Journal of
Sensor and Actuator Networks

MDPI

Article

D-FAP: Dual-Factor Authentication Protocol for Mobile Cloud Connected Devices †

Abdelrahman Abuarqoub

Department of Computer Science, Middle East University, 383 Amman 11831, Jordan; aabuarqoub@meu.edu.jo

† This paper is an extended version of my paper published in A. Abuarqoub, A Lightweight Two-Factor Authentication Scheme for Mobile Cloud Computing. In Proceedings of 3rd International Conference on Future Networks and Distributed Systems, Paris, France, 1–2 July 2019.

Received: 13 October 2019; Accepted: 16 December 2019; Published: 20 December 2019

Abstract: Emerging Mobile Cloud Computing (MCC) technologies offer a new world of promise by leveraging the quality of mobile services. With MCC, resource-constrained mobile devices could capitalize on the computation/storage resources of cloud servers via communication networks. While MCC adoption is growing significantly, several challenges need to be addressed to make MCC-based solutions scale and meet the ever-growing demand for more resource intensive applications. Security is a critical problem hindering the adoption of MCC. One of the most important aspects of MCC security is to establish authenticated communication sessions between mobile devices and cloud servers. The huge amount of data stored on mobile devices poses information security risks and privacy concerns for individuals, enterprises, and governments. The ability to establish authenticated communication sessions between mobile devices and cloud servers can resolve many security concerns. Limited computing and energy resources on mobile devices makes authentication and encryption a challenging task. In this paper, an overview of MCC authentication protocols is presented. Then, a Dual-Factor Authentication Protocol for MCC devices (D-FAP) is proposed. D-FAP aims at increasing authentication security by using multi-factors while offloading computation to the cloud to reduce battery consumption. The security of the protocol is formally verified and informal analysis is performed for various attacks. The results prove that the D-FAP is successful in mitigating various outsider and insider attacks.

Keywords: dual-factor authentication; smart card; password; mobile cloud computing; MCC security

1. Introduction

With the large-scale proliferation of smart mobile devices, an enormous amount of applications have been developed. Such applications can encounter significant challenges, including limited mobile battery life to cope with power hungry applications, insufficient bandwidth, complex, and varied mobile architecture and most importantly a potential attack vector for cybercriminal activities [1]. These applications also require high levels of storage and high processing capabilities. Mobile Cloud Computing (MCC) has been introduced to tackle these challenges. MCC combines such applications with the power-rich resources of the cloud.

Introducing MCC offers several benefits to mobile users. Firstly, MCC prolongs the lifetime of mobile devices. Offloading techniques allow transferring heavy weight computations and complex processing to the power rich cloud servers to utilise their capabilities, resulting in a significant power conservation of mobile devices. Secondly, MCC enables mobile users to store/access huge amounts of data on the cloud through wireless networks. For example, Google Photos can stores users' photos and videos on the clouds immediately after capturing and allow access to them from any connected device. This also contributes in conserving a considerable amounts of energy and storage capacity on their

mobile devices. Thirdly, MCC can efficiently support various tasks for data warehousing, managing and synchronizing multiple documents online. For instance, clouds can be used for playing online games, transcoding, or broadcasting multimedia services. Thus, all intensive processing that takes a long time when performed on mobile devices will be migrated the cloud. Finally, MCC improves reliability by backing up data and applications on a several of cloud servers. MCC can remotely provide security services such as virus scanning, malicious code detection, authentication, etc.

However, MCC poses several challenges. Mobile devices use a combination of heterogeneous wireless networks, networks that are known to be more energy intensive than their wired counterparts [2,3]. Furthermore, MCC provides the mobile device with a remote platform that is 'always online' with which the mobile device interacts frequently. The device is therefore communicating via the wireless network more often than usual, leading to increased levels of power consumption on the mobile device [2,4]. Another significant challenge for MCC is ensuring security, privacy, and trust [5,6]. Accessing cloud services grants access to a mobile device and potentially therefore access to personal or sensitive information, financial information, healthcare records, etc., which are stored on the device. Thus, more security is needed to ensure confidentiality and privacy of user's data which is stored in the cloud. Additionally, stronger authentication processes are needed to ensure access is only given to authorized persons [7–11].

Significant research has been carried out to explore ways to improve the security and efficiency of the authentication process, some of which puts heavy emphasis on the complexity and the number of authentication factors, neglecting the limitations addressed above [12–15]. Some research has attempted to address these limitations by moving the authentication processing into the cloud. However, it is well known that the process of communication can be power hungry; therefore, offloading data to the cloud to be processed seems counter-productive, especially if the mobile device is acting as a dumb terminal.

Migrating computation from the mobile device to the cloud seems to be an attractive notion as it minimizes the impact of the authentication processing on the mobile device, which in turn improves its performance. The impact of security algorithms on energy consumption was analyzed in ref. [16]. An experiment was conducted to measure the energy consumption during the processing of executing security protocols. It was found that, for a typical iPhone battery, with a capacity of 18 KJ, the experiment showed that, with encryption carried out on the device, the battery would run out of charge three times faster than when encryption was not carried out by the device.

To address the energy consumption issue, a significant amount of research has been undertaken to investigate computational offloading to improve performance and save energy for mobile devices [17]. Taking the illustrations above, the inspiration for D-FAP came from a need to address the energy consumption problem, at the same time as finding a secure and efficient method to carry out the authentication process without exerting too much pressure on the limited resources of the mobile device. In addition, aiming to achieve minimal energy consumption, two-factor authentication is implemented as it involves the minimum number of authentication factors. It has been shown that it will reduce power consumption and improve likeability and adoptability of security protocols [18].

This paper extends our previous work in ref. [19] which introduces the blueprints of a new authentication protocol that would address the mobile devices resource limitations to conserve their energy and address the security issues of the MCC. The protocol produces a secure and reliable dual factor authentication that is based on smart card and password authentication methods. In this paper, an implementation and evaluation of the proposed protocol has taken place. The performance of the protocol is evaluated via formal system tool "ProVerif" and informally analyzed for various attacks. Results demonstrate that the proposed protocol (D-FAP) is reliable and secure. D-FAP addresses the power issue of offloading the processing of the authentication to an alternative medium using a smart card, which is known to be more secure and efficient for carrying out such tasks.

The proposed work makes the following contributions: (1) D-FAP addresses the mobile devices resource limitations to conserve their energy by avoiding complicated algorithms with long and convoluted processing. (2) D-FAP addresses the security issues of the MCC even in the event of mobile

device loss. (3) The performance of D-FAP is evaluated to ensure it security against an active attackers. (4) D-FAP is compared against existing MCC authentication protocols to measure its viability.

The remainder of this paper is organized as follows. Section 2 surveys authentication protocols designed for smart mobile devices. Section 3 presents the proposed Dual-Factor Authentication Protocol for Mobile Cloud Connected Devices. Section 4 verifies the protocol and provides security analysis and comparisons. Section 5 concludes the paper.

2. An Overview of MCC Authentication Protocols

The exponential growth of the Internet interconnections has led to a significant growth of break-in attacks that can exploit vulnerabilities that exist within the applications, or in the operating system, on the mobile device [20–22]. Additionally, any attack on the cellular network can compromise the integrity of information; attacks such as International Mobile Subscriber Identity (IMSI) catchers [23], which feign man-in-the-middle attacks, can target user sensitive information, e.g., locate cellular phones, intercept communication content like text messages and phone calls. Thus, it is considered a real threat to all generations of mobile networks, breaching confidentiality through unauthorized access.

Several authentication approaches have been considered for authenticating users across many different computing platforms, including MCC. Some approaches considered the cryptographic based methods [24,25], where the authentication process is based on exchanging security keys or by using session tickets with trusted third-party entities. Other approaches focus on biometric authentication [26,27], which recognizes the user through physiological or behavioral characteristics unique to the user such as voice recognition, iris scanning, or finger prints. Approaches such as refs. [28,29], use password based authentication, which is the most widely used and accepted, due to its ease of use, its compatibility, scalability, as well as its low cost. To achieve secure, power-efficient and lightweight authentication for MCC mobile devices, dual-factor authentication protocol is introduced, in which two of the above methods are integrated ensuring minimum processing and energy consumption. The remainder of this section provides a comprehensive review of the above approaches.

2.1. Cryptographic Based Methods

Cryptographic based methods are one of the traditional methods used for authenticating users across many different computing platforms, including MCC [30]. The authentication process is based on exchanging security keys or by using session tickets with trusted third-party entities and using the same method for establishing a user's identity. A considerable research effort has been undertaken in this field.

In ref. [24], the authors proposed an authentication mechanism using an encrypted password and mobile phone token stored within a mobile phone as the authentication factors. In this mechanism, the registration and the management of the user identity information processes are undertaken by an identity provider (IdP). A Single Sign On (SSO) functionality is provided using secure Access mark Up Language and does not require a password table. This mechanism was a positive attempt to address SSO issues. The mechanism features mutual authentication between the mobile user and the IdP, which increases the security of the authentication process. Yet, the mechanism stores an encrypted password and a token on the mobile device. This could be exploited to compromise the system if the mobile device got stolen or lost.

