Hindawi
Journal of Computer Networks and Communications
Volume 2019, Article ID 8706760, 10 pages
https://doi.org/10.1155/2019/8706760

Research Article

Improving the Privacy of IoT with Decentralised Identifiers (DIDs)

Yki Kortesniemi,[1] **Dmitrij Lagutin**,[2] **Tommi Elo**,[2] **and Nikos Fotiou**[3]

[1]*Department of Computer Science, School of Science, Aalto University, P.O. Box 11000, FI-00076 Aalto, Finland*
[2]*Department of Communications and Networking, School of Electrical Engineering, Aalto University, P.O. Box 11000, FI-00076 Aalto, Finland*
[3]*Mobile Multimedia Laboratory, Department of Informatics, School of Information Sciences and Technology, Athens University of Economics and Business, 76 Patision, 104 34 Athens, Greece*

Correspondence should be addressed to Yki Kortesniemi; yki.kortesniemi@aalto.fi

Received 9 November 2018; Revised 18 January 2019; Accepted 19 February 2019; Published 20 March 2019

Academic Editor: Roberto Nardone

When IoT devices operate not only with the owner of the device but also with third parties, identifying the device using a permanent identifier, e.g., a hardware identifier, can present privacy problems due to the identifier facilitating tracking and correlation attacks. A changeable identifier can be used to reduce the risk on privacy. This paper looks at using decentralised identifiers (DIDs), an upcoming standard of self-sovereign identifiers with multiple competing implementations, with IoT devices. The paper analyses the resource requirements of running DIDs on the IoT devices and finds that even quite small devices can successfully deploy DIDs and proposes that the most constrained devices could rely on a proxy approach. Finally, the privacy benefits and limitations of using DIDs are analysed, with the conclusion that DIDs significantly improve the users' privacy when utilised properly.

1. Introduction

As Internet of Things (IoT) devices exist in multiple copies, it becomes relevant to be able to uniquely identify each device both for managing and using them. In less critical use cases, this can be based on, e.g., the device's IP address [1] or hardware identifier as in the case of RFID [2], but in cases, when it is necessary to be able to prove that it is indeed a particular device, more effective solutions such as cryptography-based identifiers can be used. If only the owner of the devices is using them, even permanent unique identifiers for the devices present no privacy problems (provided the communications with between the device and its owner cannot be monitored). However, the situation is completely different if the device has to operate with third parties—in that case, a permanent unique identifier is a privacy risk, as the device can potentially be tracked and information about the device's owner and users can be revealed. Also, if the device is at some stage sold, maintaining the same identifier would put both the old and new owners' privacy at risk.

To avoid this, technical solutions should try to keep sensitive information as private as possible. This approach is also promoted by the principles of Privacy by Design (PbD) [3] and the General Data Protection Regulation (GDPR) [4] , where both advocate minimum information collection, use of suitable technical safeguards, and so on. With identifiers of IoT devices, relevant privacy questions include the following: does the identifier persistently identify the device, does the identifier itself reveal something about the user or owner of the device, and do the identifiers used in different contexts facilitate correlating the user's data across contexts, thus affording more revealing insights to the user/owner. All these privacy problems will be maximised if each device has only a single identifier that is used for all services.

To mitigate this tracking problem, the device identifiers have to be changeable, and if the same device is used in multiple contexts simultaneously, different identifiers for each context would further improve privacy. One way of implementing these changeable identifiers is using decentralised identifiers (DIDs) [5], a new type of identifier without a central controlling party. Currently, there are multiple competing DID technologies being developed, and the standardisation activity of the Decentralised Identity Foundation [6] is promoting interoperability among many of the implementations. Currently, there are already multiple open-source initiatives for providing free DID implementations, an important consideration for, e.g., cost conscious IoT devices. Applicationwise, DIDs are being promoted as an identifier solution for individuals, organisations, and things, such as IoT devices, alike—a factor that can help simplify implementations. DIDs also support the principles of self-sovereignty [7], that is, they allow the identity owner to create, manage, and discard identifiers as they seem fitting and support the use of multiple simultaneous identifiers, thus meeting the privacy enhancing properties listed above. However, a practical limitation for utilising DIDs with IoT devices is the required resources. For more capable devices, this is no problem, but lightweight embedded devices may lack the necessary computational, energy, or even storage capability to utilise DIDs.

The contribution of this paper is to (1) analyse how DIDs can be utilised with IoT devices in a real-world use case, (2) evaluate when DIDs can be deployed on constrained IoT devices and when mitigating solutions such as proxy approach are required, and (3) analyse the privacy improvements achievable. The rest of the paper is organised as follows: Section 2 discusses identifying IoT devices and Section 3 introduces the decentralised identifiers. Section 4 then analyses the requirements for deploying DIDs on the IoT devices themselves, Section 5 proposes using a proxy approach for enabling DIDs with constrained devices, and Section 6 discusses the privacy improvements achievable with DIDs. Finally, Section 7 discusses the applicability of DIDs with IoT devices and Section 8 presents the conclusions.

2. IoT Identifiers Can Present Privacy Problems

Figure 1 illustrates a scenario, where users of electric vehicles (EVs) subscribe to a service that allows them to charge their electric vehicles in multiple locations at will. However, in order to optimise the load on the electric grid, the users are also given incentives to charge EVs only during certain periods of time or at certain location.

There are multiple actors in the scenario. Besides the user and their electric vehicle, there is a distribution system operator (DSO) managing the electrical grid in the municipality. Some of the electricity may, for instance, be produced locally in a distributed fashion, e.g., using solar panels, which means that it is available only during certain periods and it would also be optimal to consume this electricity locally in order to balance the load on the electrical grid. EVs can be charged at any of the charging stations (CSs), several of which are owned by the same charging station owner (CSO), and there can be multiple CSOs within the same municipality. Finally, there is an electricity provider (EP) running the charging service; EP buys the electricity through the DSO's grid, sells it to the consumer, and in this case also rents the charging station capacity from CSOs. It is in the provider's interest to optimise the load on the electrical grid, in order to get lower prices for the electricity transmission and offer better and cheaper service to the users (for more details of such a scenario, see e.g., the SOFIE Italian Energy Pilot description in [8]). To further facilitate offerings for energy flexibility, the electricity provider would like more information about the user, e.g., their location and vehicle driving patterns. Such an arrangement can offer financial benefits both for the vehicle owner and the electricity provider, but at the same time, it can be very privacy intrusive for the individual.

