Oliveira *et al. Journal of Internet Services and Applications* (2018) 9:24
https://doi.org/10.1186/s13174-018-0095-2

Journal of Internet Services and Applications

RESEARCH Open Access

The computer for the 21st century: present security & privacy challenges

Leonardo B. Oliveira[1*], Fernando Magno Quintão Pereira[2], Rafael Misoczki[3], Diego F. Aranha[4], Fábio Borges[5], Michele Nogueira[6], Michelle Wangham[7], Min Wu[8] and Jie Liu[9]

Abstract

Decades went by since Mark Weiser published his influential work on the computer of the 21st century. Over the years, some of the UbiComp features presented in that paper have been gradually adopted by industry players in the technology market. While this technological evolution resulted in many benefits to our society, it has also posed, along the way, countless challenges that we have yet to surpass. In this paper, we address major challenges from areas that most afflict the UbiComp revolution:

1. Software Protection: weakly typed languages, polyglot software, and networked embedded systems.
2. Long-term Security: recent advances in cryptanalysis and quantum attacks.
3. Cryptography Engineering: lightweight cryptosystems and their secure implementation.
4. Resilience: issues related to service availability and the paramount role of resilience.
5. Identity Management: requirements to identity management with invisibility.
6. Privacy Implications: sensitivity data identification and regulation.
7. Forensics: trustworthy evidence from the synergy of digital and physical world.

We point out directions towards the solutions of those problems and claim that if we get all this right, we will turn the science fiction of UbiComp into science fact.

Keywords: UbiComp, Security, Privacy, Cryptography, Forensics

1 Introduction

In 1991, Mark Weiser described a vision of the *Computer for the 21st Century* [1]. Weiser, in his prophetic paper, argued the most far-reaching technologies are those that allow themselves to disappear, vanish into thin air. According to Weiser, this oblivion is a human – not a technological – phenomenon: "Whenever people learn something sufficiently well, they cease to be aware of it," he claimed. This event is called "tacit dimension" or "compiling" and can be witnessed, for instance, when drivers react to street signs without consciously having to process the letters S-T-O-P [1].

A quarter of a century later, however, Weiser's dream is far from becoming true. Over the years, many of his concepts regarding pervasive and ubiquitous computing (UbiComp) [2, 3] have been materialized into what today we call Wireless Sensor Networks [4, 5], Internet of Things [6, 7], Wearables [8, 9], and Cyber-Physical Systems [10, 11]. The applications of these systems range from traffic accident and CO_2 emission monitoring to autonomous automobile and patient in-home care. Nevertheless, besides all their benefits, the advent of those systems per se have also brought about some drawbacks. And, unless we address them appropriately, the continuity of Weiser's prophecy will be at stake.

UbiComp poses new drawbacks because, vis-à-vis traditional computing, it exhibits an entirely different outlook [12]. Computer systems in UbiComp, for instance, feature sensors, CPU, and actuators. Respectively, this means they can hear (or spy on) the user, process her/his data (and, possibly, find out something confidential about her/him), and respond to her/his actions (or, ultimately,

*Correspondence: leonardo.barbosa@dcc.ufmg.br
[1]UFMG, Av. Antônio Carlos, 6627, Prédio do ICEx, Anexo U, sala 6330 Pampulha, Belo Horizonte, MG, Brasil
Full list of author information is available at the end of the article

expose she/he by revealing some secret). Those capabilities, in turn, make proposals for conventional computers ill-suited in the UbiComp setting and present new challenges.

In the above scenarios, some of the most critical challenges lie in the areas of Security and Privacy [13]. This is so because the market and users often pursue a system full of features at the expense of proper operation and protection; although, conversely, as computing elements pervade our daily lives, the demand for stronger security schemes becomes greater than ever. Notably, there is a dire need for a secure mechanism able to encompass all aspects and manifestations of UbiComp, across time as well as space, and in a seamless and efficient manner.

In this paper, we discuss contemporary security and privacy issues in the context of UbiComp (Fig. 1). We examine multiple research problems still open and point to promising approaches towards their solutions. More precisely, we investigate the following challenges and their ramifications.

1. Software protection in Section 2: we study the impact of the adoption of weakly typed languages by resource-constrained devices and discuss mechanisms to mitigate this impact. We go over techniques to validate polyglot software (i.e., software based on multiple programming languages), and revisit promising methods to analyze networked embedded systems.
2. Long-term security in Section 3: we examine the security of today's widely used cryptosystems (e.g., RSA and ECC-based), present some of the latest threats (e.g., the advances in cryptanalysis and quantum attacks), and explore new directions and challenges to guarantee long-term security in the UbiComp setting.
3. Cryptography engineering in Section 4: we restate the essential role of cryptography in safeguarding computers, discuss the status quo of lightweight cryptosystems and their secure implementation, and highlight challenges in key management protocols.
4. Resilience in Section 5: we highlight issues related to service availability and we reinforce the importance of resilience in the context of UbiComp.
5. Identity Management in Section 6: we examine the main requirements to promote identity management (IdM) in UbiComp systems to achieve invisibility, revisit the most used federated IdM protocols, and explore open questions, research opportunities to provide a proper IdM approach for pervasive computing.
6. Privacy implications in Section 7: we explain why security is necessary but not sufficient to ensure privacy, go over important privacy-related issues (e.g., sensitivity data identification and regulation), and discuss some tools of the trade to fix those (e.g., privacy-preserving protocols based on homomorphic encryption).
7. Forensics in Section 8 we present the benefit of the synergistic use of physical and digital evidences to facilitate trustworthy operations of cyber systems.

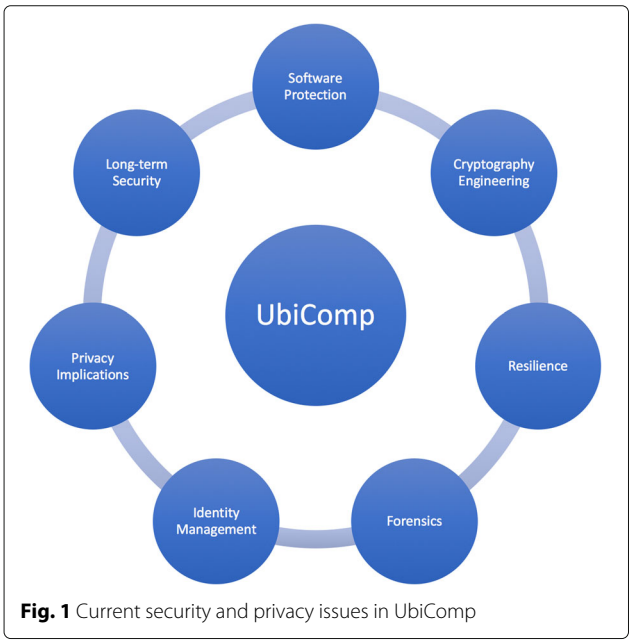

Fig. 1 Current security and privacy issues in UbiComp

We believe that only if we tackle these challenges right, we can turn the science fiction of UbiComp into science fact.

Particularly, we choose to address the areas above because they represent promising research directions e cover different aspects of UbiComp security and privacy.

2 Software protection

Modern UbiComp systems are rarely built from scratch. Components developed by different organizations, with different programming models and tools, and under different assumptions are integrated to offer complex capabilities. In this section, we analyze the software ecosystem that emerges from such a world. Figure 2 provides a high-level representation of this ecosystem. In the rest of this section, we shall focus specially on three aspects of this environment, which pose security challenges to developers: the security shortcomings of C and C++, the dominant programming languages among cyber-physical implementations; the interactions between these languages and other programming languages, and the consequences of these interactions on the distributed nature of UbiComp applications. We start by diving deeper into the idiosyncrasies of C and C++.

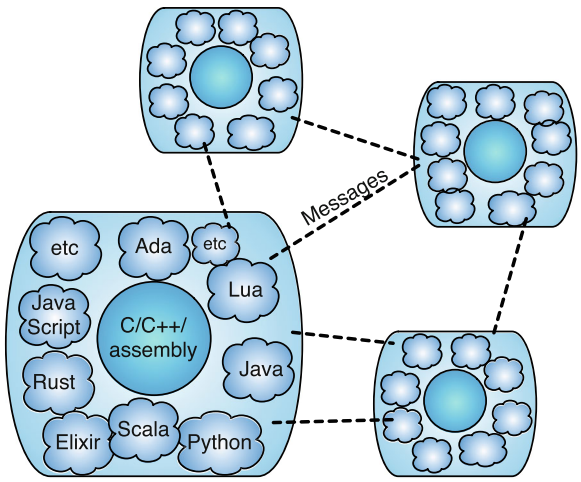

Fig. 2 A UbiComp System is formed by modules implemented as a combination of different programming languages. This diversity poses challenges to software security

2.1 Type safety

A great deal of the software used in UbiComp systems is implemented in C or in C++. This fact is natural, given the unparalleled efficiency of these two programming languages. However, if, on the one hand, C and C++ yield efficient executables, on the other hand, their weak type system gives origin to a plethora of software vulnerabilities. In programming language's argot, we say that a type system is weak when it does not support two key properties: *progress* and *preservation* [14]. The formal definitions of these properties are immaterial for the discussion that follows. It suffices to know that, as a consequence of weak typing, neither C, nor C++, ensure, for instance, bounded memory accesses. Therefore, programs written in these languages can access invalid memory positions. As an illustration of the dangers incurred by this possibility, it suffices to know that out-of-bounds access are the principle behind buffer overflow exploits.

The software security community has been developing different techniques to deal with the intrinsic vulnerabilities of C/C++/assembly software. Such techniques can be fully static, fully dynamic or a hybrid of both approaches. Static protection mechanisms are implemented at the compiler level; dynamic mechanisms are implemented at the runtime level. In the rest of this section, we list the most well-known elements in each category.

Static analyses provide a conservative estimate of the program behavior, without requiring the execution of such a program. This broad family of techniques includes, for instance, *abstract interpretation* [15], *model checking* [16] and *guided proofs* [17]. The main advantage of static analyses is the low runtime overhead, and its soundness: inferred properties are guaranteed to always hold true. However, static analyses have also disadvantages. In particular, most of the interesting properties of programs lay on undecidable land [18]. Furthermore, the verification of many formal properties, even though a decidable problem, incur a prohibitive computational cost [19].

Dynamic analyses come in several flavors: testing (KLEE [20]), profiling (Aprof [21], Gprof [22]), symbolic execution (DART [23]), emulation (Valgrind [24]), and binary instrumentation (Pin [25]). The virtues and limitations of dynamic analyses are exactly the opposite of those found in static techniques. Dynamic analyses usually do not raise false alarms: bugs are described by examples, which normally lead to consistent reproduction [26]. However, they are not required to always find security vulnerabilities in software. Furthermore, the runtime overhead of dynamic analyses still makes it prohibitive to deploy them into production software [27].

As a middle point, several research groups have proposed ways to combine static and dynamic analyses, producing different kinds of hybrid approaches to secure low-level code. This combination might yield security guarantees that are strictly more powerful than what could be obtained by either the static or the dynamic approaches, when used separately [28]. Nevertheless, negative results still hold: if an attacker can take control of the program, usually he or she can circumvent state-of-the-art hybrid protection mechanisms, such as control flow integrity [29]. This fact is, ultimately, a consequence of the weak type system adopted by languages normally seen in the implementation of UbiComp systems. Therefore, the design and deployment of techniques that can guard such programming languages, without compromising their efficiency to the point where they will no longer be adequate to UbiComp development, remains an open problem.

In spite of the difficulties of bringing formal methods to play a larger role in the design and implementation of programming languages, much has already been accomplished in this field. Testimony to this statement is the fact that today researchers are able to ensure the safety of entire operating system kernels, as demonstrated by Gerwin et al. [30], and to ensure that compilers meet the semantics of the languages that they process [31]. Nevertheless, it is reasonable to think that certain safety measures might come at the cost of performance and therefore we foresee that much of the effort of the research community in the coming years will be dedicated to making formal methods not only more powerful and expressive, but also more efficient to be used in practice.

2.2 Polyglot programming

Polyglot programming is the art and discipline of writing source code that involves two or more programming languages. It is common among implementations of cyber-physical systems. As an example, Ginga, the Brazilian protocol for digital TV, is mostly implemented in Lua and C [32]. Figure 3 shows an example of communication between a C and a Lua program. Other examples of interactions between programming languages include bindings between C and Python [33], C and Elixir [34] and the Java Native Interface [35]. Polyglot programming complicates the protection of systems. Difficulties arise due to a lack of multi-language tools and due to unchecked memory bindings between C/C++ and other languages.

An obstacle to the validation of polyglot software is the lack of tools that analyze source code written in different programming languages, under a unified framework. Returning to Fig. 3, we have a system formed by two programs, written in different programming languages. Any tool that analyzes this system as a whole must be able to parse these two distinct syntaxes and infer the connection points between them. Work has been performed towards this end, but solutions are still very preliminary. As an example, Maas et al. [33] have implemented automatic ways to check if C arrays are correctly read by Python programs. As another example, Furr and Foster [36] have described techniques to ensure type-safety of OCaml-to-C and Java-to-C bindings.

A promising direction to analyze polyglot systems is based on the idea of compilation of source code partially available. This feat consists in the reconstruction of the missing syntax and the missing declarations necessary to produce a minimal version of the original program that can be analyzed by typical tools. The analysis of code partially available makes it possible to test parts of a polyglot program in separate, in a way to produce a cohesive view of the entire system. This technique has been demonstrated to yield analyzable Java source code [37], and compilable C code [38]. Notice that this type of reconstruction is not restricted to high-level programming languages. Testimony of this fact is the notion of *micro execution*, introduced by Patrice Godefroid [39]. Godefroid's tool allows the testing of x86 binaries, even when object files are missing. Nevertheless, in spite of these developments, the reconstruction is still restricted to the static semantics of programs. The synthesis of behavior is a thriving discipline in computer science [40], but still far away from enabling the certification of polyglot systems.

C

```
#include <stdio.h>
# include <lua5.2/lua.hpp>
// Reads data from Lua, and then sends data to it.
int hello(lua_State* state) {
  int args = lua_gettop(state);
  printf("hello() was called with %d arguments:\n", args);
  for ( int n=1; n<=args; ++n) {
    printf("  arg %d: '%s'\n", n, lua_tostring(state, n));
  }
  lua_pushnumber(state, 123);
  return 1;
}
// Register a call-back function and invoke a Lua program to run it
void execute(const char* filename) {
  lua_State *state = luaL_newstate();
  luaL_openlibs(state);
  lua_register(state, "hello", hello);
  int result = luaL_loadfile(state, filename);
  result = lua_pcall(state, 0, LUA_MULTRET, 0);
}
int main(int argc, char** argv) {
  for (int n=1; n<argc; ++n) { execute(argv[n]); }
}
```

Lua

```
io.write("Calling hello() ...\n")
local value = hello("First", "Second", 112233)
io.write(string.format("hello() returned: %s\n", tostring(value)))
```

Fig. 3 Two-way communication between a C and a Lua program

2.3 Distributed programming

Ubiquitous computing systems tend to be distributed. It is even difficult to conceive any use for an application in this world that does not interact with other programs. And it is common knowledge that distributed programming opens up several doors to malicious users. Therefore, to make cyber-physical technology safer, security tools must be aware of the distributed nature of such systems. Yet, two main challenges stand in front of this requirement: the difficulty to build a holistic view of the distributed application, and the lack of semantic information bound to messages exchanged between processes that communicate through a network.