In ref. [31], a two-factor authentication scheme based on a One Time Password (OTP) was proposed. It used the mobile device for authentication alongside a 4-digit PIN, through a Java application installed on the mobile device, and this in turn generated an OTP, a secret random number which was stored on the mobile device, and a time stamp. The OTP, the random number, and the time stamp were hashed using MD5 (one of a series of message digest algorithms), and the user name and OTP were sent to the remote server for authentication. This system used a time stamp and OTP to eliminate any replay attacks. And, it proved to be lightweight by reducing the number of authentication factors to two. However, it proposed to use MD5, which is found to be vulnerable to attacks such as brute force

attacks [32]. The scheme used the Java application to generate a 4-digit number to generate the OTP. However, if the mobile was stolen or lost, an attacker could use it to impersonate the user to login to the remote system.

The authors of ref. [25] proposed a lightweight protocol for mobile cloud environment based on local mobile network authentication. The protocol was divided into two phases; registration phase, and mutual authentication phase, where the mobile user registered with their mobile service provider by entering their International Mobile Station Identification number (IMSI) and a personal secret. The service provider issued the user with an Authentication Certificate (AC) and the user used the secret key provided with the certificate to encrypt the login messages. Once the login was successful, a session key was established and used for further communications. The protocol was lightweight as it used only two-factor authentication, symmetric encryption was also low in computational costs, and it also has a low latency. However, the protocol did not establish a secure channel, which made it susceptible to attacks. The AC was stored on the mobile device so if the mobile device was lost, stolen, or compromised, the device could be used for impersonation attacks.

2.2. Biometrics Based Methods

Biometrics is considered one of the promising authentication methods due to the fact that its resistance to losses and theft. Biometric data appear to provide the ideal means of identity verification, because they are unique and permanently attached to their owner. Although using biometrics methods has many advantages, it introduces its own challenges. It introduces privacy concern, biometrics may also disclose user's sensitive information, such as race, gender, even health conditions. Additionally, unlike knowledge-based, personal information, biometric data poses significant risks because it cannot be replaced once compromised. For instance, if an attacker obtains someone's fingerprint data, the victim's identity is permanently compromised [33].

Users cannot change their biometric over and over like passwords due to the fact that human has a limited number of biometric traits, making biometric templates are hard to be replaced. Recently, many smartphones manufacturing companies have been focusing on implementing biometric technologies on smartphones (e.g., fingerprint, iris scanning, facial recognition). However, biometric systems are made up of vulnerable components such as capture devices, communication channels, and databases, which are susceptible to several attacks [34]. The protection of the biometric data is crucial since biometric systems introduce vulnerabilities that can be exploited by hackers to break into the system.

The authors of ref. [27] proposed a real-time biometric recognition system that is based on Orthogonal Line Ordinal Features (OLOF) extraction technique. The processing of the authentication is distributed on the mobile device and the server. The use of palm print that can be easily captured provides more security than any other authentication methods. However, this system could lead to a delay in the authentication process due to the intensive processing that is carried out by the mobile device (i.e., the capturing of the image, the correction of alignment functions, and the collection of data in display image and transmission). Additionally, it does not protect the biometric data from exposure during the authentication process. This method showed that using biometrics to authenticate the user may affect a protocol's ability to be lightweight.

The authors of ref. [35] implement a novel authentication verification scheme that combines human inherence factor (handwritten signature biometrics) with the standard knowledge factor (traditional user-specified password) to achieve an enhanced level of security. The major computational load is shifted on a cloud based application server so that a platform-independent user verification service with ubiquitous access becomes possible. In this scheme, Google Application Engine (GAE) is used as a cloud service provider, a hierarchical approach is used to perform signature matching and to ensure the authenticity of biometric, decision forest classifier is used. The proposed scheme is easily scalable and is available to use on mobile platforms such as smart phones and PDAs. The cost and resource requirements of the proposed SaaS are low and independent of the user end platform. However, this scheme works well in smaller group environment but not suitable for large groups.

Recently, the authors of ref. [36] proposed a novel multifactor two-server authenticated scheme under mobile cloud computing (MTSAS). In the MTSAS, it divides the authentication method and authentication means; in the meantime, the user's biometric characteristics cannot leave the user device. Thus, MTSAS avoids the fingerprint information disclosure, protects user privacy, and improves the security of the user data. In addition, considering user actual requirements, MTSAS provides the different authentication factors depending on the privacy level of the authentication. However, this scheme suffers from a problem in case of the loss of the user device, apparently, it may take the problem of data redundancy in the cloud when the user performs registration again.

2.3. Password Based Methods

Among the diverse types of two-factor authentication mechanisms, the password based is one of the most widely used and accepted, due to its ease of use, its compatibility, scalability, as well as its low cost [37]. In such authentication protocols, the cloud authenticating server stores a user chosen low entropy password in a verifier table that is used to verify the existence of the user. This password is secured by adding a random number called salt, and the resultant is hashed using a one-way hash function such as SHA, NTLM, or MD5.

The authors of ref. [26] have proposed a fuzzy vault authentication protocol. The protocol is based on digital signatures and zero-knowledge authentication. Asymmetric RSA Keys were used to provide authentication for the mobile device and the server. A fuzzy picture password method was used to provide authentication and usability, especially when users have a difficult time remembering a password required for sufficient length and randomness to be secure. Upon completing the authentication process, a secure communication channel is set up to connect the mobile device with the cloud server. The server starts exchanging Diffie–Hellman (DH) public values with the client. This gives the protocol resistance to man-in-the-middle, impersonation attacks, reply attacks, and sniffing attacks. However, the high amount processing on the client side (mobile device) leads to energy wastage and less battery lifetime. Moreover, the existence of the fuzzy image data on mobile device could be exposed, exported, or copied in the case of device loss.

The authors of ref. [28] recently proposed a message digest authentication scheme (MDA) that strategically incorporates hashing, in addition to traditional user ID and passwords, to achieve mutual authentication. MDA consists of three stages: registration, verification, and update. MDA utilizes hashing, which supplements traditional user ID and password based authentication, to ensure differentiation and sum during the verification stage. The performance of MDA was evaluated using Scyther, a widely-used security protocol analyzer. Results show that MDA can withstand several attacks, such as man-in-the-middle, and replay attacks. However, an attacker could exploit the password tables by carrying out a password guessing attack to compromise users' passwords or to discover user specific details such as the user ID and the message digest. This leads to giving the attacker the ability to use these user credentials to impersonate the legitimate user, to login to other servers in the cloud.

In order to address the password leakage from compromised remote servers, the authors of ref. [29] proposed a highly efficient cryptographic protocol by distributing the files and the user data across multiple over multiple servers in the cloud. However, in this protocol, all servers jointly validate whether the password matches or not. Moreover, the protocol suffers from password leakage on the user side. Therefore, while improving server security in general is a crucial long-term goal, helping users make stronger passwords is an immediate step that can help to protect users.

2.4. Smart Card with Password Based Methods

Password authentication with smart card is considered one of the most convenient and effective dual-factor authentication protocols in distributed systems. It assures one communicating party of the authenticity of the corresponding party by acquisition of corroborative evidence. A smart card

with a password based method has been widely deployed for various kinds of daily applications—for instant, e-banking, e-government, and e-health applications.

A mobile user may initially insert the user's smart cards into a smart card reader electrically connected with their mobile devices, and then launch a reader application on the mobile device designed to retrieve the user's identity certificate from the smart card (a first authentication factor). Thereafter, the user may launch an application or navigate to a particular website whereby the user may be required to enter a password (a second authentication factor) before access to the system is granted. For instance, many military and governmental agencies maintain policies stipulating use of smart cards by military and civilian personnel for purposes of authenticating the holder's identity (e.g., a smart card may employ public key infrastructure (PKI)) [38].

Existing smartphones are not equipped with a smart card reader; thus, they suffer from their inherent limitations in relation to two-factor authentication using smart cards. Several of companies have attempted to introduce solutions to address this limitation. Thursby Software [39], for example, has developed the PKard reader that can be connected to Android and Apple devices. PKard readers are plug-in readers that support a wide range of smartphones from different vendors. Additionally, ACS [40] has developed smart card readers that use NFC contactless technology with Bluetooth technology. However, existing standards complicate the integration of smart card devices, and do not adequately address interoperability between products from multiple vendors.

Smart card based methods have several benefits. First, it is assumed to be tamper-resistant, i.e., the secret information stored in the smart card cannot be revealed. Therefore, it reduces the security risk in case of mobile device loss or theft. Second, it represents an efficient and reliable environment for carrying out authentication, hence, saving on the battery consumption of the mobile device. Third, it adds to the security of the MCC by protecting the privacy of information.

The first smart card authentication protocol based on the password method was introduced by ref. [41]. In this protocol, the authors proposed a remote authentication system based on the Chinese remainder theorem, which consisted of three phases. The login information was stored on a smart card, which was used by the user to login to the remote server, which would in turn check and verify the user credentials and accept or reject the user accordingly. In this protocol, the password was generated by a password-generating center.

Recently, many smart card authentication protocols based on the password were proposed [42–44]. Most of these protocols are still vulnerable to ID-theft attack, offline password guessing attacks, undetectable online password guessing attacks, and user impersonation attack.

The authors of ref. [45] show that the most important aim of the smart card based password authentication protocols was to achieve 'true' two-factor security. This meant that the only user who could access the authentication server was the one with full knowledge of the password and who was in possession of the smart card. They also emphasized that any two-factor protocol design should adhere to the following security criteria:

(a) An adversary with no knowledge of the user's password, but who had full access to the user's smart card and was able to extract the security information stored on the card, should not be able to extract the user's password by performing an offline guessing attack;
(b) An adversary with full knowledge of the user's password, but who was not in possession of the user's smart card, should not be able to attempt to impersonate the user to login to the cloud server.