In this scenario, it is clear that a permanent identifier for the vehicle reveals the identity of the driver with a high probability (as there typically are only few individuals using any given vehicle). And, while the driver may have agreed to reveal this to the electricity provider, other parties such as the individual charging stations or their owners should not automatically have this information available. Thus, there is a need for the *car to be addressable with a changing identifier*, and these changes must be built in to the architecture. Furthermore, there is no reason for the provider to know the exact charging station used. The provider needs to identify only the charging station owner (for billing purposes) and the rough location of the station (on a district-level) in order to optimise the load on the grid. Hence, the user's privacy is further improved when also the individual *charging stations utilise changing identifiers*.

For the electric vehicle scenario, providing the IoT functionality with sufficient resources to utilise different identifier solutions is not a major issue neither technically nor financially. However, considering the overall IoT field presents a more challenging environment: there exists a myriad of IoT devices, some of which are extremely constrained in terms of computing power, storage space, available energy, or even sources of entropy [9] to generate secure cryptographic keys. In practice it means that these IoT devices may not be able to perform public-key-based cryptographic operations at all and may even lack a permanent storage space for the keys. Also, IoT devices often rely on wireless connectivity, which is inherently less secure and more vulnerable to privacy and security attacks. Furthemore, IoT devices may be disconnected from the Internet for a long period of time (or totally disconnected) and hence miss critical security updates. And finally, the lack of user interface makes it more difficult to manage (e.g., input necessary keys, certificates, and settings) and upgrade the device.

It is true that IoT devices are currently advancing at a rapid pace, and processors capable of supporting public-key cryptography are available at a very low cost [10]. However, many IoT devices are basically non-upgradeable as they are, e.g., embedded in the walls or other hard to reach places and have a lifetime of years or decades. Therefore, a significant number of deployed IoT devices will continue to remain constrained in the near future.

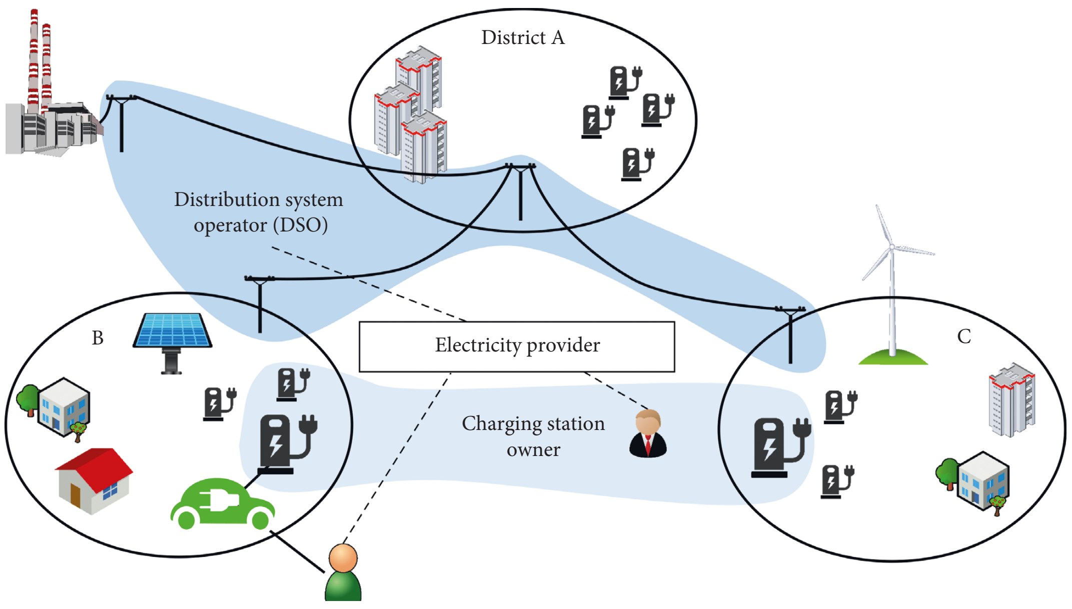


Figure 1: An overview of the electric vehicle charging scenario.

3. Decentralised Identifiers (DIDs)

For decades, individuals on the Internet have successfully carried out transactions requiring identifiers that prove the individual using the identifier is indeed the owner of the identifier. Unfortunately, as there have been no standard interoperable solutions for these identifiers, each service has been forced to create their own. More recently, some large companies, such as Google, Facebook, and Twitter, have introduced solutions known as *social logins*, where the identifiers for that company's services can also be used to login to many other services. For the individual, this reduces the number of identifiers they have to manage, but at the same time, it means that they are dependent on the service providing the social login for the identifiers, which also puts the company providing the social login in the position of monitoring the individual's use of services to the detriment of individual's privacy.

Clearly, there is a need for an identifier solution that is *controlled by the individual* and *provides sufficient privacy*. However, to provide privacy preserving pseudonymity for humans, *all addressable entities in a system* will need to support architectural level pseudonymity; otherwise, the identifier of, e.g., an IoT device may give away the identity of the individual via means of attacks, which are not directly linked to the identifiers [11].

Currently, an identifier technology receiving much attention is the decentralised identifier (DID). There are several different DID technologies in development [12], and though they started with individual goals and solutions in mind, lately many of them have adopted the approach and format of the W3C DID specification [13], being developed by the Decentralised Identity Foundation [1], thus rendering them more and more interoperable. The specification defines a DID as a random string (prefixed by "did" and a string indicating the particular DID technology), often derived from the public key used with the identifier. As an example, Figure 2 illustrates the structure of a Sovrin DID [14].

A key aspect of DIDs is that they are designed not to be dependent on a central issuing party (identity provider or IdP) that creates and controls the identifier. Instead, DIDs are created and managed by the identity owner (or a *guardian* on the owner's behalf, if the owner does not (yet) have the capacity to manage the key themselves), an approach known as *self-sovereign identity* [7].