To be accurate, the analysis of a distributed system needs to account for the interactions between the several program parts that constitute this system [41]. Discovering such interactions is difficult, even if we restrict ourselves to code written in a single programming language. Difficulties stem from a lack of semantic information associated with operations that send and receive messages. In other words, such operations are defined as part of a library, not as part of the programming language itself. Notwithstanding this fact, there are several techniques that infer communication channels between different pieces of source code. As examples, we have the algorithms of Greg Bronevetsky [42], and Teixeira et al. [43], which build a distributed view of a program's control flow graph (CFG). Classic static analyses work without further modification on this *distributed* CFG. However, the distributed CFG is still a conservative approximation of the program behavior. Thus, it forces already imprecise static analyses to deal with communication channels that might never exist during the execution of the program. The rising popularization of actor-based libraries, like those available in languages such as Elixir [34] and Scala [44] is likely to mitigate

the channel-inference problem. In the actor model channels are explicit in the messages exchanged between the different processing elements that constitute a distributed system. Nevertheless, if such model will be widely adopted by the IoT community is still a fact to be seen.

Tools that perform automatic analyses in programs rely on static information to produce more precise results. In this sense, types are core for the understanding of software. For instance, in Java and other object-oriented languages, the type of objects determines how information flows along the program code. However, despite this importance, messages exchanged in the vast majority of distributed systems are not typed. Reason for this is the fact that such messages, at least in C, C++ and assembly software, are arrays of bytes. There have been two major efforts to mitigate this problem: the addition of messages as first class values to programming languages, and the implementation of points-to analyses able to deal with pointer arithmetics in languages that lack such feature. Concerning the first front, several programming languages, such as Scala, Erlang and Elixir, incorporate messages as basic constructs, providing developers with very expressive ways to implement the actor model [45] – a core foundation of distributed programming. Even though the construction of programming abstractions around the actor model is not a new idea [45], their raising popularity seems to be a phenomenon of the 2000's, boosted by increasingly more expressive abstractions [46] and increasingly more efficient implementations [47]. In the second front, researchers have devised analyses that infer the contents [48] and the size of arrays [49] in weakly-typed programming languages. More importantly, recent years have seen a new flurry of algorithms designed to analyze C/C++ style pointer arithmetics [50–53]. The wide adoption of higher-level programming languages coupled with the construction of new tools to analyze lower-level languages is exciting. This trend seems to indicate that the programming languages community is dedicating each time more attention to the task of implementing safer distributed software. Therefore, even though the design of tools able to analyze the very fabric of UbiComp still poses several challenges to researchers, we can look to the future with optimism.

3 Long-term security

Various UbiComp systems are designed to withstand a lifespan of many years, even decades [54, 55]. Systems in the context of critical infrastructure, for example, often require an enormous financial investment to be designed and deployed in the field [56], and therefore would offer a better return on investment if they remain in use for a longer period of time. The automotive area is a field of particular interest. Vehicles are expected to be reliable for decades [57], and renewing vehicle fleets or updating features (*recalls*) increase costs for their owners. Note that modern vehicles are part of the UbiComp ecosystem as they are equipped with embedded devices with Internet connectivity. In the future, it is expected that vehicles will depend even more on data collected and shared across other vehicles/infrastructure through wireless technologies [58] in order to enable enriched driving experiences such as autonomous driving [59].

It is also worth mentioning that systems designed to endure a lifespan of several years or decades might suffer from lack of future maintenance. The competition among players able to innovate is very aggressive leading to a high rate of companies going out of business within a few years [60]. A world inundate by devices without proper maintenance will offer serious future challenges [61].

From the few aforementioned examples, it is already evident that there is an increasing need for UbiComp systems to be reliable for a longer period of time and, whenever possible, requiring as few updates as possible. These requirements have a direct impact on the security features of such systems: comparatively speaking, they would offer fewer opportunities for patching eventual security breaches than conventional systems. This is a critical situation given the intense and dynamic progresses on devising and exploiting new security breaches. Therefore, it is of utmost importance to understand what the scientific challenges are to ensure long-term security from the early stage of the design of an UbiComp system, instead of resorting to palliative measures a posteriori.

3.1 Cryptography as the core component

Ensuring long-term security is a quite challenging task for any system, not only for UbiComp systems. At a minimum, it requires that every single security component is future-proof by itself and also when connected to other components. To simplify this excessively large attack surface and still be able to provide helpful recommendations, we will focus our attention on the main ingredient of most security mechanisms, as highlighted in Section 4, i.e. Cryptography.

There are numerous types of cryptographic techniques. The most traditional ones rely on the hardness of computational problems such as integer factorization [62] and discrete logarithm problems [63, 64]. These problems are believed to be intractable by current cryptanalysis techniques and the available technological resources. Because of that, cryptographers were able to build secure instantiation of cryptosystems based on such computational problems. For various reasons (to be discussed in the following sections), however, the future-proof condition of such schemes is at stake.

3.2 Advancements in classical cryptanalysis

The first threat for the future-proof condition of any cryptosystem refers to potential advancements on cryptanalysis, i.e., on techniques aiming at solving the underlying security problem in a more efficient way (with less processing time, memory, etc.) than originally predicted. Widely-deployed schemes have a long track of academic and industrial scrutiny and therefore one would expect little or no progress on the cryptanalysis techniques targeting such schemes. Yet, the literature has recently shown some interesting and unexpected results that may suggest the opposite.

In [65], for example, Barbulescu et al. introduced a new quasi-polynomial algorithm to solve the discrete logarithm problem in finite fields of small characteristics. The discrete logarithm problem is the underlying security problem of the Diffie-Hellman Key Exchange [66], the Digital Signature Algorithm [67] and their elliptic curve variants (ECDH [68] and ECDSA [67], respectively), just to mention a few widely-deployed cryptosystems. This cryptanalytic result is restricted to finite fields of small characteristics, something that represents an important limitation to attack real-world implementations of the aforementioned schemes. However, any sub-exponential algorithm that solves a longstanding problem should be seen as a relevant indication that the cryptanalysis literature might still be subject to eventual breakthroughs.

This situation should be considered by architects designing UbiComp systems that have long-term security as a requirement. Implementations that support various (i.e. higher than usual) security levels are preferred when compared to fixed, single key size support. The same approach used for keys should be used to other quantities in the scheme that somehow impact on its overall security. In this way, UbiComp systems would be able to consciously accommodate future cryptanalytic advancements or, at the very least, reduce the costs for security upgrades.

3.3 Future disruption due to quantum attacks

Quantum computers are expected to offer dramatic speedups to solve certain computational problems, as foreseen by Daniel R. Simon in his seminal paper on quantum algorithms [69]. Some of these speedups may enable significant advancements to technologies currently limited by its algorithmic inefficiency [70]. On the other hand, to our misfortune, some of the affected computational problems are the ones currently being used to secure widely-deployed cryptosystems.

As an example, Lov K. Grover introduced a quantum algorithm [71] able to find an element in the domain of a function (of size N) which leads, with high probability, to a desired output in only $O(\sqrt{N})$ steps. This algorithm can be used to speed up the cryptanalysis of symmetric cryptography. Block ciphers of n bits keys, for example, would offer only $n/2$ bits of security against a quantum adversary. Hash functions would be affected in ways that depend on the expected security property. In more details, hash functions of n bits digests would offer only $n/3$ bits of security against collision attacks and $n/2$ bits of security against pre-image attacks. Table 1 summarizes this assessment. In this context, AES-128 and SHA-256 (collision-resistance) would not meet the minimum acceptable security level of 128-bits (of quantum security). Note that both block ciphers and hash function constructions will still remain secure if longer keys and digest sizes are employed. However, this would lead to important performance challenges. AES-256, for example, is about 40% less efficient than AES-128 (due to the 14 rounds, instead of 10).

Even more critical than the scenario for symmetric cryptography, quantum computers will offer an *exponential* speedup to attack most of the widely-deployed public-key cryptosystems. This is due to Peter Shor's algorithm [72] which can efficiently factor large integers and compute the discrete logarithm of an element in large groups in polynomial time. The impact of this work will be devastating to RSA and ECC-based schemes as increasing the key sizes would not suffice: they will need to be completely replaced.

In the field of quantum resistant public-key cryptosystems, i.e. alternative public key schemes that can withstand quantum attacks, several challenges need to be addressed. The first one refers to establishing a consensus in both academia and industry on how to defeat quantum attacks. In particular, there are two main techniques considered as capable to withstand quantum attacks, namely: post-quantum cryptography (PQC) and quantum cryptography (QC). The former is based on different computational problems believed to be so hard that not even quantum computers would be able to tackle them. One important benefit of PQC schemes is that they can be implemented and deployed in the computers currently available [73–77]. The latter (QC) depends on the existence and deployment of a quantum infrastructure, and is restricted to key-exchange purposes [78]. The limited capabilities and the very high costs for deploying quantum infrastructure should eventually lead to a consensus towards the post-quantum cryptography trend.

Table 1 Symmetric cryptography security levels

Algorithm	Classical security	Quantum security
Block cipher (n bits)	n	$n/2$
Hash Pre-Image (n bits)	n	$n/2$
Hash Collision (n bits)	$n/2$	$n/3$

There are several PQC schemes available in the literature. Hash-Based Signatures (HBS), for example, are the most accredited solutions for digital signatures. The most modern constructions [76, 77] represent improvements of the Merkle signature scheme [74]. One important benefit of HBS is that their security relies solely on certain well-known properties of hash functions (thus they are secure against quantum attacks, assuming appropriate digest sizes are used). Regarding other security features, such as key exchange and asymmetric encryption, the academic and industry communities have not reached a consensus yet, although both code-based and lattice-based cryptography literatures have already presented promising schemes [79–85]. Isogeny-based cryptography [86] is a much more recent approach that enjoys certain practical benefits (such as fairly small public key sizes [87, 88]) although it has just started to benefit from a more comprehensive understanding of its cryptanalysis properties [89]. Regarding standardization efforts, NIST has recently started a Standardization Process on Post-Quantum Cryptography schemes [90] which should take at least a few more years to be concluded. The current absence of standards represents an important challenge. In particular, future interoperability problems might arise.

Finally, another challenge in the context of post-quantum public-key cryptosystems refers to potentially new implementation requirements or constraints. As mentioned before, hash-based signatures are very promising post-quantum candidates (given efficiency and security related to hash functions) but also lead to a new set of implementation challenges, such as the task of keeping the scheme state secure. In more details, most HBS schemes have private-keys (their *state*) that evolve along the time. If rigid state management policies are not in place, a signer can re-utilize the same private-key twice, something that would void the security guarantees offered by the scheme. Recently, initial works to address these new implementation challenges have appeared in the literature [91]. A recently introduced HBS construction [92] showed how to get rid of the state management issue at the price of much larger signatures. These examples indicate potentially new implementation challenges for PQC schemes that must be addressed by UbiComp systems architects.

4 Cryptographic engineering

UbiComp systems involve building blocks of very different natures: hardware components such as sensors and actuators, embedded software implementing communication protocols and interface with cloud providers, and ultimately operational procedures and other human factors. As a result, pervasive systems have a large attack surface that must be protected using a combination of techniques.

Cryptography is a fundamental part of any modern computing system, but unlikely to be the weakest component in its attack surface. Networking protocols, input parsing routines and even interface code with cryptographic mechanisms are components much more likely to be vulnerable to exploitation. However, a successful attack on cryptographic security properties is usually disastrous due to the risk concentrated in cryptographic primitives. For example, violations of confidentiality may cause massive data breaches involving sensitive information. Adversarial interference on communication integrity may allow command injection attacks that deviate from the specified behavior. Availability is crucial to keep the system accessible by legitimate users and to guarantee continuous service provisioning, thus cryptographic mechanisms must also be lightweight to minimize potential for abuse by attackers.

Physical access by adversaries to portions of the attack surface is a particularly challenging aspect of deploying cryptography in UbiComp systems. By assumption, adversaries can recover long-term secrets and credentials that provide some control over a (hopefully small) portion of the system. Below we will explore some of the main challenges in deploying cryptographic mechanisms for pervasive systems, including how to manage keys and realize efficient and secure implementation of cryptography.

4.1 Key management

UbiComp systems are by definition heterogeneous platforms, connecting devices of massively different computation and storage power. Designing a cryptographic architecture for any heterogeneous system requires assigning clearly defined roles and corresponding security properties for the tasks under responsibility of each entity in the system. Resource-constrained devices should receive less computationally intensive tasks, and their lack of tamper-resistance protections indicate that long-term secrets should not reside in these devices. More critical tasks involving expensive public-key cryptography should be delegated to more powerful nodes. A careful trade-off between security properties, functionality and cryptographic primitives must then be addressed per device or class of devices [93], following a set of guidelines for pervasive systems:

- Functionality: key management protocols must manage lifetime of cryptographic keys and ensure accessibility to the currently authorized users, but handling key management and authorization separately may increase complexity and vulnerabilities. A promising way of combining the two services into a cryptographically-enforced access control framework is attribute-based encryption

[94, 95], where keys have sets of capabilities and attributes that can be authorized and revoked on demand.

- Communication: components should minimize the amount of communication, at risk of being unable to operate if communication is disrupted. Non-interactive approaches for key distribution [96] are recommended here, but advanced protocols based on bilinear pairings should be avoided due to recent advances on solving the discrete log problem (in the so called medium prime case [97]). These advances forcedly increase the parameter sizes, reduce performance/scalability and may be improved further, favoring more traditional forms of asymmetric cryptography.
- Efficiency: protocols should be lightweight and easy to implement, mandating that traditional public key infrastructures (PKIs) and expensive certificate handling operations are restricted to the more powerful and connected nodes in the architecture. Alternative models supporting implicit certification include identity-based [98] (IBC) and certificate-less cryptography [99] (CLPKC), the former implying inherent key escrow. The difficulties with key revocation still impose obstacles for their wide adoption, despite progress [100]. A lightweight pairing and escrow-less authenticated key agreement based on an efficient key exchange protocol and implicit certificates combines the advantages of the two approaches, providing high performance while saving bandwidth [101].
- Interoperability: pervasive systems are composed of components originating from different manufacturers. Supporting a cross-domain authentication and authorization framework is crucial for interoperability [102].

Cryptographic primitives involved in joint functionality must then be compatible with all endpoints and respect the constraints of the less powerful devices.

4.2 Lightweight cryptography

The emergence of huge collections of interconnected devices in UbiComp motivate the development of novel cryptographic primitives, under the moniker *lightweight cryptography*. The term lightweight does not imply *weaker* cryptography, but application-tailored cryptography that is especially designed to be efficient in terms of resource consumption such as processor cycles, energy and memory footprint [103]. Lightweight designs aim to target common security requirements for cryptography but may adopt less conservative choices or more recent building blocks.

As a first example, many new block ciphers were proposed as lightweight alternatives to the Advanced Encryption Standard (AES) [104]. Important constructions are LS-Designs [105], modern ARX and Feistel networks [106], and substitution-permutation networks [107, 108]. A notable candidate is the PRESENT block cipher, with a 10-year maturity of resisting cryptanalytic attempts [109], and whose performance recently became competitive in software [110].

In the case of hash functions, a design may even trade-off advanced security properties (such as collision resistance) for simplicity in some scenarios. A clear case is the construction of short Message Authentication Codes (MAC) from non-collision resistant hash functions, such as in SipHash [111], or digital signatures from short-input hash functions [112]. In conventional applications, BLAKE2 [113] is a stronger drop-in replacement to recently cryptanalyzed standards [114] and faster in software than the recently published SHA-3 standard [115].

Another trend is to provide confidentiality and authentication in a single step, through Authenticated Encryption with Associated Data (AEAD). This can be implemented with a block cipher operation mode (like GCM [116]) or a dedicated design. The CAESAR competition[1] selected new AEAD algorithms for standardization across multiple use cases, such as lightweight and high-performance applications and a defense-in-depth setting. NIST has followed through and started its own standardization process for lightweight AEAD algorithms and hash functions [2].

In terms of public-key cryptography, Elliptic Curve Cryptography (ECC) [63, 117] continues to be the main contender in the space against factoring-based cryptosystems [62], due to an underlying problem conjectured to be fully exponential in classical computers. Modern instantiations of ECC enjoy high performance and implementation simplicity and are very suited for embedded systems [118–120]. The dominance of number-theoretic primitives is however threatened by quantum computers as described in Section 3.

