Authentication and Authorization Issues in Mobile Cloud Computing: A Case Study

V. Carchiolo[1], A. Longheu[2], M. Malgeri[2], S. Ianniello[3], M. Marroccia[3] and A. Randazzo[3]
[1]*Dipartimento di Matematica ed Informatica, Università degli Studi di Catania, Catania, Italy*
[2]*Dip. di Ingegneria Elettrica, Elettronica e Informatica, Università degli Studi di Catania, Catania, Italy*
[3]*STMicroelectronics, ICT Technical Excellence Center, Catania and Naples, Italy*

Keywords: Cloud Computing, Authentication, Authorization, Accounting, Security, Mobile Devices.

Abstract: Mobile Cloud Computing incorporates Cloud computing paradigma into the mobile environment, the set of technologies that enable to access network services anyplace, anytime and anywhere. This faces many technical challenges, such as low bandwidth, availability, heterogeneity, computing offloads, data accessing, security, privacy, and trust. In this paper the MCC security solution developed and applied within the STMicroelectronics plants is presented.

1 INTRODUCTION

We are living in the Age of Networked Intelligence and our individual environment as well as company's world is being re-imagined.

The key enablers of this radical mutation include a wide range of voracious consumers, a business culture and a rise in global flows of goods, services, capitals and information. Moreover, a constellation of technologies endorses this phenomena: Internet, mobile and broadband networking, big data and analytic and smart, digitally interconnected things. This easily leads to the *Cloud Computing*, that enables consumers, small and big businesses, corporations and governments to easily access computing resources with little or no effort, upfront investment or commitment.

Mobile Cloud Computing (MCC) incorporates Cloud computing into the mobile environment, the set of technologies that enable people to access network services anyplace, anytime, and anywhere.

MCC refers to an infrastructure where both data storage and the processing occur outside of the mobile device. Mobile Cloud applications move the computing power and data storage away from mobile phones and into the Cloud, bringing application and mobile computing to a much broader range of mobile subscribers than just smartphone users.

The integration of Cloud computing and mobile networks faces many technical challenges, such as low bandwidth, availability, heterogeneity, computing offloads, data accessing, security, privacy, and trust, all enforced by the dramatic increase in the use of smartphones in recent years.

According to this scenario, a specific aspect of mobile Cloud computing security is addressed in this work: authentication and authorization. As in several works arises (Todorov, 2007) (Alizadeh et al., 2016), resources that discuss in depth authentication technologies either focus on mechanisms provided by specific products or services, or on the theory behind user authentication with complete detachment from industry solutions. Here we illustrate the MCC security solution developed and applied within the STMicroelectronics® IC manufacturer plants; the proposed system allows users authentication and authorization according to the security standards and lies in the Cloud and mobile Cloud environment introduced previously. In particular, the solution addresses the following issues

- Reduce and/or eliminate the need to store multiple passwords and multiple username for different services
- Reduce time spent re-entering passwords and username for the same identity
- Simplify the definition and management of security policies

In addition, the solution also has to take into account the complexity of both the geographical and

DOI: 10.5220/0007658602490256
In *Proceedings of the 9th International Conference on Cloud Computing and Services Science (CLOSER 2019)*, pages 249-256
ISBN: 978-989-758-365-0

organizational extension of the company with all the resulting constraints, also considering many interconnected ERP activities as supply chain (including factory automation), finance, purchasing, sales and marketing (compliant with today's cutting-edge Cloud platforms, e.g. (salesforce, 2019)) and business intelligence.

The rest of paper is organized as follows: in sections 2 and 3 we first introduce the Cloud paradigm shifting towards the MCC extension, providing an overview of relevant architectures, issues and applications and finally its security issues, in authentication and authorization. In section 4 the solution adopted within STMicroelectronics is shown, together with a description of the tools and technologies used. Our conclusions are finally discussed in section 5.

2 MCC AND RELATED ISSUES

Cloud Computing is a model for enabling ubiquitous, convenient, on demand, network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction (Peter Mell, 2009).

According to (Sanaei et al., 2014), the Mobile Cloud Computing (MCC) is "a rich mobile computing technology that leverages unified elastic resources of varied clouds and network technologies toward unrestricted functionality, storage, and mobility to serve a multitude of mobile devices anywhere, anytime ... based on the pay-as-you use principle".