Yet DIDs alone do not suffice, as some means of distributing the related public key, any later changes to the keys, or other identifier-related information are required. To this end, many of the DID solutions rely on a distributed ledger (DLT) or a blockchain for *public* DIDs (e.g., used by organisations that want to be known), whereas for *private* DIDs (e.g., used by individuals that want to remain private), an application-specific channel is used to distribute the information. Some DID technologies, e.g., Sovrin [14] and Veres One [15], are launching their own DLTs based on the Byzantine fault tolerance (BFT) consensus [16], while others rely on existing blockchains (e.g., uPort [17] is built on top of Ethereum [18, 19]). All three example technologies originally intended to use DLTs/blockchains for distributing information about DIDs belonging to individuals, organisations, and IoT devices, but the emergence of the General Data Protection Regulation (GDPR) in the EU and other similar changes have made storing personally identifiable information on a nonmutable platform such as a DLT/blockchain problematic. For this reason, Sovrin and Veres One have already excluded individuals' DIDs from the ledger and similar treatment may face the DIDs of IoT devices if they reveal personal information about their owner.

While the focus of this paper is on DIDs, a related technology, verifiable credentials (VCs), merits a mention in this context, as in many cases, in addition to identifiers, there did : sov : 3k9dg356wdcj5gf2k9bw8kfg7a

Figure 2: A Sovrin DID.

is also a need for a mechanism to associate machine verifiable properties to the identifier of an entity, e.g., in the EV use case the right to charge the vehicle at certain EP's network. Such an approach (which is analogous to traditional authorisation certificates) in the language of Decentralised Identity Foundation is known as a *verifiable credential* [20]. The relationship between an individual, their DIDs, and verifiable credentials is shown in Figure 3.

The benefits, requirements, and limitations of VCs are the topic for another paper, but the next section will among others briefly discuss the additional requirements they pose for on-device deployment.

4. Feasibility of Deploying DIDs on IoT Devices

This section considers the case of IoT devices that use DIDs directly. For devices that are extremely constrained or not trusted to store important secrets such as private keys, a proxy-based approach, where the proxy performs complex operations (e.g., public-key cryptography) on behalf of the device, is discussed in Section 5.

In order to utilise distributed identifiers and verifiable credentials, the IoT device should have (1) sufficient *performance* for cryptographic operations, (2) a sufficient amount of *energy* to perform the required operations, (3) nonvolatile *storage* space to store the code and cryptographic keys, and (4) sufficient *entropy* source to generate random cryptographic keys.

From the *performance* point of view, the most limiting factor is the performance of public-key cryptographic operations, namely, key generation, signature generation, and the signature verification. Presently, most DID solutions utilise elliptic curve cryptography (ECC) (as opposed to, e.g., RSA) due to its significantly smaller key size and the fact that all three operations are relatively fast and take roughly a similar amount of time (with RSA, key generation can take orders of magnitude longer than signature generation or verification operations). Lately, there has been much research about the performance of ECC on constrained devices. Older research [21] shows that operations with the common Ed25519 [22] signature scheme using a standard public domain NaCl [23] library on a popular 8-bit AVR microcontroller take about 23 million and 32 million cycles for signature generation and verification, respectively. Newer optimisations [24] reduce the cost of the elliptic curve point multiplication (in ECC, key and signature generation need one point multiplication, while the verification needs two) on the comparable Curve25519 from 23 million to 14 million cycles on a 8-bit device, while on a 32-bit low cost ARM Cortex-M0 core, the point multiplication uses only about 3.6 million cycles. Therefore, Cortex-M0 devices, which are available for less than half a dollar in large quantities [10] and run at up to 48 MHz, can perform up to 13 ECC operations per second. Since modern 8-bit microcontrollers run usually at 16–32 MHz, even such extremely constrained devices are able to perform all the necessary cryptographic operations for DID usage within a few seconds, which is acceptable performance for most IoT use cases. In a case where the device is even more constrained, a hardware accelerator for cryptographic functions could be used. Also, the number of cryptographic operations required can be managed with system design. If the IoT device only infrequently sends traffic (e.g., updates) to the network, each of these updates can be signed or encrypted with a public key. However, if an IoT device needs to send a significant amount of traffic to the network, it can rely on more lightweight symmetric encryption and hash-based message authentication code (HMAC) after the initial authentication and key exchange have been performed using DIDs. Therefore, it is usually not necessary for the IoT device to perform public-key operations at a high rate (tens or even hundreds of operations per second).

Finally, while the DID itself is just a simple string and easy to process as such, the related technology verifiable credentials (VC) are usually expressed in JSON format. There might be cases where the device includes a cryptographic accelerator, but is otherwise extremely constrained and, therefore, unable to parse JSON. In that case, VCs can be encoded in a more machine-friendly binary format such as BSON [25], as the VC specifications do not mandate usage of any specific encoding format. And since DIDs only utilise the ledgers for few operations, the network performance is normally not an issue even with constrained devices.

IoT devices often have only limited *energy* available, which has to be taken into account when designing security and privacy solutions. An optimised ECC implementation running on a Cortex-M0 using slightly weaker 233-bit sect233k1 curve uses only 20–34 μJ of energy for the elliptic curve point multiplication [26]. Such energy consumption is very low compared to the energy consumption of the wireless transmission or the overall consumption of the IoT device, which can easily be hundreds of μWs *or more* (the total average power consumption of a simple wireless sensor utilising Bluetooth low energy protocol is around 200–1000 μW [27].) Even with very simple 8-bit devices, energy consumption of ECC operations is reasonable, around 20 mJ per point multiplication, and an optimised hardware accelerator for cryptographic operations provides even lower energy consumption, in order of μJs per ECC point multiplication [28]. So, while there are some cases, where an extremely constrained, e.g., sensing device that only sends data very infrequently is not able to utilise public-key cryptography due to energy consumption concerns; in most IoT applications, energy consumption does not prevent usage of public-key cryptography and therefore DIDs, and in many IoT applications, such as vending machines and devices, may even be constrained in terms of, e.g., processing power while having plenty of energy available.