There has been much research in the field exploring the use of smart cards using password based authentication. A common theme in all these protocols was that they used smart cards with a static user *ID*, which was susceptible to identity theft. This was because the authentication process was carried out over an insecure channel of communication. There have been many identity violations over the years and these have raised concerns of user privacy among many users and human rights organizations.

To address this specific concern, the authors of ref. [46] introduced a very innovative and novel mechanism to protect the user identity, by employing a method to dynamically change the user *ID* for

each transaction session. This eliminated *ID* theft or impersonation attacks. Many protocols such as ref. [46], however, did not employ mutual authentication. Although this protocol claimed it was secure against most attacks, it was shown by many protocols that followed that it was vulnerable to many attacks, such as denial of service and stolen verifier attacks [47].

Subsequently, many authentication protocols followed in the footsteps of ref. [46], such as refs. [48,49]. However, the development of a fully efficient and secure authentication protocol has proven to be a difficult task to achieve due to the limitations of smart card resources, e.g., battery life and storage capacity [50].

Authentication protocols based on symmetric key primitive, such as hash-based message authentication code (HMAC), XOR, and symmetric encryption are therefore more efficient and desirable than the more expensive asymmetric key primitive, such as point multiplication, exponentiation, and pairing.

Authentication protocols based on symmetric key primitive that are more suitable for smart cards were proposed in refs. [51,52]. However, all these protocols were vulnerable to various attacks as shown in refs. [45,53].

In 2009, the authors in ref. [54] proposed a dynamic *ID*, smart card based protocol. The protocol's security was analyzed [55]. Here, the authors showed that the protocol in ref. [54] was susceptible to impersonation attacks. An authorized user (adversary) could access the system legally using their credentials, could obtain other users' secret information, and, armed with this information, the adversary could then launch an offline dictionary attack to obtain those other users' passwords.

Ref. [55] proposed a protocol to enhance the security of ref. [54], and they claimed that their protocol was fully compliant with all the requirements set out in ref. [47]. The protocol did not store the user *ID* in the server and therefore claimed that the user *ID* was fully protected by anonymizing it and not storing it on the server.

Many other protocols followed ref. [55], which analyzed and criticized each other's protocol, exposing the weaknesses and proposing their own improvements to eliminate those vulnerabilities. However, they presented their new improved protocol without proper analysis of the security and efficiency aspects, and this was then followed by other researchers then repeating the same process, criticizing those protocols again and proposing their own improvements. An example of such a protocol is one that was introduced in 2012 in ref. [56]. The authors pointed out the security weaknesses of the protocol in ref. [55], proposing their own improvements, but, as it was not thoroughly researched and analyzed as explained; it was then attacked by other researchers. This is unfortunately the ongoing situation with most other research in this field.

In 2015, the authors of ref. [57] proposed a new authentication protocol for distributed MCC services based on Elliptic curve cryptography. The protocol aims at authenticating mobile users to access cloud computing services from multiple service providers who use only a single private key. It included three entities: user, smart card generator, and Cloud Service Provider (CSP). Public and private keys are generated when the user and CSP get registered with a smart card generator. Therefore, they can authenticate each of them without the involvement of the Smart card generator. The protocol reduces authentication processing time required by communication and computation between cloud service providers and a traditional trusted third party service. It is claimed that the protocol supports mutual authentication, key exchange, user anonymity, and can resist a lot of attacks and meets general security requirements. However, the protocol is insecure against a server forgery attack.

To mitigate the security weakness of the above protocol [57], the authors of ref. [58] have developed an authentication protocol for mobile users that is also based on using elliptic curve cryptography (ECC). The proposed protocol enables users to access MCC services from multiple CSP by using a single private key. The methodology improved the authentication phase to prevent server forgery attack and was validated in ProVerif automatic cryptographic protocol verifier [59], which showed that the proposed work being more secure and robust as compared to the work of [57]. However, due to the expensive cost of the bilinear pairing, their protocol did not yield better performance.

3. D-FAP in Detail

D-FAP mainly focuses on the most general case of smart-card-based password authentication, where the participants involve a set of users and a single remote server. A common design architecture model that was suggested in ref. [45] faithfully follows many designers of similar schemes. D-FAP is based on the same model which consists of three phases, i.e., registration, login, and verification (authentication).

In the registration phase, a user chooses the login credentials and fills a registration form providing some personal information to the server. The server issues a smart card which may contain some sensitive security parameters. After completing the registration phase, the user will be able to access the server in the authentication phase. The dual-factor protocol ensures that only the user who possesses both a valid smart card and the corresponding password can be successfully verified by the server. The notation used in D-FAP protocol is provided in Table 1.

Table 1. Notation used in D-FAP.

Notation	Description
C	Random number
CF	Column number where user information stored
$h(.)$	One-way hash function
ID	User's ID
MD_u	User message digest
N_i	Nonce
PW	User's password
S	Remote server identity
SC	Smart card
SI	State identifier
T	Time stamp
ΔT	Expected valid time interval
U	User
V_i, A_i	Security parameters
X_s	Secret value of server (to be stored on server)
y	Secret value of server (to be stored in smart card)
$\oplus$	Bitwise XOR computation
$\|\|$	Concatenation operation

3.1. Registration Phase

Prior to registration, the user obtains a digital certificate by registering with a digital certificate authority. The digital certificate comprises the public key and the digital signature of the user, along with its identity information. The registration phase is carried out only once unless the user re-registers. At this stage, the user enters their user ID and chooses a password. The password will be necessary later, and, in future logins, to ensure that the user is the legitimate owner of the smart card, i.e., is not an imposter. The user ID and password are processed and sent securely to the server.

On receipt of the user ID and password the server checks its user database to see whether the user is an existing user or a new user. If it is a new user, then the server registers the user and computes security parameters that are unique to the user. Security parameters are stored in a smart card, in a process called Smart Card Personalization. Smart cards are designed to be tamper proof and the user's security parameters, along with the server's secret key, are therefore, safely stored in the smart card, making it more difficult for an adversary to access these data. The registration phase is divided into two sub-phases: user registration and server registration.

3.1.1. User Registration

The user first registers with a Cloud Service Provider (CSP). At this stage, the following events will take place:

1. U signs up with a CSP a by accessing a secure registration URL;
2. U chooses a new User ID and password, at this point, the user is prompted to load their digital certificate (the user's public key is encapsulated within the public digital certificate). The CSP will use the information provided by the certificate to identify the user, and the public key encapsulated within the certificate, to encrypt future communication with the user;
3. To enhance the security of the user password, one of the most efficient ways is to employ 'salting' and 'hashing'. Once the user enters their credentials, the server remotely:
    a. generates a random number c on the mobile device side,
    b. concatenates the random number with the password (salting),
    c. hashes the outcome with a suitable one-way hash function (hashing), i.e.,: $RPW_i = h_0(c||PW_i)$,
    d. encrypts RPW_i *and* ID_i using the server's public key;
4. U uploads $\{RPW_i, ID_i\}$ to the server using a secure encrypted channel. This can be done by using an SSL protected URL. The main purpose of this is to protect the user credentials from being sniffed by an eavesdropper.

3.1.2. Server Registration

As soon as the remote server receives the user's registration, the following events are performed by the server:

1. Record the time of receipt of registration T_{reg}. This is used to identify if the user is a registered user and, if so, when s/he was registered, and also to generate the nonce;
2. Check if the user is a new user by checking its database of existing users. This is done to protect against user duplication;
3. If the user is a new user, the server checks the validity of U's digital certificate in order to guarantee its authenticity and it checks if the certificate belongs to the user being registered (if the user is already registered, then they will be rejected);
4. Obtain U's public key from the digital certificate, which is used to encrypt future communication with the client;
5. U's access policy file, including U's usage policy and access level, is concatenated with U's digital certificate, and the result is hashed using a suitable one-way function creating the message digest $MD_u = h_0$ ($User\ policy \parallel User\ certificate$). MD_u is used for mutual authentication with the client, and it can also be used to identify the user;
6. Generate the S secret keys x_s and y. x_s is used by the remote server to generate security parameters for future communications with the smart card. y is stored by the remote server in the smart card to be used by the smart card to generate security parameters for future communication with the server;
7. Create a new user account in the user database with the following parameters stored on the database: ID_i, T_{reg}, MD_u;
8. Compute $A_i = h_0(ID_i \oplus x_s)$;
9. Compute $V_i = A_i \oplus RPW_i$;
10. Create nonce $N_i = h_0(PW_i \parallel c) \oplus h_0(x_s \parallel ID_i \parallel T_{reg})$, where x_s is S's secret key;
11. Personalize the smart card by storing V_i, A_i, N_i, MD_u, $h(.)$, y, and $\#CF$, where $\#CF$ is the column number in the database where U's records are stored;
12. Send the smart card with the reader securely to the client with instructions on how to:
    (a) use the smart card with the smart card reader,
    (b) keep the smart card secure,
    (c) change the password,
    (d) unlock the card,

(e) report any loss or theft of the card.

NB. A_i and V_i are used to enhance the security methods for the smart card, to locally authenticate the user's password, without actually storing the user's password on board the smart card.

Figure 1 illustrates D-FAP registration phase.

