.138.40	10.66.21.51	10.80.130.1	02.01:10	UTC +06.00	23.715	89.587	Banglades	Dhaka
4		10.80.130.104	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	00.05:10	UTC +06.00	24.095	90.412	Banglades	Dhaka
5		10.80.130.105	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	01.05:10	UTC +06.00	23.834	90.018	Banglades	Dhaka
⋮																		
1	MAC	10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	10.46:22	UTC +06.00	24.363	88.624	Banglades	Kustia
2	Change	10.80.130.101	B8-81-98-04-72-72	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	22.41:10	UTC +06.00	24.363	88.624	Banglades	Comilla
3		10.80.130.101	B8-81-98-04-72-73	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	22.41:10	UTC +06.00	24.363	88.624	Banglades	Comilla
⋮																		
1	Priv. IP +	10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	10.46:22	UTC +06.00	24.363	88.624	Banglades	Kustia
2	MAC	10.80.130.102	B8-81-98-04-72-72	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	22.41:10	UTC +06.00	24.363	88.624	Banglades	Comilla
3	Change	10.80.130.103	B8-81-98-04-72-73	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	22.41:10	UTC +06.00	24.363	88.624	Banglades	Comilla
⋮																		
1	OS name	10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	00.05:10	UTC +06.00	24.095	90.412	Banglades	Dhaka
2	+ Version	10.80.130.101	B8-81-98-04-72-71	CasperPC	Linux 7	64	7.0	apollo	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	01.05:10	UTC +06.00	23.834	90.018	Banglades	Dhaka
3	Change	10.80.130.101	B8-81-98-04-72-71	CasperPC	HP-UX	64	6.0	apollo	QmNpW...	103.11.138.40	10.66.21.51	10.80.130.1	02.01:10	UTC +06.00	23.715	89.587	Banglades	Dhaka
⋮																		
1	OS type	10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	32	10.0	apollo	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	10.46:22	UTC +06.00	24.363	88.624	Banglades	Dhaka
2	32/64	10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	22.41:10	UTC +06.00	24.363	88.624	Banglades	Dhaka
3	Change	10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	32	10.0	apollo	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	22.41:10	UTC +06.00	24.363	88.624	Banglades	Dhaka
⋮																		
1	New Set	10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrK...	103.11.138.40	10.66.21.51	10.80.130.1	10.46:22	UTC +06.00	24.363	88.624	Banglades	Dhaka
2	of Cookie	10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QkZrW...	103.11.138.40	10.66.21.51	10.80.130.1	22.41:10	UTC +06.00	24.363	88.624	Banglades	Dhaka
3		10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	PmNrS...	103.11.138.40	10.66.21.51	10.80.130.1	22.41:10	UTC +06.00	24.363	88.624	Banglades	Dhaka
⋮																		
1	OS:	10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	10.46:22	UTC +06.00	24.363	88.624	Banglades	Dhaka
2	name	10.80.130.101	B8-81-98-04-72-71	LinuxrPC	Linux 7	64	7.0	pluto	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	22.41:10	UTC +06.00	23.715	89.587	Banglades	Dhaka
3	+version	10.80.130.101	B8-81-98-04-72-71	HPPC	HP-UX	64	5.3	orbit	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	22.41:10	UTC +06.00	24.363	88.624	Banglades	Dhaka
4	+	10.80.130.101	B8-81-98-04-72-71	Win7PC	Windows 7	64	7.0	neptune	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	10.46:22	UTC +06.00	23.834	90.018	Banglades	Dhaka
5	usernam	10.80.130.101	B8-81-98-04-72-71	IBMPC	IBM AIX 6	64	6.0	saturn	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	22.41:10	UTC +06.00	23.715	89.587	Banglades	Dhaka
6	e	10.80.130.101	B8-81-98-04-72-71	SunPC	Sun Solaris	64	7.0	apollo	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	22.41:10	UTC +06.00	24.363	88.624	Banglades	Dhaka
⋮																		