The plethora of new primitives must be rigorously evaluated from both the security and performance point of views, involving both theoretical work and engineering aspects. Implementations are expected to consume smaller amounts of energy [121], cycles and memory [122] in ever decreasing devices and under more invasive attacks.

4.3 Side-channel resistance

If implemented without care, an otherwise secure cryptographic algorithm or protocol can leak critical information which may be useful to an attacker. Side-channel attacks [123] are a significant threat against cryptography and may use timing information, cache latency, power

and electromagnetic emanations to recover secret material. These attacks emerge from the interaction between the implementation and underlying computer architecture and represent an intrinsic security problem to pervasive computing environments, since the attacker is assumed to have physical access to at least some of the legitimate devices.

Protecting against intrusive side-channel attacks is a challenging research problem, and countermeasures typically promote some degree of *regularity* in computation. *Isochronous* or constant time implementations were among the first strategies to tackle this problem in the case of variances in execution time or latency in the memory hierarchy. The application of formal methods has enabled the first tools to verify isochronicity of implementations, such as information flow analysis [124] and program transformations [125].

While there is a recent trend towards constructing and standardizing cryptographic algorithms with some embedded resistance against the simpler timing and power analysis attacks [105], more powerful attacks such as differential power analysis [126] or fault attacks [127] are very hard to prevent or mitigate. Fault injection became a much more powerful attack methodology it was after demonstrated in software [128].

Masking techniques [129] are frequently investigated as a countermeasure to decorrelate leaked information from secret data, but frequently require robust entropy sources to achieve their goal. Randomness recycling techniques have been useful as a heuristic, but formal security analysis of such approaches is an open problem [130]. Modifications in the underlying architecture in terms of instruction set extensions, simplified execution environments and transactional mechanisms for restarting faulty computation are another promising research direction but may involve radical and possibly cost-prohibitive changes to current hardware.

5 Resilience

UbiComp relies on essential services as connectivity, routing and end-to-end communication. Advances in those essential services make possible the envisioned Weiser's pervasive applications, which can count on transparent communication while reaching the expectations and requirements of final users in their daily activities. Among user's expectations and requirements, the availability of services – not only communication services, but all services provided to users by UbiComp – is a paramount. Users more and more expect, and pay, for 24/7 available services. This is even more relevant when we think about critical UbiComp systems, such as those related to healthcare, urgency, and vehicular embedded systems.

Resilience is highlighted in this article, because it is one of the pillars of security. Resilience aims at identifying, preventing, detecting and responding to process or technological failures to recover or mitigate damages and financial losses resulted from service unavailability [131]. In general, service unavailability has been associated with non-intentional failures, however, more and more the intentional exploitation of service availability breaches is becoming disruptive and out of control, as seen in the latest Distributed Denial of Service (DDoS) attack against the company DYN, a leading DNS provider, and the DDoS attack against the company OVH, the French website hosting giant [132, 133]. The latter reached an intense volume of malicious traffic of approximately 1 TB/s, generated from a large amount of geographically distributed and infected devices, such as printers, IP cameras, residential gateways and baby monitors. Those devices are directly related to the modern concept of UbiComp systems [134] and they intend to provide ubiquitous services to users.

However, what attracts the most the attention here is the negative side effect of the ubiquity exploitation against service availability. It is fact today that the Mark Weiser's idea of *Computer for the 21st Century* has open doors to new kind of highly disruptive attacks. Those attacks are in general based on the idea of invisibility and unawareness for the devices in our homes, works, cities, and countries. But, exactly because of this, people seems to not pay enough attention to basic practices, such as change default passwords in Internet connect devices as CCTV cameras, baby monitors, smart TVs and other. This simple fact has been pointed as the main cause of the two DDoS attacks mentioned before and a report by global professional services company Deloitte suggests that Distributed Denial of Service (DDoS) attacks, that compromise exactly service availability, increased in size and scale in 2017, thanks in part to the growing multiverse of connected things[3]. They also mentioned that DDoS attacks will be more frequent, with an estimated 10 million attacks in few months.

As there is no guarantee to completely avoid these attacks, resilient solutions become a way to mitigate damages and quickly resume the availability of services. Resilience is then necessary and complementary to the other solutions we observe in the previous sections of this article. Hence, this section focuses on highlighting the importance of resilience in the context of UbiComp systems. We overview the state-of-the-art regarding to resilience in the UbiComp systems and point out future directions for research and innovation [135–138]. We also understand that resilience in these systems still requires a lot of investigations, however we believe that it was our role to raise this point to discussion through this article.

In order to contextualize resilience in the scope of UbiComp, it is important to observe that improvements on

information and communication technologies, such as wireless networking, have increased the use of distributed systems in our everyday lives. Network access is becoming ubiquitous through portable devices and wireless communications, making people more and more dependent on them. This raising dependence claims for simultaneous high level of reliability and availability. The current networks are composed of heterogeneous portable devices, communicating among themselves generally in a wireless multi-hop manner [139]. These wireless networks can autonomously adapt to changes in their environment such as device position, traffic pattern and interference. Each device can dynamically reconfigure its topology, coverage and channel allocation in accordance with changes.

UbiComp poses nontrivial challenges to resilience design due to the characteristics of the current networks, such as shared wireless medium, highly dynamic network topology, multi-hop communication and low physical protection of portable devices [140, 141]. Moreover, the absence of central entities in different scenarios increases the complexity of resilience management, particularly, when it is associated with access control, node authentication and cryptographic key distribution.

Network characteristics, as well as constraints on other kind of solutions against attacks that disrupt service availability, reinforce the fact that no network is totally immune to attacks and intrusions. Therefore, new approaches are required to promote the availability of network services. Such requirements motivate the design of resilient network services. In this work, we focus on the delivery of data from one UbiComp device to another as a fundamental network functionality and we emphasize three essential services: physical and link-layer connectivity, routing and end-to-end logical communication. However, resilience has also been observed under other perspectives. We follow the claim that resilience is achieved upon a cross-layer security solution that integrates preventive (i.e., cryptography and access control), reactive (i.e., intrusion detection systems) and tolerant (i.e., packet redundancy) defense lines in a self-adaptive and coordinated way [131, 142].

However, what are still the open challenges to achieve resilience in the UbiComp context? First of all, we emphasize the heterogeneity of devices and technologies that compose UbiComp environments. The integration from large-scale systems, such as Cloud data centers, to tiny devices, such as wearable and implantable sensors, is a huge challenge itself due to the complexity resulted from it. Then, in addition, providing integration of preventive, reactive and tolerant solutions and their adaptation is even harder in face of the different requirements of these devices, their capabilities in terms of memory and processing, and application requirements. Further, dealing with heterogeneity in terms of communication technology and protocols makes challenging the analysis of network behavior and topologies, what in conventional systems are employed to assist in the design of resilient solutions.

Another challenge is how to deal with scale. First, the UbiComp systems tend to be hyper-scale and geographically distributed. How to cope, then, with the complexity resulted from that? How to define and construct models to understand these systems and offer resilient services? Finally, we also point out as challenges the uncertainty and speed. If on the one hand, it is so hard to model, analyze and define resilient services in this complex system, on the other hand uncertainly is a norm on them, being speed and low response time a strong requirement for the applications in these systems. Hence, how to address all these elements together? How to manage them in order to offer resilient services considering diverse kind of requirements from the various applications?

All these questions lead to deep investigation and challenges. However, they also show opportunities for applied research in designing and engineering resilient systems, mainly for the UbiComp context. Particularly, if we advocate for designing resilient systems that manage the three defense lines in an adaptive way. We believe that this management can promote a great advance for applied research and for resilience.

6 Identity management

Authentication and Authorization Infrastructure (AAI) is the central element for providing security in distributed applications. AAI is a way to fulfill the security requirements in UbiComp systems. It is possible to provide identity management with this infrastructure to prevent legitimate or illegitimate users/devices to access non-authorized resources. IdM can be defined as a set of processes, technologies and policies used for assurance of identity information (e.g., identifiers, credentials, attributes), assurance of the identity of an entity (e.g., users, devices, systems), and enabling businesses and security applications [143]. Thus, IdM allows these identities to be used for authentication, authorization and auditing mechanisms [144]. A proper identity management approach is necessary for pervasive computing to be invisible to users [145]. Figure 4 provides an overview of the topics discussed in this section.

According to [143], electronic identity (eID) comprises a set of data about an entity that is sufficient to identify that entity in a particular digital context. An eID may be comprised of:

- Identifier - a series of digits, characters and symbols or any other form of data used to uniquely identify an entity (e.g., UserID, e-mail addresses, URI and IP addresses). IoT requires a global unique identifier for each entity in the network;

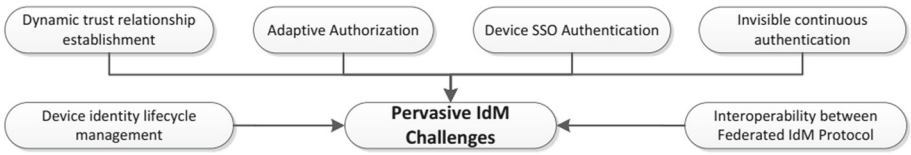

Fig. 4 Pervasive IdM Challenges

- Credentials - an identifiable object that can be used to authenticate the claimant (e.g., digital certificates, keys, tokens and biometrics);
- Attributes - descriptive information bound to an entity that specifies its characteristics.

In UbiComp systems, identity has both a digital and a physical component. Some entities might have only an online or physical representation, whereas others might have a presence in both planes. IdM requires relationships not only between entities in the same planes but also across them [145].

6.1 Identity management system

An IdM system deals with the lifecycle of an identity, which consists of registration, storage, retrieval, provisioning and revocation of identity attributes [146]. Note that the management of devices' identify lifecycle is more complicated than people's identity lifecycle due to the complexity of operational phases of a device (i.e., from the manufacturing to the removed and re-commissioned) in the context of a given application or use case [102, 147].

For example, consider a given device life-cycle. In the pre-deployment, some cryptographic material is loaded into the device during its manufacturing process. Next, the owner of the device purchases it and gets a PIN that grants the owner the initial access to the device. The device is later installed and commissioned within a network by an installer during the bootstrapping phase. The device identity and the secret keys used during normal operation are provided to the device during this phase. After being bootstrapped, the device is in operational mode. During this operational phase, the device will need to prove its identity (D2D communication) and to control the access to its resources/data. For devices with lifetimes spanning several years, maintenance cycles should be required. During each maintenance phase, the software on the device can be upgraded, or applications (running on the device) can be reconfigured. The device continues to loop through the operational phase until the device is decommissioned at the end of its lifecycle. Furthermore, the device can also be removed and re-commissioned to be used in a different system under a different owner thereby starting the lifecycle all over again. During this phase, the cryptographic material held by the device is wiped, and the owner is unbound from the device [147].

An IdM system involves two main entities: identity provider (IdP - responsible for authentication and user/device information management in a domain) and service provider (SP - also known as relying party, which provides services to user/device based on their attributes). The arrangement of these entities in an IdM system and the way in which they interact with each other characterize the IdM models, which can be traditional (isolated or silo), centralized, federated or user-centric [146].

In traditional model, IdP and SP are grouped into a single entity whose role is to authenticate and control access to their users or devices without relying on any other entity. In this model, the providers do not have any mechanisms to share this identity information with other organizations/entities. This makes the identity provisioning cumbersome for the end user or device, since the users and devices need to proliferate their sensitive data to different providers [146, 148].

The centralized model emerged as a possible solution to avoid the redundancies and inconsistencies in the traditional model and to give the user/device a seamless experience. Here, a central IdP became responsible for collecting and provisioning the user's or device's identity information in a manner that enforced the preferences of the user/device. The centralized model allows the sharing of identities among SPs and provides Single Sign-On (SSO). This model has several drawbacks as the IdP not only becomes a single point of failure but also may not be trusted by all users, devices and service providers [146]. In addition, a centralized IdP must provide different mechanisms to authenticate either users or autonomous devices to be adequate with UbiComp system requirements [149].

UbiComp systems are composed of heterogeneous devices that need to prove their authenticity to the entities they communicate with. One of the problems in this scenario is the possibility of devices being located in different security domains using distinct authentication mechanisms. An approach for providing IdM in a scenario with multiple security domains is through an AAI that uses the federated IdM model (FIM) [150, 151]. In a federation, trust relationships are

established among IdPs and SPs to enable the exchange of identity information and service sharing. Existing trust relationships guarantee that users/devices authenticated in home IdP may access protected resources provided by SPs from other federation security domains [148]. Single Sign-On (SSO) is obtained when the same authentication event can be used to access different federated services [146].

Considering the user authentication perspective, the negative points of the centralized and federated models focus primarily on the IdP, as it has full control over the user's data [148]. Besides, the user depends on an online IdP to provide the required credentials. In the federated model, users cannot guarantee that their information will not be disclosed to third parties without the users' consent [146].

The user-centric model provides the user full control over transactions involving his or her identity data [148]. In the user-centric model, the user identity can be stored on a Personal Authentication Device, such as, a smartphone or a smartcard. Users have the freedom to choose the IdPs which will be used and to control the personal information disclosed to SPs. In this model, the IdPs continue acting as a trusted third party between users and SPs. However, IdPs act according to the user's preferences [152]. The major drawback of the user-centric model is that it is not able to handle delegations. Several solutions that adopted this model combine it with FIM or centralized model, however, novel solutions prefer federated model.

6.1.1 Authentication

User and device authentication within an integrated authentication infrastructure (IdP is responsible for user and device authentication) might use a centralized IdM model [149, 153] or a traditional model [154]. Other works [155–157] proposed AAIs for IoT using the federated model, however, only for user authentication and not for device authentication. Kim et al. [158] proposes a centralized solution that enables the use of different authentication mechanisms for devices that are chosen based on device energy autonomy. However, user authentication is not provided.

Based on the traditional model, an AAI composed by a suite of protocols that incorporate authentication and access control during the entire IoT device lifecycle is proposed in [102]. Domenech et al. [151] proposes an AAI for the Web of Things, which is based on the federated IdM model (FIM) and enables SSO for users and devices. In this AAI, IdPs may be implemented as a service in a Cloud (IdPaaS - Identity Provider as a Service) or on premise. Some IoT platforms provide IdPaaS to user and device authentication such as Amazon Web Services (AWS) IoT, Microsoft Azure IoT, Google Cloud IoT platform.

Authentication mechanisms and protocols consume computational resources. Thus, to integrate an AAI into a resource constrained embedded device can be a challenge. As mentioned in Section 4.2, a set of lightweight cryptographic algorithms, which do not impose certificate-related overheads on devices, can be used to provide device authentication in UbiComp systems. There is a recent trend that investigates the benefits of using identity-based (IBC) cryptography to provide cross-domain authentication for constrained devices [102, 151, 159]. However, some IoT platforms still provide certificate-based device authentication such as Azure IoT, WSO2 or per-device public/private key authentication (RSA and Elliptic Curve algorithms) using JSON Web Tokens such as Google Cloud IoT Platform and WSO2.

Identity theft is the fastest growing crime in recent years. Currently, password-based credentials are the most used by user authentication mechanisms, despite of their weaknesses [160]. There are multiple opportunities for impersonation and other attacks that fraudulently claim another subject's identity [161]. Multi-factor authentication (MFA) is a solution created to improve the authentication process robustness and it generally combines two or more authentication factors (*something you know*, *something you have*, and *something you are*) for successful authentication [161]. In this type of authentication, an attacker needs to compromise two or more factors which makes the task more complex. Several IdPs and SPs already offer MFA to authenticate its users, however, device authentication is still an open question.