The *storage* of cryptographic keys should also not in most cases be an issue spacewise as long as a nonvolatile storage is available. ECC offers compact keys and signatures, with sizes of 256 bits and 512 bits, respectively, for the security level equivalent to 128-bit symmetric encryption. Hence, a public/private key pair would use only 64 bytes of space, and even a few kilobytes of storage space is sufficient to store multiple

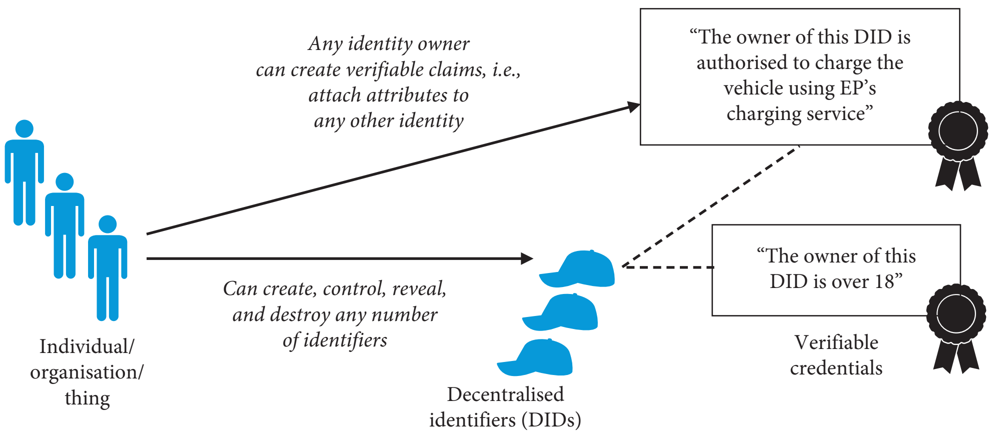

Any identity owner can create verifiable claims, i.e., attach attributes to any other identity
"The owner of this DID is authorised to charge the vehicle using EP's charging service"
"The owner of this DID is over 18"
Can create, control, reveal, and destroy any number of identifiers
Individual/ organisation/ thing
Decentralised identifiers (DIDs)
Verifiable credentials

FIGURE 3: Relationship between individuals, DIDs, and verifiable credentials.

keys or credentials. However, in some applications, storing the keys on the device can present an unacceptable security risk of key leakage unless the device utilizes, e.g., a trusted platform module (TPM). In such situation, using, e.g., a proxy solution that can act as a guardian for the keys may be a safer solution.

Finally, generating secure cryptographic keys requires a sufficient *entropy* source. This can be a challenge in the IoT environment, where the devices often have limited amount of input sources available for entropy. In that case, the entropy can be provided by a hardware-based random number generator (RNG) that is embedded in the device's processor [29]. If the hardware-based RNG is not feasible, there are several alternatives. The device's private key can be generated by another party, e.g., by the manufacturer already at the factory or by the device owner when the device is taken into use. Having the manufacturer generate the keys for all devices of that type obviously poses a security risk, thus having the owners generate the keys is already a better solution. An even better solution is if the owner (or more specifically owner's app used to initialise the device) can only act as an additional source of entropy during the initialisation process, thus letting only the device to be aware of the actual key generated.

As a proof of concept that even quite constrained devices have sufficient performance to deploy DIDs, the current uPort implementation was tested on a first-generation Raspberry Pi [30] (700 MHz BCM2835 CPU, 512 MB of RAM, released in 2012), running a Raspbian GNU/Linux 9 distribution and using the latest code of the Node.js "ethr-did" [31] package from 26th September 2018. Hundred operations were run and timed at once in each test, and the tests were repeated three times. The key pair and signature generation both took 126 ms when run separately, while generating the key pair and using it immediately to generate the signature took 230 ms overall as summarised in Table 1.

It is worth mentioning that the original Raspberry Pi is a very slow device by modern standards. It contains a single-core ARM CPU utilising the old ARMv6 instruction set, so newer ARMv7 or ARMv8 devices would offer significantly higher per-clock performance (along with more cores and higher clock speed), and devices with hardware acceleration for cryptographic operations would perform orders of magnitude faster. Also, the JavaScript and Node.js environment used by the current uPort implementation are relatively slow solutions. In comparison, a DID implementation written in C or other lower-level language would perform faster and require significantly less memory and storage space. Therefore, the setup where the tests were run can be considered a worst-case scenario. Despite all of these limitations, the current uPort Node. js implementation runs on the first-generation Raspberry Pi with acceptable performance, as the signature generation takes 126 ms. Therefore, the whole process of parsing and verifying a credential should take well below one second (in ECC, verification is 2-3 times slower than signature generation).

TABLE 1: Performance of uPort Node.js implementation on Raspberry Pi.

Operation	Time (ms)
Key pair generation	126
Signature generation	126
Key pair + signature generation	230

Overall, most devices with a 32-bit CPU can utilise DIDs with the currently available software, and more constrained devices (e.g., 8-bit microcontrollers) would be able to use DIDs with an optimised software implementation. Therefore, DIDs can be easily integrated in the electric vehicle (EV) charging service use case presented above, since IoT devices inside a vehicle would have sufficient computational resources and energy available. However, for even more constrained devices, DIDs can be used with a proxy-based approach that is discussed next.

5. Proxy-Based Approach for Constrained IoT Devices

Deploying DIDs directly on an IoT device may not always be possible nor desirable (e.g., due to limited resources or security risks), in cases where a proxy-based approach can be employed. Currently, managing an IoT device through a proxy is a typical solution (for example, Mozilla's Web of

Things gateway [32] or Amazon's Alexa smart home skill API [33]), because even though many devices have acceptable computational capabilities, they are not trusted to connect to the Internet directly. This section describes how DIDs can be introduced as a complementary function to the OAuth2-based work of the Authentication and Authorisation for Constrained Environments IETF working group [34], what new functionality they enable for IoT device/user authentication, and what security implications introducing the proxy has. The development of an actual proxy protocol and its detailed analyses are left for the future work.

The core principle of proxy-based solutions is that computationally intensive tasks and security-sensitive operations are delegated to a trusted proxy. A commonly used delegation protocol is OAuth2 [35], which enables the delegation of authentication and authorisation processes to a trusted *Authorisation Server* (AS, i.e. the proxy). The outcome of the authentication and authorisation of a user is then communicated to the *Resource Server* (RS, i.e. the IoT device) through one of the different types of *tokens* OAuth2 supports. A particular type, known as *Proof of Possession* (PoP) token, is leveraged by the Authentication and Authorisation for Constrained Environments (ACE) extension [36] in order to enable constrained devices to act as resource servers. PoP tokens include an encryption key (known as the *PoP key*) generated by the proxy and transmitted to both the user and the IoT device. This key can then be used for securing all subsequent transmissions between the user and the device. Furthermore, the ACE draft also considers IoT devices that are not connected to the Internet, in cases where the communication between the proxy and the device can be relayed through the user and it is encrypted using a secret key preshared during setup (referred to as the *psk*).