Figure A.3: Dataset Part-3



		Machine/Host Bound Parameters								Network Bound Parameters				Location Bound Parameters				
S/N	Remarks	Private IP	MAC	Hostname	OS name	32/64	version	OS user	Cookie	Public IP	DNS	Gateway	Time	Time zone	Latitude	Longitude	Country	City
1	Macine + Location Change	10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	22.41:10	UTC +06.00	24.414	89.008	Banglades	Dhaka
2		10.80.130.102	B8-81-98-04-72-72	LinuxrPC	Linux 7	64	7.0	pluto	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	23.48:16	UTC +01.00	24.363	88.624	Ireland	Dublin
3		10.80.130.103	B8-81-98-04-72-73	HPPC	HP-UX	64	5.3	orbit	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	23.55.12	UTC +02.00	24.363	88.624	Austria	Veinna
4		10.80.130.104	B8-81-98-04-72-74	Win7PC	Windows 7	64	7.0	neptune	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	00.05:10	UTC +01.00	24.095	90.412	UK	London
5		10.80.130.105	B8-81-98-04-72-75	IBMPC	IBM AIX 6	64	6.0	saturn	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	04.01:10	UTC -07.00	13.082	80.271	Canada	Vancouver
6		10.80.130.106	B8-81-98-04-72-76	SunPC	Sun Solaris	64	7.0	apollo	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	22.41:10	UTC +06.00	24.363	88.624	Banglades	Comilla
⋮																		
1	Machine + Network Change	10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	10.46:22	UTC +06.00	24.095	90.412	Banglades	Dhaka
2		10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.41	10.66.21.52	10.80.130.2	10.46:22	UTC +06.00	24.095	90.412	Banglades	Dhaka
3		10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNpW...	103.11.138.42	10.66.21.53	10.80.130.3	10.46:22	UTC +06.00	24.095	90.412	Banglades	Dhaka
4		10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.43	10.66.21.54	10.80.130.4	10.46:22	UTC +06.00	24.095	90.412	Banglades	Dhaka
5		10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.44	10.66.21.55	10.80.130.5	10.46:22	UTC +06.00	24.095	90.412	Banglades	Dhaka
6		10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.45	10.66.21.56	10.80.130.6	10.46:22	UTC +06.00	24.095	90.412	Banglades	Dhaka
⋮																		
1	Location + Network Change	10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	00.05:10	UTC +06.00	24.095	90.412	Banglades	Gazipur
2		10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.41	10.66.21.52	10.80.130.2	01.05:10	UTC +06.00	23.834	90.018	Banglades	Tangail
3		10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNpW...	103.11.138.42	10.66.21.53	10.80.130.3	02.01:10	UTC +06.00	23.715	89.587	Banglades	Pabna
4		10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.43	10.66.21.54	10.80.130.4	8:45:22	UTC +06.00	24.414	89.008	Banglades	Natore
5		10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.44	10.66.21.55	10.80.130.5	10.46:22	UTC +06.00	24.363	88.624	Banglades	Kustia
6		10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.45	10.66.21.56	10.80.130.6	22.41:10	UTC +06.00	24.363	88.624	Banglades	Comilla
⋮																		
1	Machine	10.80.130.101	B8-81-98-04-72-71	CasperPC	Windows 10	64	10.0	apollo	QmNrW...	103.11.138.40	10.66.21.51	10.80.130.1	00.05:10	UTC +06.00	24.095	90.412	Banglades	Gazipur
2	+	10.80.130.102	B8-81-98-04-72-72	LinuxrPC	Linux 7	64	7.0	pluto	QmNrW...	103.11.138.41	10.66.21.52	10.80.130.2	01.05:10	UTC +06.00	23.834	90.018	Banglades	Tangail
3	Location	10.80.130.103	B8-81-98-04-72-73	HPPC	HP-UX	64	5.3	orbit	QmNrW...	103.11.138.42	10.66.21.53	10.80.130.3	02.01:10	UTC +06.00	23.715	89.587	Banglades	Pabna
4	+	10.80.130.104	B8-81-98-04-72-74	Win7PC	Windows 7	64	7.0	neptune	QmNrW...	103.11.138.43	10.66.21.54	10.80.130.4	8:45:22	UTC +06.00	24.414	89.008	Banglades	Natore
5	Network	10.80.130.105	B8-81-98-04-72-75	IBMPC	IBM AIX 6	64	6.0	saturn	QmNrW...	103.11.138.44	10.66.21.55	10.80.130.5	10.46:22	UTC +06.00	24.363	88.624	Banglades	Kustia
6	Change	10.80.130.106	B8-81-98-04-72-76	SunPC	Sun Solaris	64	7.0	apollo	QmNrW...	103.11.138.45	10.66.21.56	10.80.130.6	22.41:10	UTC +06.00	24.363	88.624	Banglades	Comilla