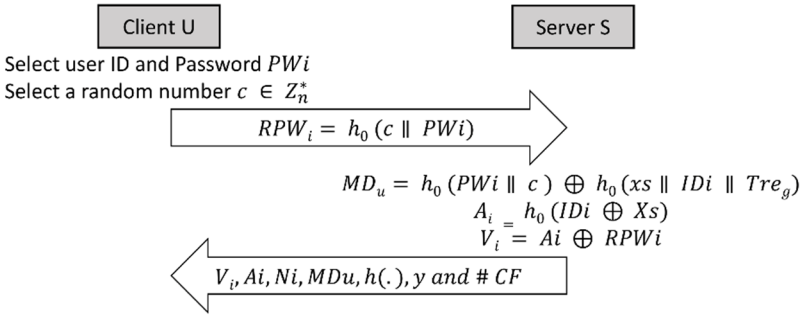

Figure 1. D-FAP registration phase.

3.2. Login Phase

The login process is used when the user intends to connect to the remote server. The user inserts his smart card in to the smart card reader and enters his/her user *ID* and password *PW*. The smart card validates the user credentials. If the user fails to enter the correct credentials, the smart card should initiate a credential's failure procedure to avoid password guessing attacks. Password guessing attacks may lead to impersonation attacks, or attempts by an attacker to change the password which can lead to a denial of service attack. However, if the user enters valid credentials, the smart card will generate an authentication message and send it to the server over the insecure channel.

It is noted that many design schemes tend to generate the authentication message without checking the validity of the user's credential first, making those schemes vulnerable to impersonation attacks. This needs to be avoided in the new design. The login phase comprises both the local and remote procedures.

3.2.1. Local Authentication between the *U* and *SC*

As *U* inserts the *SC* in to the smart card reader, *U* is prompted by the *SC* to enter their *ID* and password *PW*. The *SC* checks the validity of *U*'s *ID* and *PW* by:

1. Concatenating the newly entered *PW*′ with a random number *c* and hashing them;
2. Computing $A'_i = V_i \oplus h_0\ (PW'_i \| c)$;
3. Comparing A'_i with the A_i stored in its memory. If $A'_i = A_i$, then *SC* proceeds to the remote login operation, else the operation is terminated with the request for *U* to re-enter the password.

3.2.2. Remote Login between *SC* and *S*

Once the user is locally authenticated by the *SC*, the *SC* carries out the remote login process through the following steps:

1. *SC* generates $T_k = ID_i \oplus h_0(PW_i \| c)$;
2. T_k is used as a seed for Crypto Pseudo Random Number Generator (CPRNG) to generate *Kauth*, which is used to encrypt MD_u;
3. T_k is used to encrypt $\{\{MD_u\} \| SI\}$, where *SI* is the State Identifier;
4. To protect the anonymity of the user, CID_i is generated by creating a dynamic identifier, i.e., $CID_i = h_0(PW_i \| c) \oplus h_0(N_i \oplus y \oplus T)$;
5. *SC* computes $B_i = h_0\ (ID_i \oplus h_0(PW_i \| c))$;
6. *SC* computes $C_i = (T \oplus N_i \oplus B_i \oplus y)$, where *T* is the current time;

7. *SC* computes $D_i = \#CF \oplus \{\{MD_u\} \oplus SI\}$;
8. *SC* encrypts the message using's public key $\{CID_i, N_i, C_i, D_i, T\}$;
9. *SC* sends the encrypted message to the server *S*.

Because CID_i is dynamic, it means that the user *ID* cannot be predicted. c is random which makes $h_0\ (PW_i \parallel c)$ unpredictable and, therefore, user *ID* theft or a password guessing attack would be extremely difficult to achieve.

Figure 2 illustrates D-FAP login phase.

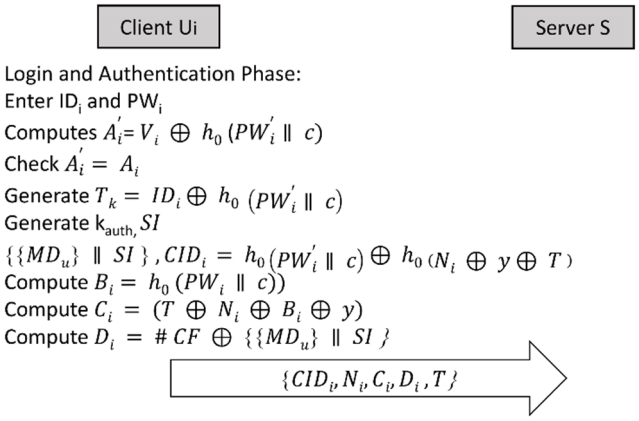

Figure 2. D-FAP login phase.

3.3. Verification Phase

The authentication phase commences when the authentication message is received by the server. The time difference between the time of arrival of the message and the time that the message was sent by the client is calculated. This time difference is important to avoid replay attacks. The server carries out multiple checks to validate the authenticity of the message. Once done, the next step is for the server to establish trust with the client. This is done by performing mutual authentication. The server initiates the mutual authentication process by sending a mutual authentication message to the client. The client, represented by the smart card, checks the mutual authentication message and, if it is authentic, then the smart card computes a session key to be used for future communications between the server and the client.

3.3.1. Server Authentication

Once *S* receives the authentication message, it carries out the following tasks:

1. Register the time of message arrival $T*$;
2. Calculate the time difference between arrival time $T*$ and authentication message sent time T, extracted from the received authentication message, i.e., $\Delta T = T* - T$. If the time window is acceptable, the message is accepted, else it is rejected. The time difference between the time of arrival of the message and the time the message sent is important to avoid replay attacks;
3. Extract *U*'s password $h_0(PW_i \parallel c) = CID \oplus h_0(N_i \oplus y \oplus T)$;
4. Compute $B_i = h_0(CID_i \oplus h_0(PW_i \parallel c))$;
5. Compute $C_i = (T \oplus N_i \oplus B_i \oplus y)$;
6. Check if $C_i* = C_i$. If so, then it proceeds to the next step, else, the user is rejected. This process is used to ensure the authenticity of the message received from *U*;
7. Use the shared column reference *#CF* to locate *U*'s record in the server database;
8. Use T_K to decrypt $\{\{MD_u\} \parallel SI\}$ in D_i to obtain *SI* and $\{MD_u\}$;
9. T_K is also used as a seed for *CPRNG* to generate *Kauth*, which is used to decrypt $\{MD_u\}$;

10. The decrypted MD_u is compared with the user MD_u stored in the server database. If they match, then the user is known to be a legitimate user, else the user authentication message is rejected;
11. For mutual authentication, S signs MD_u using the cloud's private key;
12. S uses U's public key to encrypt $\{T', sign\ \{MD_u\}, S\}$ and sends to the user, where T' is the current server time, and S is the server identity.

NB. and S in $\{T', sign\ \{MD_u\}, S\}$ are used to prevent impersonation and replay attacks.

3.3.2. Mobile Mutual Authentication

The process involving the response to the mutual authentication message sent by the server. When the mobile receives the mutual authentication message from S, the smart card SC carries out the following tasks:

1. Using the SC's private key, SC decrypts $\{T', sign\{MD_u\}, S\}$;
2. SC checks time $\Delta T = T'' - T'$ where T'' is the time of message arrival. If the time window is acceptable, the message is accepted, else it is rejected;
3. Using the cloud's public key, SC decrypts the signed message $sign\{MD_u\}$;
4. The decrypted MD_u is checked against the smart card's stored MD_u. If they match, then mutual authentication had been accomplished; otherwise, the message will be rejected.

Figure 3 illustrates D-FAP verification phase.

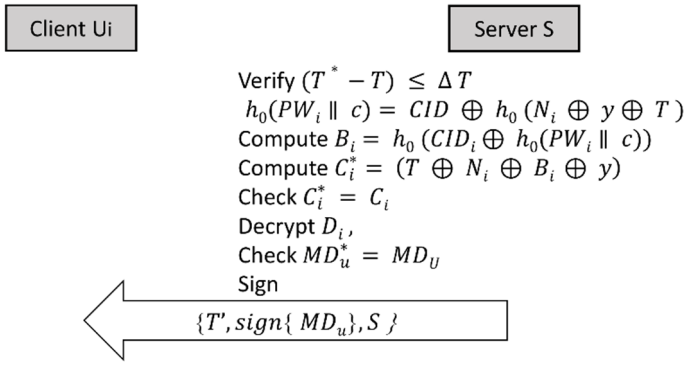

Figure 3. D-FAP verification phase.

3.3.3. Password Change Procedure

The following steps present the password change procedure:

1. The user U_i inserts the smart card into the smart card reader, and submits the current ID_i and PW_i, and requests to change the password
2. The user is prompted twice for a new PW_i*;
3. The Smart card computes $N_i^* = N_i \oplus h_0(PW_i \| c) \oplus h_0\left(PW_i' \| c\right)$;
4. The Smart card computes $V_i^* = A_i \oplus h_0\left(PW_i' \| c\right)$;
5. Then, once N_i will be replaced with N_i* and V_i will be replaced with V_i*. The password has been changed with the new password PW_i' and terminates the operation.

3.3.4. Lost Smart Card Revocation Procedure

An important feature of smart card-based authentication is revoking the lost smart cards without changing the user's identities. In a situation where the smart card is lost or stolen, the user must immediately request a cancellation of the card. The CSP validates the user's credentials supplied at initial registration. At that time, the user is required to supply their user digital certificate to provide

additional security for the CSP to be satisfied that it is a legitimate user and not an imposter. The CSP immediately revokes the lost smart card, and the user is requested to re-register with the CSP and with a new user's *ID*. To protect the user from impersonation attacks, the user is not allowed to use any of their previous 10 user *IDs* or passwords for the new registration.

