Capítulo 1

O Computador para o Século 21: Desafios de Segurança e Privacidade após 25 Anos[1]

Leonardo B. Oliveira[a], **Fernando Magno Quintão Pereira**[a], **Rafael Misoczki**[b], **Diego F. Aranha**[c], **Fábio Borges**[d], **Michele Nogueira**[e], and **Michelle Wangham**[f]

[a]Universidade Federal de Minas Gerais (UFMG)
[b]Intel Corporation
[c]Universidade Estadual de Campinas (Unicamp)
[d]Laboratório Nacional de Computação Científica (LNCC)
[e]Universidade Federal do Paraná (UFPR)
[f]Universidade do Vale do Itajaí (UNIVALI)

Resumo

Passaram décadas desde Mark Weiser publicou seu trabalho influente sobre como seria um computador do século 21. Ao longo dos anos, alguns dos recursos da UbiComp apresentados nesse mincurso foram sendo gradualmente adotados pelas indústrias no mercado de tecnologia. Embora tal evolução tecnológica tenha resultado em muitos benefícios para a nossa sociedade, ela também colocou, ao longo do caminho, incontáveis desafios que ainda temos que transpor. Neste minicurso, abordamos grandes desafios de duas áreas que mais afligem a revolução UbiComp: segurança e privacidade. Em particular, examinamos problemas abertos em proteção de software, segurança de longo prazo, engenharia de criptografia, implicações de privacidade, resiliência cibernética e gestão de identidade. Nós também apontamos direções promissoras para as soluções desses problemas. Acreditamos que se compreendermos tanto os desafios como os novos rumos corretamente, então conseguiremos tornar a ficção científica de UbiComp em ciência de fato.

[1]Uma versão preliminar deste trabalho foi apresentada em ICCCN 2017 [Oliveira et al. 2017].

1.1. Introdução

Em 1991, Mark Weiser descreveu uma visão do Computador para o Século 21 [Weiser 1991]. Weiser, em seu artigo profético, argumentou que as tecnologias mais abrangentes são aquelas que se tornam transparentes. De acordo com Weiser, esse esquecimento é um fenômeno humano – não tecnológico –: "Sempre que as pessoas aprendem algo suficientemente bem, eles deixam de estar cientes disso", afirmou. Este evento é chamado de "dimensão tácita"ou "compilação"e pode ser testemunhado, por exemplo, quando os motoristas reagem aos sinais de rua sem conscientemente ter que processar as letras P-A-R-E [Weiser 1991].

Um quarto de século depois, no entanto, o sonho de Weiser está longe de se tornar realidade. Ao longo dos anos, muitos dos seus conceitos sobre Computação Ubíqua (UbiComp) [Weiser 1993, Lyytinen and Yoo 2002] foram materializados no que hoje chamamos de Redes de Sensores Sem Fio (*Wireless Sensor Networks* – WSNs [Estrin et al. 1999, Pottie and Kaiser 2000]), Internet das Coisas (*Internet of Things* – IoT [Ashton 2009, Atzori et al. 2010]), Computação Vestível (Wearable Computing [Mann 1997, Martin and Healey 2007]), e Sistemas-Ciber-Físicos (*Cyber-Physical Systems* – CPSs [Lee 2006, Rajkumar et al. 2010]). As aplicações desses sistemas variam de monitoramento de acidentes de trânsito e de emissões de CO_2 para o carro autônomo, a cuidados para pacientes domésticos. No entanto, além de todos os seus benefícios, o advento desses sistemas também provoca vulnerabilidades. E, a menos que as abordemos adequadamente, o futuro da profecia de Weiser estará em jogo.

A UbiComp apresenta novas vulnerabilidades porque exibe uma perspectiva totalmente diferente da computação tradicional [Abowd and Mynatt 2000]. Os elementos computacionais em UbiComp, por exemplo, apresentam sensores, CPU e atuadores. Respectivamente, isso significa que eles conseguem ouvir (ou espionar) você, processar seus dados (e, possivelmente, descobrir algo sigiloso sobre você), e responder às suas ações (ou, em última análise, te expor revelando algum segredo). Estas capacidades, por sua vez, fazem propostas para computadores convencionais inadequados no contexto de UbiComp e, por fim, apresentam novos desafios.

Muitos desses desafios estão nas áreas de Segurança e Privacidade [Stajano 2002]. Isto ocorre porque o mercado e os usuários muitas vezes buscam um sistema cheio de recursos à custa de operação e proteção adequadas; embora, contraditoriamente, na medida que elementos computacionais permeiam nossas vidas, a demanda por esquemas de segurança mais fortes se torna ainda maior [Souza et al. 2017]. Particularmente, existe uma necessidade extrema de um mecanismo de segurança capaz de abarcar todos os aspectos e manifestações de UbiComp, tanto no tempo como no espaço, e de maneira perfeita e eficiente.

Neste minicurso, discutimos questões contemporâneas de segurança e privacidade no contexto da UbiComp. Fazemos uma revisão bibliográfica e examinamos vários problemas de pesquisa ainda em abertos e apresentamos rumos promissores para suas soluções. Mais precisamente, ao longo deste minicurso, investigaremos os desafios abaixo e suas ramificações (vide Figura 1.1).

Figura 1.1. Tópicos abordados no minicurso

Proteção de Software (Seção 1.2) Linguagens de programação são a ferramenta fundamental para a construção de software. Assim, a implementação de sistemas confiáveis e seguros depende, não somente, da habilidade de programadores, mas também das linguagens de programação usadas. Uma das partes deste minicurso irá cobrir essa dependência intrínseca entre linguagens de programação e segurança de software. Iremos analisar as dificuldades que as linguagens fracamente tipadas impõem à construção de software seguro, e estudaremos como programação poliglota. O advento da Internet das Coisas tem tornado a programação poliglota cada vez mais comum. Finalmente, abordaremos também os desafios que a programação distribuída traz para a detecção de vulnerabilidades de software, mostrando como ferramentas modernas podem ser usadas para mitigar tais problemas [Teixeira et al. 2015].

Segurança de Longo Prazo (Seção 1.3) Diversos sistemas de UbiComp são projetados para oferecer uma vida útil de vários anos, até mesmo décadas [Poovendran 2010, Conti 2006]. No contexto de infraestrutura crítica, por exemplo, um enorme investimento financeiro pode ser necessário para que sistemas sejam projetados e efetivamente implantados. Tais sistemas ofereceriam um melhor retorno sobre investimento caso permanecessem em funcionamento por mais tempo. A área automotiva, a qual pertence a UbiComp dado que carros atualmente dispõem de diversos componentes eletrônicos (sendo alguns já conectados à Internet e futuramente ainda mais dependente dessa troca de dados [DoT 2013]), é outra área que demanda sistemas de longa vida útil, uma vez que veículos devem ser funcionais por várias décadas. Renovar frotas veiculares e até mesmo atualizar características pontuais de um veículo (processo conhecido como *recall*) levam ao aumento de custo por parte do usuário, ressaltando a importância de se oferecer soluções robustas por um longo prazo.

Uma maneira efetiva de reduzir a vida útil desses sistemas consiste em comprometer as técnicas de segurança da informação por eles utilizadas. Neste contexto, iremos examinar os principais fatores de risco que ameaçam técnicas criptográficas atuais quando integradas a sistemas com longa expectativa de vida útil. Entre tais ameaças, discutiremos avanços recentes de técnicas de criptanálise, a futura disrupção causada por ataques de computadores quânticos contra criptossistemas convencionais e os novos desafios de implementação de criptossistemas resistentes a ataques de computadores quânticos. Por fim, apresentaremos direções (e inerentes desafios) em como se garantir que sistemas estejam preparados para oferecer segurança de longo prazo em UbiComp.

Engenharia Criptográfica (Seção 1.4). Os sistemas UbiComp envolvem blocos de construção de naturezas muito diferentes: componentes de hardware como sensores e atuadores, software embarcado implementando protocolos de comunicação, interface com provedores de computação em nuvem e, finalmente, procedimentos operacionais e outros fatores humanos. Como resultado, sistemas pervasivos exibem uma ampla superfície de ataque que deve ser protegida utilizando uma combinação de técnicas. A criptografia é peça fundamental nessa proteção, mas é improvável que seja o componente mais fraco. Os protocolos de rede, as rotinas de tratamento de entrada e mesmo o código de interface com mecanismos criptográficos são componentes muito mais propensos a serem vulneráveis à exploração. No entanto, um ataque bem-sucedido em propriedades de segurança criptográficas costuma ser desastroso. Essa seção discutirá boas práticas e problemas de pesquisa no projeto e implementação de mecanismos criptográficos em computação ubíqua, de forma que criptografia cumpra seu papel de representar o componente mais robusto de um sistema computacional moderno.

Resiliência Cibernética (Seção 1.5). A disponibilidade dos serviços essenciais, tais como comunicação fim-a-fim, roteamento e conectividade, na UbiComp é um dos principais pilares da segurança, em conjunto com a confidencialidade e integridade dos dados. A resiliência cibernética visa identificar, prevenir, detectar e responder a processos ou falhas tecnológicas, e recuperar ou reduzir danos e as suas perdas financeiras [Lima et al. 2009]. A violação da disponibilidade dos serviços vem sendo cada vez mais frequente, vide o último ataque de negação de serviço contra a empresa DYN, um dos principais provedores de DNS, e o ataque de negação de serviço contra a OVH, a gigante empresa francesa. O primeiro chegou a um intenso volume de tráfego malicioso de aproximadamente 1 Tbps, gerado a partir de dispositivos que compõem os modernos sistemas UbiComp [ddo]. Como não há nenhuma maneira garantida de evitar estes ataques, as empresas devem se concentrar na criação de soluções mais resistentes aos seus danos e rapidamente retomar a disponibilidade dos serviços. Nos concentraremos nessa seção em apresentar a importância da resiliência cibernética no contexto de sistemas UbiComp e apresentar uma visão geral do estado da arte e direções futuras de pesquisa e inovação [Santos et al. 2017, Macedo et al. 2016, Soto and Nogueira 2015, Lipa et al. 2015].

Gestão de Identidade (Seção 1.6). Um sistema de gestão de identidades (GId) consiste na integração de políticas, processos de negócios e tecnologias de identificação, autenticação, autorização e auditoria. Uma abordagem apropriada de gestão de identidade

(pervasiva) é necessária para que a computação ubíqua seja invisível para usuários [Arias-Cabarcos et al. 2015]. Os sistemas UbiComp são compostos de dispositivos heterogêneos que precisam provar a sua autenticidade para as entidades com as quais se comunicam. Um problema neste cenário é garantir a autenticidade não apenas de usuários mas também de dispositivos que se comunicam entre si e que podem estar localizados em domínios administrativos de segurança diferentes [Wangham et al. 2013]. A segurança naturalmente consome recursos e, assim, acrescentar soluções de IdM em dispositivos embarcados com restrições computacionais pode ser um desafio [Oliveira et al. 2011]. Os dispositivos empregados em sistemas UbiComp, normalmente, atuam no mundo físico. Isso significa que uma ameaça à segurança de um dispositivo pode ter impactos no mundo físico. Ou seja, há o risco de que uma violação influencie o mundo físico ao ponto de atentar contra o bem-estar e até à vida das pessoas [Wu et al. 2017, Domenech et al. 2016]. Nessa seção, os principais desafios para prover GId serão discutidos, por meio da análise das principais IAAs e dos trabalhos recentes que tratam dos problemas de pequisa levantados. Por fim, oportunidades de pesquisa e de inovação nesta área serão indicadas.

Implicações na Privacidade (Seção 1.7) A segurança – ou, em particular, a confidencialidade – é uma condição necessária mas não suficiente para um protocolo garantir Privacidade [Borges 2016b]. Para garantir a segurança, os sistemas UbiComp devem implementar mecanismos de proteção como canais de comunicação seguros, de modo que apenas os remetentes autorizados cifrem mensagens para os receptores autorizados para decrifrá-las. Para garantir a privacidade, é importante determinar as fontes relevantes de vazamento e empregar protocolos de preservação da privacidade para manipular dados confidenciais [Borges de Oliveira 2017b]. Tais protocolos geralmente precisam usar criptossistemas para garantir que o receptor possa obter as informações necessárias sem inferir dados confidenciais sobre o remetente.