6.1.2 Authorization

In UbiComp system, a security domain can have client devices and SPs devices (SP embedded). In this context, physical devices and online providers can offer services. Devices join and leave, SPs appear and disappear, and access control must adapt itself to maintain the user perception of being continuously and automatically authenticated [145]. The data access control provided by AAI embedded in the device is also a significant requirement. Since these devices are cyber-physical systems (CPS), a security threat against these can likely impact the physical world. Thus, if a device is improperly accessed, there is a chance that this violation will affect the physical world risking people's well-being and even their lives [151].

Physical access control systems (PACS) provide access control to physical resources, such as buildings, offices or any other protected areas. Current commercial PACS are based on traditional IdM model and usually use low-cost devices such as smart cards. However, there is a trend to threat PACS as a (IT) service, i.e. unified physical and digital access [162]. Considering IoT scenarios, the translation

of SSO authentication credentials for PACS across multiple domains (in a federation), is also a challenge due to interoperability, assurance and privacy concerns.

In the context of IoT, authorization mechanisms are based on access control models used in classic Internet such as Discretionary model, for example Access Control List (ACL) [163]), Capability Based Access Control (CapBAC) [164, 165], Role Based Access Control (RBAC) [156, 166, 167] and Attribute Based Access Control (ABAC) [102, 168, 169]. ABAC and RBAC are the models better aligned to federated IdM and UbiComp systems. As proposed in [151], an IdM system that supports different access control models, such as RBAC and ABAC, can more easily adapt to the needs of the administration processes in the context of UbiComp.

Regarding policy management models to access devices, there are two approaches: provisioning [151, 170] and *outsourcing* [150, 151, 171, 172]. In provisioning, the device is responsible for the authorization decision making, which requires the policy to be in a local base. In this approach, Policy Enforcement Point (PEP), which controls the access to the device, and Policy Decision Point (PDP) are both in the same device. In outsourcing, the decision making takes place outside the device, in a centralized external service, that replies to all policy evaluation requests from all devices (PEPs) of a domain. In this case, the decision making can be offered as a service (PDPaaS) in the cloud or on premise [151].

For constrained devices, the provisioning approach is robust since it does not depend on an external service. However, in this approach, the decision making and the access policy management can be costly for the device. The outsourcing approach simplifies the policy management, but it has communication overhead and single point of failure (centralized PDP).

6.2 Federated identity management system

The IdM models guide the construction of policies and business processes for IdM systems but do not indicate which protocols or technologies should be adopted. SAML (Security Assertion Markup Language) [173], OAuth2 [174] and OpenId Connect specifications stand out in the federated IdM context [175, 176] and are adequate for UbiComp systems. SAML, developed by OASIS, is an XML-based framework for describing and exchanging security information between business partners. It defines syntax and rules for requesting, creating, communicating and using SAML Assertions, which enables SSO across domain boundaries. Besides, SAML can describe authentication events that use different authentication mechanisms [177]. These characteristics are very important for the interoperability between security technologies of different administrative domains to be accomplished. According to [151, 178, 179], the first step toward achieving interoperability is the adoption of SAML. However, XML-based SAML is not a lightweight standard and has a high computational cost for IoT resource-constrained devices [176].

Enhanced Client and Proxy (ECP), a SAML profile, defines the security information exchange that involves clients who do not use a web browser and consequently allows device SSO authentication. Nevertheless, ECP requires SOAP protocol, which is not suitable due to its high computational cost [180]. Presumably, due to its computational cost, this profile is still not widely used in IoT devices.

OpenID Connect (OIDC) is an open framework that adopts user-centric and federated IdM models. It is decentralized, which means no central authority approves or registers SPs. With OpenID, an user can choose the OpenID Provider (IdP) he or she wants to use. OpenID Connect is a simple identity layer on top of the OAuth 2.0 protocol. It allows Clients (SPs) to verify user or device identity based on the authentication performed by an Authorization Server (OpenID Provider), as well as to obtain basic profile information about the user or device in an interoperable and REST-like manner [181]. OIDC uses JSON-based security token (JWT) that enables identity and security information to be shared across security domains, consequently it is a lightweight standard and suitable for IoT. Nevertheless, it is a developing standard that requires more time and enterprise acceptance to become a established standard [176].

An IoT architecture based on OpenID, which treats authentication and access control in a federated environment was proposed in [156]. Devices and users may register at a trusted third party of the home domain, which helps the user's authentication process. In [182], OpenId connect is used for authentication and authorization of users and devices and to establish trust relationships among entities in an ambient assisted living environment (medical devices acting as a SP), in a federated approach.

SAML and OIDC are used for user authentication in Cloud platforms (Google, AWS, Azure). FIWARE platform[4] (an open source IoT platform), via Keyrock Identity Management Generic Enabler, which brings support to SAML and OAuth2-based for authentication of users. However, platforms usually use certification-based or token-based certification for device authentication using a centralized or traditional model. In future works, it may be interesting to perform practical investigations on SAML (ECP profile with different lightweight authentication mechanisms) and OIDC for various types of IoT devices and cross-domain scenarios and compare them with current authentication solutions.

OAuth protocol[5] is an open authorization framework that allows an user/ application to delegate Web

resources to a third-party without sharing its credentials. With OAuth protocol it is possible to use a Json Web Token or a SAML assertion as a means for requesting an OAuth 2.0 access token as well as for client authentication [176]. Fremantle et al. [150] discusses the use of OAuth for IoT applications that use MQTT protocol, which is a lightweight message queue protocol (publish/subscribe model) for small sensors and mobile devices.

A known standard for authorization in distributed systems is XACML (eXtensible Access Control Markup Language). XACML is a language based on XML for authorization policy description and request/response for access control decisions. Authorization decisions may be based on user/device attributes, on requested actions, and environment characteristics. Such features enable the building of flexible authorization mechanisms. Furthermore, XACML is generic, regardless of the access control model used (RBAC, ABAC) and enables the use of a local authorization decision making (provisioning model) or by an external service provider (outsourcing model). Another important aspect is that there are profiles and extensions that provide interoperability between XACML and SAML [183].

6.3 Pervasive IdM challenges

Current federation technologies rely on preconfigured static agreements, which are not well-suited for the open environments in UbiComp scenarios. These limitations negatively impact scalability and flexibility [145]. Trust establishment is the key for scalability. Although FIM protocols can cover security aspects, dynamic trust relationship establishment are open question [145]. Some requirements, such as usability, device authentication and the use of lightweight cryptography, were not properly considered in Federated IdM solutions for UbiComp systems.

Interoperability is another key requirement for successful IdM system. UbiComp systems integrates heterogeneous devices that interact with humans, systems in the Internet, and with other devices, which leads to interoperability concerns. These systems can be formed by heterogeneous domains (organizations) that go beyond the barriers of a Federation with the same AAI. The interoperability between federations that use different federated identity protocols (SAML, OpenId and OAuth) is still a problem and also a research opportunity.

Lastly, IdM systems for UbiComp systems must appropriately protect user information and adopt proper personal data protection policies. Section 7 discusses the challenges to provide privacy in UbiComp systems.

7 Privacy implications

UbiComp systems tend to collect a lot of data and generate a lot of information. Correctly used, information generates innumerable benefits to our society that has provided us with a better life over the years. However, the information can be used for illicit purposes, just as computer systems are used for attacks. Protecting private information is a great challenge that can often seem impractical, for instance, protecting customers' electrical consumption data from their electricity distribution company [184–186].

Ensuring security is a necessary condition for ensuring privacy, for instance, if the communication between clients and a service provider is not secure, then privacy is not guaranteed. However, it is not a sufficient condition, for instance, the communication is secure, but a service provider uses the data in a not allowed way. We can use cryptography to ensure secure as well as privacy. Nevertheless, even though one uses encrypted communication, the metadata from the network traffic might reveal private information. The first challenge is to find the extend of the data relevance and the impact of data leakage.

7.1 Application scenario challenges

Finding which data might be sensitive is a challenging task. Some cultures classify some data as sensitive when others classify the same data as public. Another challenge is to handle regulations from different countries.

7.1.1 Identifying sensitive data

Classifying what may be sensitive data might be a challenging task. The article 12 of the Universal Declaration of Human Rights proclaimed by the United Nations General Assembly in Paris on 10 December 1948 states: *No one shall be subjected to arbitrary interference with his privacy, family, home, or correspondence, nor to attacks upon his honor and reputation. Everyone has the right to the protection of the law against such interference or attacks.* Lawmakers have improved privacy laws around the world. However, there is still plenty of room for improvements, specially, when we consider data from people, animals, and products. Providers can use such data to profile and manipulate people and market. Unfair competitors might use private industrial data to get advantages over other industries.

7.1.2 Regulation

UbiComp systems tend to run worldwide. Thus, their developers need to deal with several laws from distinct cultures. The abundance of laws is a challenge for international institutions. The absence of laws too. On the one hand, the excess of laws compels institutions to handle a huge bureaucracy to follow several laws. On the other hand, the absence of laws causes unfair competition because unethical companies can use private data to get

advantages over ethical companies. Business models must use privacy-preserving protocols to ensure democracy and avoid a surveillance society (see [187]). Such protocols are the solution for the dilemma between privacy and information. However, they have their own technological challenges.

7.2 Technological challenges

We can deal with already collected data from legacy systems or private-by-design data that are collected by privacy-preserving protocols, for instance, databases used in old systems and messages from privacy-preserving protocols, respectively. If a scenario can be classified as both, we can just tackle it as an already collected data in the short term.

7.3 Already collected data

One may use a dataset for information retrieval while keeping the anonymity of the true owners' data. One may use data mining techniques over a private dataset. Several techniques are used in privacy preserving data mining [188]. ARX Data Anonymization Tool[6] is a very interesting tool for anonymization of already collected data. In the following, we present several techniques used to provide privacy in already collected data.

7.3.1 Anonymization

Currently, we have several techniques for anonymization and to evaluate the level of anonymization, for instance, k-anonymity, l-diversity, and t-closeness [189]. They use a set E from data indistinguishable for an identifier in a table.

The method k-anonymity suppresses table columns or replace them for keeping each E with at least k registers. It seems safe, but only 4 points marking the position on the time are enough to identify uniquely 95% of the cellphone users [190].

The method l-diversity requires that each E have at least l values "well-represented" for each sensitive column. Well-represented can be defined in three ways:

1. at least l distinct values for each sensitive column;
2. for each E, the Shannon entropy is limited, such that $H(E) \geqslant \log_2 l$, where $H(E) = -\sum_{s\in S} \Pr(E,s)\log_2(\Pr(E,s))$, S is the domain of the sensitive column, and $\Pr(E,s)$ is the probability of the lines in E that have sensitive values s;
3. the most common values cannot appear frequently, and the most uncommon values cannot appear infrequently.

Note that some tables do not have l distinct sensitive values. Furthermore, the table entropy should be at least $\log_2 l$. Moreover, the frequency of common and uncommand values usually are not close to each other.

We say that E is t-closeness if the distance between the distribution of a sensitive column E end the distribution of column in all the table is not more than a threshold t. Thus, we say that a table has t-closeness if every E in a table have t-closeness. In this case, the method generates a trade-off between data usefulness and privacy.

7.3.2 Differential privacy

The idea of differential privacy is similar to the idea of indistinguishability in cryptography. For defining it, let ϵ be a positive real number and $\mathcal{A}$ be a probabilistic algorithm with a dataset as input. We say that $\mathcal{A}$ is ϵ-differentially private if for every dataset D_1 and D_2 that differ in one element, and for every subset S of the image of $\mathcal{A}$, we have $\Pr[\mathcal{A}(D_1) \in S] \leq e^{\epsilon} \times \Pr[\mathcal{A}(D_2) \in S]$, where the probability is controlled for the algorithm randomness.

Differential privacy is not a metric in the mathematical sense. However, if the algorithms keep the probabilities based on the input, we can construct a metric d to compare the distance between two algorithms with $d(\mathcal{A}_1, \mathcal{A}_2) = |\epsilon_1 - \epsilon_2|$. In this way, we can determine if two algorithms as equivalents $\epsilon_1 = \epsilon_2$, and we can determine the distance from an ideal algorithm computing

$$d(\mathcal{A}_1, \mathcal{A}_{\text{ideal}}) = |\epsilon_1 - 0|.$$

7.3.3 Entropy and the degree of anonymity

The degree of anonymity g can be measured with the Shannon entropy $H(X) = \sum_{i=1}^{N}\left[p_i \cdot \log_2\left(\frac{1}{p_i}\right)\right]$, where $H(X)$ is the network entropy, N is the number of nodes, and p_i is the probability for each node i. The maximal entropy happens when the probability is uniform, i.e., all nodes are equiprobably $1/N$, hence $H_M = \log_2(N)$. Therefore, the anonymity degree g is defined by $g = 1 - \frac{H_M - H(X)}{H_M} = \frac{H(X)}{H_M}$.

Similar to differential privacy, we can construct a metric to compare the distance between two networks computing $d(g_1, g_2) = |g_1 - g_2|$. Similarly, we can compare if they are equivalent $g_1 = g_2$. Thus, we can determine the distance from an ideal anonymity network computing $d(g_1, g_{\text{ideal}}) = |g_1 - 1|$.

The network can be replaced by a dataset, but in this model, each register should have a probability.

7.3.4 Complexity

Complexity analysis also can be used as a metric to measure the time required in the best case for retrieving information from an anonymized dataset. It can also be used in private-by-design data as the time required to break a privacy-preserving protocol. The time measure

can be done with asymptotical analysis or counting the number of steps to break the method.

All techniques have their advantages and disadvantages. However, even though the complexity prevents the leakage, even though the algorithm has differential privacy, even though the degree of anonymity is the maximum, privacy might be violated. For example, in an election with 3 voters, if 2 collude, then the third voters will have the privacy violated independent of the algorithm used. In [191], we find how to break protocols based on noise for smart grids, even when they are provided with the property of differential privacy.

Cryptography should ensure privacy in the same way that ensures security. An encrypted message should have maximum privacy metrics as well as cryptography ensures for security. We should use the best algorithm that leaks privacy and compute its worst-case complexity.

7.3.5 Probability

We can use probabilities to measure the chances of leakage. This approach is independent of algorithm used to protect privacy.

For example, consider an election with 3 voters. If 2 voters cast yes and 1 voter cast no, an attacker knows that the probability of a voter cast yes is 2/3 and for no is 1/3. The same logics applies if the number of voters and candidates grow.

Different from the case of yes and no, we may keep the privacy from valued measured. For attackers to discover the time series of three points, they represent each point for a number of stars, i.e., symbols $\star$. Thus, attackers can split the total number of stars in three boxes. Let the sum of the series be 7, a probability would be $\boxed{\star\star\star\star}\boxed{\star}\boxed{\star\star}$. For simplicity, attackers can split the stars by bars instead of boxes. Hence, $\star\star\star\star \mid \star \mid \star\star$ is the same solution. With such notation, the binomial of 7 stars plus 2 bars chosen 7 stars determines the possible number of solutions, i.e., $\binom{7+2}{7} = \frac{9!}{7!(9-7)!} = 36$.

Generalizing, if t is the number of points in a time series and s its sum, then the number of possible time series for the attackers to decide the correct is determined by s plus $t-1$ chosen s, i.e.,

$$\binom{s+t-1}{s} = \frac{(s+t-1)!}{(t-1)!\,s!} = \binom{s+t-1}{i-1}. \quad (1)$$

If we collect multiple time series, we can form a table, e.g., a list of candidates with the number of votes by states. The tallyman cold reveal only the total number of voter by state and the total number of votes by candidate, who could infer the possible number of votes by state [191]. Data from previous elections may help the estimation. The result of the election could be computed over encrypted data in a much more secure way than anonymization by k-anonymity, l-diversity, and t-closeness. Still, depending on the size of the table and its values, the time series can be found.

In general, we can consider measurements instead of values. Anonymity techniques try to reduce the number of measurements in the table. Counterintuitively, smaller the number of measurements, bigger the chances of discover them [191].