4. D-FAP Security Analysis

4.1. Formal Verification by ProVerif

The formal security analysis uses an automated analysis tool called ProVerif [59]. ProVerif is a known automatic verifier for cryptographic protocols that verifies the security properties of secrecy, authentication and observational equivalences, under the assumption that the cryptographic primitives are idealized. Digital signatures, hash functions, signature proofs, etc. are suitable for analyzing an authentication protocol. Recently, many researchers [60–62] have verified the authentication in the user authentication protocol using ProVerif. The formal security analysis shows the results of verifying and analyzing the security of D-FAP protocol using ProVerif.

In the ProVerif attacker model, it is assumed that an attacker *A* has full control of the communication channels between the communicating parties. *A* is able to monitor, capture, modify, compose, send, and resend any communication messages on the communication channels. This means that *A* is able to fake any messages and impersonate any of the communicating entities. It can know the password of the user *U*, or the information stored on the smart card, but not both.

Table 2 elaborates channels, Constants, events, and functions used in D-FAP protocol. We use two channels of communication between the user *U* and the server *S*. A secured channel *prCH* that is used for registration, and a public channel *pubCH* which is an unsecured channel used for the login and authentication processes. The constant used in the communication process is the password. Two events are modelled for the Client *U*, an initialization event *initiate_ Clinet*, and a termination event *term_ Client*. Similarly, for the remote server *S* are *initiate_ RemS* and *term _ RemS*. The rest of the functions represent the various cryptographic or logical operations which are modelled in ProVerif as constructors and destructors.

Table 2. Define values and functions.

(* Channel *)	free pubCH: channel. free prCH: channel [Private].
(*Constants *)	Const PWi: bitstring [Private].
(* Events *)	event initiate_ Client (bitstring). event term_ Client (bitstring). event initiate_ RemS (bitstring). event term _ RemS (bitstring).
(* Functions *)	fun owh (bitstring): bitstring fun concat (bitstring, bitstring): bitstring fun Exr (bitstring, bitstring): bitstring fun SyEnc (bitstring, bitstring): bitstring fun pubk (seckey): secpubkey fun aenc (bitstring, privkey) fun sigpk (SSkey): sigpkey fun sig (bitsting, bitstring)

4.1.1. Simulation Processes

This subsection shows the modeling of the processes carried out by the two parties, *pUser*, the User process and *pServer*, the server process.

1. Registration phase

(∗ User_ process registration ∗)

$Let\ pUser = new\ ID_i : bitstring;$
$New\ c : bitstring;$
$Out\ (prCH,\ (ID_i, owh(concat\ (c.PW_i)));$
$in(prCH, (bV_i : bitstring, bN_i : bitstring, bA_i : bitstring, bMD_u : bitstring, by, b\#CF));$

In the registration phase, the *pUser* process is used to communicate securely to the server using the secure channel ID_i and $owh(concat(c, PW_i)$. The server checks the time T_{reg} and computes $V_i\ A_i, N_i, N_i$, and MD_u to *pUser*.

(∗ Server_ process registration ∗)

$Let\ pServer = new\ v : bitstring;$
$In\ (prCH.(VID_i : bitstring, vRPW_i : bitstring));$
$New\ Up : bitstring;$
$New\ Uc : bitstring;$
$Let\ MD_u = owh\ (concat(Up, Uc));$
$New x_s : bitstring;$
$Let\ A_i = owh(Exr(vID_i, x_s);$
$Let\ V_i = Exr(A_i, vRPW_i);$
$New\ Tr : bitstring;$
$Let\ N_i = Exr(vRPW_i, owh(concat(x_s, vID_i, Tr));$
$New\ y : bitstring;$
$New\ \#\ CF : bitstring;$
$Out\ (prCH, (V_i,\ A_i,\ N_i,\ MD_u,\ y,\ \#CF));$

2. Login phase

(* *pUser Login* ∗)

$let\ RPW_i' = owh(concat(c, PW_i))$
$let\ T_k' = (ID_i, RPW_i')$
$let\ A_i' = (bV_i, RPW_i', T_k')$
$if\ yA_i = A'_i\ then$
$let\ T_k = Exr(ID_i, RPW_i)$
$new\ Kauth : bitstring;$
$let\ EMD_u = (MD_u, Kauth)$
$new\ SI : bitstring$
$let\ E = SyE(Exr(EMD_u, SI), T_k)in$
$let\ B_i = owh(Exr(ID_i, RPW_i))in$
$new\ T : bitstring;$
$let\ C_i = Exr(T, bN_i, B_i, by)in$
$let\ CID_i = Exr(RPW_i, owh(Exr(bN_i, by, T)in$
$let\ D_i = Exr(b\#CF, E)in$
$out(pubCH, (CID, bN_i, C_i, D_i, T));$

3. Authentication phase

(∗ pServer Authentication ∗)

$Event\ init\_Server(vID_i);$
$in(pubCH(vCID_i : bitstring,\ vbN_i : bitstring,\ vC_i : bitstring,\ vD_i : bitstring,\ vT : bitstring));$
$new\ vT : bitstring;$
$let\ RPW_i' = Exr(vCID_i, owh(N_i,\ y,\ vT))$

$let\ B_i' = owh(Exr(vCID_i,\ RPW_i'))$
$let\ C_i' = Exr(vT, N_i, B_i', y)$
$if\ C_i' = C_i\ then$
$let\ E = Exr(vD_i, \#CF) in$
$let\ E' = SyD(E, T_k) in$
$new\ r : bitstring;$
$let\ r = ((MD_u), SI)$
$let\ SyE(MD_u) = Exr(r, SI)$
$new\ Kauth : bitstring;$
$let\ MD_u' = SyD(MD_u, Kauth)$
$if\ MD_u' = MD_u\ then$
$new\ T' : bitstring;$
$new\ sskey : bitstring$
$let\ a\ = (MD_u, sskey)$
$new\ upkey : bitstring;$
$new\ S : bitstring;$
$let\ c = aenc((T',\ sign(MD_u),\ S), upkey)$
$out(c);$
$event\ term\_Server(xID_i);$

(∗ pUser Mutual Authentication ∗)

$in(pubCH(vc))$
$New\ uskey : bitstring;$
$let\ d = adec(vc, uskey)$
$new\ spkey : bitstring;$
$let\ MD_u' = checksign(Sign(MD_u, Spkey))$
If $MD_u' = MD_u$ then
Let $P_i = owh(contact(CID_i, y))$
Let S_k = $owh(contact(D,\ p_i, y))$
Event term_User(ID_i);

4.1.2. Simulation Results

To guarantee that an attacker A is unable to obtain the user credentials, the following queries are run to test the security of the authentication process.

$$query\ id : bitstring : inj-event\ (term_{Client}(id)) ==>\ inj\ -\ event\ (intiate_{Client}(id)),$$

$$query\ id : bitstring : inj-\ event\ (term\_RemS))\ ==>\ inj\ -\ event\ (intiate\_\ RemS\ (id)).$$

Upon successful authentication, the session secret key S_k is used for future communication between the user and the server. To ensure that an attacker A is unable to discover the session key, the session key is given a private value. To further test the secrecy of S_k, the following queries are used:

$free\ S_k : bitstring\ [private]$
$query\ attacker\ (S_k).$

If the security queries result in a 'false', then an attack is possible and the attacker is able to discover the session key S_k. While in case the query result in a 'True', then the query was proven and the attack was not successful. The simulation results of D-FAP protocol are shown to be accurate for all events (see Table 3).

Table 3. ProVerif results.

Parameter	ProVerif Output
S_k	Secure
$event(term\_RemS(id)) \Rightarrow event(intiate\_RemS\ (id))$	True
$event(term\_Client(id-1212)) \Rightarrow event(intiate\_Client\ (1212))$	True
$not\ attacker(S_k)$	True

ProVerif has been used to test D-FAP for validating secrecy of the session key and the authentication between the user and the remote server. ProVerif queries are used to search for any valid security breaches that can be exploited by an attacker. The secrecy property of the D-FAP is tested to ensure that an attacker is unable to reach encrypted authentication messages that have been exchanged between the communicating parties using the insecure channel.

The adversary model used by the ProVerif tool to impersonate an adversary A, who engages with all the participants of the protocol that share the same insecure channel, is also implemented.

The results of the simulation show that both the user and server events start and terminate successfully and the security queries are applied successfully to the session key.

As shown in Table 3, in the results from the simulation of D-FAP, the secrecy of the session key S_k is maintained, and the attacker A is unable to obtain it. It also verifies the secrecy of the exchanged authentication messages, and shows that an attacker is unable to access these messages.

The results show that an attacker A, with a sequence of actions, cannot execute $event(initiate\_Server\ (id))$ before $event(term\_Server\ (id))$ and, therefore, cannot breach the security of the server. Furthermore, the same attacker A is not able to breach the user authentication by executing $event(intiate\_User\ (1212))$ before $event(term\_User(id-1212))$.

The results also show that the session key is secure and is not revealed to the attacker. Therefore, the claim that D-FAP maintains full secrecy has been proven by simulation.

4.2. Informal Security Analysis

In this section, we provide an informal security analysis of the proposed protocol to test its agility to withstand various possible attacks found in the current literature.

1. Denial of Service attack (DoS)

DoS is an intentional cyberattack carried out by an attacker who has already gained access to the smart card. This attack usually leads to excessive consumption of computational resources, damage of configuration information, or interruption of connection [63].