Figure A.4: Dataset Part-4

Table A.1: Dataset with Cumulative Summation of Authentication Parameter Set and Probable Output(Y)

HostTotal	NetTotal	LocTotal	LearnTotal	Probale(Y)
0	0	0	0	0
100	100	100	90	1
100	100	90	90	1
100	100	90	88	1
90	100	90	87	1
100	100	90	90	1
100	100	90	70	0
100	100	90	90	1
100	100	90	81	1
90	100	90	80	1
100	0	70	90	1
100	0	70	60	0
80	0	70	88	1
100	40	70	89	1
100	0	70	75	0
100	0	70	85	1
100	0	70	85	1
90	20	70	70	0
30	0	0	82	0
60	0	0	78	0
20	0	0	82	0
20	0	0	81	0
85	80	0	60	0
20	0	0	80	0
30	0	0	85	0
40	0	0	82	0
0	0	0	83	0
60	0	0	82	0
90	100	90	80	1
...	...	...	...	...

A.2 Dataset with Cumulative Summation of Parameter Set

A.3 Code Snippets

Client Class

```
import java.net.*;
import java.io.*;
import java.text.SimpleDateFormat;
import java.util.*;
import java.net.InetAddress;
import com.maxmind.geoip.*;
class GetLocationExample {
 public ServerLocation getLocation(String ipAddress){
 File file = new File("GeoLiteCity.dat");
 return getLocation(ipAddress, file); }
 public ServerLocation getLocation(String ipAddress, File file){
 ServerLocation serverLocation = null;
 try{
  serverLocation = new ServerLocation();
  LookupService lookup = new
        LookupService(file,LookupService.GEOIP_MEMORY_CACHE);
  com.maxmind.geoip.Location locationServices=lookup.getLocation(ipAddress);
  serverLocation.setCountryCode(locationServices.countryCode);
  serverLocation.setCountryName(locationServices.countryName);
  serverLocation.setRegion(locationServices.region);
  serverLocation.setRegionName(regionName.regionNameByCode(
  locationServices.countryCode, locationServices.region));
  serverLocation.setCity(locationServices.city);
  serverLocation.setPostalCode(locationServices.postalCode);
  serverLocation.setLatitude(String.valueOf(locationServices.latitude));
  serverLocation.setLongitude(String.valueOf(locationServices.longitude));
  } catch (IOException e){ System.err.println(e.getMessage());}
  return serverLocation;
 }
}
class HardwareAddress
```

```
{ public static String getMacAddress() throws UnknownHostException,
 SocketException
 {InetAddress ipAddress = InetAddress.getLocalHost();
NetworkInterface networkInterface=NetworkInterface.getByInetAddress(ipAddress);
 byte[] macAddressBytes = networkInterface.getHardwareAddress();
 StringBuilder macAddressBuilder = new StringBuilder();
for (int macAddressByteIndex=0;macAddressByteIndex<macAddressBytes.length;
      macAddressByteIndex++)
 { String macAddressHexByte =
              String.format("%02X",macAddressBytes[macAddressByteIndex]);
   macAddressBuilder.append(macAddressHexByte);
if(macAddressByteIndex!=macAddressBytes.length-1){macAddressBuilder.append(":");}
 }
 return macAddressBuilder.toString();
 }
}
public class PAP {
 public final static int SOCKET_PORT = 12345;
 public final static String SERVER = "127.0.0.1";  // localhost
 public final static String FILE_TO_SEND = "FileToSend.txt";
 public static void main(String[] args) throws IOException {
 File fileToSend = new File("fileToSend.txt");//Contains pap data
 if (fileToSend.exists()){
       RandomAccessFile raf = new RandomAccessFile(fileToSend, "rw");
       raf.setLength(0); }
  FileWriter fileWriter = new FileWriter(fileToSend,true);
  fileWriter.flush();
  BufferedWriter writer=new BufferedWriter(fileWriter);
  writer.flush();
//writing private IP address of current device
  InetAddress localhost = InetAddress.getLocalHost();
//System.out.println("System IP Address:"+(localhost.getHostAddress()).trim());
  writer.write("Private IP Address  :" + (localhost.getHostAddress()).trim());
  writer.newLine();