MCC is actually an extension of cloud computing and incorporates mobile computing and wireless networking aiming to provide typical cloud services to the mobile consumers. Substantial difference with the classic cloud computing service is that in MCC execution time and energy consumption are fundamental aspects and both are significantly improved by transferring execution of resource-intensive application from the hosting mobile device to the cloud-based resource, therefore mobile devices exploiting MCC are not required high resources. In addition to data processing, in MCC also data storage is shifted to cloud servers. Migrating these two workloads made MCC an useful solution to obtain higher-level service performance on mobile devices despite their (local) resources.

2.1 Security

MCC joins several technologies, therefore it inherits their pros and cons.

Here we want to focus on security issues, therefore we first consider generic cloud services, shifting afterwards to MCC case. Cloud services can be generally considered high risky since they may:

- permit risky behaviour such as anonymous use
- lack basic security features such as encryption in transit and admin activity logging
- have sneaky terms and conditions that put data at risk

Cloud-based information systems are exposed to threats that can have adverse effect on organizational assets; risks to be addressed occur at three levels: organizational, mission and business process and information system level. As well as traditional information systems, in cloud-based information systems risks must be managed throughout the system development life cycle. Moreover, in a cloud ecosystem, the complex relationship among cloud actors and their processes require an integrated, ecosystem-wide risk management framework.

When considering MCC, although several security mechanisms have been developed, it is still not safe enough for many users to offload their personal data to the cloud. This perception is stronger than in Cloud Computing because of the problems inherent in the mobile environment.

While different measures can be implemented to improve security of the services offered in the MCC, its vulnerabilities are still related to technologies it is based on (mobile computing, wireless networks, and cloud computing), in particular:

1. Security of mobile devices

   At this level, the main concern is the security of handheld devices which have open operating systems, third-party applications, and wireless access to the Internet anywhere, anytime, therefore, they are vulnerable to the threats and risks as in PCs and desktops. There are different approaches to lessen the security issues related to mobile devices. Antimalware programs are run on the devices to identify and delete Trojan horse, viruses, and worms. Periodically updating the OS and downloading applications from known vendors such as Google, Apple, and Microsoft can also help. Also, unexplained links should not be tried, receiving data transmission from strange phones should be avoided, new, unauthorized software should not be installed, and the interface of Bluetooth, Wi-Fi, and so on, should be shut down. If a device is stolen or lost, there must be some remote data wiping technique so that the data cannot be misused. The mobile device can also use hardware-based encryption techniques for internal and external memory support.

2. Security of Wireless Communication Channel.

   This level deals with issues related to securing the wireless communication channel between the mobile and cloud servers. Mobile devices access their resources and services deployed in the cloud environment through communication channels from cloud servers. This increases the number of wireless application protocol gateways and IP multimedia subsystem equipment in the IP network, giving rise to many new security threats in the mobile Internet. These mobile terminals access phone service, SMS, and other Internet services using 3G, 4G, Wi-Fi, WiMax, and Bluetooth. These broad access methods may increase security risks associated with networks, causing information leakage and malicious attacks.

   To protect data from leakage during communication between mobile device and cloud environment, several solutions are available. The mobile users mainly encrypt data white transmitting into the cloud so that an adversary cannot understand or even be able to get the data. Secure transmission protocols such as TLS/SSL or VPNs can be used to transfer data. Socket programming is also used for secure transmission of sensitive data in a cloud environment. Also, public key encryption is used for protecting from Man-In-The-Middle attacks. Strong password and biometric authentication should be used to enhance data security during transmission. Even the rough access points in public places should be avoided for security reasons. Switching off the wireless interfaces, such as Wi-Fi and Bluetooth, after using the mobile device will also help.

3. Security of Cloud Infrastructure.

   At this level, users offload their data to the cloud and lose control over this data. Cloud computing is based on virtualization technology, and if there is some vulnerability in the virtualization software, the data of one user on the same physical server can be leaked to that of other users. There is also the necessity of proper access control and data management, according to the needs of the consumer. Different techniques and mechanisms for the protection of data in the cloud are provided by cloud services vendors. To increase the trust of customers for storing data in the cloud server, it should provide privacy, authentication, confidentiality, and availability of services. These security mechanisms must be strong enough to handle attacks by adversaries and hackers. There must be also a mechanism to recover the user's data if lost or erased by an attacker. There should also be a secure and efficient key management mechanism for the cloud environment. Cloud should use an implicit authentication technique to reduce the risk of fraud in a mobile cloud.