In some approaches, if the client requests to change their password, the smart card will not verify that the old name they input is similar to the original name. Hence, the attacker can invoke the password change procedure by inputting PW_i^{new}, replacing the original password PW_i. Therefore, the service will be denied for the legitimate client.

In D-FAP, the client credentials are authenticated by the smart card, which compares the user ID and password to the prestored credentials. The user is then requested to reenter the old password and only then will the request for a new password be prompted.

2. Masquerade attack

This attack is a type of spoofing attack that utilizes the vulnerabilities of security protocols to masquerade as one of the two parties to the communication [63]. Hence, an attacker uses a forged identity to gain unauthorized access. This can lead to modification or fabrication attack. In D-FAP, the login message is generated only if $C_i = (T \oplus N_i \oplus B_i \oplus y)$. The attacker could record $\{CID,\ N_i,\ D_i,\ T\}$ by eavesdropping on the communication between the user and the server, and could then extract CID_i from the message. However, the CID is dynamic in nature, which is different in each session and

encrypted using the server public key, and can only decrypted by the server's secret key. Also, N_i and D_i can only be computed by the user. The attacker can try to impersonate the server by recording $\{T', sign\{MD_u\}, S\}$, and trying to replay the message to the user. However, this is not possible since the identity S will be different.

3. Parallel session attack

The attacker applies new runs of authentication process using knowledge gathered from the initial run. Messages from these new runs of the protocol are replayed in the initial run. This allows an attacker to masquerade a legitimate user in another session [64].

Consider an attacker records $\{T', sig\{MD_u\}, S\}$, and then waits for the next authentication session, until the server sends it mutual authentication message back to the user. The attacker intercepts this message and replaces with his own message to the user, trying to impersonate the server.

For this to work, the attacker needs access to the private key of the user and the time stamp algorithm. The attacker can then decrypt the message and add in his time stamp $T*$. However, he cannot input his identity in place of the server ID S. Even if the attacker inputs his identity, the message will be $\{T*, sign\{MD_u\}.A\}$, which will be rejected, as the identity of the sender is different than the identity of the server.

4. Playback attack

The playback cure network communication intercepts it, and then fraudulently delays or resends it to either parties to masquerade as the server or the client. This can lead to injecting false messages, data corruption, or faking authentication credentials [65].

This is not possible with the proposed protocol due to the strong digital signatures that include time stamps. The server and the client check the time spent by the conveyed message between the two parties, taking into consideration the network delays. When this time exceeds the acceptable time limit, the message would be rejected.

5. Smart card loss attack

This attack can lead to changing the smart card password, or guessing the password of the user using password guessing attacks, or masquerading the legitimate user to login to the system [47]. When a smart card is lost or stolen, an attacker may access these parameters $\{V_i, A_i, N_i, MD_u, h(.), y\}$. In a protocol that is vulnerable to this type of attack, the attacker can use the details obtained from the card to perform various attacks such as password guessing attack, or impersonation attack, to login to the cloud. However, the login message would be $\{CID_i, N_i, C_i, D_i, T\}$. For the attacker to find the value of the hashed password from $h_0(PW_i \| c) = CID \oplus h_0(N_i \oplus y \oplus T)$, the attacker needs access to y, c and $h(.)$, to guess PW_i, hash it, and then calculate $CID \oplus h_0(N_i \oplus\ y \oplus T)$. This cannot be performed in polynomial time. Furthermore, the smart card consists of the following parameters $\{V_i, A_i, N_i, MD_u, h(.), y\}$. If it is lost or stolen, the only parameter that can reveal PW_i is V_i, which is $V_i = A_i \oplus\ RPW_i$. However, the attacker needs knowledge of T_k. This is not achievable in polynomial time. Therefore, an attacker cannot impersonate the owner of the card to login to the cloud. As a result, the proposed protocol is resistant to smart card loss attack.

6. Stolen verifier attack

In this attack, an attacker steals the password-verifier from the server's database and applies an offline guessing attack on it to get the client's exact password and, hence, he can masquerade as a legitimate client or the server, or can obtain the secret information [66]. D-FAP does not use password verifier tables, hence, it is immune to this type of attack.

7. Reflection attack

This attack is performed on mutual authentication protocol in which the attacker tricks the server into revealing the secret to its own challenge. The attacker creates a parallel session to initiate a valid session with the server. The attacker masquerades the legitimate user to request a login session from the server. The server attempts to authenticate the attacker by sending it a challenge, sending back a challenge, and waiting for a response. The attacker initiates another session with the server and sends the challenge. The server responds to the challenge; the attacker uses the response in the original session which will be validated by the server. Therefore, the attacker obtains access to the system resources with the privileges of a legitimate user [67].

This attack specifically targets protocols that use the challenge and response authentication system, in which an attacker uses the same protocol in both directions. D-FAP is immune to this type of attacks, since it does not use a challenge and response authentication system.

8. Insider attack

Insider attack is launched by someone with authorized system access who is purposely compromising the security. This can lead to violating the confidentiality, integrity, or availability of the system [68]. In a real-world environment, it is a common practice that many users use the same passwords to access different applications or servers for their convenience of remembering long passwords and use them easily. However, if the system manager or a privileged insider of S has known the passwords of U_i, he may try to impersonate U_i by accessing other servers where U_i could be a registered user. In D-FAP, the user ID and password are secured at every stage of the process, i.e., the password is never revealed in any of the processing during the authentication process.

D-FAP does not employ a password verifier table. It uses Dynamic ID that is dynamically changed each time the user logs in. Therefore, it can resist the insider attack and stolen verifier attack.

9. Password guessing attack

The adversary could perform brute force attack to compromise the low entropy password with the eavesdrops on the communication between the user and the server record [69]. The login message would be $\{CID_i, N_i, C_i, D_i, T\}$. For the attacker to find the value of the hashed salted password from $h_0(PW_i \| c) = CID \oplus h_0(N_i \oplus y \oplus T)$, the attacker needs access to y, c, and $h(.)$, to guess a PW_i, salt it with c, hash it, and then calculate $CID \oplus h_0(N_i \oplus y \oplus T)$. This cannot be done in polynomial time and has a low probability of achievement.

The smart card consists of the following parameters $\{V_i, A_i, N_i, MD_u, h(.), y\}$. If it is lost or stolen, the only parameter that can reveal PW_i is V_i, which is $V_i = A_i \oplus RPW_i$. However, the attacker needs knowledge of T_k. This is not achievable in polynomial time, Thus, password guessing attack is fruitless for D-FAP.

4.3. D-FAP Security Comparisons

The performance of D-FAP has been compared against two of its best rivals in the literature [57] and [58]. These protocols are similar to the D-FAP in spirit, but different in approach. Both protocols contribute to authentication in the MCC environment using smart cards and password methods.

When comparing D-FAP with the protocols proposed in refs. [57,58], using the security properties from the previous section, it has been shown that D-FAP is secure against all the specified attacks. However, an analysis of both protocols shows that they suffer from many security weaknesses. For instance, they suffer from smart card loss attack, and masquerade attacks. These vulnerabilities are due to the lack of defense mechanisms to counteract against smart card loss. D-FAP has presented a fully detailed procedure to protect against these attacks in the event of smart card loss.

A provision for generating a session key for future communication between the communicating parties is defined in D-FAP, while ref. [57] does not do this. It is important that both communicating

parties maintain security after successful authentication, to be able to exchange messages in complete secrecy, and to maintain confidentiality and privacy. D-FAP is shown to maintain secrecy for all the secret keys including the session key. However, in ref. [57], if an attacker has stolen the server's secret key xs, the attacker can easily compute the user's hashed password $h(PW_i) \oplus h(x_i)$ and impersonate that legitimate user.

Moreover, except for our scheme, other listed schemes [57,58] are proved to be susceptible to an insider attack. D-FAP does not employ a password verifier table. It uses a Dynamic *ID*, which is dynamically changed each time the user performs login. It is therefore extremely difficult for an insider, such as an administrator, to learn the legitimate user *ID* and password in order to masquerade as the user and gain access to other servers that the user has access to within the cloud.

Finally, R-TFAP is shown to be efficient for parallel session attacks because of the different message structures between the user and the server. Refs. [57,58] are vulnerable to these types of attacks; this is due to the ability to compute the session key by an unauthorized person using some eavesdropped communication.

In designing D-FAP, measures to overcome the weaknesses identified when analyzing the strengths and weaknesses of other similar protocols in the literature review have been implemented. This has added to the security, efficiency, strength, and attractiveness of the protocol. Table 4 presents the comparisons between D-FAP and other related authentication protocols.

Table 4. Comparisons of security properties with related works.

Attack	Tsai et al. [57]	Chaudhary et al. [58]	D-FAP
Resist offline password Guessing attack	yes	yes	yes
Prevent playback attack	yes	yes	yes
Minimize denial of service attack	yes	yes	yes
Prevent insider attack	no	no	yes
Prevent of masquerade attack	no	yes	yes
Prevent stolen verifier attacks	yes	yes	yes
Prevent reflection attack	yes	yes	yes
Prevent parallel session attack	no	yes	yes
Prevent smart card loss attack	no	no	yes

5. Conclusions

The insecure nature and heterogeneity of wireless communications have complicated the security and privacy needs in MCC. Energy limitations in mobile devices increase the demand for a lightweight security mechanism. An essential part of such a security mechanism is authentication, which secures any system against unauthorized access. Authentication should be lightweight, in order to minimize computation and communication costs.