The solution proposed in ACE draft specifies the following protocols (depicted in Figure 4):

(1) *Access Token Request*. User requests an Access Token from the proxy.

(2) *Access Token Response.* If the proxy accepts the request, it returns the Access Token (which includes the PoP key) and the PoP key encrypted with the psk. The acceptance process is left application specific and it may involve, e.g., authenticating the user and evaluating an access control policy.

(3) *Resource Request*. The authorised user requests access to a protected resource on the IoT device. The user and the IoT device mutually authenticate using the PoP key.

(4) *Protected Resource*. If the request from the user is authorised, the IoT device fulfills the request and returns a response.

It should be noted that the messages of the first two protocols are exchanged over a secure communication channel (depicted with a red square in Figure 4). Furthermore, it is assumed that a setup phase has taken place during which the psk has been generated by the proxy and installed to the device (denoted with a dotted line in Figure 4). The setup phase usually takes place only once during the lifetime of the IoT device-proxy relationship (provided that the psk is not compromised).

Based on the above, OAuth2 delegation forms a practical base for a proxy-based solution that enables DIDs for constrained devices. This also introduces two interesting use cases: (i) user to device authentication using DIDs and (ii) device to user authentication using DIDs.

User to Device Authentication. With OAuth2 delegation, the user authentication can be viewed as a two-step process: (1) initially, the user is authenticated to the proxy (using the Access Token Request protocol), and if this process is successful, the user obtains a token with a PoP key; (2) the user *proves* to the IoT device that they have been authenticated (and authorised) using the obtained token and PoP key. Now, the OAuth standard does not specify how the first step is implemented, instead it is left as a design choice for the application developers. Since the proxy is a powerful and trusted node, DID-based authentication and authorisation can be used to implement this step with the Access Token Request protocol and the rest of the protocols do not have to be modified in any way.

Device to User Authentication. As already discussed, the proxy and the IoT device are configured with a preshared secret key. This key is used for encrypting keying material (the PoP key) transferred from the proxy to the IoT device via the user. The same PoP key is also provided to the user in cleartext: the fact that the IoT device and the user end up using the same key is a proof that the user communicates with a legitimate IoT device. Of course, this conclusion is based on the assumptions that (i) the user can verify the identity of the proxy and (ii) the user can verify that the IoT device has indeed delegated the authorisation and authentication processes to that proxy. Unfortunately, the implementation of those verification processes is again left to application providers. Here, DIDs can be used for verifying that both these conditions hold. In particular, an IoT device can be associated with a DIDs managed by the proxy (i.e., in DID terminology, the proxy act as a guardian) or the proxy can be authorised to speak on behalf of the device's DID (the proxy can prove this, e.g., with a verifiable credential issued by the device owner). During the Access Token Request protocol, a user can then *challenge* the proxy and request the appropriate verifiable credentials: a successful completion of this process means that the user is interacting with a proxy that acts on behalf of the desired IoT device.

6. How Far Can DIDs Improve Privacy?

The DID movement promotes the idea of using anonymous or pseudonymous identifiers and using a different identifier for each service and even switching identifiers at suitable intervals, which makes it significantly more difficult to correlate the user's activities in different services. However, this alone is not enough to protect privacy, as same care has to be taken with all related technologies; wallets, credentials, etc. should all try to minimise the information leaked through normal use and prevent unauthorised access to the information. An example of possible leak is credential

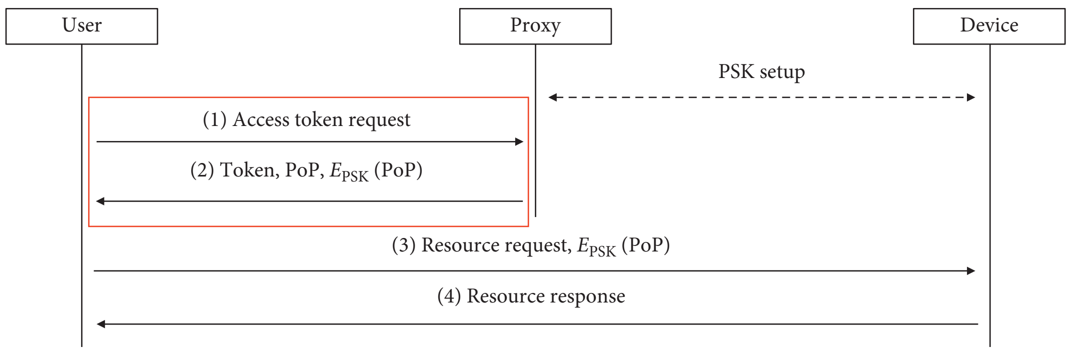


Figure 4: OAuth2 delegation for constrained devices.

validation: if it has to be done with the credential issuer, the issuer has an opportunity to learn, where and when the credentials are being used. Solutions such as short-lived credentials [37, 38] or revocation accumulators [39], which make the revocation information publicly available, avoid this problem.

A major design choice in many systems is how much privacy the different parties in the system has against each other. In the electric vehicle charging use case, the electricity provider (EP) and the charging stations (CSs) are public entities, who do not need to hide their identity, while the individuals using the service to charge their vehicle have a much stronger need for privacy. In this scenario, the EP would typically know the vehicle owner's identity as they are being billed for the charging, but the EP would also have to authorise each new vehicle DID to use the CSs in the network, so EP would automatically also know all the different DIDs used by the car. Of course, depending on the agreement with the owner of the car, the owner might have more than one vehicle they charge with the service, and therefore all the DIDs may not refer to the same vehicle.

However, the individual CSs, which could be operated by many different organisations, would not need to know more about the vehicle than that it is authorised by the EP, which will remunerate the CS owner for the energy. So, the vehicle owner should have more privacy against the charging stations. Depending on how the system is implemented, the CSs would then report back to the EP, how much each vehicle has charged and when, but potentially also at which CS. The latter, however, creates privacy issues, as over time, the EP can mine this information for the location of the vehicle and the habits of its owner. One solution is that changing DIDs are also used for CS identities as shown in Figure 5, in cases where the CS owner would only report the district where the user has charged the vehicle to the EP (the location on the district-level is necessary for optimising the load on the grid), but the exact identity (and therefore the location) of the CS used would remain hidden from the EP, which would further improve user's privacy. The DSO that operates the grid would be separated from end-users, and it would not learn vehicle DIDs or even users' real identities.