Acreditamos que se compreendermos tanto os desafios como os novos rumos corretamente, então conseguiremos tornar a ficção científica da UbiComp em Ciência e realidade de fato.

1.2. Proteção de Software

Os sistemas modernos de computação ubíqua raramente são criados a partir do zero. Os componentes desenvolvidos por diferentes organizações, com diferentes modelos e ferramentas de programação, e sob diferentes pressupostos, são integrados para oferecer recursos complexos aos seus usuários. Essa complexidade traz consigo desafios de segurança, que serão analisados neste capítulo. Para este fim, nesta seção, analisamos o ecossistema de software que caracteriza o campo de computação ubíqua. A Figura 1.2 fornece uma representação de alto nível desse ecossistema. Nos concentramos especialmente em três aspectos desse ambiente, que impõem desafios de segurança aos desenvolvedores: deficiências de segurança de C e C++, as linguagens de programação dominantes em implementações de sistemas ciber-físicos; as interações entre essas linguagens com outras e as consequências dessas interações na natureza distribuída das aplicações ubíquas. Começamos nosso estudo mergulhando mais profundamente nas idiossincrasias das linguagens C e C++.

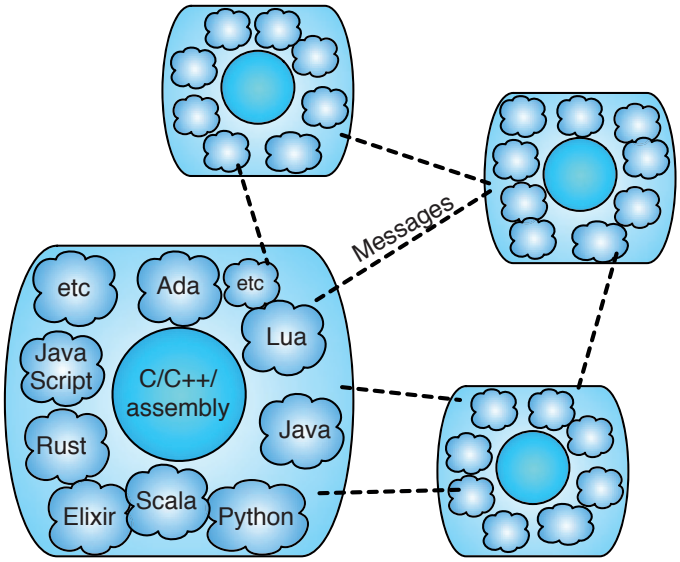


Figura 1.2. Um sistema de computação ubíqua é formado por módulos implementados como uma combinação de diferentes linguagens de programação. Essa diversidade traz consigo desafios para a construção de software seguro.

1.2.1. Segurança de tipos

Uma grande parte do software usado nos sistemas ubíquos é implementado em C ou em C++. Esse fato é natural, dada a eficiência incomparável dessas linguagens de programação. No entanto, se C e C++ são eficientes, por um lado, seu sistema de tipos, conhecido como *tipagem fraca* dá origem a uma infinidade de vulnerabilidades de software. Em jargão de linguagens de programação, dizemos que um sistema de tipo é fraco quando ele não suporta duas propriedades principais: *progresso* e *preservação* [Pierce 2002]. As definições formais dessas propriedades são imateriais para a discussão que segue. Basta saber que, como consequência da tipagem fraca, nem C, nem C++, protegem, por exemplo, a memória contra acessos inválidos. Portanto, os programas escritos nessas linguagens podem ler ou escrever posições inválidas de memória. Como uma ilustração dos perigos incorridos por esta possibilidade, basta lembrar que o acesso fora de limites alocados é o princípio por trás de um tipo de ataque conhecido como estouro de buffer.

A comunidade de segurança de software vem desenvolvendo diferentes técnicas para lidar com as vulnerabilidades intrínsecas de C e C++, e também de linguagens Assembly. Tais técnicas podem ser totalmente estáticas, totalmente dinâmicas ou um híbrido de ambas. Os mecanismos de proteção estática são implementados no nível do compilador; mecanismos dinâmicos são implementados no nível do ambiente de execução. No restante desta seção, listamos os elementos mais conhecidos em cada categoria.

As análises estáticas fornecem uma estimativa conservadora do comportamento do programa, sem exigir a sua execução. Esta ampla família de técnicas inclui, por exemplo, *Interpretação Abstrata* [Cousot and Cousot 1977], *verificação de modelos* [McMillan 1993] e *provas guiadas* [Leroy 2009]. A principal vantagem das análises estáticas é a baixa sobrecarga em tempo de execução, e sua solidez: as proprieda-

des inferidas são garantidas para sempre serem verdadeiras. No entanto, análises estáticas também apresentam desvantagens. Em particular, a maioria das propriedades interessantes dos programas é indecidível [Rice 1953]. Além disso, a verificação de muitas propriedades formais, ainda que decidível, implica um custo computacional proibitivo [Wilson and Lam 1995].

As análises dinâmicas vêm em vários sabores: teste (KLEE [Cadar et al. 2008]), perfilamento (Aprof [Coppa et al. 2012], Gprof [Graham et al. 1982]), execução simbólica (DART [Godefroid et al. 2005]), emulação (Valgrind [Nethercote and Seward 2007]), e instrumentação binária (Pin [Luk et al. 2005]). As virtudes e limitações das análises dinâmicas são exatamente o oposto daquelas encontrados em técnicas estáticas. Análises dinâmicas geralmente não geram falsos alarmes: os erros são descritos por exemplos, que normalmente levam a uma reprodução consistente [Rimsa et al. 2011]. No entanto, elas não precisam sempre encontrar vulnerabilidades de segurança em programas. Além disso, a sobrecarga de tempo de execução das análises dinâmicas ainda as torna proibitivas para serem implantadas em software de produção [Serebryany et al. 2012].

Como um meio termo, vários grupos de pesquisa propuseram formas de combinar análises estáticas e dinâmicas, produzindo diferentes tipos de abordagens híbridas para proteger código de baixo nível. Esta combinação pode render garantias de segurança que são estritamente mais poderosas do que o que poderia ser obtido por técnicas estáticas ou a dinâmicas, quando usado separadamente [Russo and Sabelfeld 2010]. No entanto, os resultados negativos ainda são válidos: se um atacante pode assumir o controle de um programa, geralmente ele ou ela pode contornar proteções híbridas de última geração, tais como a integridade do fluxo de controle [Carlini et al. 2015]. Esse fato é, em última análise, uma consequência do sistema de tipo fraco adotado por linguagens normalmente vistas na implementação de sistemas de computação ubíqua. Portanto, o projeto e implantação de técnicas que podem proteger tais linguagens de programação, sem comprometer sua eficiência, continua a ser um problema aberto.

1.2.2. Programação Poliglota

A *Programação Poliglota* é a arte e a disciplina de escrever código fonte envolvendo duas ou mais linguagens de programação. É comum entre implementações de sistemas ciberfísicos. Como exemplo, Ginga, o protocolo brasileiro para TV digital, é principalmente implementado em Lua e C [Soares et al. 2007]. A Figura 1.3 mostra um exemplo de comunicação entre um programa C e um programa Lua. Outros exemplos de interações entre linguagens de programação incluem ligações entre C e Python [Maas et al. 2016] e C e Elixir [Fedrecheski et al. 2016]. Outro exemplo de programação poliglota é o uso de *Java Native Interface* [Rellermeyer et al. 2008], uma forma de combinar código Java com código escrito em outras linguagens. A programação poliglota complica a proteção de sistemas. Dificuldades surgem devido à falta de ferramentas multilínguas e devido à falta de controle de ligações de memória entre C/C++ e outras linguagens.

Um obstáculo à validação do software Polyglot é a falta de ferramentas para analisar o código-fonte escrito em diferentes linguagens de programação, sob uma estrutura unificada. Retornando à Figura 1.3, temos um sistema formado por dois programas es-