Offloading personal information, data, and application to the remote cloud as well as in communication channel raise various questions regarding security, privacy, and trust; they involve both the communication and cloud level described before and can be outlined as follows:

- Regulatory compliance: Cloud service providers should have external audits and security certification
- Privileged user access: When sensitive data get offloaded to the cloud, it may appear that the data are no longer under the direct physical, logical, and personal control of the user owner of data.
- Data location: Exact physical location of user's data is not transparent, and this may result in confusion in particular authorities and commitments on local privacy needed.
- Recovery: Cloud providers should provide proper recovery management schemes for data and services when a technical fault or disaster arises.
- Data segregation: As cloud data are usually stored in a shared space in a multi-tenant environment, each user's data should be separated and isolated from the others with efficient encryption methodologies.
- Long-term viability: It must be ensured that user's data would be safe and accessible even in the event the cloud company itself goes out of business.
- Investigative support: For multiple customers, logging and data may be co-located. Thus, it may be vital, but hard, to predict any inappropriate or illegal activity.

Installing and running local security software can help get rid of various kind of malicious codes such as viruses, worms, bugs and so on.

2.2 Authentication, Authorization and Accounting

Whether a security system serves the purposes of information asset protection or provides for general security outside the scope of IT, it is common to have three main security processes:

- Authentication is used to determine and validate user identity. It is often referred to as identification and authentication, the former provides user identity to the security system (typically in the form of a user ID), the latter is the process of validating user identity by verifying user-provided credential (e.g. password, PIN, certificates); in particular, the *supplicant* (authenticating user client) provides its identity and evidences for it to the authenticator (i.e. the server), that uses the Security authority/database as the storage mechanism to check user credentials. Practical authentication process can be session-based (i.e. using cookies) or HTTP-based (Basic or Digest Authentication). Moreover, the growth of applications endorse the concept of *Single Sign On* (SSO), i.e. a unified authentication for access infrastructure and service. Two SSO mechanisms, namely OpenId and OAuth, are beginning to attract users.

  OpenID(Recordon and Reed, 2006)(OpenID Foundation, 2019) is an open standard and decentralized authentication protocol. It was promoted by the non-profit OpenID Foundation, it allows users to be authenticated by co-operating sites (known as Relying Parties or RP) using a third party service, eliminating the need for webmasters to provide their own ad hoc login systems, and allowing users to log into multiple unrelated websites without having to have a separate identity and password for each.

  OAuth(Parecki, 2019) is a protocol framework that attempts to facilitate interactions between applications in a properly secure manner. In particular, while OpenId just checks the user's identity and sends back a response confirming it, OAuth sends back a token. This token can be used in subsequent requests almost as a substitute for a valid user session (avoiding password sharing), to allow access to restricted resources. The 2.0 version of OAuth extends beyond the sort of data exchange between Web sites, and can be applied to authorizing mobile and desktop clients to access restricted resources, and to authorization performed by client-side scripts without an intervening server application
- Authorization provides users the access to resources that they are allowed to have and prevents users from accessing resources that they are not allowed to access. Typical implementations are *account-based*, where rights to perform operations are associated with individual user accounts, or *role-based*, where users are linked to a specific role for which rights are granted.
- Accounting provides an audit trail of user actions. This is sometimes referred to as auditing.

3 AUTHENTICATION IN MCC

User authentication in Mobile Cloud Computing is a critical requirement in securing cloud-based computations and communications and with authorization are one of the most important security issues for MCC users. Both mobile device and cloud server should authenticate each other in order to secure the communication when the user with its device accesses the cloud from any geographical location in the world, using different networks and various mobile devices.

Adoption of MCC highly necessitates robust and effective authentication solution by which users can utilize the cloud-based services with any mobile devices (e.g. smartphones, tablets) anytime, anywhere, with low computing cost of the native resource. There is a substantial difference between MCC and typical mobile device authentication, in the first environment the mobile devices connect to the Internet to perform authentication. Moreover, the hungry-resource part of authentication mechanism can be moved and processed by the cloud servers.

User authentication in MCC is the process of identifying and validating the mobile user to ensure that it is legitimate to access mobile cloud resources. Due to limited resources of mobile devices, optimal MCC authentication mechanisms should be lightweight with the least possible computing, memory, and storage requirements.