However, even without a permanent identifier to link individual transactions, correlation attacks can be mounted also using any other data relating to the use of services, e.g., information the user has decided to reveal to the services, information about the devices, e.g., network hardware or IP address, or usage patterns. In the electric vehicle charging use case, the type and model of the vehicle and the characteristics of the battery system in the vehicle are examples of information that CS could use to identify the vehicle despite changing identifiers. Indeed, it is evident that the ability of anonymisation techniques to effectively protect privacy is questioned by recent studies (e.g., [40]) and evidenced by consequences of privacy breaches such as those involving AOL [41] and Netflix [42]. These attacks should not come up as a surprise, since it is well known that an individual can be uniquely identified by the combination of generic attributes (e.g., 87% of the US population can be uniquely identified by the combination of ZIP code, gender, and the date of birth [43]).

So, in reference to the privacy questions in the introductions, using multiple and changeable DIDs, it is possible to avoid persistently identifying the device; the identifiers do not reveal anything about the user or owner of the device, and the use of a different identifier in each contexts renders it harder to correlate the user's data across contexts. Thus, the takeaway is that changeable anonymous/pseudonymous identifiers are definitely required, but not sufficient, to prevent correlation attacks from identifying individual users.

7. Discussion

Decentralised identifiers provide significant privacy benefits in cases where IoT devices communicate directly with the outside world, especially when the device operates with multiple parties in different contexts. The analysis shows that DIDs can be deployed directly even on fairly constrained IoT devices (e.g., 8-bit MCUs), so they are a feasible identifier solution for many use cases. And, when the device is more constrained, a proxy-based approach described in Section 5 can be used: with it, IoT devices only have to perform symmetric encryption operations, which are computationally much easier to implement. Furthermore, IoT devices learn nothing about a user's DID, hence even if an IoT device is compromised, users' privacy is protected. Finally, it is important to highlight that no modification to OAuth2 is required for this approach: the OAuth standard does not specify how a user and a proxy mutually authenticate;

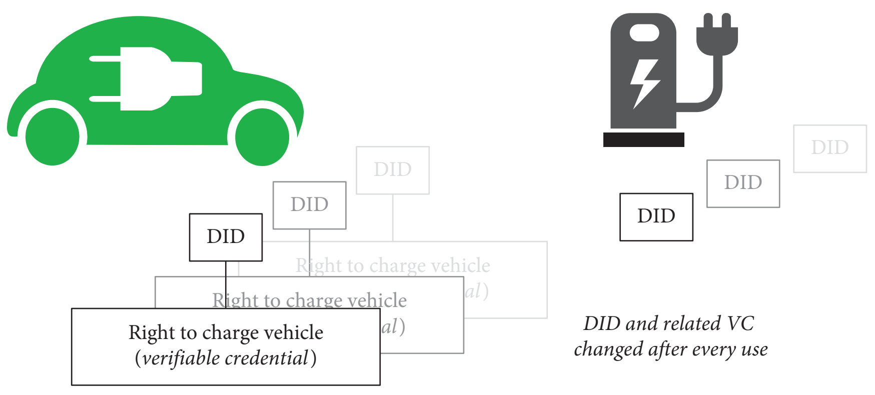

DID
DID
DID
Right to charge vehicle
Right to charge vehicle
Right to charge vehicle (verifiable credential)
DID
DID
DID
DID and related VC changed after every use

Figure 5: Regularly changing DIDs and VCs for all parties makes tracking and correlation harder.

instead, it leaves it as design choice for the application developers, so DIDs can be used for achieving this goal.

Using DIDs for devices is beneficial also because the same technology can be used for the identifiers of individuals, organisations, and devices, thus simplifying the system implementation and further hiding what type of entity is behind the identifier. Also, multiple open source implementations of the DID standards are under development, which makes it easier for manufacturers to support DIDs in their future solutions. A further benefit of widely used standard libraries is that they are less likely to have significant bugs, and any bugs are likely to be fixed faster. Using privacy enhancing solutions such as DIDs in all systems even when not strictly required helps spread the privacy enhancing solutions and avoids costly system redesigns if the privacy requirements later become stricter. So, for instance, DIDs could also be used for local device management, even if that at the moment does not necessarily require such privacy protection.

In the proxy-based approach, introducing a new component (the proxy) into the system does affect the security and privacy properties of the solution, as the new component may contain faults or be susceptible to compromising. At the same time, it may be easier to protect the proxy with firewalls and other security systems than all the IoT devices which may be left to fend for themselves. Now, for the device owner to trust the proxy, they have to be able to rely on the proxy to be well implemented and not to have malicious intentions. However, the owner is also free to choose the most suitable proxy they want to trust, so they are in the position to take the necessary steps to assure themselves of the trustworthiness of the proxy.

The risks introduced by the proxy also depend on what the system is used for in the first place and how much the proxy is trusted. For instance, is the proxy acting as the guardian for the IoT device (and does it, therefore, have access to the related key) or is the guardian some other entity, e.g, the device owner? If, for instance, the proxy is only acting on behalf of the device, e.g., issuing access tokens, it does this with its own identifier and, therefore, has no need to have access to the device's private key, thus limiting the amount of trust to the level of standard OAuth.

8. Conclusions

This paper has shown that DIDs are a suitable solution for privacy enhancing identifiers of IoT devices. With many devices, the DIDs can be implemented on the devices themselves, while a proxy approach (e.g., based on OAuth-ACE) can be used for the more constrained devices or when, e.g., the security risks of storing cryptographic keys on the device are too high. The privacy evaluation shows that, in many cases, having privacy enhancing identifiers for the IoT devices is necessary for protecting the privacy of the users and owners of the devices, but similar care has to be taken with all other elements of the system to truly protect the privacy of the individuals.

Future work in this area includes, e.g., designing and evaluating a proxy solution for DIDs, an evaluation of the benefits and drawbacks of DIDs compared to other identifier solutions, including roles/attributes-based solutions (e.g., [44]), token-based solutions (e.g., OpenID as used in [45]), capabilities-based solutions (e.g., [46]), and others. Another area for future work is studying how the use of DIDs could enhance the general IoT authorisation problem, i.e., how the user gets access to some device, which is managed by some party. Furthermore, most DIDs solutions leverage DLTs and the blockchain technology: the use of DLTs combined with DIDs in the context of the Internet of Things is a promising, yet challenging, research direction.