This paper introduces a new authentication protocol, Dual-Factor Authentication Protocol for mobile cloud connected devices (D-FAP), that addresses the mobile devices resource limitations to conserve their energy and address the security issues of the MCC. D-FAP produces a secure and lightweight dual-factor authentication that is based on smart card and password authentication methods. It is shown that D-FAP satisfies the security requirements for user authentication and session key agreement and resists various kinds of known attacks.

6. Future Work and Suggested Applications

D-FAP has the potential to be developed further to meet demands. Recommendations for future development include: D-FAP could be developed further to be used to access enterprise systems in an MCC environment. For instance, supporting PKard Readers which can be connected to mobile devices. Furthermore, since the majority of today's smartphones are equipped with a fingerprint identity sensor, the password authentication method can be replaced with a biometric method.

Although this would improve the security even further, it would increase the energy consumption and hence reduce the efficiency. Moreover, D-FAP could be optimized for IoT systems.

It is envisaged that D-FAP could be applied across a variety of different industries. This could include: Businesses that support Bring Your Own Device (BYOD) policy, as it enhances security of access and mobility; or the banking industry, as it increases security and, because it reduces the number of factors used in the authentication process, it reduces the access time, thereby increasing its likeability; or the health care industry, which deals with highly sensitive data requiring high levels of privacy and confidentiality; or high security industries such as in government and defense.

Funding: This research received no external funding.

Acknowledgments: The author is grateful to the Middle East University, Amman, Jordan for the financial support granted to cover the publication fee of this research article.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Srivastava, S. Smartphone Triggered Security Challenges—Issues, Case Studies and Prevention. In *Cyber Security in Parallel and Distributed Computing*; John Wiley & Sons, Inc.: Hoboken, NJ, USA, 2019; pp. 187–206. [CrossRef]
2. Cui, Y.; Ma, X.; Wang, H.; Stojmenovic, I.; Liu, J. A Survey of Energy Efficient Wireless Transmission and Modeling in Mobile Cloud Computing. *Mob. Netw. Appl.* **2013**, *18*, 148–155. [CrossRef]
3. Ateya, A.; Muthanna, A.; Gudkova, I.; Abuarqoub, A.; Vybornova, A.; Koucheryavy, A. Development of Intelligent Core Network for Tactile Internet and Future Smart Systems. *J. Sens. Actuator Netw.* **2018**, *7*, 1. [CrossRef]
4. Ateya, A.; Muthanna, A.; Vybornova, A.; Gudkova, I.; Gaidamaka, Y.; Abuarqoub, A.; Algarni, A.; Koucheryavy, A. Model Mediation to Overcome Light Limitations—Toward a Secure Tactile Internet System. *J. Sens. Actuator Netw.* **2019**, *8*, 6. [CrossRef]
5. Baker, T.; Asim, M.; Dermott, Á.M.; Iqbal, F.; Kamoun, F.; Shah, B.; Alfandi, O.; Hammoudeh, M. A secure fog-based platform for SCADA-based IoT critical infrastructure. *Software* **2019**. [CrossRef]
6. Muthanna, A.; Ateya, A.; Khakimov, A.; Gudkova, I.; Abuarqoub, A.; Samouylov, K.; Koucheryavy, A. Secure and Reliable IoT Networks Using Fog Computing with Software-Defined Networking and Blockchain. *J. Sens. Actuator Netw.* **2019**, *8*, 15. [CrossRef]
7. Irshad, A.; Chaudhry, S.A.; Shafiq, M.; Usman, M.; Asif, M.; Ghani, A. A provable and secure mobile user authentication scheme for mobile cloud computing services. *Int. J. Commun. Syst.* **2019**, *32*, e3980. [CrossRef]
8. Mo, J.; Hu, Z.; Chen, H.; Shen, W. An Efficient and Provably Secure Anonymous User Authentication and Key Agreement for Mobile Cloud Computing. *Wirel. Commun. Mob. Comput.* **2019**, *2019*. [CrossRef]
9. Atwady, Y.; Hammoudeh, M. A Survey on Authentication Techniques for the Internet of Things. In Proceedings of the International Conference on Future Networks and Distributed Systems, Cambridge, UK, 19–20 July 2017.
10. Wang, P.; Li, B.; Shi, H.; Shen, Y.; Wang, D. Revisiting Anonymous Two-Factor Authentication Schemes for IoT-Enabled Devices in Cloud Computing Environments. *Secur. Commun. Netw.* **2019**, *2019*. [CrossRef]
11. Epiphaniou, G.; Walshe, M.; Al-Khateeb, H.; Hammoudeh, M.; Katos, V.; Dehghantanha, A. Non-Interactive Zero Knowledge Proofs for the Authentication of IoT Devices in Reduced Connectivity Environments. *Ad Hoc Netw.* **2019**, *95*, 101988.
12. Balasubramanian, N.; Balasubramanian, A.; Venkataramani, A. Energy Consumption in Mobile Phones: A Measurement Study and Implications for Network Applications. In Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement 2009, Chicago, IL, USA, 4–6 November 2009; pp. 280–293. [CrossRef]
13. Aloraini, A.; Hammoudeh, M. A Survey on Data Confidentiality and Privacy in Cloud Computing. In Proceedings of the International Conference on Future Networks and Distributed Systems, Cambridge, UK, 19–20 July 2017; pp. 1–7.
14. Belguith, S.; Kaaniche, N.; Hammoudeh, M. Analysis of attribute-based cryptographic techniques and their application to protect cloud services. *Trans. Emerg. Telecommun. Technol.* **2019**, e3667. [CrossRef]

15. Fathi, R.; Salehi, M.A.; Leiss, E.L. User-Friendly and Secure Architecture (UFSA) for Authentication of Cloud Services. In Proceedings of the 2015 IEEE 8th International Conference on Cloud Computing, New York, NY, USA, 27 June–2 July 2015; pp. 516–523.
16. Potlapally, N.R.; Ravi, S.; Raghunathan, A.; Jha, N.K. Analyzing the energy consumption of security protocols. In Proceedings of the 2003 International Symposium on Low Power Electronics and Design, New York, NY, USA, 27–27 August 2003; pp. 30–35.
17. Forman, G.H.; Zahorjan, J. The challenges of mobile computing. *Computer* **1994**, *27*, 38–47. [CrossRef]
18. Liao, I.E.; Cheng-Chi, L.; Min-Shiang, H. Security enhancement for a dynamic ID-based remote user authentication scheme. In Proceedings of the International Conference on Next Generation Web Services Practices (NWeSP'05), Seoul, Korea, 22–26 August 2005; p. 4.
19. Abuarqoub, A. A Lightweight Two-Factor Authentication Scheme for Mobile Cloud Computing. In Proceedings of the 3rd International Conference on Future Networks and Distributed Systems, Paris, France, 1–2 July 2019; pp. 1–7.
20. Carlin, A.; Hammoudeh, M.; Aldabbas, O. Intrusion Detection and Countermeasure of Virtual Cloud Systems—State of the Art and Current Challenges. *Int. J. Adv. Comput. Sci. Appl.* **2015**, *6*. [CrossRef]
21. Ghafir, I.; Prenosil, V.; Svoboda, J.; Hammoudeh, M. A Survey on Network Security Monitoring Systems. In Proceedings of the 2016 IEEE 4th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW), Vienna, Austria, 22–24 August 2016; pp. 77–82.
22. Teing, Y.-Y.; Homayoun, S.; Dehghantanha, A.; Choo, K.-K.R.; Parizi, R.M.; Hammoudeh, M.; Epiphaniou, G. Private Cloud Storage Forensics: Seafile as a Case Study. In *Handbook of Big Data and IoT Security*; Dehghantanha, A., Choo, K.-K.R., Eds.; Springer International Publishing: Cham, UK, 2019; pp. 73–127. [CrossRef]
23. Khan, M.S.A.; Mitchell, C.J. Trashing IMSI catchers in mobile networks. In Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks, Boston, MA, USA, 18–20 July 2017; pp. 207–218.
24. Binu, S.; Mohan, A.; Deepak, K.T.; Manohar, S.; Misbahuddin, M.; Raj, P. A proof of concept implementation of a mobile based authentication scheme without password table for cloud environment. In Proceedings of the 2015 IEEE International Advance Computing Conference (IACC), Bangalore, India, 12–13 June 2015; pp. 1224–1229.
25. Momeni, M. A Lightweight Authentication Scheme for Mobile Cloud Computing. *Int. J. Comp. Sci. Bus. Inf.* **2014**, *14*, 153–160.
26. Schwab, D.; Yang, L. Entity authentication in a mobile-cloud environment. In Proceedings of the Eighth Annual Cyber Security and Information Intelligence Research Workshop, Oak Ridge, TN, USA, 8–10 January 2013; pp. 1–4.
27. Moço, N.F.; Técnico, I.S.; Telecomunicações, I.D.; Correia, P.L. Smartphone-based palmprint recognition system. In Proceedings of the 2014 21st International Conference on Telecommunications (ICT), Lisbon, Portugal, 4–7 May 2014; pp. 457–461.
28. Dey, S.; Sampalli, S.; Ye, Q. MDA: Message digest-based authentication for mobile cloud computing. *J. Cloud Comput.* **2016**, *5*, 18. [CrossRef]
29. Camenisch, J.; Lehmann, A.; Neven, G. Optimal Distributed Password Verification. In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, 12–16 October 2015; pp. 182–194.
30. Moffat, S.; Hammoudeh, M.; Hegarty, R. A Survey on Ciphertext-Policy Attribute-based Encryption (CP-ABE) Approaches to Data Security on Mobile Devices and its Application to IoT. In Proceedings of the International Conference on Future Networks and Distributed Systems (ICFNDS'17), Cambridge, UK, 19–20 July 2017; p. 34. [CrossRef]
31. Yang, G.; Wong, D.S.; Wang, H.; Deng, X. Two-factor mutual authentication based on smart cards and passwords. *J. Comput. Syst. Sci.* **2008**, *74*, 1160–1172. [CrossRef]
32. Xie, T.; Liu, F.; Feng, D. Fast Collision Attack on MD5. *IACR Cryptol. ePrint Arch.* **2013**, *2013*, 170.
33. Boone, G.; Huang, J.; Spiegeleire, S.D.; Sweijs, T. *Future Issue Biometrics: The Uncertainty of Identification Authentication: 2010–2020*; The Hague Centre for Strategic Studies: The Hague, The Netherlands, 2009.
34. Ahmad, S.; Mohd Ali, B.; Wan Adnan, W.A. Technical issues and challenges of biometric applications as access control tools of information security. *Int. J. Innov. Comput. Inf. Control* **2012**, *8*, 7983–7999.