3.1 MCC versus CC

Although the goal still remains to minimize security threats, authentication mechanisms in MCC somehow differs from those adopted in non-mobile cloud computing. Some of the most important security threats to mobile users are information leakage, denial of service, malfunction and loss of device. These threats have to be included in security threats that can manifest as attacks via the services offered through the wireless network, including network profiling, information leakages, session hijacking, and jamming.

The capabilities and limitations of mobile devices introduce some challenges for developing effective and efficient authentication system. The most relevant causes of discrepancy are linked with Network heterogeneity, Resource limitations, Mobile and device sensors, High mobility (Carchiolo et al., 2019). In table 1 the main differences between authentication requirements and principles in MCC and cloud

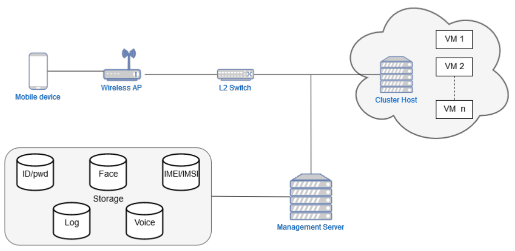

Figure 1: Multi-factor authentication architecture.

computing are shown. In the following, MCC authentication is discussed, both *cloud-side* as well as the *user-side* approach.

Table 1: MCC vs CC comparison.

Metrics	MCC	CC
Resource limitation	challenge	not effective
Mobile device feature	opportunity	not effective
High mobility	challenge	not effective
Network heterogeneity	challenge	challenge
Wired or wireless comm.	challenge	not effective

3.2 Cloud-side MCC Authentication

As intuition suggests, in this case authentication is performed on the cloud server and this is considered more flexible, efficient, and adjustable compared to other methods due to the (possibly) unlimited resources of cloud servers.

Such authentication can be either identity- or context- based; in identity-based methods, several credentials as unique ID, password, and biometrics can be used; their combination leads to two-factor authentication (easier) or even multi-factor (safer) authentication (Jeong et al., 2015)(Mohsin et al., 2017).

This architecture (Fig.1) includes four main entities, i.e. mobile device, storage, a management server, and a cluster host. Usually, Transport Layer Security (TLS) protocol is used for communications between the authentication service provider and wireless access point. Five parameters are exploited to authenticate the mobile user: ID/password, International Mobile Equipment Identity (IMEI), International Mobile Subscriber Identity (IMSI), voice and face recognition; for these last two note that privacy issues must be addressed, though biometric methods seems more appealing to users than passwords(Braz and Robert, 2006)).

The cluster host distributes these five parameters to individual virtual machines (VM) to improve performance of the authentication process. The management server handles the load balancing on the VMs in the clustered host. Users information are protected in case of loss and theft using IMEI and IMSI; these are encrypted with a hash function to avoid their clear exposure.

As opposed to identity-based, in context-based methods users are authenticated by analyzing multiple passive user information and requires minimal user interaction. However, the accuracy is lower than identity-based method because the authentication procedure depends on the accuracy of pattern analysis results.

3.3 User-side MCC Authentication

In user-side authentication method, steps are processed in mobile devices that nowadays largely support such approach thanks to their significant performance improvement. Also User-side authentication methods can be identify- or context-based; like identity-based methods in cloud-side, it uses user identity information to authenticate the user, but here is the mobile device that processes and analyzes user attributes to check user authentication instead of cloud servers. User sensitive information as biometrics are stored locally in the mobile device during authentication procedure, which increase the privacy and security issues; applications can be found in (Oh et al., 2011) and (Schwab and Yang, 2013).

Context-based user-side methods are similar to the corresponding cloud-side methods; the only difference lies in that the mobile device processes and evaluates user information instead of cloud server. Generally, a context-based authentication mechanism needs more computation resource compared to the identity-based methods, thus introducing performance issue due to resource limitations of mobile devices. Therefore, context-based user-side authentication methods are less appropriate in MCC compared to cloud-based methods. In addition, various types of user sensitive information are stored in mobile devices, increasing users privacy risk due to device loss compared to the more reliable cloud environment.