Data Availability

The performance data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The research reported here has been partially undertaken in the context of projects SOFIE (Secure Open Federation for Internet Everywhere), which has received funding from the European Union's Horizon 2020 Research and Innovation programme under grant agreement no. 779984, and TrustNet (Trust Network for Distributed Personal Data Management), which has received funding from Business Finland under grant no. 3387/31/2017.

References

[1] R. Want, B. N. Schilit, and S. Jenson, "Enabling the internet of things," *Computer*, vol. 48, no. 1, pp. 28–35, 2015.

[2] A. Juels, "RFID security and privacy: a research survey," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 2, pp. 381–394, 2006.

[3] A. Cavoukian, *Privacy by Design: The 7 Foundational Principles*, Information and Privacy Commissioner of Ontario, Toronto, Canada, 2009, https://www.ipc.on.ca/wp-content/uploads/Resources/7foundationalprinciples.pdf.

[4] Regulation (Eu) 2016/679 of the European Parliament and of the Council–on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation), 2018, https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679.

[5] D. Reed, "Decentralized Identifiers (DIDs) v0.11–data model and syntaxes for decentralized identifiers (DIDs)," W3C Community Group Draft Report, 2018, https://w3c-ccg.github.io/did-spec/.

[6] Decentralized identity foundation," 2018, https://identity.foundation/.

[7] C. Allen, "The path to self-sovereign identity," 2016, http://www.lifewithalacrity.com/2016/04/the-path-to-self-sovereign-identity.html.

[8] I. Oikonomidis: Baseline system and measurements, SOFIE Deliverable D5.1, 2018, http://media.voog.com/0000/0042/0957/files/SOFIE_D5.1-Baseline_System_and_Measurements.pdf.

[9] J. Wurn, K. Hoang, O. Arias, A. R. Sadeghi, and Y. Jin, "Security analysis on consumer and industrial IoT devices," in *Proceedings of the 21st Asia and South Pacific Design Automation Conference (ASP-DAC)*, IEEE, Wanchai, Hong Kong, China, January 2016.

[10] STM32F030F4, *Mainstream ARM Cortex-M0 Value Line MCU with 16 Kbytes Flash, 48 MHz CPU*, STMicroelectronics, Geneva, Switzerland, 2017, https://www.st.com/content/st_com/en/products/microcontrollers/stm32-32-bit-arm-cortex-mcus/stm32-mainstream-mcus/stm32f0-series/stm32f0x0-value-line/stm32f030f4.html.

[11] J. R. Rao and P. Rohatgi, "Can pseudonymity really guarantee privacy?," in *Proceedings of the 9th USENIX Security Symposium Denver*, Denver, CO, USA, August 2000.

[12] Blockchain and identity: projects/companies working on blockchain and identity," https://github.com/peacekeeper/blockchain-identity.

[13] D. Reed, "Decentralized identifiers (DIDs) v0.1, data model and syntaxes for decentralized identifiers (DIDs)," Draft Community Group Report, 2018, https://w3c-ccg.github.io/did-spec/.

[14] Sovrin Foundation, *Identity For All*, Sovrin Foundation, Northampton, MA, USA, 2018, https://sovrin.org/.

[15] VeresOne, "VeresOne: a globally interoperable blockchain for identity," https://veres.one/.

[16] M. Castro and B. Liskov, *Practical Byzantine Fault Tolerance*, Vol. 99, OSDI, New Orleans, LA, USA, 1999.

[17] Uport, "Open identity system for the decentralized web," https://www.uport.me.

[18] V. Buterin, "A next-generation smart contract and decentralized application platform," 2013, https://github.com/ethereum/wiki/wiki/White-Paper.

[19] G. Wood, "Ethereum: a secure decentralised generalised transaction ledger," Ethereum Yellow Paper, 2014, https://github.com/ethereum/yellowpaper.

[20] M. Sporny, D. C. Burnett, D. Longley, and G. Kellogg, "Verifiable credentials data model 1.0–expressing verifiable information on the web," W3C Editor's Draft, 2018, https://w3c.github.io/vc-data-model.

[21] M. Hutter and P. Schwabe, "NaCl on 8-bit AVR microcontrollers," in *Progress in Cryptology–AFRICACRYPT 2013, Lecture Notes in Computer Science*, A. Youssef, A. Nitaj Youssef, and A. E. Hassanien, Eds., Vol. 7918, Springer, Berlin, Heidelberg, 2013.

[22] D. J. Bernstein, N. Duif, T. Lange, P. Schwabe, and B.-Y. Yang, "High-speed high-security signatures," *Journal of Cryptographic Engineering*, vol. 2, no. 2, pp. 77–89, 2012.

[23] NaCl: Networking and Cryptography library, https://nacl.cr.yp.to/.

[24] M. Düll, B. Haase, G. Hinterwälder, M. Hutter et al., "High-speed Curve25519 on 8-bit, 16-bit, and 32-bit microcontrollers," *Designs, Codes and Cryptography*, vol. 77, no. 2-3, pp. 493–514, 2015.

[25] BSON (Binary JSON) Serialization, http://bsonspec.org/.

[26] R. de Clercq, L. Uhsadel, A. Van Herrewege, and I. Verbauwhede, "Ultra low-power implementation of ECC on the ARM Cortex-M0+," in *Proceedings of 51st ACM/EDAC/IEEE Design Automation Conference (DAC)*, San Francisco, CA, USA, June 2014.

[27] D. Oletic and V. Bilas, "System-level power consumption analysis of the wearable asthmatic wheeze quantification," *Journal of Sensors*, vol. 2018, Article ID 6564158, 18 pages, 2018.

[28] R. Azarderakhsh, K. U. Jarvinen, and M. Mozaffari-Kermani, "Efficient algorithm and architecture for elliptic curve cryptography for extremely constrained secure applications," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 61, no. 4, pp. 1144–1155, 2014.

[29] B. Sunar, W. J. Martin, and D. R. Stinson, "A provably secure true random number generator with built-in tolerance to active attacks," *IEEE Transactions on Computers*, vol. 56, no. 1, pp. 109–119, 2006.

[30] BCM2835, Raspberry Pi Documentation, Raspberry Pi Foundation, https://www.raspberrypi.org/documentation/hardware/raspberrypi/bcm2835/README.md.