If we consider privacy by design, we do not have already collected data.

7.4 Private-by-design data

Messages is the common word for private-by-design data. Messages are transmitted data, processed, and stored. For privacy-preserving protocols, individual messages should not be leaked. CryptDB[7] is an interesting tool, which allows us to make queries over encrypted datasets. Although messages are stored in a dataset, they are encrypted messages with the users' keys. To keep performance reasonable, privacy-preserving protocols aggregate or consolidate messages and solve a specific problem.

7.4.1 Computing all operators

In theory, we can compute a Turin machine over encrypted data, i.e., we can use a technique called fully homomorphic encryption [192] to compute any operator over encrypted data. The big challenge of fully homomorphic encryption is performance. Hence, constructing a fully homomorphic encryption for many application scenarios is a herculean task. The most usual operation is addition. Thus, most privacy-preserving protocols use additive homomorphic encryption [193] and DC-Nets (from "Dining Cryptographers") [194]. Independent of the operation, the former generates functions, and the latter generates families of functions. We can construct an asymmetric DC-Net based on an additive homomorphic encryption [194].

7.4.2 Trade-off between enforcement and malleability

The privacy enforcement has a high cost. With DC-Nets, we can enforce privacy. However, every encrypted message need to be considered in the computation for users to decrypt and to access the protocol output. It is good for privacy but bad for fault tolerance. For illustration, consider an election where all voters need to vote. Homomorphic encryption enables protocols to decrypt and output even missing an encrypted message. Indeed, it enables the decryption of a single encrypted message. Therefore, homomorphic encryption cannot ensure privacy. For illustration, consider an election where one can read and change all votes. Homomorphic encryption techniques are malleable, and DC-Nets are non-malleable. On the one hand, mailability simplifies the process and improve fault tolerance but disables privacy enforcement. On the other hand, non-mailability enforces privacy but complicates

the process and diminishes fault tolerance. In addition, the key distribution with homomorphic encryption is easier than with DC-Net schemes.

7.4.3 Key distribution

Homomorphic encryption needs a public-private key pair. Who owns the private key controls all the information. Assume that a receiver generates the key pair and send the public key to the senders in a secure communication channel. Thus, senders will use the same key to encrypt their messages. Since homomorphic encryption schemes are probabilistic, sender can use the same key to encrypt the same message that their encrypted messages will be different from each other. However, the receiver does not know who sent the encrypted messages.

DC-Net needs a private key for each user and a public key for the protocol. Since DC-Nets do not require senders and receiver, the users are usually named participants. They generate their own private key. Practical symmetric DC-Nets need that participants send a key to each other in a secure communication channel. Afterward, each participant has a private key given by the list of shared keys. Hence, each participant encrypts computing $\mathfrak{M}_{i,j} \leftarrow \text{Enc}\left(m_{i,j}\right) = m_{i,j} + \sum_{o \in \mathcal{M}-\{i\}} \text{Hash}\left(s_{i,o} \,||\, j\right) - \text{Hash}\left(s_{o,i} \,||\, j\right)$, where $m_{i,j}$ is the message sent by the participant i in the time j, Hash is a secure hash function predefined by the participants, $s_{i,o}$ is the secret key sent from participant i to participant o, similarly, $s_{o,i}$ is the secret key received by i from o, and $||$ is the concatenation operator. Each participant i can send the encrypted message $\mathfrak{M}_{i,j}$ to each other. Thus, participants can decrypt the aggregated encrypted messages computing $\text{Dec} = \sum_{i \in \mathcal{M}} \mathfrak{M}_{i,j} = \sum_{i \in \mathcal{M}} m_{i,j}$. Note that if one or more messages are missing, the decryption is infeasible. Asymmetric DC-Nets do not require a private key based on shared keys. Each participant simply generates a private key. Subsequently, they use a homomorphic encryption or a symmetric DC-Net to add their private keys generating the decryption key.

Homomorphic encryption schemes have low overhead than DC-Nets for setting up keys and for distributing them. Symmetric DC-Nets need $O\left(I^2\right)$ messages to set up the keys, where I is the number of participants. Figure 5 depicts the messages to set up keys using (a) symmetric DC-Nets and (b) homomorphic encryption. Asymmetric DC-Nets can be settled easier than symmetric DC-Nets with the price of trusting the homomorphic encryption scheme.

7.4.4 Aggregation and consolidation

The aggregation and consolidation with DC-Nets are easier than with homomorphic encryption. Using DC-Nets, participants can just broadcast their encrypted messages or just send directly to an aggregator. Using homomorphic encryption, senders cannot send encrypted messages directly to the receiver, who can decrypt individual messages. Somehow, senders should aggregate the encrypted messages, and the receiver should receive only the encrypted aggregation, which is a challenge in homomorphic encryption and trivial in DC-Nets due to the trade-off described in Section 7.4.2. In this work, we are referencing DC-Nets as fully connected DC-Nets. For non-fully connected DC-Nets, aggregation is based on trust and generates new challenges. Sometimes, aggregation and consolidation are used as synonym. However, consolidation is more complicated and generates more elaborate information than the aggregation. For example, the aggregation of the encrypted textual messages is just to join them, while the consolidation of encrypted textual messages generates a speech synthesis.

7.4.5 Performance

Fully homomorphic encryption tends to have big keys and requires a prohibitive processing time. On the contrary, asymmetric DC-Nets and partially homomorphic encryption normally use modular multi-exponentiations, which can be computed in logarithmic time [195]. Symmetric DC-Nets are efficient only for a small number of participants, because each participant need an iteration over the number of participants to encrypt a message. The number of participants is not relevant for asymmetric DC-Nets and for homomorphic encryption.

8 Forensics

Digital forensics is a branch of forensic science addressing the recovery and investigation of material found in digital devices. Evidence collection and interpretation play a key role in forensics. Conventional forensic approaches separately address issues related to computer forensics and information forensics. There is, however, a growing trend in security and forensics research that utilizes interdisciplinary approaches to provide a rich set of forensic capabilities to facilitate the authentication of data as well as the access conditions including who, when, where, and how.

In this trend, there are two major types of forensic evidences [196]. One type is intrinsic to the device, the information processing chain, or the physical environment, in such forms as the special characteristics associated with specific types of hardware or software processing or environment, the unique noise patterns as a signature of a specific device unit, certain regularities or correlations related to certain device, processing or their combinations, and more. Another type is extrinsic approaches, whereby specially designed data are proactively injected into the signals/data or into the physical world and later

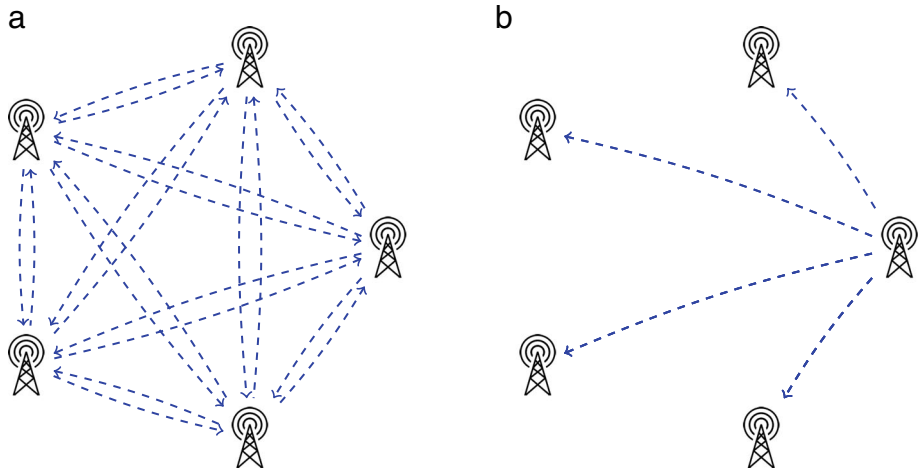

Fig. 5 Setting up the keys. **a** Symmetric DC-Nets **b** Homomorphic encryption

extracted and examined to infer or verify the hosting data's origin, integrity, processing history, or capturing environment.

In mid of the convergence between digital and physical systems with sensors, actuators and computing devices becoming closely tied together, an emerging framework has been proposed as Proof-Carrying Sensing (PCS) [197]. This was inspired by Proof-Carrying Code, a trusted computing framework that associates foreign executables with a model to prove that they have not been tampered with and they function as expected. In the new UbiComp context involving cyber physical systems where mobility and resource constraints are common, the physical world can be leveraged as a channel that encapsulates properties difficult to be tampered with remotely, such as proximity and causality, in order to create a challenge-response function. Such a Proof-Carrying Sensing framework can help authenticate devices, collected data, and locations, and compared to traditional multifactor or out-of-band authentication mechanisms, it has a unique advantage that authentication proofs are embedded in sensor data and can be continuously validated over time and space at without running complicated cryptographic algorithms.

In terms of the above-mentioned intrinsic and extrinsic view point, the physical data available to establish a mutual trust in the PCS framework can be intrinsic to the physical environment (such as temperature, luminosity, noise, electrical frequency), or extrinsic to it, for example, they are actively injected by the device into the physical world. By monitoring the propagation of intrinsic or extrinsic data, a device can confirm its reception by other devices located within its vicinity. The challenge in designing and securely implementing such protocols can be addressed by the synergy of combined expertises such as signal processing, statistical detection and learning, cryptography, software engineering, and electronics.

To help appreciate the intrinsic and extrinsic evidences in addressing the security and forensics in UbiComp that involves both digital and physical elements, we now discuss two examples. Consider first an intrinsic signature of power grids. The electric network frequency (ENF) is the supply frequency of power distribution grids, with a nominal value of 60Hz (North America) or 50Hz (Europe). At any given time, the instantaneous value of ENF usually fluctuates around its nominal value as a result of the dynamic interaction between the load variations in the grid and the control mechanisms for power generation. These variations are nearly identical in all locations of the same grid at a given time due to the interconnected nature of the grid. The changing values of instantaneous ENF over time forms an ENF signal, which can be intrinsically captured by audio/visual recordings (Fig. 6) or other sensors [198, 199]. This has led to recent forensic applications, such as validating the time-of-recording of an ENF-containing multimedia signal and estimating its recording location using concurrent reference signals from power grids based on the use of ENF signals.

Next, consider the recent work by Satchidanandan and Kumar [200] introducing a notion of watermarking in a cyber-physical system, which can be viewed as a class of extrinsic signatures. If an actuator injects into the system a properly designed probing signal that is unknown in advance to other nodes in the system, then based on the knowledge of the cyber-physical system's dynamics and other properties, the actuator can examine the sensors' report about the signals at various points and can potentially infer whether there is malicious activity in the system or not, and if so, where and how.

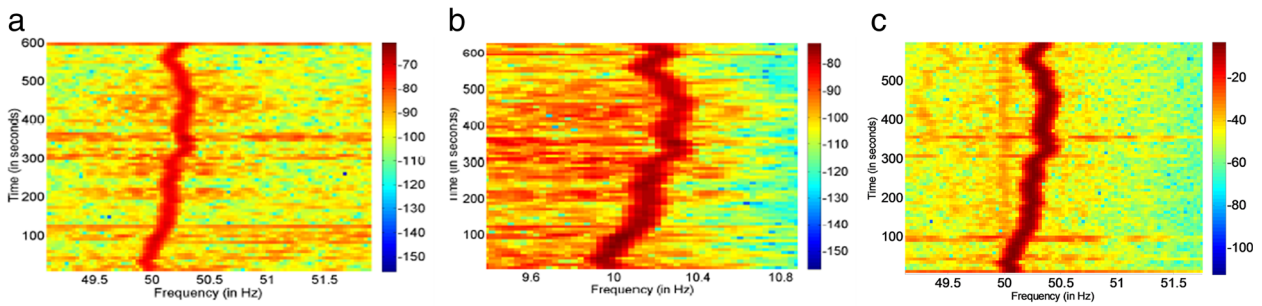

Fig. 6 An example of intrinsic evidence related to the power grid. Showing here are spectrograms of ENF signals in concurrent recordings of **a** audio, **b** visual, and **c** power main. Cross-correlation study can show the similarity between media and power line reference at different time lags, where a strong peak appears at the temporal alignment of the matching grid

A major challenge and research opportunity lies on discovering and characterizing suitable intrinsic and extrinsic evidences. Although qualitative properties of some signatures are known, it is important to develop quantitative models to characterize the normal and abnormal behavior in the context of the overall system. Along this line, the exploration of physical models might yield analytic approximations of such properties; and in the meantime, data-driven learning approaches can be used to gather statistical data characterizing normal and abnormal behaviors. Building on these elements, a strong synergy across the boundaries of traditionally separate domains of computer forensics, information forensics, and device forensics should be developed so as to achieve comprehensive capabilities of system forensics in UbiComp.

9 Conclusion

In the words of Mark Weiser, Ubiquitous Computing is "the idea of integrating computers seamlessly into the world at large" [1]. Thus, far from being a phenomenon from this time, the design and practice of UbiComp systems were already being discussed one quarter of a century ago. In this article, we have revisited this notion, which permeates the most varied levels of our society, under a security and privacy point of view. In the coming years, these two topics will occupy much of the time of researchers and engineers. In our opinion, the use of this time should be guided by a few observations, which we list below:

- UbiComp software is often produced as the combination of different programming languages, sharing a common core often implemented in a type-unsafe language such as C, C++ or assembly. Applications built in this domain tend to be distributed, and their analysis, i.e., via static analysis tools, needs to consider a holistic view of the system.
- The long-life span of some of these systems, coupled with the difficulty (both operational and cost-wise) to update and re-deploy them, makes them vulnerable to the inexorable progress of technology and cryptanalysis techniques. This brings new (and possibly disruptive) players to this discussion, such as quantum adversaries.
- Key management is a critical component of any secure or private real-world system. After security roles and key management procedures are clearly defined for all entities in the framework, a set of matching cryptographic primitives must be deployed. Physical access and constrained resources complicate the design of efficient and secure cryptographic algorithms, which are often amenable to side-channel attacks. Hence, current research challenges in the space include more efficient key management schemes, in particular supporting some form of revocation; the design of lightweight cryptographic primitives which facilitate correct and secure implementation; cheaper side-channel resistance countermeasures made available through advances in algorithms and embedded architectures.
- Given the increasing popularization of UbiComp systems, people become more and more dependent on their services for performing different commercial, financial, medical and social transactions. This rising dependence requires simultaneous high level of reliability, availability and security. This observation strengthens the importance of the design and implementation of resilient UbiComp systems.
- One of the main challenges to providing pervasive IdM is to ensure the authenticity of devices and users and adaptive authorization in scenarios with multiple and heterogeneous security domains.
- Several databases currently store sensitive data. Moreover, a vast number of sensors are constantly

collecting new sensitive data and storing them in clouds. Privacy-preserving protocols are being designed and perfected to enhance user's privacy in specific scenarios. Cultural interpretations of privacy, the variety of laws, big data from legacy systems in clouds, processing time, latency, key distribution and management, among other aforementioned are challenges for us to develop privacy-preserving protocols.

– The convergence between the physical and digital systems poses both challenges and opportunities in offering forensic capabilities to facilitate the authentication of data as well as the access conditions including who, when, where, and how; a synergistic use of intrinsic and extrinsic evidences with interdisciplinary expertise will be the key.

Given these observations, and the importance of ubiquitous computing, it is easy to conclude that the future holds fascinating challenges waiting for the attention of the academia and the industry.

Finally, note the observations and the predictions presented in this work regarding how UbiComp may evolve represent our view of the field based on the technology landscape today. New scientific discoveries, technology inventions as well as economic, social, and policy factors may lead to new and/or different trends in the technology evolutionary paths.