4 AUTHENTICATION IN STMicroelectronics

The case study of STMicroelectronics authentication mechanism, presented in this section, complaints with constraints and specifications of such a complex corporate context. In our proposal we use a CMS-style web application using the PHP framework Laravel(Taylor, 2017)(Bean, 2015) for what concerns the application and OAuth 2 for the authentication and authorization process. A relevant factor to be considered was that this mechanism must work with both mobile devices as well as PCs desktop, and moreover, it must also work from an Intranet network through a corporate firewall. Using a web application is therefore a suitable choice, because it can be used via mobile and non-mobile devices, and it also relies on standard HTTP/HTTPS protocols communicability through corporate intranet firewalls. Since the security of the entire application was a key aspect of the specification, a SSL/TLS public certificate was adopted to support HTTPS using *Let's Encrypt*(Internet Security Research Group, 2017) a free, automated, and open certificate authority endorsed by the non-profit Internet Security Research Group (ISRG).

As shown in Fig. 2, the client may be any device (also mobile) with browsing capabilities. Clients' requests are forwarded through a dynamic DNS service provided by *No-IP*(Vitalwerks Internet Solutions, 2017) to the Web Server, hosted by Amazon EC2 instance(Amazon Web Services, 2017). Deployment of the application on the server is realized using SFTP protocol.

As cited previously, the application development frameworks adopted is Laravel, free, opensource and PHP based framework that complies with model–view–controller (MVC) architectural pattern as frequently occurs.

In addition, Laravel also provides a simple, convenient way to authenticate with OAuth providers using Laravel Socialite(Taylor, 2019) using *Facebook*, *Twitter*, *Google*, *LinkedIn*, *GitHub* and *Bitbucket*. In particular, every request that an application sends to the Google+ API needs to identify the application to Google. Either an API key, or an OAuth 2.0 client ID can be used. A client ID should be used when calls are made on behalf of a given user. Google supports incremental authorization, which enables applications to request initial permissions at sign-in and later additional permissions can be granted, typically just before they are needed. This feature can improve the sign-in conversion rate if initial scopes are configured to be only the minimum that application requires to get the user started.

Twitter API Authentication Model instead supplies either user or app-only authentication. In the former, a user access token grants permission to app's API calls, whereas in the latter API requests are issued without user context information.

Laravel HTTP requests handling aims to provide authentication and it is carried out with a routing mechanism that operates in two phases, one to redirect the user to the OAuth provider, and another to receive the callback from the provider after authentication.

To access Socialite, the Socialite facade is used, together with Auth facade to access various Auth methods as shown in the listing 2:

```
namespace App\Http\Controllers;
use App\Role;
use App\User;
use Illuminate\Routing\Controller;
use Illuminate\Support\Facades\Auth;
use Laravel\Socialite\Facades\Socialite;

class AuthController extends Controller
{
    /**
     * @param $provider
     * @return Response
     */
    public function redirectToProvider($provider)
    {
        return Socialite::driver($provider)->redirect();
    }

    /**
     * @param $provider
     * @return Response
     */
    public function handleProviderCallback($provider)
    {
        try {
            $userContext = Socialite::driver($provider)
                ->user();
            $authUser = $this->findOrCreateUser(
                $userContext);
            Auth::login($authUser, true);
            return redirect('home');
        }
        catch (\Exception $e) {
            return back();
        }
    }

    private function findOrCreateUser($userContext)
    {
        if ($authUser = User::where('email',
            $userContext->email)->first()) {
            return $authUser;
        }
        else {
            $newUser = new User;
            $newUser->email = $userContext->email;
            $newUser->name = $userContext->name;
            $newUser->password = bcrypt(self::
                generate_password());
            $newUser->save();
            $defaultRole = Role::where('name', 'guest')
                ->first();
            $newUser->addRole($defaultRole);
            return $newUser;
        }
        throw new \Exception();
    }
}
```

Listing 1: AuthController.php.

The `redirectToProvider` sends user info to the OAuth provider, while the `handleProviderCallback` method will read the incoming request and retrieves the user's information from the provider; finally, `findOrCreateUser` method looks up the user object in the database returning it if present, or creating a new user otherwise.