C

```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

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

Figura 1.3. Comunicação bidirecional entre um programa escrito em C e um programa escrito em Lua

critos em diferentes linguagens de programação. Qualquer ferramenta que analise este sistema como um todo deve ser capaz de analisar essas duas sintaxes distintas e inferir os pontos de conexão entre eles. Alguns trabalhos já foram realizados para esse fim, mas as soluções ainda são muito preliminares. Como exemplo, Maas *et al.* [Maas et al. 2016] implementaram modos automáticos para verificar se os arranjos C são lidos corretamente por programas Python. Como outro exemplo, Furr e Foster [Furr and Foster 2008] descreveram técnicas para proteger ligações entre OCaml e C e entre Java e C.

Uma direção promissora para analisar sistemas poliglotas é baseada na idéia de compilação do código fonte parcialmente disponível. Esta façanha consiste na reconstrução da sintaxe faltante, o que inclui as declarações necessárias para produzir uma versão mínima do programa original que pode ser analisado por ferramentas tradicionais. A análise do código parcialmente disponível possibilita testar partes de um programa poliglota em separado, de forma a produzir uma visão coesa do sistema inteiro. Pesquisadores já demonstraram que essa técnica produz código fonte analisável. A título de exemplo, cita-se o trabalho de Dagenais *et al* [Dagenais and Hendren 2008]. Observe que esse tipo de reconstrução não se restringe a linguagens de programação de alto nível. Testemunho deste fato é a noção de *micro-execução*, introduzida por Patrice Godefroid [Godefroid 2014]. A ferramenta construída por Godefroid permite o teste de código binário x86, mesmo

quando arquivos de objeto estão faltando. No entanto, apesar desta evolução, a reconstrução ainda é restrito à semântica estática de programas. A síntese do comportamento é uma disciplina próspera no computador Ciência [Manna and Waldinger 1971], mas ainda está longe de permitir a certificação de sistemas poliglotas.

1.2.3. Programação Distribuída

Os sistemas de computação ubíquos tendem a ser distribuídos. É mesmo difícil conceber qualquer uso para uma aplicação neste mundo que não interage com outros programas. E, é de conhecimento geral que a programação distribuída abre várias portas para usuários mal-intencionados. Portanto, para tornar a tecnologia ciberfísica mais segura, as ferramentas de segurança devem ser conscientes da natureza distribuída de tais sistemas. No entanto, dois desafios principais estão diante dessa exigência: a dificuldade de construir uma visão holística da aplicação distribuída e a falta de informação semântica vinculada a mensagens trocadas entre processos que se comunicam através de uma rede.

Para ser eficaz, a análise de um sistema distribuído precisa ser responsável pela interações entre as diversas partes do programa que constituem esse sistema [López et al. 2015]. Descobrir tais interações é difícil, mesmo quando nos restringimos à código escrito em uma única linguagem de programação. As dificuldades advêm da falta de informação semântica associada às operações que enviam e recebem mensagens. Em outras palavras, tais operações são definidas como parte de uma biblioteca, não como parte da linguagem de programação em si. Não obstante este fato, existem várias técnicas que inferem canais de comunicação entre diferentes partes do código-fonte. Como exemplos, temos os algoritmos de Greg Bronevetsky [Bronevetsky 2009], E Teixeira *et al.* [Teixeira et al. 2015], que criam uma visão distribuída do gráfico de fluxo de controle de um programa (CFG[2]). A beleza de tais técnicas vêm do fato de que as análises estáticas clássicas funcionam sem mais modificações nesse *CFG distribuído*. No entanto, o CFG distribuído ainda é uma aproximação conservadora do comportamento do programa. Assim, ele ainda leva análises estáticas, já imprecisas, a terem de lidar com canais de comunicação que talvez nunca existam durante a execução real de um programa.

As ferramentas que realizam análises automáticas em programas dependem de informações estáticas para produzir resultados mais precisos. Nesse sentido, os tipos são fundamentais para a compreensão dos programas. Por exemplo, em Java e outras linguagens de programação orientadas a objetos, o tipo de objetos determina como a informação flui ao longo do código do programa. No entanto, apesar desta importância, as mensagens trocadas na grande maioria dos sistemas distribuídos não possuem tipos. A razão para isso é o fato de que tais mensagens, pelo menos em C, C++ e linguagens de montagem, são arranjos de *bytes*. Para mitigar esse problema, pesquisadores vêm desenvolveram análises que inferem o conteúdos [Cousot et al. 2011] e o tamanho de arranjos [Nazaré et al. 2014] em linguagens de programação fracamente tipadas. No entanto, muitos problemas ao longo dessa direção de pesquisa permanecem abertos. Por exemplo, a análise de arranjos em linguagens de programação de baixo nível requer frequentemente a capacidade de lidar com a aritmética do ponteiro. Análises estáticas que lidam com ponteiros ainda sofrem severas restrições que dificultam a análise de programas re-

[2]Sigla transcrita do inglês *Control Flow Graph*

ais [Paisante et al. 2016]. Portanto, o projeto de ferramentas capazes de analisar software usado em sistemas de computação ubíqua ainda é um problema aberto.

1.3. Segurança de Longo Prazo

Diversos sistemas de UbiComp são projetados para oferecer uma vida útil de vários anos, até mesmo décadas [Poovendran 2010, Conti 2006]. Por exemplo, sistemas no contexto de infraestrutura crítica podem necessitar de um enorme investimento financeiro para serem projetados e efetivamente implantados [Rinaldi et al. 2001]. Logo, tais sistemas ofereceriam um melhor retorno sobre investimento caso pudessem permanecer em uso por um maior período de tempo. A área automotiva é outra área de particular interesse no que concerne a segurança de longo prazo. Veículos devem ser funcionais e confiáveis por vários anos, geralmente décadas [BTS 2017]. A renovação de frotas de veículos e até mesmo a atualização de algumas de suas características (*recall*) implicam, obviamente, em aumento de custos para seus usuários. Note que veículos fazem parte do ecossistema UbiComp pois são equipados com vários dispositivos eletrônicos embarcados, dos quais alguns já oferecem conectividade à Internet. Futuramente, tais veículos dependerão ainda mais fortemente de dados coletados e compartilhados por meio de tecnologias de rede sem fio [DoT 2013] a fim de oferecer experiências mais ricas de condução, tais como as esperadas por veículos autônomos [Maurer et al. 2016].

Vale também ressaltar que os sistemas concebidos de forma a oferecer uma vida útil de vários anos ou décadas podem estar sujeitos à escassez de manutenções futuras. Por exemplo, a concorrência na indústria de tecnologia é bastante agressiva, levando assim a uma alta taxa de empresas que saem do mercado dentro de poucos anos [Patel 2015]. Logo, o cenário de um mundo inundado por dispositivos dos quais ninguém é responsável pelo seu bom funcionamento e operação adequada certamente oferecerá importantes desafios [Jacobsson et al. 2016].

Os exemplos mencionados acima, dentre muitos outros possíveis, ilustram um crescente interesse em se projetar sistemas de UbiComp que sejam confiáveis por mais tempo e, sempre que possível, exigindo o menor número possível de atualizações. Tais requisitos têm um impacto direto sobre os mecanismos de segurança desses sistemas: um número relativamente menor de oportunidades para corrigir eventuais brechas de segurança do que em sistemas convencionais. Esta é uma situação crítica dado o intenso e dinâmico campo de pesquisa de criação e exploração de brechas de segurança. Portanto, é de extrema importância entender quais são os desafios em se prover mecanismos de segurança de longo prazo desde o início do projeto de um sistema para, desta forma, não precisar recorrer a medidas paliativas *a posteriori*.

1.3.1. A Criptografia como Componente Central

Garantir segurança de longo prazo é uma tarefa bastante desafiadora para qualquer sistema, não apenas para os sistemas de UbiComp. No mínimo, isso exige que cada componente de segurança seja à prova de ataques futuros tanto individualmente quanto quando considerando a sua integração/conexão com outros sistemas. O ingrediente fundamental dos mecanismos de segurança é a Criptografia, como será destacado na Seção 1.4 e, portanto, esse será o foco de nossa avaliação.

As técnicas de criptografia tradicionais dependem de problemas computacionais, tais como a fatoração de números inteiros [Rivest et al. 1978] e o problema do logaritmo discreto [Miller 1985, Koblitz 1987]. Acredita-se que esses problemas sejam intratáveis pelas técnicas de criptoanálise e recursos tecnológicos atuais, o que possibilitou alguns criptógrafos a criarem instâncias atualmente seguras destes criptossistemas. Por várias razões (a serem discutidas a seguir), no entanto, tais construções tendem a oferecer sérios riscos em um futuro não tão distante.

1.3.2. Avanços em Criptoanálise Clássica

A primeira ameaça para a segurança de longo prazo de qualquer criptossistema se refere a potenciais avanços em criptoanálise, ou seja, em técnicas que visam a resolver o problema de segurança subjacente a esses criptossistemas de forma mais eficiente do que atualmente (e.g., com menor tempo de processamento, menos memória, entre outros). Esquemas amplamente utilizados oferecem uma longa história de escrutínio acadêmico e industrial e, portanto, seria de se esperar pouco ou nenhum progresso nas técnicas de criptoanálise desses esquemas. No entanto, a literatura mostra alguns resultados recentes interessantes que podem sugerir o oposto.

Em [Barbulescu et al. 2014], por exemplo, os autores introduziram um novo algoritmo de complexidade quase-polinomial para resolver o problema do logaritmo discreto em corpos finitos de pequenas características. Vale relembrar que se espera que problemas matemáticos subjacentes de criptossistemas tenham complexidade exponencial para um atacante não dispondo de alçapões para a resolução de tal problema, como a chave privada associada. O problema do logaritmo discreto é, de maneira geral, o problema de segurança subjacente do algoritmo de troca de chaves *Diffie-Hellman* [Diffie and Hellman 2006], do algoritmo de assinatura digital DSA [NIST 2013] e de suas variantes em curva elíptica (ECDH [NIST 2006] e ECDSA [NIST 2013], respectivamente), apenas para mencionar alguns criptossistemas amplamente implantados. É importante ressaltar que o resultado criptoanalítico mencionado acima é restrito a corpos finitos de pequena característica, o que representa uma limitação importante para atacar implementações reais dos esquemas mencionados. No entanto, qualquer algoritmo sub-exponencial que resolva um problema conhecido de longa data pode ser visto como uma indicação relevante de que a literatura em criptoanálise está, e sempre estará, sujeita a eventuais avanços importantes.

Tais observações sobre futuros avanços em criptoanálise devem ser consideradas pelos arquitetos de sistemas de UbiComp que tenham a segurança de longo prazo como um pré-requisito. Por exemplo, implementações que suportem vários níveis de segurança são preferíveis quando comparadas àquelas que suportem apenas um nível de segurança (e.g., tamanho de chave) fixo. Desta forma, os sistemas de UbiComp devem conscientemente acomodar recursos para lidar com futuros avanços em criptoanálise ou, pelo menos, reduzir os custos referentes a atualizações de segurança.

1.3.3. Futura Disrupção Devido A Ataques Quânticos

Espera-se que computadores quânticos ofereçam melhorias dramáticas às técnicas para resolver certos problemas computacionais, conforme mostrado por Daniel R. Simon em seu artigo seminal sobre algoritmos quânticos [Simon 1994]. Por um lado, tal área de

pesquisa poderá promover avanços consideráveis em diversas tecnologias atualmente limitadas devido à ineficiência de seus algoritmos [Knill 2010]. Entretanto, por outro lado, alguns dos problemas afetados são utilizados para proteger criptossistemas amplamente implantados e que, portanto, deveriam ser sempre de difícil resolução.