Endnotes

[1] https://competitions.cr.yp.to/caesar.html

[2] https://csrc.nist.gov/projects/lightweight-cryptography

[3] Deloitte's annual Technology, Media and Telecommunications Predictions 2017 report: https://www2.deloitte.com/content/dam/Deloitte/global/Documents/Technology-Media-Telecommunications/gx-deloitte-2017-tmt-predictions.pdf

[4] https://www.fiware.org.

[5] OAuth 2.0 core authorization framework is described by IETF in RFC 6749 and other specifications and profiles.

[6] https://arx.deidentifier.org/

[7] https://css.csail.mit.edu/cryptdb/

Abbreviations

AAI: Authentication and Authorization Infrastructure; ABAC: Attribute Based Access Control; ACL: Access Control List; AES: Advanced Encryption Standard; CapBAC: Capability Based Access Control; CFG: control flow graph; CLPKC: Certificateless cryptography; DDoS: Distributed Denial of Service; ECC: Elliptic Curve Cryptography; ECP: Enhanced Client and Proxy; eID: Electronic identity; ENF: Electric network frequency; FIM: Federated Identity Management Model; HBS: Hash-Based Signatures; IBC: Identity-based; IdM: Identity Management; IdP: Identity Provider; IdPaaS: Identity Provider as a Service; IoT: Internet of things; MAC: Message Authentication Codes; MFA: Multi-factor authentication; PACS: Physical access control systems; PCS: Proof-Carrying Sensing; PDP: Policy Decision Point; PDPaaS: Policy Decision Point as a Service; PEP: Policy Enforcement Point; PKIs: Public key infrastructures; PQC: Post-quantum cryptography; QC: Quantum cryptography; RBAC: Role Based Access Control; SP: Service Provider; SSO: Single Sign-On; UbiComp: Pervasive and ubiquitous computing; XACML: eXtensible Access Control Markup Language

Acknowledgments

We would like to thank Artur Souza for contributing with fruitful discussions to this work.

Funding

This work was partially supported by the CNPq, NSF, RNP, FAPEMIG, FAPERJ, and CAPES.

Availability of data and materials

Data sharing not applicable to this article as no datasets were generated or analyzed during the current study.

Authors' contributions

All authors wrote and reviewed the manuscript. Mainly, LBO focused on the introduction and the whole paper conception, FM focused on Software Protection, RM focused on Long-Term Security, DFA focused on Cryptographic Engineering, MN focused on Resilience, MW focused on Identity Management, FB focused on Privacy, MW focused on Forensics, JL focused on the conclusion and the whole paper conception. All authors read and approved the final manuscript.

Authors' information

Leonardo B. Oliveira is an associate professor of the CS Department at UFMG, a visiting associate professor of the CS Department at Stanford, and a research productivity fellow of the Brazilian Research Council (CNPq). Leonardo has been awarded the Microsoft Research Ph.D. Fellowship Award, the IEEE Young Professional Award, and the Intel Strategic Research Alliance Award. He published papers on the security of IoT/Cyber-Physical Systems in publication venues like IPSN and SenSys, and he is the (co)inventor of an authentication scheme for IoT (USPTO Patent Application No. 62287832). Leonardo served as General Chair and TPC Chair of the Brazilian Symposium on Security (SBSeg) in 2014 and 2016, respectively, and as a member in the Advisory Board of the Special Interest Group on Information and Computer System Security (CESeg) of the Brazilian Computer Society. He is a member of the Technical Committee of Identity Management (CT-GId) of the Brazilian National Research and Education Network (RNP).

Fernando M. Q. Pereira is an associate professor at UFMG's Computer Science Department. He got his Ph.D at the University of California, Los Angeles, in 2008, and since then does research in the field of compilers. He seeks to develop techniques that let programmers to produce safe, yet efficient code. Fernando's portfolio of analyses and optimizations is available at http://cuda.dcc.ufmg.br/. Some of these techniques found their way into important open source projects, such as LLVM, PHC and Firefox.

Rafael Misoczki is a Research Scientist at Intel Labs, USA. His work is focused on post-quantum cryptography and conventional cryptography. He contributes to international standardization efforts on cryptography (expert member of the USA delegation for ISO/IEC JTC1 SC27 WG2, expert member of INCITS CS1, and submitter to the NIST standardization competition on post-quantum cryptography). He holds a PhD degree from Sorbonne Universités (University of Paris - Pierre et Marie Curie), France (2013). He also holds an MSc. degree in Electrical Engineering (2010) and a BSc. degree in Computer Science (2008), both from the Universidade de São Paulo, Brazil.

Diego F. Aranha is an Assistant Professor in the Institute of Computing at the University of Campinas (Unicamp). He holds a PhD degree in Computer Science from the University of Campinas and has worked as a visiting PhD student for 1 year at the University of Waterloo. His professional experience is in Cryptography and Computer Security, with a special interest in the efficient implementation of cryptographic algorithms and security analysis of real world systems. Coordinated the first team of independent researchers capable of detecting and exploring vulnerabilities in the software of the Brazilian voting machine during controlled tests organized by the electoral authority. He received the Google Latin America Research Award for research on privacy twice, and the MIT TechReview's Innovators Under 35 Brazil Award for his work in electronic voting.

Fábio Borges is Professor in the doctoral program at Brazilian National Laboratory for Scientific Computing (LNCC in Portuguese). He holds a Ph.D. degree in Doctor of Engineering (Dr.-Ing.) in the Department of Computer Science at TU Darmstadt, a master's degree in Computational Modeling at LNCC, and a bachelor's degree in mathematics at Londrina State University (UEL). Currently, he is developing research at the LNCC in the field of Algorithms, Security, Privacy, and Smart Grid. Further information is found at http://www.lncc.br/~borges/ .

Michele Nogueira is an Associate Professor of the Computer Science Department at Federal University of Paraná. She received her doctorate in Computer Science from the UPMC — Sorbonne Universités, Laboratoire d'Informatique de Paris VI (LIP6) in 2009. Her research interests include wireless networks, security and dependability. She has been working on providing resilience to self-organized, cognitive and wireless networks by adaptive and opportunistic approaches for many years. Dr. Nogueira was one of the pioneers in addressing survivability issues in self-organized wireless networks, being her works "A Survey of Survivability in Mobile Ad Hoc Networks" and "An Architecture for Survivable Mesh Networking" her prominent scientific contributions. She is an Associate Technical Editor for the IEEE Communications Magazine and the Journal of Network and Systems Management. She serves as Vice-chair for the IEEE ComSoc - Internet Technical Committee. She is an ACM and IEEE Senior Member.

Michelle S. Wangham is a Professor at University of Vale do Itajaí (Brazil). She received her M.Sc. and Ph.D. on Electrical Engineering from the Federal University of Santa Catarina (UFSC) in 2004. Recently, she was a Visiting Researcher at University of Ottawa. Her research interests are vehicular networks, security in embedded and distributed systems, identity management, and network security. She is a consultant of the Brazilian National Research and Education Network (RNP) acting as the coordinator of Identity Management Technical Committee (CT-GID) and member of Network Monitoring Technical Committee. Since 2013, she is coordinating the GIdLab project, a testbed for R&D in Identity Management.

Min Wu received the B.E. degree (Highest Honors) in electrical engineering - automation and the B.A. degree (Highest Honors) in economics from Tsinghua University, Beijing, China, in 1996, and the Ph.D. degree in electrical engineering from Princeton University in 2001. Since 2001, she has been with the University of Maryland, College Park, where she is currently a Professor and a University Distinguished Scholar-Teacher. She leads the Media and Security Team, University of Maryland, where she is involved in information security and forensics and multimedia signal processing. She has coauthored two books and holds nine U.S. patents on multimedia security and communications. Dr. Wu coauthored several papers that won awards from the IEEE, ACM, and EURASIP, respectively. She also received an NSF CAREER award in 2002, a TR100 Young Innovator Award from the MIT Technology Review Magazine in 2004, an ONR Young Investigator Award in 2005, a ComputerWorld "40 Under 40" IT Innovator Award in 2007, an IEEE Mac Van Valkenburg Early Career Teaching Award in 2009, a University of Maryland Invention of the Year Award in 2012 and in 2015, and an IEEE Distinguished Lecturer recognition in 2015–2016. She has served as the Vice President-Finance of the IEEE Signal Processing Society (2010–2012) and the Chair of the IEEE Technical Committee on Information Forensics and Security (2012–2013). She is currently the Editor-in-Chief of the IEEE Signal Processing Magazine. She was elected IEEE Fellow for contributions to multimedia security and forensics.

Jie Liu Dr. Jie Liu is a Principal Researcher at Microsoft AI and Research Redmond, WA. His research interests root in sensing and interacting with the physical world through computing. Examples include time, location, and energy awareness, and Internet/Intelligence of Things. He has published broadly in areas such as sensor networking, embedded devices, mobile and ubiquitous computing, and data center management. He has received 6 best paper awards in top academic conferences in these fields. In addition, he holds more than 100 patents. He is the Steering Committee chair of Cyber-Physical-System (CPS) Week, and ACM/IEEE IPSN, and a Steering Committee member of ACM SenSys. He is an Associate Editor of ACM Trans. on Sensor Networks, was an Associate Editor of IEEE Trans. on Mobile Computing, and has chaired a number of top-tier conferences. Among other recognitions, he received the Leon Chua Award from UC Berkeley in 2001; Technology Advance Award from (Xerox) PARC in 2003; and a Gold Star Award from Microsoft in 2008. He received his Ph.D. degree from Electrical Engineering and Computer Sciences, UC Berkeley in 2001, and his Master and Bachelor degrees from Department of Automation, Tsinghua University, Beijing, China. From 2001 to 2004, he was a research scientist in Palo Alto Research Center (formerly Xerox PARC). He is an ACM Distinguished Scientist and an IEEE Senior Member.

Competing interests
The authors declare that they have no competing interests.

Publisher's Note

Author details
[1]UFMG, Av. Antônio Carlos, 6627, Prédio do ICEx, Anexo U, sala 6330 Pampulha, Belo Horizonte, MG, Brasil. [2]Federal University of Minas Gerais Belo Horizonte, Campinas, Brasil. [3]Intel Labs, Hillsboro, Campinas, Brasil. [4]University of Campinas, Campinas, Brasil. [5]National Laboratory for Scientific Computing, Petrópolis, Campinas, Brasil. [6]Federal University of Paraná, Curitiba, Campinas, Brasil. [7]Universidade do Vale do Itajaí, Florianópolis, Campinas, Brasil. [8]University of Maryland, Maryland, USA. [9]Microsoft Research, Redmond, MD, USA.

Received: 13 April 2018 Accepted: 27 August 2018
Published online: 04 December 2018

References

1. Weiser M. The computer for the 21st century. Sci Am. 1991;265(3): 94–104.
2. Weiser M. Some computer science issues in ubiquitous computing. Commun ACM. 1993;36(7):75–84.
3. Lyytinen K, Yoo Y. Ubiquitous computing. Commun ACM. 2002;45(12): 63–96.
4. Estrin D, Govindan R, Heidemann JS, Kumar S. Next century challenges: Scalable coordination in sensor networks. In: MobiCom'99. New York: ACM; 1999. p. 263–70.
5. Pottie GJ, Kaiser WJ. Wireless integrated network sensors. Commun ACM. 2000;43(5):51–8.
6. Ashton K. That 'Internet of Things' Thing. RFiD J. 2009;22:97–114.
7. Atzori L, Iera A, Morabito G. The internet of things: a survey. Comput Netw. 2010;54(15):2787–805.
8. Mann S. Wearable computing: A first step toward personal imaging. Computer. 1997;30(2):25–32.
9. Martin T, Healey J. 2006's wearable computing advances and fashions. IEEE Pervasive Comput. 2007;6(1):14–6.
10. Lee EA. Cyber-physical systems-are computing foundations adequate. In: NSF Workshop On Cyber-Physical Systems: Research Motivation, Techniques and Roadmap, volume 2. Citeseer; 2006.
11. Rajkumar RR, Lee I, Sha L, Stankovic J. Cyber-physical systems: the next computing revolution. In: 47th Design Automation Conference. ACM; 2010.
12. Abowd GD, Mynatt ED. Charting past, present, and future research in ubiquitous computing. ACM Trans Comput Human Interact (TOCHI). 2000;7(1):29–58.
13. Stajano F. Security for ubiquitous computing. Hoboken: Wiley; 2002.
14. Pierce BC. Types and programming languages, 1st edition. Cambridge: The MIT Press; 2002.
15. Cousot P, Cousot R. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In: POPL. New York: ACM; 1977. p. 238–52.
16. McMillan KL. Symbolic model checking. Norwell: Kluwer Academic Publishers; 1993.
17. Leroy X. Formal verification of a realistic compiler. Commun ACM. 2009;52(7):107–15.
18. Rice HG. Classes of recursively enumerable sets and their decision problems. Trans Amer Math Soc. 1953;74(1):358–66.
19. Wilson RP, Lam MS. Efficient context-sensitive pointer analysis for c programs. In: PLDI. New York: ACM; 1995. p. 1–12.
20. Cadar C, Dunbar D, Engler D. KLEE: Unassisted and automatic generation of high-coverage tests for complex systems programs. In: OSDI. Berkeley: USENIX; 2008. p. 209–24.
21. Coppa E, Demetrescu C, Finocchi I. Input-sensitive profiling. In: PLDI. New York: ACM; 2012. p. 89–98.
22. Graham SL, Kessler PB, McKusick MK. gprof: a call graph execution profiler (with retrospective). In: Best of PLDI. New York: ACM; 1982. p. 49–57.