[31] uPort, "Ethr-DID library," https://github.com/uport-project/ethr-did.

[32] Mozilla IoT–things gateway," https://iot.mozilla.org/gateway/.

[33] Amazon, "Understand the smart home skill API," https://developer.amazon.com/docs/smarthome/understand-the-smart-home-skill-api.html.

[34] Authentication and Authorization for Constrained Environments, IETF WG, https://datatracker.ietf.org/wg/ace.

[35] D. Hardt, "The OAuth 2.0 authorization framework," in *RFC 6749*, IETF, Fremont, CA, USA, 2012.

[36] L. Seitz, G. Selander, and E. Wahlstro, "Authentication and authorization for constrained environments (ACE) using the OAuth 2.0 framework (ACE-OAuth)," in *RFC Draft*, IETF, Fremont, CA, USA, 2018.

[37] C. Ellison, "Simple public key infrastructure certificate theory," in *RFC 2693*, IETF, Fremont, CA, USA, 1999.

[38] J. Koponen, "Internet access through WLAN with XML encoded SPKI certificates," in *Proceedings of NordSec 2000*, Reykjavik, Iceland, October 2000.

[39] J. Camenisch, M. Kohlweiss, and C. Soriente, “An accumulator based on bilinear maps and efficient revocation for anonymous credentials,” Cryptology ePrint Archive, Report 2008/539, https://eprint.iacr.org/2008/539.

[40] P. Ohm, “Broken promises of privacy: responding to the surprising failure of anonymization,” *UCLA Law Review*, vol. 57, p. 1701, 2010.

[41] M. Barbaro and T. J. Zeller, *A Face is Exposed for AOL Searcher, No. 4417749*, New York Times, New York, NY, USA, 2006.

[42] A. Narayanan and V. Shmatikov, “Robust de-anonymization of large sparse datasets,” in *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 111–125, Oakland, CA, USA, May 2008.

[43] L. Sweeney, “Simple demographics often identify people uniquely,” in *Data Privacy Working Paper 3*, Carnegie Mellon, Pittsburgh, PA, USA, 2000.

[44] C.-T. Lee, “A smart energy system with distributed access control,” in *Proceedings of IEEE International Conference on Internet of Things*, Taipei, Taiwan, September 2014.

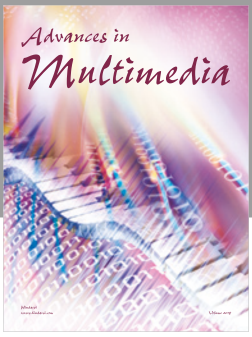
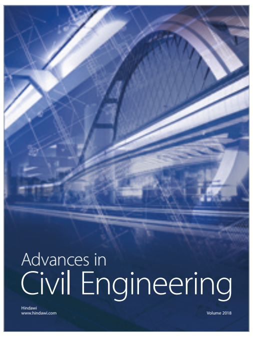
[45] A. Blazquez, V. Tsiatsis, and K. Vandikas, “Performance evaluation of OpenID connect for an IoT information marketplace,” in *Proceedings of 2015 IEEE 81st Vehicular Technology Conference (VTC Spring)*, pp. 1–6, Glasgow, Scotland, UK, May 2015.

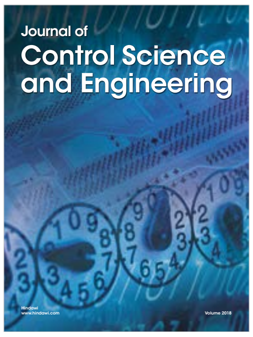
[46] B. Anggorojati, “Capability-based access control delegation model on the federated IoT network,” in *Proceedings of 15th International Symposium on Wireless Personal Multimedia Communications, WPMC 2012*, pp. 604–608, Taipei, Taiwan, September 2012.

Journal of
Engineering

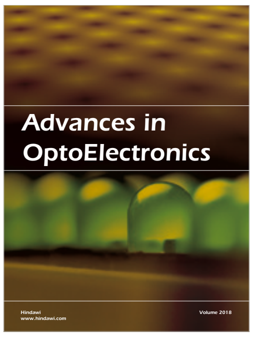
The Scientific World Journal

International Journal of
Rotating Machinery

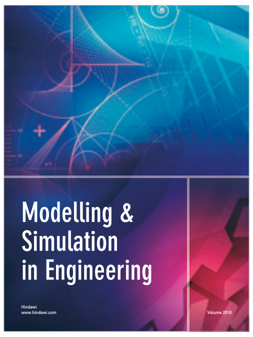
Journal of
Sensors

Advances in
Multimedia

Advances in
Civil Engineering

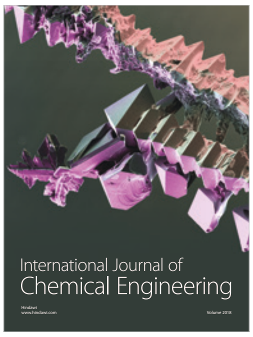
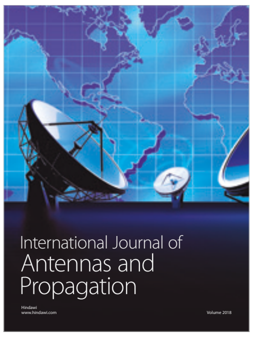

Journal of
Control Science and Engineering

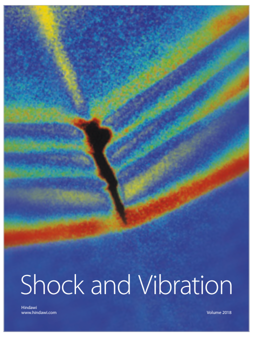
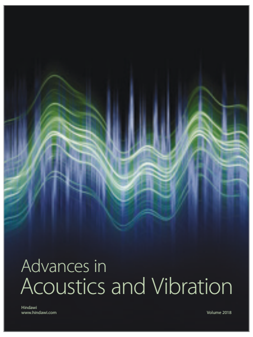
Journal of
Robotics

Journal of
Electrical and Computer Engineering

Advances in
OptoElectronics

VLSI Design

International Journal of
Navigation and Observation

Modelling & Simulation in Engineering

International Journal of
Aerospace Engineering

International Journal of
Chemical Engineering

International Journal of
Antennas and Propagation

Active and Passive
Electronic Components

Shock and Vibration

Advances in
Acoustics and Vibration