Em 1996, Lov K. Grover propôs um algoritmo quântico [Grover 1996] capaz de encontrar com alta probabilidade um elemento no domínio de uma função (a qual tenha N possíveis saídas) que leva a uma saída específica em apenas $O(\sqrt{N})$ passos. Este algoritmo é extremamente importante, uma vez que pode ser utilizado para acelerar a criptoanálise da vários criptossistemas simétricos.

Devido ao algoritmo de Grover, cifras de bloco com chave de n bits, por exemplo, oferecerão apenas $n/2$ bits de segurança contra um adversário quântico, ao invés dos n bits de segurança contra um adversário clássico (não-quântico). As funções de hash também serão afetadas por esse ataque de diferentes formas, dependendo da propriedade de segurança almejada. Por exemplo, funções de hash com saída de tamanho n bits oferecerão apenas $n/3$ bits de segurança contra ataques de colisão e $n/2$ bits de segurança contra ataques de pré-imagem, ao considerar um adversário quântico. Isso seria um nível de segurança sensivelmente menor do que contra um adversário clássico: $n/2$ e n para resistência à colisão e pré-imagem, respectivamente. Tabela 1.1 resume esta avaliação. Neste contexto, AES-128 e SHA-256 (considerando resistência à colisão) não atingiriam o nível de segurança considerado mínimo (128 bits de segurança) e, portanto, não deveriam ser utilizados. É importante ressaltar, no entanto, que tanto as construções de cifras de bloco quanto de funções de hash sobreviverão a esse ataque quântico desde que tamanhos apropriados de chaves e de saída de função de hash sejam utilizados (e.g., AES-256 e SHA-384). Infelizmente, chaves e saída de função de hash maiores acarretarão em importantes desafios de desempenho. AES-256, por exemplo, é cerca de 40% menos eficiente do que o AES-128 (devido ao diferente número de *rounds*, 14 ao invés de 10).

Algoritmo	**Segurança Clássica**	**Segurança Quântica**
Cifra de Bloco (n bits)	n	$n/2$
Hash Pré-Imagem (n bits)	n	$n/2$
Hash Colisão (n bits)	$n/2$	$n/3$

Tabela 1.1. Níveis de Segurança para Criptografia Simétrica

Ainda mais crítico do que o cenário para a criptografia simétrica será o cenário para a criptografia assimétrica (também conhecida como criptografia de chaves públicas). Os computadores quânticos oferecerão uma vantagem exponencial para atacar a maioria dos criptossistemas de chave pública utilizados em larga escala atualmente. Isto ocorrerá devido ao algoritmo de Peter Shor [Shor 1997] que permite fatorar eficientemente (em tempo polinomial) números inteiros grandes e também calcular o logaritmo discreto de um elemento em grandes grupos. O impacto deste trabalho será devastador para esquemas tais como RSA e ECC, por exemplo, pois o aumento de tamanho de chaves não resolveria efetivamente esta fragilidade. Em resumo, tais criptossistemas de chaves públicas baseados em fatoração e logaritmo discreto precisarão ser completamente aposentados e, portanto, substituídos.

Existem alternativas criptográficas que são resistentes a ataques quânticos. No entanto, diversos desafios ainda precisam ser superados. O primeiro se refere ao estabelecimento de um consenso na academia e na indústria sobre quais alternativas são as mais promissoras. De maneira geral, existem duas principais técnicas que oferecem funcionalidades similares à criptografia de chaves públicas e que são capazes de sobreviver a ataques quânticos, a saber: criptografia pós-quântica (PQC, do acrônimo em inglês *Post-Quantum Cryptography*) e criptografia quântica. O primeiro é baseado em problemas computacionais clássicos que devem ser difíceis mesmo para computadores quânticos. Os esquemas PQC podem ser implementados nos computadores atuais [McEliece 1978, Merkle 1979, Regev 2005, Buchmann et al. 2011, McGrew et al. 2017], apenas para mencionar alguns poucos trabalhos. Já a outra alternativa (criptografia quântica) depende de toda uma infra-estrutura quântica a ser implantada e pode ser usado principalmente para fins de troca de chaves [Bennett and Brassard 1984]. A dificuldade e os altos custos da implantação de infra-estrutura quântica necessários à criptografia quântica favorecem o crescente interesse em *criptografia pós-quântica*.

Os criptossistemas pós-quânticos são baseados em diferentes problemas computacionais. No contexto de assinaturas digitais, assinaturas baseadas em hash são soluções bastante promissoras. Tais esquemas [Buchmann et al. 2011, McGrew et al. 2017] são variantes do esquema de assinaturas baseadas em hash proposto por Ralph C. Merkle [Merkle 1979] e sua segurança depende exclusivamente de certas propriedades já bastante estudadas de funções hash. Como funções de hash são resistentes a ataques de computadores quânticos (dependendo do tamanho da saída, conforme discutido acima), tais esquemas de assinaturas também serão. Diferentemente do contexto de assinaturas digitais, outras funcionalidades tais como esquemas de troca de chaves e criptografia assimétrica, ainda não têm um consenso na comunidade.

Ainda não existem padrões para criptossistemas pós-quânticos. Entretanto, esforços de padronização já foram iniciados tais como o liderado pelo *National Institute of Standards and Technology* (NIST) [NIST 2016]. Este processo deve levar pelo menos mais alguns anos até ser concluído. Note que a ausência atual de padrões pode representar um desafio de interoperabilidade aos sistemas que precisam oferecer segurança de longo prazo.

Outro desafio no contexto dos criptossistemas pós-quânticos de chave pública se refere aos diferentes requisitos de implementação. Como mencionado anteriormente, as assinaturas baseadas em hash são alternativas pós-quânticos interessantes (em termos de eficiência e segurança associada somente a funções de hash), mas também trazem um conjunto de desafios de implementação completamente novo. Em particular, assinaturas baseadas em hash são esquemas classificados como *stateful*, o que nesse contexto significa que, entre a emissão de duas assinaturas, o sistema precisa atualizar a sua chave privada. Caso políticas robustas de proteção e gerenciamento desse *state* não estejam em vigor, um assinante poderá forçar com que chave privada não seja atualizada, algo que anularia totalmente as garantias de segurança oferecidas pelo esquema. Recentemente, trabalhos iniciais para enfrentar esses novos desafios de implementação começaram a aparecer na literatura [McGrew et al. 2016]. Também relacionado a esse ponto, uma construção recente [Bernstein et al. 2015] mostrou como construir assinaturas baseadas em hash sem *state*. Entretanto, essa proposta vem associada com um importante aumento no tamanho

das assinaturas. Estes exemplos indicam que os novos criptossistemas trarão novos desafios de implementação, os quais devem ser corretamente considerados pelos arquitetos de sistemas de UbiComp.

1.4. Engenharia Criptográfica

A criptografia é uma peça fundamental na proteção de qualquer sistema computacional moderno, mas raramente o componente mais fraco em sua superfície de ataque. Enquanto essa posição é normalmente ocupada por fatores humanos e vulnerabilidades de *software*, mecanismos criptográficos concentram risco e um ataque com sucesso às suas propriedades termina sendo desastroso. Por exemplo, violações de confidencialidade causadas por mecanismos fracos de encriptação podem causar vazamentos massivos de dados envolvendo informação sensível. A interferência maliciosa na integridade de comunicação pode permitir ataques de injeção de comandos que forçam o sistema a desviar do comportamento esperado. A disponibilidade dos serviços é também crítica para manter o sistema acessível a usuários legítimos e garantir fornecimento contínuo do serviço, portanto mecanismos criptográficos precisam também ser leves para minimizar o potencial de abuso por atacantes.

O acesso físico por adversários a partes da superfície de ataque são um aspecto particularmente desafiador de se implementar criptografia em sistemas ubíquos. Por construção, os adversários podem ser capazes de recuperar chaves de longa duração e credenciais que terminam por fornecer certo controle sobre porções do sistema. A seguir alguns dos principais desafios na engenharia de mecanismos criptográficos serão explorados no contexto de computação ubíqua, incluindo cuidados com implantação segura, soluções para gerência de chaves e implementação eficiente de criptografia.

1.4.1. Boas Práticas

A segurança real de uma técnica criptográfica depende de uma série de premissas, que precisam ser satisfeitas para a técnica funcionar a contento. O projeto teórico de uma técnica criptográfica começa com a detecção de um *problema* computacionalmente difícil, que pode variar desde a simples seleção de uma cadeia de bits dentro de um espaço grande de chaves, caso da criptografia simétrica, até problemas difíceis em Teoria dos Números em que se baseia a criptografia assimétrica. Após detectado um problema computacionalmente difícil, é importante projetar um *algoritmo* criptográfico que utiliza o problema como fonte de confiança; e um *protocolo* em que múltiplas partes executam diferentes algoritmos para obter algum tipo de funcionalidade conjunta. A análise formal de algoritmos e protocolos permite relacionar a segurança fornecida por estes com a dificuldade de resolver os problemas subjacentes. Considerando aspectos mais aplicados de engenharia, o esquema para distribuição de chaves determina como chaves criptográficas são geradas, entregues a cada uma das partes participantes e posteriormente armazenadas. Por fim, o protocolo precisa ser concretizado em uma implementação em software ou hardware para ser implantado em produção. O ciclo de vida típico de uma aplicação criptográfica está ilustrado na Figura 1.4.

Há muitos cuidados que precisam ser tomados durante a implantação de uma técnica criptográfica:

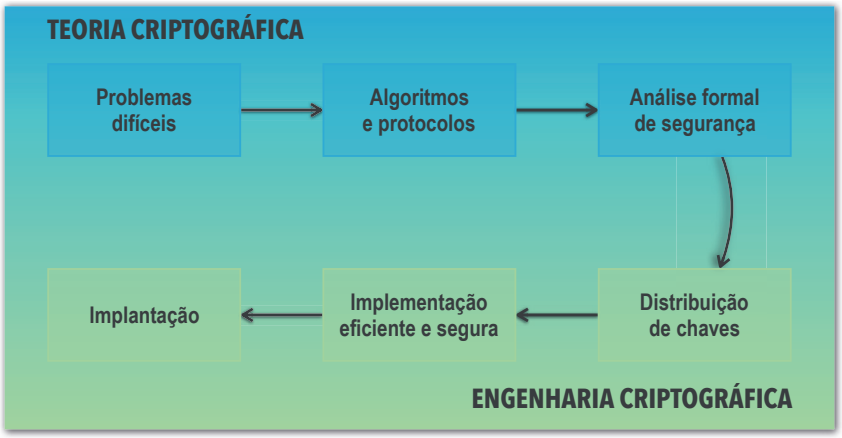


Figura 1.4. Tópicos abordados no minicurso

- **Geração de chaves:** uma fonte de entropia de alta qualidade precisa ser determinada na plataforma alvo para geração de números aleatórios. A escolha de gerador tipicamente depende da combinação de linguagem de programação, sistema operacional e bibliotecas de suporte, mas tem se tornado mais fácil em plataformas modernas. Não é preciso dizer que a prática comum de utilizar tomadas de tempo ou outros valores previsíveis é altamente desaconselhada [Aranha et al. 2014]. Recomendações incluem a utilização do gerador `/dev/urandom` do sistema operacional GNU/Linux ou da instrução `RDRAND` introduzida recentemente pela Intel.

- **Escolha de algoritmos e parâmetros:** é fundamental determinar com precisão quais as propriedades dentre sigilo, autenticação e integridade precisam ser fornecidas pelo sistema; e posteriormente quais os algoritmos que serão responsáveis por fornecer essas propriedades. Encriptação utilizando cifras de bloco e fluxo fornecem confidencialidade, funções de resumo fornecem integridade; autenticadores e assinaturas digitais fornecem autenticação de mensagem e origem, respectivamente. Resta definir quais os tamanhos de chaves e parâmetros para esses algoritmos, após exame cuidadoso do conhecimento na literatura e estimativas de dificuldade de solução dos problemas subjacentes [3].

- **Escolha de bibliotecas criptográficas:** é preferível reusar implementações disponíveis a customizar implementações de terceiros. Utilizar bibliotecas criptográficas como a NaCl[4], com interfaces amigáveis que demandem menor conhecimento especializado do desenvolvedor também é recomendável [Bernstein et al. 2012b].

- **Distribuição de chaves:** é desafiador determinar com chaves criptográficas serão distribuídas entre os múltiplos participantes. É importante evitar o compartilhamento direto de chaves simétricas ou sua inserção direta em partes desprotegidas do sistema, sendo favorável utilizar protocolos de acordos de chaves baseados em criptografia assimétrica [Oliveira et al. 2011]. Desta forma, o problema de compartilhamento de segredo se converte na obtenção de chaves públicas autênticas,

[3] https://www.keylength.com
[4] https://nacl.cr.yp.to/

possivelmente tratado com o uso de certificados digitais rigorosamente validados. Outra alternativa é transformar o problema de compartilhamento de segredo em autenticação, onde um nó restrito em recursos obtém acesso a uma chave criptográfica armazenada em um nó mais poderoso após satisfazer requisitos de autenticação, talvez com ajuda do usuário final.