OAuth2 server is a service provider allowing access to API endpoints for the authorized user or authorized applications. This is obatianed by integrating *OAuth2orize*, an authorization server toolkit for

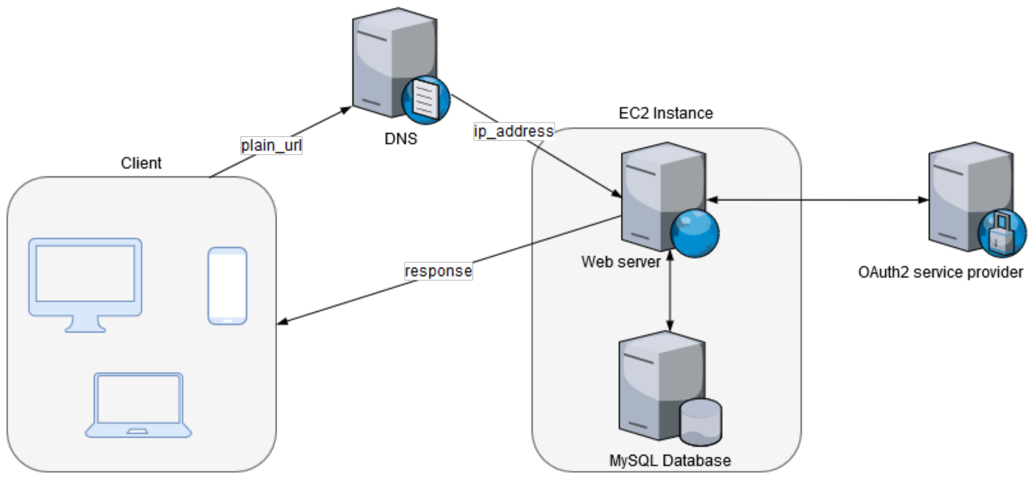

Figure 2: Application's architecture.

Node.js, into the application. It provides a suite of middleware that, combined with Passport authentication strategies and application-specific route handlers, can be used to settle up a server that implements the OAuth 2.0 protocol.

OAuth 2.0 defines an authorization framework, allowing an extensible set of authorization grants to be exchanged for access tokens. Implementations can choose what grant types to support, by using the bundled middleware (to support common types) or plugins (to support extension types). It provides a large number of modules for interoperability with various services. One of these in particular, *Mongoose*(mongoose, 2017), was used to connect with the non-relational database *MongoDB*(MongoDB, Inc, 2019). Using a non-relational database, such as MongoDB gives several benefits as high performance, low latency, simplicity, scalability and rapid object-relational mapping.

OAuth2orize requires session state for the express application in order to properly complete the authorization transaction. In order to do this, the express-session package was needed to use it in the express application.

```
// Register authorization code grant type
server.grant(oauth2orize.grant.code(function(client,
      redirectUri, user, ares, callback) {
  // Create a new authorization code
  var code = new Code({
    value: uid(16),
    clientId: client._id,
    redirectUri: redirectUri,
    userId: user._id
  });

  // Save the auth code and check for errors
  code.save(function(err) {
    if (err) { return callback(err); }

    callback(null, code.value);
  });
}));
```

Listing 2: controllers/oauth2.js.

When a client redirects a user to user authorization endpoint, an authorization transaction is initiated. To complete the transaction, the user must authenticate and approve the authorization request. Because this may involve multiple HTTP request/response exchanges, the transaction is stored in the session and a new authorization code model for the user and application client it is stored in MongoDB so it can be accessed to exchanging for an access token.

5 CONCLUSIONS

In this paper, the MCC authentication and authorization issues were discussed, from Cloud computing questions to specific MCC scenario. We then presented the solution developed and applied within the STMicroelectronics IC manufacturer plants. Further works include measurements and performance assessment of the proposed solution, and also its improvement by introducing stronger mechanisms as trustworthiness (Buzzanca et al., 2017), (Gai et al., 2016). Another interesting challenge is related to the exploitation of techniques derived from complex networks and communities (Song et al., 2018), useful for the study of reliability and robustness of the proposed MCC solution (Carchiolo et al., 2015) (Carchiolo et al., 2018).

ACKNOWLEDGEMENTS

This work has been supported by the Universitá degli Studi di Catania, ”Piano della Ricerca 2016/2018 Linea di intervento 2”.

REFERENCES

Alizadeh, M., Abolfazli, S., Zamani, M., Baaaharun, S., and Sakurai, K. (2016). Authentication in mobile cloud computing: A survey. *Journal of Network and Computer Applications*, 61:59–80.

Amazon Web Services (2017). Amazon elastic compute cloud - ec2. //https//aws.amazon.com/it/ec2/. Last acc. 14 June 2017.

Bean, M. (2015). *Laravel 5 Essentials*. Packt Publishing.