  //appending public IP address of current device
  String systemipaddress = "";
try{URL url_name = new URL("http://bot.whatismyipaddress.com");
BufferedReader sc=new BufferedReader(new InputStreamReader(url_name.openStream()));
//reads system IPAddress
  systemipaddress = sc.readLine().trim();
  } catch (Exception e){systemipaddress = "Cannot Execute Properly"; }
//System.out.println("Public IP Address: " + systemipaddress +"\n");
  writer.append("Public IP Address   :" + systemipaddress);
  writer.newLine();
//writing MAC address of current device
  HardwareAddress hwa=new HardwareAddress();
try{//System.out.println("Mac: " + hwa.getMacAddress());
  writer.append("MAC Address :" + hwa.getMacAddress());
  writer.newLine();
   } catch (SocketException e) {e.printStackTrace();}
//writing OS attributes //System.getProperties().list(System.out);
  System.getProperty("user.name"));
  writer.append("OS NAME :"+ System.getProperty("os.name"));
  writer.newLine();
  writer.append("OS VERSION :" + System.getProperty("os.version"));
  writer.newLine();
  writer.append("USERNAME :" + System.getProperty("user.name"));
  writer.newLine();
//writing current timestamp
  String timeStamp =
         new SimpleDateFormat("yyyy.MM.dd.HH.mm.ss").format(new Date());
  String time=timeStamp.substring(11,16);
  writer.append("Current TIME :" +time);
  writer.newLine();//writing the location of current device
  GetLocationExample obj = new GetLocationExample();
  ServerLocation location = obj.getLocation(systemipaddress);
  //System.out.println(location);
  writer.append("Longitude :" + location.getLongitude());
```

```
 writer.newLine();
 writer.append("Latitude :" + location.getLatitude());
 writer.newLine();
 writer.append("Country :" + location.getCountryName());
 writer.newLine();
 writer.append("City :" + location.getCity());
 writer.newLine();
 //writing Cookie data
 CookieManager cookieManager = new CookieManager();
 cookieManager.setCookiePolicy(CookiePolicy.ACCEPT_ALL);
 CookieHandler.setDefault(cookieManager);
 //creates url for the given string
 URL url = null;
try{
url = new URL("https://www.mycloud.com/");
//open’s a connection with the url and returns urlConnection object
URLConnection  urlConnection = url.openConnection();
//get’s the contents from this url specifies
urlConnection.getContent();
   } catch (MalformedURLException e) {e.printStackTrace();}
  catch (IOException e) {e.printStackTrace(); }
//returns the cookie store(bunch of cookies)
CookieStore cookieStore = cookieManager.getCookieStore();
//getting cookies in the form of List of type HttpCookie
List<HttpCookie> listOfcookies = cookieStore.getCookies();
for(HttpCookie httpCookie: listOfcookies){
   writer.append("Cookie Name : "+httpCookie.getName());
   writer.newLine();
   writer.append("Cookie Value : "+httpCookie.getValue());
   writer.newLine();
   }
   writer.close();//sending to SERVER side
   FileInputStream fis; BufferedInputStream bis = null;
   OutputStream os = null; Socket sock = null;
 try{
  sock = new Socket(SERVER, SOCKET_PORT);
  System.out.println("Connecting...");
  byte [] mybytearray  = new byte [(int)fileToSend.length()];
  fis = new FileInputStream(fileToSend);
  bis = new BufferedInputStream(fis);
  bis.read(mybytearray,0,mybytearray.length);
  os = sock.getOutputStream();
 System.out.println("Sending"+FILE_TO_SEND+"("+mybytearray.length+"bytes)");
  os.write(mybytearray,0,mybytearray.length);
  os.flush();
  System.out.println("Sent.");
  } finally{
            if (bis != null) bis.close();
            if (os != null) os.close();
            if (sock!=null) sock.close();
            }
} }
```