A seguir, algumas das questões acima são tratadas em maior nível de detalhe, como gerência de chaves criptográficas, recomendações de algoritmos criptográficos leves e implementação segura.

1.4.2. Gerência de Chaves

Por definição, sistemas ubíquos são plataformas heterogênas, conectando dispositivos de poder computacional e capacidade de armazenamento massivamente distintas. Projetar uma arquitetura criptográfica para qualquer sistema heterogêneo requer determinar claramente os papéis e propriedades de segurança para cada entidade no sistema. Dispositivos carentes em recursos devem receber tarefas computacionalmente menos intensivas para executar, e sua falta de proteções contra ataques físicos indicam que segredos de longo prazo não devem residir nesses dispositivos. Tarefas mais críticas envolvendo criptografia de chave pública, como acordos de chaves e assinaturas digitais, devem ser portanto delegadas aos componentes mais poderosos em recursos disponíveis. Um compromisso cuidadoso entre propriedades de segurança, funcionalidade e primitivas criptográficas deve então ser construído para cada dispositivo ou classe de dispositivos [Barker et al. 2012], levando em consideração uma série de recomendações:

- **Eficiência:** protocolos de gerência de chaves devem ser leves e fáceis de implementar, direcionando a interface com Infra-estruturas de Chaves Públicas (ICPs) e operações custosas de tratamento de certificados digitais para os componentes mais poderosos da arquitetura. Modelos alternativos com suporte a certificação implícita que evitam esses problemas incluem *criptografia baseada em identidades* [Boneh and Franklin 2003] e *sem certificados* [Al-Riyami and Paterson 2003], ressaltando que a utilização de criptografia baseada em identidades implica custódia inerente de chaves criptográficas por uma autoridade central. As dificuldades com revogação de chaves ainda impõem obstáculos para adoção irrestrita desses modelos, apesar do progresso recente no projeto desses mecanismos [Boldyreva et al. 2012].

- **Funcionalidade:** protocolos para gerência de chaves devem gerenciar todo o tempo de vida de chaves criptográficas e garantir acessibilidade apenas para usuários autorizados em um certo instante de tempo. Entretanto, gerenciar chaves criptográficas e autorizar usuários em processos separados pode aumentar a complexidade e introduzir vulnerabilidades. Uma forma promissora de se combinar os dois serviços de segurança com controle de acesso de forma a satisfazer garantias criptográficas formais é a *criptografia baseada em atributos* [Waters 2011, Liu and Wong 2016], onde chaves possuem um conjunto de atributos que descrevem capacidades passíveis de autorização ou revogação sob demanda.

- **Comunicação:** componentes devem minimizar a quantidade de tráfego transmitido, sob risco de se tornarem inoperantes caso o meio de comunicação seja atacado ou se torne indisponível. Abordagens não-interativas para distribuição de chaves [Oliveira et al. 2011] são desejáveis aqui, mas protocolos avançados baseados em emparelhamentos bilineares sobre curvas elípticas devem ser cautelosamente evitados após avanços recentes no cálculo do logaritmo discreto [Kim and Barbulescu 2016]. Esses avanços forçam o aumento no tamanho dos parâmetros e reduzem desempenho e escalabilidade, favorecendo formas mais tradicionais de criptografia assimétrica.

- **Interoperabilidade:** sistemas pervasivos são compostos por componentes com naturezas distintas e origens em múltiplos fabricantes. Desta forma, mecanismos de autenticação e autorização precisam suportar diferentes domínios de aplicação para fornecer interoperabilidade entre esses dispositivos [Neto et al. 2016].

Primitivas criptográficas que participam em qualquer tipo de funcionalidade conjunta fornecida por um sistema ubíquo devem então ser compatíveis com todos os componentes e respeitar as limitações de dispositivos equipados com menos recursos.

1.4.3. Criptografia Leve

O surgimento de ecossistemas gigantescos de dispositivos conectados motiva o desenvolvimento de primitivas criptográficas adequadas a esse cenário, sob o termo *criptografia leve*. Esse termo não denota criptografia *fraca*, mas técnicas criptográficas customizadas para uma aplicação específica e cuidadosamente projetadas para serem eficientes em termos do consumo de recursos como ciclos de processamento, capacidade energética e armazenamento em memória [Mouha 2015]. Técnicas criptográficas leves procuram satisfazer requisitos de segurança usuais para algoritmos criptográficos de sua classe, mas podem adotar escolhas menos conservadoras de projeto ou construções mais recentemente introduzidas na literatura.

Como um primeiro exemplo, várias cifras de bloco para encriptação simétrica foram propostas como alternativas leves ao algoritmo *Advanced Encryption Standard* (AES) [Daemen and Rijmen 2002]. Construções importantes são *LS-Designs* [Grosso et al. 2014], redes modernas de Feistel e Adição-Rotação-Soma (ARX) [Dinu et al. 2016], e redes de substituição-permutação [Albrecht et al. 2014, Beierle et al. 2016]. Um candidato notável é a cifra de bloco PRESENT, que vem se tornando mais eficiente [Reis et al. 2017] mesmo com a maturidade alcançada após resistir a 10 anos de criptanálise [Bogdanov et al. 2007].

No caso de funções de resumo criptográficas, um projeto pode trocar propriedades de segurança avançadas (como resistência a colisões) por simplicidade em alguns cenários. Um exemplo claro dessa idéia é a construção de códigos curtos para autenticação de mensagens (*Message Authentication Codes – MACs*) a partir de funções de resumo que não são resistentes a colisões, com no caso do algoritmo SipHash [Aumasson and Bernstein 2012], ou assinaturas digitais construídas a partir de funções de resumo com entradas curtas [Kölbl et al. 2016]. Em aplicações convencionais que exigem resistência a colisões, a função BLAKE2 [Aumasson et al. 2013] tem se apre-

sentado na prática como candidata mais interessante para substituir a função padronizada SHA-1 recentemente quebrada [Stevens et al. 2016] do que o próprio padrão SHA-3 [].

Outra tendência importante é fornecer confidencialidade e autenticação em um único passo, por meio de algoritmos para cifração autenticada (*Authenticated Encryption with Associated Data – AEAD*). Essa técnica pode ser implementada como um modo de operação de uma cifra de bloco (*Galois Counter Mode – GCM* [McGrew and Viega 2004]) ou um projeto dedicado. Espera-se que a competição CAESAR [5] selecione um novo algoritmo AEAD para padronização até o final do ano. A derivação de chaves simétricas a partir de senhas pode ser realizada por uma função de derivação de chaves criptográficas que consome quantidades configuráveis de ciclos e memória, como o algoritmo Argon2 recentemente selecionado como finalista na competição *Password Hashing Competition* [Biryukov et al. 2016].

Em termos de criptografia assimétrica, criptografia de curvas elípticas (*Elliptic Curve Cryptography – ECC*) [Miller 1985, Koblitz 1988] continua sendo o principal candidato para aplicação nesse espaço, em contraste com criptossistemas baseados na fatoração de inteiros [Rivest et al. 1978]. As principais vantagens vêm da complexidade conjecturada como completamente exponencial do problema do logaritmo discreto em curvas elípticas, que permite diminuir o tamanho de chaves criptográficas e parâmetros de domínio. Instanciações modernas de ECC fornecem alto desempenho e simplicidade de implementação absolutamente compatíveis com sistemas embarcados [Bernstein 2006, Bernstein et al. 2012a, Costello and Longa 2015].

A longo prazo, a dominância de primitivas criptográficas com dificuldade baseada em problemas de Teoria dos Números é entretanto ameaçada por computadores quânticos, conforme descrito na Seção 1.3. É importante notar que essa miríade de novas primitivas sendo propostas precisa ser rigorosamente avaliada tanto do ponto de vista de segurança quanto desempenho, requerendo tanto trabalho teórico quanto aspectos de engenharia. Implementações devem consumir quantidades cada vez menores de energia [Banik et al. 2015], ciclos de processamento e memória [Dinu et al. 2015] em dispositivos cada vez menores sob ataques cada vez mais invasivos.

1.4.4. Resistência a Ataques de Canal Lateral

Caso implementados sem cuidado, um algoritmo criptográfico ou protocolo originalmente seguro pode vazar informação crítica útil a um adversário. Ataques de canal lateral [Kocher 1996] são uma ameaça significativa à segurança criptográfica e podem utilizar informação de tempo, latência da hierarquia de memória, consumo de energia e emanações eletromagnéticas para recuperar segredos criptográficos. Esses ataques emergem da interação entre implementação e arquitetura computacional subjacente e representam um problema de segurança intrínseco a ambientes de computação pervasiva, já que assume-se que o atacante possui acesso físico irrestrito a pelo menos alguns dos dispostiivos legítimos em operação.

Proteger contra ataques de canal lateral é um problema de pesquisa desafiador e contramedidas tipicamente fornecem algum grau adicional de regularidade du-

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

rante a computação. Implementações *isócronas* ou em tempo constante estão entre as primeiras estratégias de defesa para esses problema de segurança quando incidem sobre variações no tempo de execução ou tempo de resposta da hierarquia de memória. Técnicas simples envolvem eliminação de desvios condicionais com condição baseada em informação secreta e endereçamento de tabelas utilizando índices calculados a partir de segredos, dado que estas podem ou não estar armazenadas em memória *cache* [Faz-Hernández et al. 2015]. A aplicação de métodos formais tem desenvolvido as primeiras ferramentas para análise estática de execução em tempo constante da implementação, como análise de fluxo de informação [Rodrigues et al. 2016] e transformações entre programas [Almeida et al. 2016]. Análise dinâmica é normalmente realizada com heurísticas [Reparaz et al. 2017, Langley 2010], mas igualmente importante para detectar variações de tempo introduzidas pelo compilador [Kaufmann et al. 2016] ou mesmo execução do conjunto de instruções em uma certa arquitetura [Großschädl et al. 2009, Pornin 2017].

Enquanto há uma tendência recente na direção de construir e padronizar algoritmos criptográficos com alguma resistência embutida contra ataques simples de tempo ou potência [Grosso et al. 2014], ataques mais poderosos como análise diferencial de potência [Kocher et al. 1999] ou injeção de falhas [Biham and Shamir 1997] são difíceis de prevenir ou mitigar para implementações em software. Em particular, ataques de injeção de falhas se tornaram muito mais poderosos após serem demonstrados como de execução factível em software [Kim et al. 2014].

Técnicas de mascaramento [Ishai et al. 2003] são frequentemente investigadas como uma contramedida para eliminar a correlação entre informação vazada e informação secreta, mas frequentemente requerem fontes robustas de entropia para atingir seu objetivo. Reciclagem de aleatoriedade tem sido útil como uma heurística para suavizar esse requisito, mas análise formal da seguranças dessa abordagem é um problema em aberto [Balasch et al. 2014]. Modificações na arquitetura subjacente para introduzir extensões no conjunto de instruções, simplificar o ambiente de execução ou fornecer mecanismos transacionais para reiniciar computação manipulada são outra direção promissora de pesquisa, mas podem envolver alterações radicais às arquiteturas computacionais em uso, talvez proibitivas para sistemas embarcados restritos em recursos.

1.5. Resiliência Cibernética