35. Khan, S.H.; Akbar, M.A. Multi-Factor Authentication on Cloud. In Proceedings of the 2015 International Conference on Digital Image Computing: Techniques and Applications (DICTA), 23–25 November 2015; pp. 1–7.
36. Han, Z.; Yang, L.; Wang, S.; Mu, S.; Liu, Q. Efficient Multifactor Two-Server Authenticated Scheme under Mobile Cloud Computing. *Wirel. Commun. Mob. Comput.* **2018**, *2018*. [CrossRef]
37. Ghafir, I.; Saleem, J.; Hammoudeh, M.; Faour, H.; Prenosil, V.; Jaf, S.; Jabbar, S.; Baker, T. Security threats to critical infrastructure: The human factor. *J. Supercomput.* **2018**, *74*, 4986–5002. [CrossRef]
38. Fiducia, K.J.; Thomas, J.F.; Schmerge, P.S. Mobile Enterprise Smartcard Authentication. U.S. Patent 9,083,703, 14 July 2015.
39. Kard, P. Thursby Software. Available online: http://www.thursby.com/ (accessed on 12 December 2019).
40. ACS. Advanced Card Systems Ltd. Available online: https://www.acs.com.hk/en/ (accessed on 12 December 2019).
41. Chang, C.; Wu, T. Remote password authentication with smart cards. *IEE Proc.* **1991**, *138*, 165–168. [CrossRef]
42. Radhakrishnan, N.; Karuppiah, M. An efficient and secure remote user mutual authentication scheme using smart cards for Telecare medical information systems. *Inform. Med. Unlocked* **2018**. [CrossRef]
43. Chen, C.-L.; Deng, Y.-Y.; Tang, Y.-W.; Chen, J.-H.; Lin, Y.-F. An Improvement on Remote User Authentication Schemes Using Smart Cards. In Proceedings of the second International Conference on Mobile Ad-Hoc and Sensor Networks, Hong Kong, China, 13–15 December 2006; pp. 416–423.
44. Zhao, Y.; Li, S.; Jiang, L. Secure and Efficient User Authentication Scheme Based on Password and Smart Card for Multiserver Environment. *Sec. Comm. Netw.* **2018**, *2018*, 9178941. [CrossRef]
45. Wang, D.; He, D.; Wang, P.; Chu, C. Anonymous Two-Factor Authentication in Distributed Systems: Certain Goals Are Beyond Attainment. *IEEE Trans. Dependable Secur. Comput.* **2015**, *12*, 428–442. [CrossRef]
46. Das, M.L.; Saxena, A.; Gulati, V.P. A dynamic ID-based remote user authentication scheme. *IEEE Trans. Consum. Electron.* **2004**, *50*, 629–631. [CrossRef]
47. Madhusudhan, R.; Mittal, R.C. Dynamic ID-based remote user password authentication schemes using smart cards: A review. *J. Netw. Comput. Appl.* **2012**, *35*, 1235–1248. [CrossRef]
48. Yoon, E.-J.; Yoo, K.-Y. Improving the Dynamic ID-Based Remote Mutual Authentication Scheme. In Proceedings of the OTM Confederated International Conferences, On the Move to Meaningful Internet Systems 2006, Montpellier, France, 29 October–3 November 2006; pp. 499–507.
49. Lee, C.-C.; Hwang, M.-S.; Yang, W.-P. A flexible remote user authentication scheme using smart cards. *SIGOPS Oper. Syst. Rev.* **2002**, *36*, 46–52. [CrossRef]
50. Zhu, Y.; Ma, D.; Huang, D.; Hu, C. Enabling secure location-based services in mobile cloud computing. In Proceedings of the Second ACM SIGCOMM Workshop on Mobile Cloud Computing, Hong Kong, China, 16 August 2013; pp. 27–32.
51. Chaudhry, S.A.; Farash, M.S.; Naqvi, H.; Kumari, S.; Khan, M.K. An enhanced privacy preserving remote user authentication scheme with provable security. *Secur. Commun. Netw.* **2015**, *8*, 3782–3795. [CrossRef]
52. Chen, C.-T.; Lee, C.-C. A two-factor authentication scheme with anonymity for multi-server environments. *Secur. Commun. Netw.* **2015**, *8*, 1608–1625. [CrossRef]
53. Chaudhry, S. Comment on Robust and Efficient Password Authenticated Key Agreement with User Anonymity for Session Initiation Protocol Based Communications. *IET Commun.* **2015**, *9*, 1034. [CrossRef]
54. Wang, Y.-Y.; Liu, J.-Y.; Xiao, F.-X.; Dan, J. A more efficient and secure dynamic ID-based remote user authentication scheme. *Comput. Commun.* **2009**, *32*, 583–585. [CrossRef]
55. Wen, F.; Li, X. An improved dynamic ID-based remote user authentication with key agreement scheme. *Comput. Electr. Eng.* **2012**, *38*, 381–387. [CrossRef]
56. Tang, H.-B.; Liu, X.-S. Cryptanalysis of a dynamic ID-based remote user authentication with key agreement scheme. *Int. J. Commun. Syst.* **2012**, *25*, 1639–1644. [CrossRef]
57. Tsai, J.; Lo, N. A Privacy-Aware Authentication Scheme for Distributed Mobile Cloud Computing Services. *IEEE Syst. J.* **2015**, *9*, 805–815. [CrossRef]
58. Chaudhry, S.A.; Kim, I.L.; Rho, S.; Farash, M.S.; Shon, T. An improved anonymous authentication scheme for distributed mobile cloud computing services. *Clust. Comput.* **2019**, *22*, 1595–1609. [CrossRef]
59. Blanchet, B.; Cheval, V.; Allamigeon, X.; Smyth, B.; Sylvestre, M. ProVerif: Cryptographic Protocol Verifier in the Formal Model. Available online: https://prosecco.gforge.inria.fr/personal/bblanche/proverif/manual.pdf (accessed on 12 December 2019).

60. Wu, F.; Xu, L.; Kumari, S.; Li, X. An improved and provably secure three-factor user authentication scheme for wireless sensor networks. *Peer-to-Peer Netw. Appl.* **2018**, *11*, 1–20. [CrossRef]
61. Lee, H.; Lee, D.; Moon, J.; Jung, J.; Kang, D.; Kim, H.; Won, D. An improved anonymous authentication scheme for roaming in ubiquitous networks. *PLoS ONE* **2018**, *13*. [CrossRef]
62. Ryu, J.; Lee, H.; Kim, H.; Won, D. Secure and Efficient Three-Factor Protocol for Wireless Sensor Networks. *Sensors* **2018**, *18*, 4481. [CrossRef]
63. Chen, Q.; Zhang, C.; Zhang, S. Overview of Security Protocol Analysis. In *Secure Transaction Protocol Analysis: Models and Applications*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 17–71.
64. Shieh, W.; Wang, M. A New Parallel Session Attack to Khan-Zhang's Authentication Scheme. In Proceedings of the 2008 3rd International Conference on Innovative Computing Information and Control, DaLian, China, 18–20 June 2008; p. 154.
65. Yeh, K.-H.; Su, C.; Lo, N.W.; Li, Y.; Hung, Y.-X. Two robust remote user authentication protocols using smart cards. *J. Syst. Softw.* **2010**, *83*, 2556–2565. [CrossRef]
66. Ku, W.C.; Tsai, H.; Tsaur, M.J. Stolen-verifier attack on an efficient smartcard-based one-time password authentication scheme. *IEICE Trans. Commun.* **2004**, *87*, 2374–2376.
67. Sumitra, B.; Pethuru, C.R.; Misbahuddin, M. A survey of cloud authentication attacks and solution approaches. *Int. J. Innov. Res. Comput. Commun. Eng.* **2014**, *2*, 6245–6253.
68. Zhan, J.; Fan, X.; Han, J.; Gao, Y.; Xia, X.; Zhang, Q. CIADL: Cloud insider attack detector and locator on multi-tenant network isolation: An OpenStack case study. *J. Ambient Intell. Humaniz. Comput.* **2019**. [CrossRef]
69. Ding, Y.; Horster, P. Undetectable on-line password guessing attacks. *SIGOPS Oper. Syst. Rev.* **1995**, *29*, 77–86. [CrossRef]