Server Class

```
import javafx.util.Pair;
import java.io.*;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.ArrayList;
class ProfileAttribute{
String privateIP,publicIP,mac,osname,osver,username,time;
String longtd,latd,country,city;
 String[] strings = new String[12];
 ArrayList<String> a=new ArrayList<>();
 ArrayList<Pair<String,String>> cookieData = new ArrayList<>();
 public void assign() {
 strings = new String[]{"Private IP Address", "Public IP Address",
  "MAC Address", "OS Name", "OS version", "User Name", "Time",
   "Longitude", "Latitude", "Country", "City", "Cookies"};
    privateIP=a.get(0);
    publicIP=a.get(1);
    mac=a.get(2);
    osname=a.get(3);
    osver=a.get(4);
    username=a.get(5);
    time=a.get(6);
    longtd=a.get(7);
    latd=a.get(8);
    country=a.get(9);
    city=a.get(10);
for(int i=0;i<11;i++) System.out.println(strings[i]+":"+a.get(i));
System.out.println(strings[11]+": ");
 for(int i=11;i<a.size();i+=2){
    cookieData.add(new Pair<>(a.get(i),a.get(i+1)));
    System.out.println("Cookie Name: "+a.get(i));
    System.out.println("Cookie Data: "+a.get(i+1)); }
 }
}
public class Server {
 public final static int SOCKET_PORT = 13267;// we may change this
 //public final static String SERVER = "127.0.0.1";  // localhost
 public final static String
 FILE_TO_RECEIVED = "E:\\file_received.txt";
 public final static int FILE_SIZE = 6022386; // file size
 // should bigger than the file to be downloaded
 public static void main (String [] args ) throws IOException {
 int bytesRead;
 int current = 0;
 FileOutputStream fos = null;
 BufferedOutputStream bos = null;
 Socket sock = null;
 ServerSocket serverSocket=null;
 try{
 serverSocket = new ServerSocket(SOCKET_PORT);
 System.out.println("Waiting...");
 while(sock==null){ sock = serverSocket.accept();}
 System.out.println("Connection Accepted"); //Receive file
    byte [] mybytearray  = new byte [FILE_SIZE];
    InputStream is = sock.getInputStream();
    fos = new FileOutputStream(FILE_TO_RECEIVED);
    bos = new BufferedOutputStream(fos);
    bytesRead = is.read(mybytearray,0,mybytearray.length);
    current = bytesRead;
do{bytesRead=is.read(mybytearray,current,(mybytearray.length-current));
   if(bytesRead >= 0) current += bytesRead;
  } while(bytesRead > -1);
  bos.write(mybytearray, 0 , current);
  bos.flush();
System.out.println("File"+FILE_TO_RECEIVED+"Downloaded("+current+"bytes read)");
  }
 finally{ if (fos != null) fos.close();
          if (bos != null) bos.close();
          if (sock != null) sock.close(); }
ProfileAttribute profileAttribute=new ProfileAttribute();//read the file
BufferedReader br=new BufferedReader(new FileReader("E:\\file_received.txt"));
try{
   String line = br.readLine();
   while (line != null) {
   profileAttribute.a.add(line.substring(line.indexOf(':')+1));
   line = br.readLine();}
   profileAttribute.assign();
  }finally { br.close();}
 } }
```

Bibliography

[1] A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, and I. Stoica, “Above the clouds: A berkeley view of cloud computing,” *Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Rep. UCB/EECS*, vol. 28, no. 13, p. 2009, 2009.

[2] G. D. Tormo, F. G. Mármol, and G. M. Pérez, “Identity management in cloud systems,” in *Security, Privacy and Trust in Cloud Systems*, pp. 177–210, Springer, Sep 2014.

[3] D. Balfanz, R. Chow, O. Eisen, M. Jakobsson, S. Kirsch, S. Matsumoto, J. Molina, and P. van Oorschot, “The future of authentication,” *IEEE Security & Privacy*, vol. 10, no. 1, pp. 22–27, 2012.

[4] A. N. Toosi, R. N. Calheiros, and R. Buyya, “Interconnected cloud computing environments: Challenges, taxonomy, and survey,” *ACM Computing Surveys (CSUR)*, vol. 47, no. 1, p. 7, 2014.

[5] P. A. Boampong and L. A. Wahsheh, “Different facets of security in the cloud,” in *Proceedings of the 15th Communications and Networking Simulation Symposium*, (San Diego, CA, USA), p. 5, Mar. 26-30, 2012.

[6] J. Somorovsky, M. Heiderich, M. Jensen, J. Schwenk, N. Gruschka, and L. Lo Iacono, “All your clouds are belong to us: security analysis of cloud management interfaces,” in *Proceedings of the 3rd ACM workshop on Cloud computing security workshop*, pp. 3–14, ACM, 2011.