A disponibilidade de serviços essenciais no contexto da computação ubíqua (UbiComp) é um dos principais pilares da segurança, juntamente com a confidencialidade e a integridade dos dados. A resiliência visa identificar, prevenir, detectar e responder a falhas tecnológicas ou processuais, a fim de recuperar ou reduzir danos e perdas financeiras [Lima et al. 2009]. A violação da disponibilidade do serviço está se tornando cada vez mais frequente e disruptiva. Um exemplo é o último ataque de Negação de Serviço Distribuído (DDoS) contra a provedora de nomes DYN e o ataque DDoS contra a empresa OVH, o gigante provedor francês de hospedagem de sites. O primeiro atingiu um volume intenso, e inédito, de tráfego malicioso de aproximadamente 1.2 Tbps, gerado a partir de uma grande quantidade de dispositivos geograficamente distribuídos e infectados, como impressoras, câmeras IP, *gateways* residenciais e monitores para bebês, que compõem os modernos sistemas UbiComp [ddo]. Como não existe uma maneira garantida

de evitar esses ataques, as soluções resilientes tornam-se uma forma de mitigar os danos e retomar rapidamente a disponibilidade de serviços. Esta seção se concentra em apresentar a importância da resiliência no contexto dos sistemas UbiComp, aborda o estado da arte, os requisitos de resiliência, e aponta as futuras direções para pesquisa e inovação [Santos et al. 2017, Macedo et al. 2016, Soto and Nogueira 2015, Lipa et al. 2015].

As melhorias nas tecnologias da informação e da comunicação, como as redes sem fio, aumentaram a importância dos sistemas distribuídos em nossas vidas diárias. A primeira década do século 21 testemunhou o aumento dos sistemas de grande escala, como as redes sociais de escala global, os mercados globais de publicidade, e o comércio global [Delic 2016]. O acesso à rede está se tornando onipresente através de dispositivos portáteis e comunicações sem fio, tornando as pessoas cada vez mais dependentes dela. Isso aumenta a demanda por um alto nível de confiabilidade, disponibilidade e segurança simultânea em transações comerciais, financeiras, médicas e sociais suportadas pela UbiComp.

A Internet de coisas (IoT), as redes veiculares e as redes de corporais sem fio são exemplos de sistemas distribuídos comuns e que compõem a UbiComp. Essas redes são compostas por dispositivos portáteis heterogêneos, comunicando-se entre eles, em geral, de maneira multi-salto e através de tecnologias de comunicação sem fio [Zhang et al. 2008]. Os pesquisadores vêm trabalhando para que essas redes sem fio possam se adaptar de forma autônoma às mudanças em seu ambiente, como por exemplo às mudanças na posição do dispositivo, o padrão de tráfego e à interferência. Cada dispositivo pode reconfigurar dinamicamente sua topologia, cobertura e alocação do canal de acordo com as mudanças [Cremonezi et al. 2017]. Além disso, em geral, nenhuma entidade centralizada controla a rede, exigindo uma abordagem de gerenciamento descentralizada [Lima et al. 2009, Nogueira 2009].

Como já enfatizado ao longo deste capítulo, a segurança é crucial para os sistemas UbiComp, particularmente quando eles oferecem suporte a aplicativos sensíveis à segurança como em contextos militares, de segurança pública, finanças e de saúde. As ameaças de segurança aproveitam falhas de protocolos e vulnerabilidades dos sistemas operacionais, bem como as características da rede. Essas redes representam desafios não triviais para o projeto de segurança devido às suas características, como o meio sem fio compartilhado, a topologia de rede altamente dinâmica, a comunicação multi-salto e a baixa proteção física dos dispositivos portáteis [Salem and Hubaux 2006, Yang et al. 2004]. Além disso, a ausência de entidades centrais aumenta a complexidade das operações de gerenciamento de segurança, particularmente o controle de acesso, a autenticação dos dispositivos e a distribuição de chaves criptográficas, tais como discutido na Seção 1.4.

Várias soluções de segurança vêm sendo propostas [Hu et al. 2003, Lou et al. 2004, Marti et al. 2000, Papadimitratos and Haas 2003, Refaei et al. 2005, Yi et al. 2001, Zapata 2002]. Essas soluções empregaram duas linhas de defesa bem definidas como preventiva ou reativa, nas quais a primeira tenta prevenir ataques por criptografia, autenticação e mecanismos de controle de acesso, e a última procura detectar intrusões e reagir de acordo [Lima et al. 2009, Nogueira 2009]. No entanto, naturalmente cada mecanismo de segurança aborda problemas específicos com limitações para lidar com diferentes tipos de ataques e intrusões. As defesas preventivas, por exemplo, são

vulneráveis a dispositivos infectados que participam de operações da rede e do sistema, enquanto que as reativas em sua maioria funcionam eficientemente somente contra ataques ou intrusões conhecidos.

Devido às limitações das linhas de defesa preventiva e reativa, os pesquisadores propuseram complementá-las através da abordagem de tolerância a intrusões [Yang et al. 2004], tal como ilustrado na Figura 1.5. Esta linha de defesa tem como objetivo melhorar a resiliência da rede e do ciberespaço contra ataques e intrusões usando técnicas de tolerância a falhas, geralmente redundância e mecanismos de recuperação.

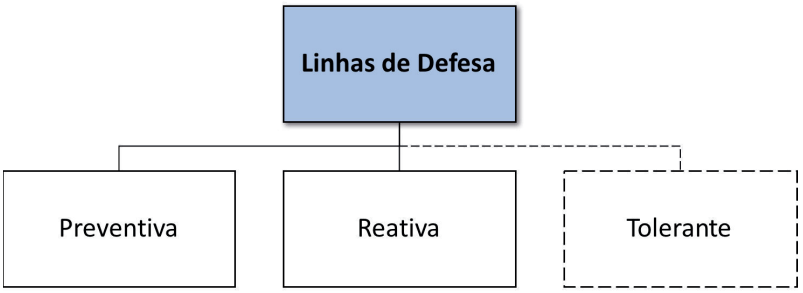


Figura 1.5. Linhas de Defesa: Composição da Resiliência [Nogueira 2009]

As características da rede e do ciberespaço, e as limitações nas linhas de defesa, reforçam o fato de que nenhum sistema é totalmente imune a ataques e intrusões. Portanto, novas abordagens são necessárias para garantir a integridade, confidencialidade, autenticação e, especialmente, a disponibilidade de serviços. Esses aspectos motivam o projeto de serviços resilientes. A resiliência é a capacidade de um sistema oferecer serviços essenciais, mesmo em face de ataques e intrusões [Laprie et al. 2004, Lima et al. 2009, Nogueira 2009]. Neste minicurso focamos na comunicação (a entrega de dados de um dispositivo na UbiComp para outro) em seus diferentes níveis como serviço essencial para o contexto da UbiComp. Desta forma, destacamos três serviços: conectividade física e de camada de enlace, roteamento e comunicação lógica de fim-a-fim. Seguimos a afirmação de que a resiliência é alcançada quando uma solução integra e gerencia as linhas de defesa preventivas, reativas e tolerantes de uma maneira auto-adaptativa e coordenada [Lima et al. 2009, Nogueira 2009]. A seguir listamos os requisitos para atingir resiliência no contexto UbiComp diante do nosso ponto de vista.

1.5.1. Requisitos de Resiliência no Contexto UbiComp

Os requisitos de resiliência podem variar substancialmente de acordo com o escopo do sistema, a criticidade dos serviços oferecidos e as consequências da interrupção do serviço [Ellison et al. 1997, Linger et al. 1998]. As redes no contexto da UbiComp apresentam diversas funções, operações e serviços influenciados direcionados pelos requisitos das aplicações. Em uma situação crítica em que partes do sistema são comprometidas por ataques ou intrusões, a prioridade é dada para manter a conectividade de rede em três níveis principais: camada de física, roteamento e comunicação fim-a-fim. Portanto, para alcançar a resiliência, alguns requisitos devem ser atingidos na engenharia de soluções resilientes. Os requisitos apontam para funções ou recursos necessários para melhorar a capacidade da rede e do sistema de fornecer ou recuperar os serviços essenciais mesmo

diante de ataques ou intrusões.

Neste capítulo, categorizamos os requisitos de resiliência em dois grupos: os requisitos relacionados aos *serviços essenciais* e aqueles requisitos relacionados às *características da rede/sistema*. Os requisitos no primeiro grupo são resumidos como:

- **heterogeneidade** - uma abordagem de resiliência deve considerar a heterogeneidade dos dispositivos, da tecnologia de comunicação e da capacidade de recursos dos dispositivos.
- **autoconfiguração** - a rede e o sistema devem ser capazes de alterar dinamicamente os valores dos parâmetros das conexões, dos dispositivos e dos protocolos, bem como dos mecanismos de segurança, como regras de *firewall* e os limites dos sistemas de reputação, por exemplo.
- **autoadaptação** - capacidade da rede ou sistema de se ajustar em resposta às condições, como mobilidade, e aos requisitos das atividades, como níveis exigidos de Qualidade de Experiência (QoE) pelos usuários.
- **eficiência** - a abordagem de resiliência deve suportar o uso eficiente de recursos dos dispositivos e da rede, como uso energético do dispositivo e largura de banda da rede quando uma falha maliciosa é suspeita ou acontecendo.
- **controle de acesso** - os mecanismos devem controlar o acesso dos dispositivos na rede, bem como monitorar suas atividades.
- **proteção** - os mecanismos de segurança precisam ser gerenciados e combinados para proteger a comunicação em todas as camadas da pilha de protocolos.
- **integridade**, **confidencialidade** e **não-repúdio** – princípios de segurança que precisam ser assegurados para a comunicação.
- **redundância** - a rede deve tolerar e mitigar os ataques por meio de técnicas de tolerância a intrusões, como protocolos duplos ou operações de criptografia, uso simultâneo de rotas múltiplas e outros.
- **robustez** - a rede e o sistema devem continuar seus serviços mesmo durante uma eventual desconexão.

O segundo grupo inclui requisitos de resiliência associados às características gerais das redes ou sistema que compõem o contexto da UbiComp, tais como:

- **descentralização** - a resiliência na UbiComp deve ser fornecida por uma abordagem descentralizada para evitar um ponto de falha, sensível a ataques.
- **auto-organização** - os mecanismos para apoiar a resiliência devem ser auto-organizados sem exigir intervenção humana diante das necessárias mudanças para se adaptar às condições da rede e do sistema.

- **escalabilidade** - os mecanismos para fornecer resiliência devem considerar a variabilidade e o crescimento no número total de dispositivos.
- **autodiagnóstico** - a rede e o sistema devem monitorar-se e detectar falhas, indisponibilidades, mau-comportamento ou dispositivos maliciosos.
- **autorrecuperação** - as abordagens resilientes devem evitar disrupções de conectividade e recuperar a rede de problemas que possam ter acontecido. Eles também devem encontrar uma maneira alternativa de usar recursos e reconfigurar dispositivos, redes ou protocolos para mantê-los em operação normal.
- **auto-otimização** - os mecanismos devem otimizar o uso de recursos da rede e do sistema, minimizando a latência e mantendo a qualidade do serviço.

1.5.2. Questões em Aberto

Quais são os desafios abertos para alcançar a resiliência no contexto UbiComp? Em primeiro lugar, enfatizamos a heterogeneidade dos dispositivos e tecnologias que compõem os ambientes UbiComp. A integração de sistemas de grande escala, como os centros de dados Cloud, para dispositivos minúsculos, como sensores portáteis e implantáveis, é um enorme desafio, devido à complexidade resultante. Então, além disso, a integração das três linhas de defesa e sua adaptação é ainda mais difícil diante dos diferentes requisitos desses dispositivos, suas capacidades em termos de memória e processamento, e os requisitos das aplicações. Ainda, lidar com a heterogeneidade em termos de tecnologia e protocolos de comunicação torna mais difícil analisar o comportamento e a topologia da rede, o que poderia ser usado para auxiliar no projeto de soluções resilientes.

Outro desafio é como lidar com a escala. Primeiro, os sistemas UbiComp tendem a ser de grande escala e geograficamente distribuídos. Como lidar com a complexidade resultante disso? Como definir e construir modelos para entender esses sistemas e oferecer serviços resilientes? Finalmente, destacamos como desafios a incerteza e a velocidade. Se, por um lado, é tão difícil modelar, analisar e definir serviços resilientes neste sistema complexo, por outro lado incerteza é uma norma sobre eles, sendo a velocidade e o baixo tempo de resposta um forte requisito para as aplicações nesses sistemas. Por isso, como abordar esses elementos juntos? Como gerenciá-los para oferecer serviços resilientes considerando diversos tipos de requisitos das várias aplicações?

Todas essas questões levam a profundas investigações e desafios. No entanto, eles também mostram oportunidades para pesquisa aplicada na concepção e engenharia de sistemas resilientes, principalmente para o contexto UbiComp. Particularmente, se nos concentramos na concepção de sistemas resilientes que possam coordenar de forma adaptativa as três linhas de defesa podem ser um grande avanço para a pesquisa aplicada e para a resiliência dos sistemas e das redes.