Braz, C. and Robert, J.-M. (2006). Security and usability: The case of the user authentication methods. In *Proc. of IHM 06*, pages 199–203, New York, NY, USA. ACM.

Buzzanca, M., Carchiolo, V., Longheu, A., Malgeri, M., and Mangioni, G. (2017). Direct trust assignment using social reputation and aging. *Journal of Ambient Intelligence and Humanized Computing*, 8(2):167–175.

Carchiolo, V., Grassia, M., Longheu, A., Malgeri, M., and Mangioni, G. (2018). Exploiting long distance connections to strengthen network robustness. In *IDCS*.

Carchiolo, V., Longheu, A., Malgeri, M., Iannello, S., Marroccia, M., and Randazzo, A. (2019). Cloud in mobile platforms: managing authentication/authorization. In *Advances in Intelligent Systems and Computing*. In Press.

Carchiolo, V., Longheu, A., Malgeri, M., and Mangioni, G. (2015). The cost of trust in the dynamics of best attachment. *Computing and Informatics*, 34:167–184.

Gai, K., Qiu, M., and Elnagdy, S. A. (2016). A novel secure big data cyber incident analytics framework for cloud-based cybersecurity insurance. In *2016 IEEE 2nd BigDataSecurity, HPSC, and IDS*, pages 171–176.

Internet Security Research Group (2017). Let's Encrypt. https://letsencrypt.org/. Last acc. 14 June 2017.

Jeong, Y.-S., Park, J. S., and Park, J. H. (2015). An efficient authentication system of smart device using multi factors in mobile cloud service architecture. *Intl Journal of Communication Systems*, 28(4):659–674.

Mohsin, J. K., Han, L., Hammoudeh, M., and Hegarty, R. (2017). Two factor vs multi-factor, an authentication battle in mobile cloud computing environments. In *Proceedings of ICFNDS '17*, pages 39:1–39:10, New York, NY, USA. ACM.

MongoDB, Inc (2019). MongoDB Atlas. https://www.mongodb.com. Last acc. 12 Febraury 2019.

mongoose (2017). mongoose ODM. http://mongoosejs.com/. Last acc. 15 September 2017.

Oh, D.-S., Kim, B.-H., and Lee, J.-K. (2011). *A Study on Authentication System Using QR Code for Mobile Cloud Computing Environment*, pages 500–507. Springer Berlin Heidelberg, Berlin, Heidelberg.

OpenID Foundation (2019). OpenID. http://openid.net/. Last acc. 19 Feb. 2019.

Parecki, A. (2019). OAuth. https://oauth.net. Last acc. 19 Feb. 2019.

Peter Mell, T. G. (2009). The NIST Definition of Cloud Computing. https://www.nist.gov/sites/default/files/documents/itl/cloud/cloud-def-v15.pdf. Last acc. 13 June 2017.

Recordon, D. and Reed, D. (2006). Openid 2.0: A platform for user-centric identity management. In *Proc. of DIM '06*, pages 11–16, New York, NY, USA. ACM.

salesforce (2019). SalesForce. https://www.salesforce.com. Last acc. 5 Feb. 2019.

Sanaei, Z., Abolfazli, S., Gani, A., and Buyya, R. (2014). Heterogeneity in mobile cloud computing: Taxonomy and open challenges. *IEEE Communications Surveys Tutorials*, 16(1):369–392.

Schwab, D. and Yang, L. (2013). Entity authentication in a mobile-cloud environment. In *Proc. of CSIIRW '13*, pages 42:1–42:4, New York, NY, USA. ACM.

Song, Z., Sun, Y., Yan, H., Wu, D., Niu, P., and Wu, X. (2018). Robustness of smart manufacturing information systems under conditions of resource failure: A complex network perspective. *IEEE Access*, 6:3731–3738.

Taylor, O. (2017). Laravel - The PHP Framework for Web Artisans. https://laravel.com/. Last acc. 30 Aug. 2017.

Taylor, O. (2019). Laravel Socialite. https://laravel.com/docs/5.7/socialite. Last acc. 18 Febraury 2019.

Todorov, D. (2007). *Mechanics of User Identification and Authentication: Fundamentals of Identity Management*. CRC Press.

Vitalwerks Internet Solutions, L. (2017). No-IP. https://www.noip.com/. Last acc. 15 Sept. 2017.