[7] C. Modi, D. Patel, B. Borisaniya, A. Patel, and M. Rajarajan, “A survey on security issues and solutions at different layers of cloud computing,” *The journal of supercomputing*, vol. 63, no. 2, pp. 561–592, 2013.

[8] C. A. Ardagna, R. Asal, E. Damiani, and Q. H. Vu, “From security to assurance in the cloud: A survey,” *ACM Computing Surveys (CSUR)*, vol. 48, no. 1, p. 2, 2015.

[9] N. Alomar, M. Alsaleh, and A. Alarifi, “Social authentication applications, attacks, defense strategies and future research directions: a systematic review,” *IEEE Communications Surveys & Tutorials*, vol. 99, 2017.

[10] M. Alizadeh, S. Abolfazli, M. Zamani, S. Baharun, and K. Sakurai, “Authentication in mobile cloud computing: A survey,” *Journal of Network and Computer Applications*, vol. 61, pp. 59–80, 2016.

[11] K. Walsh and J. Manferdelli, “Intra-cloud and inter-cloud authentication,” in *Cloud Computing (CLOUD), 2017 IEEE 10th International Conference on*, pp. 318–325, IEEE, 2017.

[12] C. S. Vorugunti, "Ppmuas: A privacy preserving mobile user authentication system for cloud environment utilizing big data features," in *IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, pp. 1–6, IEEE, 2016.

[13] H. Jeong and E. Choi, "User authentication using profiling in mobile cloud computing," *AASRI Procedia*, vol. 2, pp. 262–267, 2012.

[14] U. M. A. Naushahi, "Profile-based access control in cloud computing environments with applications in health care systems," 2016.

[15] E. Shi, Y. Niu, M. Jakobsson, and R. Chow, "Implicit authentication through learning user behavior," in *International Conference on Information Security*, pp. 99–113, Springer, 2010.

[16] F. Aloul, S. Zahidi, and W. El-Hajj, "Two factor authentication using mobile phones," in *IEEE/ACS International Conference on Computer Systems and Applications*, (Rabat, Morocco), pp. 641–644, May 10-13, 2009.

[17] B. Sumitra, C. Pethuru, and M. Misbahuddin, "A survey of cloud authentication attacks and solution approaches," *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 2, no. 10, pp. 6245–6253, 2014.

[18] S. Malladi, J. Alves-Foss, and R. B. Heckendorn, "On preventing replay attacks on security protocols." University of Idaho, Moscow, Project funded in part by DARPA under grant MDA972-00-1-0001, Jan., 2002.

[19] A. Jesudoss and N. Subramaniam, "A survey on authentication attacks and countermeasures in a distributed environment," *Indian Journal of Computer Science and Engineering (IJCSE)*, vol. 5, no. 2, pp. 71–77, 2014.

[20] B. Nolan, "Identity theft attacks & countermeasures." GIAC Security Essentials Certificatio(GSEC) Practical Assignment Version 1.4c Option 1, SANS Institute, Oct., 2004.

[21] S. Lee, T. Y. Kim, and H.-J. Lee, "Mutual authentication scheme for cloud computing," in *Future Information Communication Technology and Applications*, pp. 149–157, Springer, May 2013.

[22] P. A. Boampong and L. A. Wahsheh, "Different facets of security in the cloud," in *Proceedings of the 15th Communications and Networking Simulation Symposium*, (Orlando, Florida), Society for Computer Simulation International, Mar. 26-30, 2012.

[23] C. Powell, T. Aizawa, and M. Munetomo, "Design of an sso authentication infrastructure for heterogeneous intercloud environments," (Luxembourg), IEEE 3rd International Conference on Cloud Networking (CloudNet), Oct. 8-10, 2014.

[24] J. M. Alves, T. G. Rodrigues, D. W. Beserra, J. C. Fonseca, P. T. Endo, and J. Kelner, "Multi-factor authentication with openid in virtualized environments," *IEEE Latin America Transactions*, vol. 15, no. 3, pp. 528–533, 2017.

[25] S. Ziyad and A. Kannammal, “A multifactor biometric authentication for the cloud,” in *Computational Intelligence, Cyber Security and Computational Models*, pp. 395–403, New Delhi, India: Springer, 2014.