1.6. Gestão de Identidade

Uma forma de atender aos requisitos de segurança de sistema UbiComp é por meio de uma infraestrutura de autenticação e de autorização (*Authentication and Authorization Infrastructure* – AAI). AAI é conhecida como o elemento central para prover a segurança

em aplicações distribuídas. Com esta infraestrutura, é possível implantar a gestão de identidade – GId (do inglês, *Identity Management* - IdM) de forma a impedir que usuários ou dispositivos (ilegítimos ou não) tenham acesso aos recursos que estes não estão autorizados [Lopez et al. 2004]. A gestão de identidade pode ser entendida como o conjunto de processos e tecnologias usados para garantir a identidade de uma entidade (usuário, dispositivo ou sistema), garantir as informações de uma identidade (identificadores, credenciais e atributos) e para prover procedimentos de autenticação, autorização e auditoria [ITU 2009]. Uma abordagem apropriada de gestão de identidade é necessária para que UbiComp seja invisível para usuários [Arias-Cabarcos et al. 2015].

De acordo com [ITU 2009], uma identidade eletrônica pode ser definida como um conjunto de dados sobre uma entidade que é suficiente para identificar esta entidade em um contexto digital particular. Uma identidade é constituída de:

- Identificador – conjunto de dígitos, caracteres e símbolos ou qualquer outra forma de dados usados para identificar unicamente a identidade de uma entidade (p.ex., UserID, e-mail, URI e endereço IP). IoT requer um identificador único e global para cada entidade da rede;
- Credenciais – um objeto que pode ser usado para provar uma identidade. Exemplo de credenciais incluem certificados digitais X.509 assinados por uma autoridade certificadora, senhas, *tokens* entre outras;
- Atributos – conjunto de dados descritivos ligado a uma entidade que especifica suas características. Por exemplo, nome completo, endereço domiciliar e data de nascimento.

1.6.1. Sistema de Gestão de Identidade

Um sistema de GId lida com todo o ciclo de vida da identidade, que consiste de seu registro, armazenamento, recuperação, provisionamento e revogação de atributos da identidade [Bhargav-Spantzel et al. 2007]. Vale destacar que gerenciar o ciclo de vida da identidade de dispositivos é mais complexo e custoso do que gerenciar a identidade de pessoas [Garcia-Morchon et al. 2017]. Um sistema de GId é caraterizado pelos seguintes elementos [Bhargav-Spantzel et al. 2007]:

- Entidade: usuário, dispositivo ou serviço que deseja acessar um recurso;
- Provedor de identidade (*Identity Provider* – IdP) - responsável por gerenciar identidades de entidades e pelo processo de autenticação;
- Provedor de Serviço (*Service Provider* – SP) - oferece recursos as entidades autorizadas.

A disposição destes elementos em um sistema de GId e a forma com que estes interagem entre si caraterizam os modelos de gestão de identidades, sendo estes classificados como [Bhargav-Spantzel et al. 2007]: tradicional (isolado ou silo), centralizado, federado e centrado no usuário.

Os sistemas UbiComp são compostos de dispositivos heterogêneos que precisam provar a sua autenticidade para as entidades com as quais se comunicam. Um problema neste cenário é garantir a autenticidade de dispositivos e de usuários que se comunicam entre si e que podem estar localizados em domínios administrativos de segurança diferentes, bem como podem utilizar mecanismos de autenticação e autorização distintos [Wangham et al. 2013].

Uma das maneiras de prover a GId em um cenário com múltiplos domínios de segurança é por meio de uma AAI baseada no modelo de gestão de identidades federadas. Em uma federação relações de confiança são estabelecidas entre IdPs e SPs para possibilitar a troca de informações relacionadas a identidade e o compartilhamento de serviços. Conforme ilustrado na Figura 1.6, neste modelo, os acordos de confiança existentes garantem que usuários/dispositivos autenticados em seus IdPs de origem podem acessar recursos protegidos providos por SPs de outros domínios da federação. A autenticação única (*Single Sign-On – SSO*) é obtida quando o mesmo evento de autenticação pode ser usado para acessar diferentes serviços federados [Bhargav-Spantzel et al. 2007].

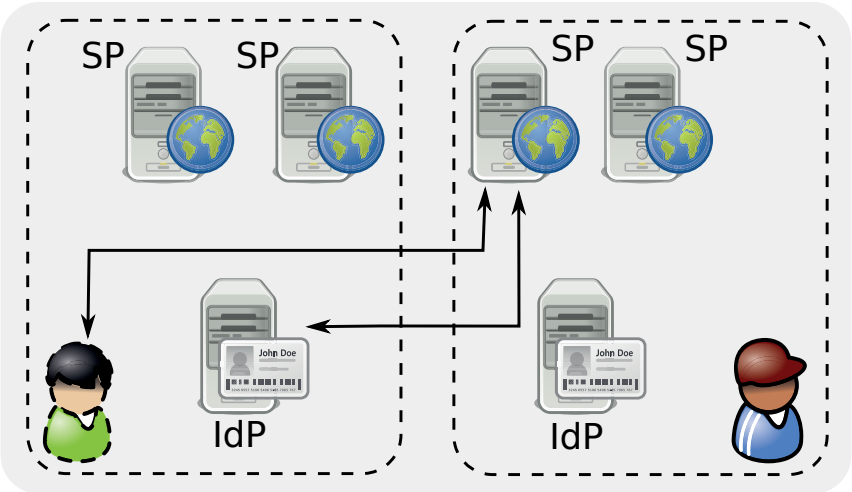


Figura 1.6. Modelo de Identidades Federadas [Wangham et al. 2010]

No modelo federado, a tradução de credenciais de autenticação para sistemas de controle de acesso físico (do inglês, *Physical Access Control Systems* - PACS), ou seja, acesso físico e acesso lógico federado unificado, também é um desafio.

1.6.1.1. Autenticação

A autenticação de dispositivos e de usuários em uma mesma infraestrutura é considerada em [Nguyen et al. 2010, Hanumanthappa and Singh 2012] usando o modelo de GId centralizado e em [Gusmeroli et al. 2013] usando o modelo de GId tradicional. Em [Akram and Hoffmann 2008, Liu et al. 2012, Ndibanje et al. 2014], as IAAs propostas para IoT, adequadas aos sistemas UbiComp, seguem o modelo federado, entretanto, tratam apenas da autenticação SSO de usuários, não da autenticação de dispositivos. Assim, a gestão de identidades federadas de dispositivos ainda é um desafio de pesquisa.

[Kim et al. 2015] propuseram uma solução centralizada que usa diferentes mecanismos de autenticação de dispositivos que são selecionados tendo como base a autonomia de energia do dispositivo. Entretanto, esta solução não provê autenticação de usuários. Baseado no modelo tradicional, uma AAI composta por um conjunto de protocolos que provê autenticação e controle de acesso durante todo o ciclo de vida de um dispositivo da IoT é proposto em [Neto et al. 2016]. [Domenech et al. 2016] propõem uma AAI para Web das Coisas, baseada no modelo de identidades federadas, que provê na mesma infraestrutura a autenticação única de usuários e de dispositivos, por meio de diferentes mecanismos de autenticação. Nesta AAI, um IdP pode ser implementado como um serviço na nuvem (IdPaaS) ou *on premise*

Os mecanismos e protocolos de autenticação consomem recursos computacionais, portanto, integrar uma AAI em um dispositivo com recursos limitados pode ser um desafio. Conforme mencionado na seção 1.4.3, um conjunto de criptografias leves, que não impõe sobrecargas relacionadas ao uso de certificados em dispositivos, pode ser usado para fornecer autenticação de dispositivo em sistemas UbiComp. Existe uma tendência recente que investiga os benefícios do uso da criptografia baseada em identidade (*Identity-Based Cryptography* –IBC) para fornecer autenticação entre domínios para dispositivos restritos [Markmann et al. 2015, Neto et al. 2016, Domenech et al. 2016].

A autenticação multi-fator (*Multi-Factor Authentication* – MFA) é uma solução criada para melhorar a robustez do processo de autenticação e, geralmente, combina dois ou mais fatores de autenticação ("algo que você sabe", "algo que você possui"e "algo que você é") [NIST 2017]. Em [Brainard et al. 2006], é apresentado outro fator: “alguém que você conhece” – relações humanas para intermediar a autenticação de um usuário. Neste tipo de autenticação, um invasor precisa comprometer dois ou mais fatores, o que torna a tarefa mais complexa. Muitos IdPs e SPs já oferecem multi-fator para autenticar seus usuários, porém, para para autenticar dispositivos ainda não há soluções amplamente aceitas.

1.6.1.2. Autorização

Em sistemas UbiComp, consideramos que um domínio de segurança pode ter dispositivos cliente e dispositivos SP, que são dispositivos com um provedor de serviço incorporado. Neste contexto, dispositivos físicos e SPs on-line podem oferecer serviços. Neste sistemas dinâmicos, os dispositivos entram e saem da rede, os SPs aparecem e desaparecem, logo o controle de acesso deve se adaptar para manter a percepção do usuário de estar sendo automaticamente e continuamente autenticado [Arias-Cabarcos et al. 2015]. O controle de acesso fornecido pela AAI embarcada no dispositivo também é um requisito significativo. Como esses dispositivos funcionam no mundo real, uma ameaça de segurança contra esses dispositivos pode afetar o mundo físico. Assim, se um dispositivo for acessado de forma incorreta, há uma chance de que essa violação afete o mundo físico arriscando o bem-estar das pessoas e até mesmo a vida [Domenech et al. 2016].

No contexto da IoT, os mecanismos de autorização são baseados em modelos de controle de acesso usados na Internet clássica, como o discricionário (Lista de Controle de Acesso - ACL [Guinard et al. 2010]), Controle de Acesso Baseado em Capacidades (*Ca-*

pability)- CapBAC [Rotondi et al. 2011, Mahalle et al. 2013], Controle de Acesso Baseado em Papel - RBAC [De Souza et al. 2008, Liu et al. 2012, Jindou et al. 2012] e Controle de Acesso Baseado em Atributos - ABAC [Han and Li 2012, Zhang and Liu 2011, Neto et al. 2016]. ABAC e RBAC são os modelos mais alinhados aos sistemas de identidades federadas [Wangham et al. 2013]. Conforme proposto em [Domenech et al. 2016], um sistema de GId que suporte diferentes modelos de controle de acesso, como RBAC e ABAC, pode se adaptar mais facilmente às necessidades dos processos de administração no contexto da UbiComp.

Em relação ao modelo de gestão de políticas de acesso aos dispositivos, as soluções de autorização para sistemas UbiComp podem implementar duas abordagens: *provisioning* [Gardel et al. 2013, Domenech et al. 2016] ou *outsourcing* [Alam et al. 2011, Seitz et al. 2013, Fremantle et al. 2014, Domenech et al. 2016]. Na abordagem *provisioning*, a tomada de decisão de autorização, ocorre no próprio dispositivo, o que exige que a política esteja em um repositório local. Nesta abordagem, o *Policy Enforcement Point* (PEP), que controla os acessos ao dispositivo, e o *Policy Decison Point* estão juntos em cada dispositivo do domínio. Na abordagem *outsourcing*, a tomada de decisão de acesso ocorre fora do ambiente computacional do dispositivo (SP), em um serviço externo centralizado, que responde a pedidos de avaliação de políticas de todos os guardiões de recursos (PEPs) dos dispositivos (SPs) de um domínio. Neste caso, a tomada de decisão pode ser oferecida como um serviço (*PDPaaS - Policy Decison Point as a Service*) na nuvem ou *on premise* conforme proposto em [Domenech et al. 2016].

Para dispositivos com restrições computacionais, a abordagem de *provisioning*, apesar de ser mais robusta, por não depender de um serviço externo, pode ser muito custosa tanto para processar a tomada de decisão quanto para administração das políticas de autorização. Já a abordagem de *outsourcing*, que simplifica a gestão das políticas de autorização em um domínio, tem como desvantagem o sobrecusto de comunicação e a criação de um ponto de único de falhas (PDP centralizado).

1.6.2. Protocolos de Gestão de Identidade Federada

Os modelos de GId guiam a construção de políticas e processos de negócios para sistemas GId, mas não indicam quais protocolos ou tecnologias devem ser adotados. SAML 2.0 (Security Assertion Markup Language) [Committee et al. 2012], OpenId Connect [Foundation 2014] e OAuth 2 [Hardt 2012a] se destacam no contexto de identidades federadas [Maler and Reed 2008].

SAML, desenvolvido pela OASIS, é uma estrutura baseada em XML para descrever e trocar informações de segurança entre parceiros de negócios. Esta define a sintaxe e as regras para solicitar, criar, comunicar e usar asserções SAML, que possibilita a autenticação SSO por diversos domínios administrativos. Além disso, SAML pode descrever eventos de autenticação que usam diferentes mecanismos de autenticação [OASIS 2005]. Essas características são muito importantes para que a interoperabilidade entre domínios administrativos diferentes seja alcançado. Segundo [Paci et al. 2009], o primeiro passo para alcançar a interoperabilidade é a adoção do SAML.

Um dos perfis de utilização do SAML é o ECP (*Enhanced Client and Proxy*). O ECP define a troca de informações de segurança envolvendo clientes ativos que não usam

um navegador web e permite a autenticação única de dispositivos. Contudo, o ECP exige o uso do protocolo SOAP nas trocas de mensagens, o que muitas pode não ser adequado devido ao seu alto custo computacional [Zeng et al. 2011].