23. Godefroid P, Klarlund N, Sen K. Dart: directed automated random testing. In: PLDI. New York: ACM; 2005. p. 213–23.
24. Nethercote N, Seward J. Valgrind: a framework for heavyweight dynamic binary instrumentation. In: PLDI. New York: ACM; 2007. p. 89–100.
25. Luk C-K, Cohn R, Muth R, Patil H, Klauser A, Lowney G, Wallace S, Reddi VJ, Hazelwood K. Pin: Building customized program analysis tools with dynamic instrumentation. In: PLDI. New York: ACM; 2005. p. 190–200.
26. Rimsa AA, D'Amorim M, Pereira FMQ. Tainted flow analysis on e-SSA-form programs. In: CC. Berlin: Springer; 2011. p. 124–43.
27. Serebryany K, Bruening D, Potapenko A, Vyukov D. Addresssanitizer: a fast address sanity checker. In: ATC. Berkeley: USENIX; 2012. p. 28.
28. Russo A, Sabelfeld A. Dynamic vs. static flow-sensitive security analysis. In: CSF. Washington: IEEE; 2010. p. 186–99.
29. Carlini N, Barresi A, Payer M, Wagner D, Gross TR. Control-flow bending: On the effectiveness of control-flow integrity. In: SEC. Berkeley: USENIX; 2015. p. 161–76.
30. Klein G, Elphinstone K, Heiser G, Andronick J, Cock D, Derrin P, Elkaduwe D, Engelhardt K, Kolanski R, Norrish M, Sewell T, Tuch H, Winwood S. sel4: Formal verification of an os kernel. In: SOSP. New York: ACM; 2009. p. 207–20.
31. Jourdan J-H, Laporte V, Blazy S, Leroy X, Pichardie D. A formally-verified c static analyzer. In: POPL. New York: ACM; 2015. p. 247–59.
32. Soares LFG, Rodrigues RF, Moreno MF. Ginga-NCL: the declarative environment of the brazilian digital tv system. J Braz Comp Soc. 2007;12(4):1–10.
33. Maas AJ, Nazaré H, Liblit B. Array length inference for c library bindings. In: ASE. New York: ACM; 2016. p. 461–71.
34. Fedrecheski G, Costa LCP, Zuffo MK. ISCE. Washington: IEEE; 2016.
35. Rellermeyer JS, Duller M, Gilmer K, Maragkos D, Papageorgiou D, Alonso G. The software fabric for the internet of things. In: IOT. Berlin, Heidelberg: Springer-Verlag; 2008. p. 87–104.
36. Furr M, Foster JS. Checking type safety of foreign function calls. ACM Trans Program Lang Syst. 2008;30(4):18:1–18:63.
37. Dagenais B, Hendren L. OOPSLA. New York: ACM; 2008. p. 313–28.
38. Melo LTC, Ribeiro RG, de Araújo MR, Pereira FMQ. Inference of static semantics for incomplete c programs. Proc ACM Program Lang. 2017;2(POPL):29:1–29:28.
39. Godefroid P. Micro execution. In: ICSE. New York: ACM; 2014. p. 539–49.
40. Manna Z, Waldinger RJ. Toward automatic program synthesis. Commun ACM. 1971;14(3):151–65.
41. López HA, Marques ERB, Martins F, Ng N, Santos C, Vasconcelos VT, Yoshida N. Protocol-based verification of message-passing parallel programs. In: OOPSLA. New York: ACM; 2015. p. 280–98.
42. Bronevetsky G. Communication-sensitive static dataflow for parallel message passing applications. In: CGO. Washington: IEEE; 2009. p. 1–12.
43. Teixeira FA, Machado GV, Pereira FMQ, Wong HC, Nogueira JMS, Oliveira LB. Siot: Securing the internet of things through distributed system analysis. In: IPSN. New York: ACM; 2015. p. 310–21.
44. Lhoták O, Hendren L. Context-sensitive points-to analysis: Is it worth it? In: CC. Berlin, Heidelberg: Springer; 2006. p. 47–64.
45. Agha G. An overview of actor languages. In: OOPWORK. New York: ACM; 1986. p. 58–67.
46. Haller P, Odersky M. Actors that unify threads and events. In: Proceedings of the 9th International Conference on Coordination Models and Languages. COORDINATION'07. Berlin, Heidelberg: Springer-Verlag; 2007. p. 171–90.
47. Imam SM, Sarkar V. Integrating task parallelism with actors. In: OOPSLA. New York: ACM; 2012. p. 753–72.
48. Cousot P, Cousot R, Logozzo F. A parametric segmentation functor for fully automatic and scalable array content analysis. In: POPL. New York: ACM; 2011. p. 105–18.
49. Nazaré H, Maffra I, Santos W, Barbosa L, Gonnord L, Pereira FMQ. Validation of memory accesses through symbolic analyses. In: OOPSLA. New York: ACM; 2014.
50. Paisante V, Maalej M, Barbosa L, Gonnord L, Pereira FMQ. Symbolic range analysis of pointers. In: CGO. New York: ACM; 2016. p. 171–81.
51. Maalej M, Paisante V, Ramos P, Gonnord L, Pereira FMQ. Pointer disambiguation via strict inequalities. In: Proceedings of the 2017 International Symposium on Code Generation and Optimization, *CGO '17*. Piscataway: IEEE Press; 2017. p. 134–47.
52. Maalej M, Paisante V, Pereira FMQ, Gonnord L. Combining range and inequality information for pointer disambiguation. Sci Comput Program. 2018;152(C):161–84.
53. Sui Y, Fan X, Zhou H, Xue J. Loop-oriented pointer analysis for automatic simd vectorization. ACM Trans Embed Comput Syst. 2018;17(2):56:1–56:31.
54. Poovendran R. Cyber-physical systems: Close encounters between two parallel worlds [point of view]. Proc IEEE. 2010;98(8):1363–6.
55. Conti JP. The internet of things. Commun Eng. 2006;4(6):20–5.
56. Rinaldi SM, Peerenboom JP, Kelly TK. Identifying, understanding, and analyzing critical infrastructure interdependencies. IEEE Control Syst. 2001;21(6):11–25.
57. US Bureau of Transportation Statistics BTS. Average age of automobiles and trucks in operation in the united states. 2017. Accessed 14 Sept 2017.
58. U.S. Department of Transportation. IEEE 1609 - Family of Standards for Wireless Access in Vehicular Environments WAVE. 2013.
59. Maurer M, Gerdes JC, Lenz B, Winner H. Autonomous driving: technical, legal and social aspects. Berlin: Springer; 2016.
60. Patel N. 90% of startups fail: Here is what you need to know about the 10%. 2015. https://www.forbes.com/sites/neilpatel/2015/01/16/90-of-startups-will-fail-heres-what-you-need-to-know-about-the-10/. Accessed 09 Sept 2018.
61. Jacobsson A, Boldt M, Carlsson B. A risk analysis of a smart home automation system. Futur Gener Comput Syst. 2016;56(Supplement C):719–33.
62. Rivest RL, Shamir A, Adleman LM. A method for obtaining digital signatures and public-key cryptosystems. Commun ACM. 1978;21(2):120–6.
63. Miller VS. Use of elliptic curves in cryptography. In: CRYPTO, volume 218 of Lecture Notes in Computer Science. Berlin: Springer; 1985. p. 417–26.
64. Koblitz N. Elliptic curve cryptosystems. Math Comput. 1987;48(177):203–9.
65. Barbulescu R, Gaudry P, Joux A, Thomé E. A heuristic quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic. In: EUROCRYPT 2014. Berlin: Springer; 2014. p. 1–16.
66. Diffie W, Hellman M. New directions in cryptography. IEEE Trans Inf Theor. 2006;22(6):644–54.
67. Barker E. Federal Information Processing Standards Publication (FIPS PUB) 186-4 Digital Signature Standard (DSS). 2013.
68. Barker E, Johnson D, Smid M. Special publication 800-56A recommendation for pair-wise key establishment schemes using discrete logarithm cryptography. 2006.
69. Simon DR. On the power of quantum computation. In: Symposium on Foundations of Computer Science (SFCS 94). Washington: IEEE Computer Society; 1994. p. 116–23.
70. Knill E. Physics: quantum computing. Nature. 2010;463(7280):441–3.
71. Grover LK. A fast quantum mechanical algorithm for database search. In: Proceedings of ACM STOC 1996. New York: ACM; 1996. p. 212–19.
72. Shor PW. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J Comput. 1997;26(5):1484–509.
73. McEliece RJ. A public-key cryptosystem based on algebraic coding theory. Deep Space Netw. 1978;44:114–6.
74. Merkle RC. Secrecy, authentication and public key systems / A certified digital signature. PhD thesis, Stanford. 1979.
75. Regev O. On lattices, learning with errors, random linear codes, and cryptography. In: Proceedings of ACM STOC '05. STOC '05. New York: ACM; 2005. p. 84–93.
76. Buchmann J, Dahmen E, Hülsing A. Xmss - a practical forward secure signature scheme based on minimal security assumptions. In: Yang B-Y, editor. PQCrypto. Berlin: Springer; 2011. p. 117–29.
77. McGrew DA, Curcio M, Fluhrer S. Hash-Based Signatures. Internet Engineering Task Force (IETF). 2017. https://datatracker.ietf.org/doc/html/draft-mcgrew-hash-sigs-13. Accessed 9 Sept 2018.
78. Bennett CH, Brassard G. Quantum cryptography: public key distribution and coin tossing. In: Proceedings of IEEE ICCSSP'84. New York: IEEE Press; 1984. p. 175–9.
79. Bos J, Costello C, Ducas L, Mironov I, Naehrig M, Nikolaenko V, Raghunathan A, Stebila D. Frodo: Take off the ring! practical, quantum-secure key exchange from LWE. Cryptology ePrint Archive, Report 2016/659. 2016. http://eprint.iacr.org/2016/659.

80. Alkim E, Ducas L, Pöppelmann T, Schwabe P. Post-quantum key exchange - a new hope. Cryptology ePrint Archive, Report 2015/1092. 2015. http://eprint.iacr.org/2015/1092.
81. Misoczki R, Tillich J-P, Sendrier N, PBarreto LSM. MDPC-McEliece: New McEliece variants from moderate density parity-check codes. In: IEEE International Symposium on Information Theory – ISIT'2013. Istambul: IEEE; 2013. p. 2069–73.
82. Hoffstein J, Pipher J, Silverman JH. Ntru: A ring-based public key cryptosystem. In: International Algorithmic Number Theory Symposium. Berlin: Springer; 1998. p. 267–88.
83. Bos J, Ducas L, Kiltz E, Lepoint T, Lyubashevsky V, Schanck JM, Schwabe P, Stehlé D. Crystals–kyber: a CCA-secure module-lattice-based KEM. IACR Cryptol ePrint Arch. 2017;2017:634.
84. Aragon N, Barreto PSLM, Bettaieb S, Bidoux L, Blazy O, Deneuville J-C, Gaborit P, Gueron S, Guneysu T, Melchor CA, Misoczki R, Persichetti E, Sendrier N, Tillich J-P, Zemor G. BIKE: Bit flipping key encapsulation. Submission to the NIST Standardization Process on Post-Quantum Cryptography. 2017. https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Round-1-Submissions.
85. Barreto PSLM, Gueron S, Gueneysu T, Misoczki R, Persichetti E, Sendrier N, Tillich J-P. Cake: Code-based algorithm for key encapsulation. In: IMA International Conference on Cryptography and Coding. Berlin: Springer; 2017. p. 207–26.
86. Jao D, De Feo L. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In: International Workshop on Post-Quantum Cryptography. Berlin: Springer; 2011. p. 19–34.
87. Costello C, Jao D, Longa P, Naehrig M, Renes J, Urbanik D. Efficient compression of sidh public keys. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. Berlin: Springer; 2017. p. 679–706.
88. Jao D, Azarderakhsh R, Campagna M, Costello C, DeFeo L, Hess B, Jalali A, Koziel B, LaMacchia B, Longa P, Naehrig M, Renes J, Soukharev V, Urbanik D. SIKE: Supersingular isogeny key encapsulation. Submission to the NIST Standardization Process on Post-Quantum Cryptography. 2017. https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Round-1-Submissions.
89. Galbraith SD, Petit C, Shani B, Ti YB. On the security of supersingular isogeny cryptosystems. In: International Conference on the Theory and Application of Cryptology and Information Security. Berlin: Springer; 2016. p. 63–91.
90. National Institute of Standards and Technology (NIST). Standardization Process on Post-Quantum Cryptography. 2016. http://csrc.nist.gov/groups/ST/post-quantum-crypto/. Accessed 9 Sept 2018.
91. McGrew D, Kampanakis P, Fluhrer S, Gazdag S-L, Butin D, Buchmann J. State management for hash-based signatures. In: International Conference on Research in Security Standardization. Springer; 2016. p. 244–60.
92. Bernstein DJ, Hopwood D, Hülsing A, Lange T, Niederhagen R, Papachristodoulou L, Schneider M, Schwabe P, Wilcox-O'Hearn Z. SPHINCS: Practical Stateless Hash-Based Signatures. Berlin, Heidelberg: Springer Berlin Heidelberg; 2015. p. 368–97.
93. Barker E, Barker W, Burr W, Polk W, Smid M. Recommendation for key management part 1: General (revision 3). NIST Spec Publ. 2012;800(57): 1–147.
94. Waters B. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In: Public Key Cryptography. LNCS, 6571 vol. Berlin: Springer; 2011. p. 53–70.
95. Liu Z, Wong DS. Practical attribute-based encryption: Traitor tracing, revocation and large universe. Comput J. 2016;59(7):983–1004.
96. Oliveira LB, Aranha DF, Gouvêa CPL, Scott M, Câmara DF, López J, Dahab R. Tinypbc: Pairings for authenticated identity-based non-interactive key distribution in sensor networks. Comput Commun. 2011;34(3):485–93.
97. Kim T, Barbulescu R. Extended tower number field sieve: A new complexity for the medium prime case. In: CRYPTO (1). LNCS, 9814 vol. Berlin: Springer; 2016. p. 543–71.
98. Boneh D, Franklin MK. Identity-based encryption from the weil pairing. SIAM J Comput. 2003;32(3):586–615.
99. Al-Riyami SS, Paterson KG. Certificateless public key cryptography. In: ASIACRYPT. LNCS, 2894 vol. Berlin: Springer; 2003. p. 452–73.
100. Boldyreva A, Goyal V, Kumar V. Identity-based encryption with efficient revocation. IACR Cryptol ePrint Arch. 2012;2012:52.
101. Simplício Jr. MA, Silva MVM, Alves RCA, Shibata TKC. Lightweight and escrow-less authenticated key agreement for the internet of things. Comput Commun. 2017;98:43–51.
102. Neto ALM, Souza ALF, Cunha ÍS, Nogueira M, Nunes IO, Cotta L, Gentille N, Loureiro AAF, Aranha DF, Patil HK, Oliveira LB. Aot: Authentication and access control for the entire iot device life-cycle. In: SenSys. New York: ACM; 2016. p. 1–15.
103. Mouha N. The design space of lightweight cryptography. IACR Cryptol ePrint Arch. 2015;2015:303.
104. Daemen J, Rijmen V. The Design of Rijndael: AES - The Advanced Encryption Standard. Information Security and Cryptography. Berlin: Springer; 2002.
105. Grosso V, Leurent G, Standaert F-X, Varici K. Ls-designs: Bitslice encryption for efficient masked software implementations. In: FSE. LNCS, 8540 vol. Berlin: Springer; 2014. p. 18–37.
106. Dinu D, Perrin L, Udovenko A, Velichkov V, Großschädl J, Biryukov A. Design strategies for ARX with provable bounds: Sparx and LAX. In: ASIACRYPT (1). LNCS, 10031 vol. Berlin: Springer; 2016. p. 484–513.
107. Albrecht MR, Driessen B, Kavun EB, Leander G, Paar C, Yalçin T. Block ciphers - focus on the linear layer (feat. PRIDE). In: CRYPTO (1). LNCS, 8616 vol. Berlin: Springer; 2014. p. 57–76.
108. Beierle C, Jean J, Kölbl S, Leander G, Moradi A, Peyrin T, Sasaki Y, Sasdrich P, Sim SM. The SKINNY family of block ciphers and its low-latency variant MANTIS. In: CRYPTO (2). LNCS, 9815 vol. Berlin: Springer; 2016. p. 123–53.
109. Bogdanov A, Knudsen LR, Leander G, Paar C, Poschmann A, Robshaw MJB, Seurin Y, Vikkelsoe C. PRESENT: an ultra-lightweight block cipher. In: CHES. LNCS, 4727 vol. Berlin: Springer; 2007. p. 450–66.
110. Reis TBS, Aranha DF, López J. PRESENT runs fast - efficient and secure implementation in software. In: CHES, volume 10529 of Lecture Notes in Computer Science. Berlin: Springer; 2017. p. 644–64.
111. Aumasson J-P, Bernstein DJ. Siphash: A fast short-input PRF. In: INDOCRYPT. LNCS, 7668 vol. Berlin: Springer; 2012. p. 489–508.
112. Kölbl S, Lauridsen MM, Mendel F, Rechberger C. Haraka v2 - efficient short-input hashing for post-quantum applications. IACR Trans Symmetric Cryptol. 2016;2016(2):1–29.
113. Aumasson J-P, Neves S, Wilcox-O'Hearn Z, Winnerlein C. BLAKE2: simpler, smaller, fast as MD5. In: ACNS. LNCS, 7954 vol. Berlin: Springer; 2013. p. 119–35.
114. Stevens M, Karpman P, Peyrin T. Freestart collision for full SHA-1. In: EUROCRYPT (1). LNCS, 9665 vol. Berlin: Springer; 2016. p. 459–83.
115. NIST Computer Security Division. SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions. FIPS Publication 202, National Institute of Standards and Technology, U.S. Department of Commerce, May 2014.
116. McGrew DA, Viega J. The security and performance of the galois/counter mode (GCM) of operation. In: INDOCRYPT. LNCS, 3348 vol. Berlin: Springer; 2004. p. 343–55.
117. Koblitz N. A family of jacobians suitable for discrete log cryptosystems. In: CRYPTO, volume 403 of LNCS. Berlin: Springer; 1988. p. 94–99.
118. Bernstein DJ. Curve25519: New diffie-hellman speed records. In: Public Key Cryptography. LNCS, 3958 vol. Berlin: Springer; 2006. p. 207–28.
119. Bernstein DJ, Duif N, Lange T, Schwabe P, Yang B-Y. High-speed high-security signatures. J Cryptographic Eng. 2012;2(2):77–89.
120. Costello C, Longa P. Four$\mathbb{Q}$: Four-dimensional decompositions on a $\mathbb{Q}$-curve over the mersenne prime. In: ASIACRYPT (1). LNCS, 9452 vol. Berlin: Springer; 2015. p. 214–35.
121. Banik S, Bogdanov A, Regazzoni F. Exploring energy efficiency of lightweight block ciphers. In: SAC. LNCS, 9566 vol. Berlin: Springer; 2015. p. 178–94.
122. Dinu D, Corre YL, Khovratovich D, Perrin L, Großschädl J, Biryukov A. Triathlon of lightweight block ciphers for the internet of things. NIST Workshop on Lightweight Cryptography. 2015.
123. Kocher PC. Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems. In: CRYPTO. LNCS, 1109 vol. Berlin: Springer; 1996. p. 104–13.
124. Rodrigues B, Pereira FMQ, Aranha DF. Sparse representation of implicit flows with applications to side-channel detection. In: Zaks A,