[26] A. I. Ali A. Yassin, Hai Jin and D. Zou, “Anonymous password authentication scheme by using digital signature and fingerprint in cloud computing,” (Xiangtan, China), pp. 282–289, IEEE International Conference on Cloud and Green Computing (CGC), Nov. 1-3, 2012.

[27] A. A. Yassin, A. A. Hussain, and K. A.-A. Mutlaq, “Cloud authentication based on encryption of digital image using edge detection,” in *IEEE International Symposium on Artificial Intelligence and Signal Processing (AISP)*, (Mashhad, Iran), pp. 1–6, Mar. 3-5, 2015.

[28] S. Nagaraju and L. Parthiban, “Secauthn: Provably secure multi-factor authentication for the cloud computing systems,” *Indian Journal of Science and Technology*, vol. 9, March 2016.

[29] K. B. Raja, R. Raghavendra, M. Stokkenes, and C. Busch, “Multi-modal authentication system for smartphones using face, iris and periocular,” (Phuket, Thailand), pp. 143–150, IEEE International Conference on Biometrics (ICB), May 19-22, 2015.

[30] B. S. Al-Attab and H. Fadewar, “Authentication scheme for insecure networks in cloud computing,” in *IEEE International Conference on Global Trends in Signal Processing, Information Computing and Communication (ICGTSPICC)*, (Jalgaon, India), pp. 158–163, Dec. 22-24, 2016.

[31] Ö. M. Candan and A. Levi, “Robust two-factor smart card authentication,” (Istanbul, Turkey), IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom), Jun. 5-8, 2017.

[32] W.-i. Bae and J. Kwak, “Smart card-based secure authentication protocol in multi-server iot environment,” *Multimedia Tools and Applications, Springer*, pp. 1–19, 2017.

[33] A. Huszti and N. Oláh, “A simple authentication scheme for clouds,” in *IEEE Conference on Communications and Network Security (CNS)*, (Philadelphia, PA, USA), pp. 565–569, Oct. 17-19, 2016.

[34] R. Chow, M. Jakobsson, R. Masuoka, J. Molina, Y. Niu, E. Shi, and Z. Song, “Authentication in the clouds: a framework and its application to mobile users,” in *Proceedings of the 2010 ACM workshop on Cloud computing security workshop*, (Chicago, Illinois, USA), pp. 1–6, Oct. 08 - 08, 2010.

[35] W. Liu, A. S. Uluagac, and R. Beyah, “Maca: A privacy-preserving multi-factor cloud authentication system utilizing big data,” in *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, (Toronto, ON, Canada), pp. 518–523, Apr. 27 - May 2, 2014.

[36] A. S. Uluagac, W. Liu, and R. Beyah, “A multi-factor re-authentication framework with user privacy,” in *IEEE Conference on Communications and Network Security (CNS)*, (San Francisco, CA, USA), pp. 504–505, Oct. 29-31, 2014.

[37] M. H. Ibrahim, S. Kumari, A. K. Das, and V. Odelu, “Attribute-based authentication on the cloud for thin clients,” *The Journal of Supercomputing, Springer*, pp. 1–33, Jan., 2017.

[38] T. Bhattasali, K. Saeed, N. Chaki, and R. Chaki, “A survey of security and privacy issues for biometrics based remote authentication in cloud,” in *IFIP International Conference on Computer Information Systems and Industrial Management*, pp. 112–121, Springer, 2014.

[39] J.-C. Liou and S. Bhashyam, “A feasible and cost effective two-factor authentication for online transactions,” in *IEEE 2nd International Conference on Software Engineering and Data Mining (SEDM)*, (Chengdu, China), pp. 47–51, IEEE, June 23-25, 2010.

[40] M. Hoekstra, R. Lal, P. Pappachan, V. Phegade, and J. Del Cuvillo, “Using innovative instructions to create trustworthy software solutions.,” *HASP, Proceedings of the 2nd International Workshop on Hardware and Architectural Support for Security and Privacy*, vol. 11, June 23-24, 2013.

[41] J. L. Hernandez-Ardieta, A. I. Gonzalez-Tablas, J. M. De Fuentes, and B. Ramos, “A taxonomy and survey of attacks on digital signatures,” *Computers & security*, vol. 34, pp. 67–112, 2013.