Hermenegildo MV, editors. Proceedings of the 25th International Conference on Compiler Construction, CC 2016, Barcelona, Spain, March 12-18, 2016. New York: ACM; 2016. p. 110–20.
125. Almeida JB, Barbosa M, Barthe G, Dupressoir F, Emmi M. Verifying constant-time implementations. In: USENIX Security Symposium. Berkeley: USENIX Association; 2016. p. 53–70.
126. Kocher PC, Jaffe J, Jun B. Differential power analysis. In: CRYPTO. LNCS, 1666 vol. Springer; 1999. p. 388–97.
127. Biham E, Shamir A. Differential fault analysis of secret key cryptosystems. In: CRYPTO. LNCS, 1294 vol. Berlin: Springer; 1997. p. 513–25.
128. Kim Y, Daly R, Kim J, Fallin C, Lee J-H, Lee D, Wilkerson C, Lai K, Mutlu O. Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors. In: ISCA. Washington, DC: IEEE Computer Society; 2014. p. 361–72.
129. Ishai Y, Sahai A, Wagner D. Private circuits: Securing hardware against probing attacks. In: CRYPTO. LNCS, 2729 vol. Springer; 2003. p. 463–81.
130. Balasch J, Gierlichs B, Grosso V, Reparaz O, Standaert F-X. On the cost of lazy engineering for masked software implementations. In: CARDIS. LNCS, 8968 vol. Berlin: Springer; 2014. p. 64–81.
131. Nogueira M, dos Santos AL, Pujolle G. A survey of survivability in mobile ad hoc networks. IEEE Commun Surv Tutor. 2009;11(1):66–77.
132. Mansfield-Devine S. The growth and evolution of ddos. Netw Secur. 2015;2015(10):13–20.
133. Thielman S, Johnston C. Major Cyber Attack Disrupts Internet Service Across Europe and US. https://www.theguardian.com/technology/2016/oct/21/ddos-attack-dyn-internet-denial-service. Accessed 3 July 2018.
134. DDoS attacks: For the hell of it or targeted – how do you see them off? http://www.theregister.co.uk/2016/09/22/ddos_attack_defence/. Accessed 14 Feb 2017.
135. Santos AA, Nogueira M, Moura JMF. A stochastic adaptive model to explore mobile botnet dynamics. IEEE Commun Lett. 2017;21(4):753–6.
136. Macedo R, de Castro R, Santos A, Ghamri-Doudane Y, Nogueira M. Self-organized SDN controller cluster conformations against DDoS attacks effects. In: 2016 IEEE Global Communications Conference, GLOBECOM, 2016, Washington, DC, USA, December 4–8, 2016. Piscataway: IEEE; 2016. p. 1–6.
137. Soto J, Nogueira M. A framework for resilient and secure spectrum sensing on cognitive radio networks. Comput Netw. 2015;79:313–22.
138. Lipa N, Mannes E, Santos A, Nogueira M. Firefly-inspired and robust time synchronization for cognitive radio ad hoc networks. Comput Commun. 2015;66:36–44.
139. Zhang C, Song Y, Fang Y. Modeling secure connectivity of self-organized wireless ad hoc networks. In: IEEE INFOCOM. Piscataway: IEEE; 2008. p. 251–5.
140. Salem NB, Hubaux J-P. Securing wireless mesh networks. IEEE Wirel Commun. 2006;13(2):50–5.
141. Yang H, Luo H, Ye F, Lu S, Zhang L. Security in mobile ad hoc networks: challenges and solutions. IEEE Wirel Commun. 2004;11(1):38–47.
142. Nogueira M. SAMNAR: A survivable architecture for wireless self-organizing networks. PhD thesis, Université Pierre et Marie Curie - LIP6. 2009.
143. ITU. NGN identity management framework: International Telecommunication Union (ITU); 2009. Recommendation Y.2720.
144. Lopez J, Oppliger R, Pernul G. Authentication and authorization infrastructures (aais): a comparative survey. Comput Secur. 2004;23(7):578–90.
145. Arias-Cabarcos P, Almenárez F, Trapero R, Díaz-Sánchez D, Marín A. Blended identity: Pervasive idm for continuous authentication. IEEE Secur Priv. 2015;13(3):32–39.
146. Bhargav-Spantzel A, Camenisch J, Gross T, Sommer D. User centricity: a taxonomy and open issues. J Comput Secur. 2007;15(5):493–527.
147. Garcia-Morchon O, Kumar S, Sethi M, Internet Engineering Task Force. State-of-the-art and challenges for the internet of things security. Internet Engineering Task Force; 2017. https://datatracker.ietf.org/doc/html/draft-irtf-t2trg-iot-seccons-04.
148. Torres J, Nogueira M, Pujolle G. A survey on identity management for the future network. IEEE Commun Surv Tutor. 2013;15(2):787–802.
149. Hanumanthappa P, Singh S. Privacy preserving and ownership authentication in ubiquitous computing devices using secure three way authentication. In: Proceedings. International Conference on Innovations in Information Technology (IIT); 2012. p. 107–12.
150. Fremantle P, Aziz B, Kopecký J, Scott P. Federated identity and access management for the internet of things. In: 2014 International Workshop on Secure Internet of Things; 2014. p. 10–17.
151. Domenech MC, Boukerche A, Wangham MS. An authentication and authorization infrastructure for the web of things. In: Proceedings of the 12th ACM Symposium on QoS and Security for Wireless and Mobile Networks, Q2SWinet '16. New York: ACM; 2016. p. 39–46.
152. Birrell E, Schneider FB. Federated identity management systems: A privacy-based characterization. IEEE Secur Priv. 2013;11(5):36–48.
153. Nguyen T-D, Al-Saffar A, Huh E-N. A dynamic id-based authentication scheme. In: Proceedings. Sixth International Conference on Networked Computing and Advanced Information Management (NCM), 2010. 2010. p. 248–53.
154. Gusmeroli S, Piccione S, Rotondi D. A capability-based security approach to manage access control in the internet of things. Math Comput Model. 2013;58:1189–205.
155. Akram H, Hoffmann M. Supports for identity management in ambient environments-the hydra approach. In: Proceedings. 3rd International Conference on Systems and Networks Communications, 2008. ICSNC'08. 2008. p. 371–7.
156. Liu J, Xiao Y, Chen CLP. Authentication and access control in the internet of things. In: Proceedings. 32nd International Conference on Distributed Computing Systems Workshops (ICDCSW) 2012. 2012. p. 588–92.
157. Ndibanje B, Lee H-J, Lee S-G. Security analysis and improvements of authentication and access control in the internet of things. Sensors. 2014;14(8):14786–805.
158. Kim Y-P, Yoo S, Yoo C. Daot: Dynamic and energy-aware authentication for smart home appliances in internet of things. In: Consumer Electronics (ICCE), 2015 IEEE International Conference on. 2015. p. 196–7.
159. Markmann T, Schmidt TC, Wählisch M. Federated end-to-end authentication for the constrained internet of things using ibc and ecc. SIGCOMM Comput Commun Rev. 2015;45(4):603–4.
160. Dasgupta D, Roy A, Nag A. Multi-factor authentication. Cham: Springer International Publishing; 2017. p. 185–233.
161. NIST. Digital Identity Guidelines. NIST Special Publication 800-63-3. 2017. https://doi.org/10.6028/NIST.SP.800-63-3.
162. Dzurenda P, Hajny J, Zeman V, Vrba K. Modern physical access control systems and privacy protection. In: 2015 38th International Conference on Telecommunications and Signal Processing (TSP). 2015. p. 1–5.
163. Guinard D, Fischer M, Trifa V. Sharing using social networks in a composable web of things. In: Proceedings. 8th IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010. 2010. p. 702–7.
164. Rotondi D, Seccia C, Piccione S. Access control & IoT: Capability based authorization access control system. In: Proceedings. 1st IoT International Forum; 2011.
165. Mahalle PN, Anggorojati B, Prasad NR, Prasad R. Identity authentication and capability based access control (iacac) for the internet of things. J Cyber Secur Mob. 2013;1(4):309–48.
166. Moreira Sá De Souza L, Spiess P, Guinard D, Köhler M, Karnouskos S, Savio D. Socrades: A web service based shop floor integration infrastructure. In: The internet of things. Springer; 2008. p. 50–67.
167. Jindou J, Xiaofeng Q, Cheng C. Access control method for web of things based on role and sns. In: Proceedings. IEEE 12th International Conference on Computer and Information Technology (CIT), 2012. Washington: IEEE Computer Society; 2012. p. 316–21.
168. Han Q, Li J. An authorization management approach in the internet of things. J Inf Comput Sci. 2012;9(6):1705–13.
169. Zhang G, Liu J. A model of workflow-oriented attributed based access control. Int J Comput Netw Inf Secur (IJCNIS). 2011;3(1):47–53.
170. do Prado Filho TG, Vinicius Serafim Prazeres C. Multiauth-wot: A multimodal service for web of things athentication and identification. In: Proceedings of the 21st Brazilian Symposium on Multimedia and the Web, WebMedia '15. New York: ACM; 2015. p. 17–24.
171. Alam S, Chowdhury MMR, Noll J. Interoperability of security-enabled internet of things. Wirel Pers Commun. 2011;61(3):567–86.
172. Seitz L, Selander G, Gehrmann C. Authorization framework for the internet-of-things. In: Proceedings. IEEE 14th International Symposium and Workshops on a World of Wireless, Mobile and Multimedia Networks (WoWMoM). Washington, DC: IEEE Computer Society; 2013. p. 1–6.

173. OASIS. Saml v2.0 executive overview. 2005. https://www.oasis-open.org/committees/download.php/13525/sstc-saml-exec-overview-2.0-cd-01-2col.pdf.
174. Hardt D. The oauth 2.0 authorization framework. RFC 6749, RFC Editor; 2012. http://www.rfc-editor.org/rfc/rfc6749.txt.
175. Maler E, Reed D. The venn of identity: Options and issues in federated identity management. IEEE Secur Priv. 2008;6(2):16–23.
176. Naik N, Jenkins P. Securing digital identities in the cloud by selecting an apposite federated identity management from saml, oauth and openid connect. In: 2017 11th International Conference on Research Challenges in Information Science (RCIS). Piscataway: IEEE; 2017. p. 163–74.
177. OASIS. Authentication context for the oasis security assertion markup language (saml) v2.0. 2005. http://docs.oasis-open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf.
178. Paci F, Ferrini R, Musci A, Jr KS, Bertino E. An interoperable approach to multifactor identity verification. Computer. 2009;42(5):50–7.
179. Pöhn D, Metzger S, Hommel W. Géant-trustbroker: Dynamic, scalable management of saml-based inter-federation authentication and authorization infrastructures. In: Cuppens-Boulahia N, Cuppens F, Jajodia S, El Kalam AA, Sans T, editors. ICT Systems Security and Privacy Protection. Berlin, Heidelberg: Springer Berlin Heidelberg; 2014. p. 307–20.
180. Zeng D, Guo S, Cheng Z. The web of things: A survey. J Commun. 2011;6(6). http://ojs.academypublisher.com/index.php/jcm/article/view/jcm0606424438.
181. The OpenID Foundation. Openid connect core 1.0. 2014. http://openid.net/specs/openid-connect-core-1_0.html.
182. Domenech MC, Comunello E, Wangham MS. Identity management in e-health: A case study of web of things application using openid connect. In: 2014 IEEE 16th International Conference on e-Health Networking, Applications and Services (Healthcom). Piscataway: IEEE; 2014. p. 219–24.
183. OASIS. Extensible access control markup language (xacml) version 3.0. 2013. http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.pdf.
184. Borges F, Demirel D, Bock L, Buchmann JA, Mühlhäuser M. A privacy-enhancing protocol that provides in-network data aggregation and verifiable smart meter billing. In: ISCC. USA: IEEE; 2014. p. 1–6.
185. Borges de Oliveira F. Background and Models. Cham: Springer International Publishing; 2017. p. 13–23.
186. Borges de Oliveira F. Reasons to Measure Frequently and Their Requirements. Cham: Springer International Publishing; 2017. p. 39–47.
187. Holvast J. The Future of Identity in the Information Society, volume 298 of IFIP Advances in Information and Communication Technology. In: Matyáš V, Fischer-Hübner S, Cvrček D, Švenda P, editors. Berlin: Springer Berlin Heidelberg; 2009. p. 13–42.
188. Toshniwal D. Privacy preserving data mining techniques privacy preserving data mining techniques for hiding sensitive data hiding sensitive data: a step towards open data open data. Singapore: Springer Singapore; 2018. p. 205–12.
189. Li N, Li T, Venkatasubramanian S. t-closeness: Privacy beyond k-anonymity and l-diversity. In: 2007 IEEE 23rd International Conference on Data Engineering. USA: IEEE; 2007. p. 106–15.
190. De Montjoye Y-A, Hidalgo CA, Verleysen M, Blondel VD. Unique in the crowd: The privacy bounds of human mobility. Sci Rep. 2013;3:1–5.
191. Borges de Oliveira F. Quantifying the aggregation size. Cham: Springer International Publishing; 2017. p. 49–60.
192. Gentry C. A Fully Homomorphic Encryption Scheme. Stanford: Stanford University; 2009. AAI3382729.
193. Borges de Oliveira F. A Selective Review. Cham: Springer International Publishing; 2017. p. 25–36.
194. Borges de Oliveira F. Selected Privacy-Preserving Protocols. Cham: Springer International Publishing; 2017. p. 61–100.
195. Borges F, Lara P, Portugal R. Parallel algorithms for modular multi-exponentiation. Appl Math Comput. 2017;292:406–16.
196. Stamm MC, Wu M, Liu KJR. Information forensics: An overview of the first decade. IEEE Access. 2013;1:167–200.
197. Wu M, Quintão Pereira FM, Liu J, Ramos HS, Alvim MS, Oliveira LB. New directions: Proof-carrying sensing — Towards real-world authentication in cyber-physical systems. In: Proceedings of ACM Conf. on Embedded Networked Sensor Systems (SenSys). New York: ACM; 2017.
198. Grigoras C. Applications of ENF analysis in forensic authentication of digital audio and video recordings. J Audio Eng Soc. 2009;57(9):643–61.
199. Garg R, Varna AL, Hajj-Ahmad A, Wu M. "seeing" enf: Power-signature-based timestamp for digital multimedia via optical sensing and signal processing. TIFS. 2013;8(9):1417–32.
200. Satchidanandan B, Kumar PR. Dynamic watermarking: Active defense of networked cyber–physical systems. Proc IEEE. 2017;105(2):219–40.