O OpenID Connect é um *framewok* aberto que adota o modelo GId federado e centrado no usuário. Este é descentralizado, o que significa que nenhuma autoridade central aprova ou registra SPs. Com o OpenID, um usuário pode escolher o Provedor OpenID (IdP) que ele quer usar. O OpenID Connect é uma camada de identidade simples em cima do protocolo OAuth 2.0, que permite que os Clientes (SPs) verifiquem a identidade do usuário ou do dispositivo com base na autenticação realizada em um Servidor de Autorização OAuth (Provedor OpenID), bem como obtenha informações básicas do perfil do usuário ou dispositivo, de uma maneira interoperável e REST [Foundation 2014].

O protocolo OAuth [Hardt 2012b] é um *framework* de autenticação e autorização aberto que permite que um usuário/aplicação compartilhe recursos na web (delegue acesso a um recurso) com terceiros sem ter que compartilhar sua credencial de autenticação. Com o protocolo OAuth 2.0 é possível autorizar o acesso a esses recursos por um tempo determinado. Em [Fremantle et al. 2014], os autores exploram o uso de OAuth para sistemas IoT que usam o protocolo MQTT, um protocolo de fila de mensagem leve (*publish/subscribe*) para pequenos sensores e dispositivos móveis.

Um padrão conhecido para prover autorização em sistemas distribuídos é o XACML (*eXtensible Access Control Markup Language*). O XACML é uma linguagem baseada em XML para descrição de políticas de autorização e para requisição/resposta de decisões de controle de acesso. Decisões de autorização podem ser baseadas em atributos do usuário/dispositivo, das ações solicitadas e das características do ambiente. Estas funcionalidades possibilitam a criação de mecanismos de autorização flexíveis. Além disso, o XACML é genérico, independentemente do modelo de controle de acesso usado (RBAC, ABAC) e permite o uso do modelo de tomada de decisão local (*provisioning*) ou de um provedor de serviço externo (modelo *outsourcing*). Outro aspecto importante é que existem perfis e extensões que fornecem interoperabilidade entre XACML e SAML [OASIS 2013].

1.6.3. Outros Desafios da Gestão de Identidade Pervasiva

As tecnologias de federação atuais dependem de acordos estáticos pré-configurados entre suas entidades (IdPs, SPs), que não são adequados para ambientes abertos como os ambientes de computação ubíqua. Ambientes de UbiComp são dinâmicos, multi-domínios e com múltiplos SPs. Estas pre-configurações afetam negativamente a escalabilidade e a flexibilidade das soluções. O estabelecimento dinâmico de confiança é a chave para a escalabilidade. Embora os protocolos de identidade federada citados possam cobrir aspectos de segurança, os desafios de usabilidade e confiança (*trust*) são questões em aberto [Arias-Cabarcos et al. 2015].

A interoperabilidade é outro requisito fundamental para gestão de identidade pervasiva. Os sistemas UbiComp podem ser formados por domínios heterogêneos (organizações) que vão além das barreiras de uma Federação (com o mesmo protocolo de GId). A interoperabilidade entre federações baseadas no protocolo SAML foi tratada em alguns trabalhos [Silva et al. 2015, Souza and Wangham 2015], porém, alguns requisitos

para computação ubíqua (usabilidade, estabelecimento dinâmico de relações de confiança, autenticação de dispositivos e uso de criptografia leve) não foram considerados nestes trabalhos. A interoperabilidade entre diferentes protocolos de identidades federadas (SAML, OpenId Connect e OAuth) ainda é um problema em aberto e uma oportunidade de pesquisa.

Por fim, sistemas de GId federadas para sistemas Ubicomp devem proteger adequadamente as informações do usuário e devem aderir adequadamente às política de privacidade definidas para os dados do mesmo. A seção a seguir trata especificamente dos desafios para prover privacidade em sistemas UbiComp.

1.7. Implicações de privacidade

Sistemas de UbiComp tendem a coletar uma grande quantidade de dados e gerar muita informação. Usada de forma correta, a informação gera inúmeros benefícios para nossa sociedade que vem nos propiciando uma vida melhor ao longo dos anos. No entanto, a informação pode ser usada para fins ilícitos, assim como sistemas computacionais são usados para ataques. Proteger informações privadas é um grande desafio que muitas vezes pode parecer impraticável, por exemplo, proteger da companhia de energia elétrica local os dados de consumo elétrico gerados pelos respectivos clientes [Borges et al. 2014, Borges de Oliveira 2017a, Borges de Oliveira 2017d]

O texto desta seção apresenta uma visão ampla das questões de privacidade. Para uma introdução mais detalhada, recomendamos a leitura do trabalho *Introdução à Privacidade:Uma Abordagem Computacional* [Borges 2016a] apresentado nesta mesma série.

1.7.1. Desafios do cenário da aplicação

Quando remetentes e receptores são considerados no contexto de um cenário de aplicação particular, há dois grandes desafios. O primeiro é a identificação de dados sensíveis quem podem diferir de cultura para cultura. O segundo é como lidar com várias leis e regulamentações potencialmente conflitantes.

1.7.1.1. Identificando dados sensíveis

Decidir quais dados podem ser sensíveis pode ser uma tarefa desafiadora. O artigo 12 da Declaração Universal dos Direitos Humanos proclamada pela Assembleia Geral das Nações Unidas em Paris, em 10 de dezembro de 1948, afirma: *"Ninguém deve ser submetido a uma interferência arbitrária em sua privacidade, família, lar ou correspondência, nem ataques a sua honra e reputação. Todos têm o direito à proteção da lei contra tais interferências ou ataques."* Em geral, é necessária mais atenção aos dados que permitem o monitoramento de produtos, animais e pessoas. Esses dados podem ser usados para criar um perfil comportamental que permite predição e manipulação. Os protocolos práticos de preservação da privacidade permitem ao receptor obter informações consolidadas ou coletar dados agregados por adição ou concatenação, anonimamente. Por exemplo, os dados de localização apresentam vários problemas de privacidade e, portanto, poucas pessoas compartilham sua localização [Gu et al. 2016], embora os smartphones possam divulgar dados mais sensíveis do que a localização [Ge et al. 2016]. Os sistemas

de UbiComp na área de saúde processam dados sensíveis [Liu et al. 2014], e as câmeras de segurança podem transmitir dados confidenciais [Tekeoglu and Tosun 2015]. Os desafios de privacidade não têm a ver apenas com os cidadãos, mas também as indústrias [Sadeghi et al. 2015]. Mesmo que uma mensagem seja criptografada, os metadados - ou seja, o tráfego de rede - podem revelar informações confidenciais [Hu et al. 2016].

1.7.1.2. Regulamentação

Os regulamentos sobre privacidade estão a crescer em todo o mundo e as leis variam de país para país. Isso é um desafio para as instituições internacionais que desenvolvem sistemas de UbiComp comercializá-los em todo o mundo de acordo com a lei. Outro problema pode ser a ausência de lei, permitindo que os concorrentes desenvolvam estratégias baseadas em informações privadas que aumentam a vantagem no mercado em relação a corporações mais éticas. Em geral, violações da privacidade podem mudar uma eleição e podem levar a uma sociedade de vigilância [Holvast 2009]. Independentemente dos cenários de aplicação, os protocolos de preservação da privacidade enfrentam seus próprios desafios tecnológicos.

1.7.2. Desafios tecnológicos

Em casos de termos uma base de dados pronta para ser anonimizada, ou seja, os dados sensíveis já foram identificados, então podemos usar alguns softwares para localizar os dados e anonimizar a base [Prasser and Kohlmayer 2015, Dai et al. 2009, Poulis et al. 2015]. As técnicas de k-anonimato, l-diversidade, e t-proximidade [Li et al. 2007] estão nas bases de muitas ferramentas.

Os desafios são maiores quando os dados sensíveis são continuamente coletados e transmitidos em uma rede, em vez de ficarem estáticos em um banco de dados. Em séries temporais, poderíamos medir a privacidade de acordo com as possibilidades de encontramos os números em parte da série [Borges de Oliveira 2017c].

1.7.2.1. Computação de todos os operadores

Em teoria, qualquer operador pode ser computado sobre dados criptografados [Gentry 2009], tais técnicas são conhecidas como criptografia totalmente homomórfica. Na prática, é um desafio construir um sistema de criptografia totalmente homomórfica para muitos cenários de aplicativos. Os pesquisadores geralmente desenvolvem protocolos de preservação da privacidade baseados em criptografia homomórfica aditiva [Borges de Oliveira 2017f] e DC-Nets (de "Dining Cryptographers") [Borges de Oliveira 2017e]. Ambas as técnicas computam o operador de adição sobre dados criptografados. No entanto, as primeiras são funções, e estas são famílias de funções. Dada uma criptografia homomórfica aditiva, podemos construir uma DC-Net assimétrica [Borges de Oliveira 2017e].

1.7.2.2. Troca entre aplicação e maleabilidade

DC-Nets pode impor privacidade, garantindo que ninguém pode decifrar mensagens, a menos que todas as mensagens cifradas tenham sido levadas em consideração. A criptografia homomórfica não impõe a privacidade, ou seja, as mensagens individuais podem ser decifradas. Isso acontece porque as DC-Nets não são maleáveis e a criptografia homomórfica é maleável. Por um lado, os esquemas de DC-Net permitem que os invasores alterem mensagens criptografadas e usem outras mensagens criptografadas para deduzir o valor associado a outras pessoas no processo de agregação. Por outro lado, é mais fácil gerar e distribuir chaves para sistemas de criptografia homomórfica do que para esquemas DC-Net.

1.7.2.3. Distribuição de chave

Usando a criptografia homomórfica, o receptor precisa apenas gerar um par de chaves público-privado e transmitir a chave pública para os remetentes de forma autenticada, o que compartilhará a mesma chave para criptografar suas mensagens. Embora as mensagens possam ter o mesmo conteúdo, as mensagens criptografadas serão diferentes umas das outras porque os esquemas de criptografia homomórfica são probabilísticos. Usando esquemas DC-Net, um protocolo não precisa de um receptor porque os remetentes podem ser os receptores. Por esse motivo, vamos apenas chamá-los de participantes, que geram suas chaves privadas. Para redes DC simétricas, cada participante compartilha duas chaves entre si, ou seja, envia um segredo para e recebe outro segredo entre si. Depois, cada participante tem uma chave privada dada pela lista de segredos. Por isso, cada participante criptografa uma mensagem $m_{i,j}$ usando a função

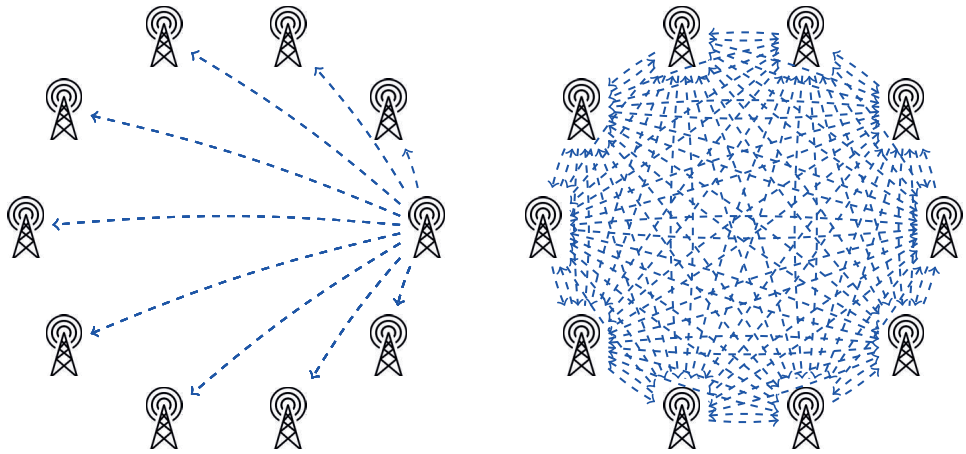
$$\mathfrak{M}_{i,j} \leftarrow \mathrm{Enc}(m_{i,j}) = m_{i,j} + \sum_{o \in \mathscr{M} - \{i\}} \mathrm{Hash}(s_{i,o} \,||\, j) - \mathrm{Hash}(s_{o,i} \,||\, j),$$

onde $m_{i,j}$ é a mensagem enviada pelo participante i no tempo j, Hash é uma função de hash segura predefinida pelos participantes, $s_{i,o}$ é o segredo enviado do participante i para o participante o, da mesma forma, $s_{o,i}$ é o segredo recebido por i de o e $||$ é o operador de concatenação. Cada participante i pode enviar a mensagem criptografada $\mathfrak{M}_{i,j}$ uns para os outros. Assim, eles podem decifrar as mensagens criptografadas agregadas que computam

$$\mathrm{Dec} = \sum_{i \in \mathscr{M}} \mathfrak{M}_{i,j} = \sum_{i \in \mathscr{M}} m_{i,j}.$$

Ninguém pode decifrar se faltar uma ou mais mensagens. Para DC-Nets assimétricas, cada participante gera uma chave privada usada para cifrar e, em seguida, eles adicionam as chaves usando uma criptografia homomórfica ou DC-net simétrica para gerar a chave de decifrar, que pode ser tornada pública.

Observe que os esquemas de criptografia homomórfica tendem a ter baixa sobrecarga para configurar chaves e para distribuí-las. O receptor das mensagens criptografadas agregadas apenas gera um par de chaves pública-privada e envia a chave pública para cada remetente. DC-Nets simétricas precisam de $O(I^2)$ mensagens para configurar as chaves, onde I é o número de participantes. Figura 1.7 representa as mensagens para configurar chaves usando (a) criptografia homomórfica e (b) DC-nets simétricas.



(a) Criptografia homomórfica (b) DC-Nets simétricas

Figura 1.7. Configurando as chaves

1.8. Conclusão

Nas palavras de Mark Weiser, a Ubiquitous Computing é "a ideia de integrar computadores perfeitamente no mundo em geral" [Weiser 1991]. Assim, longe de ser um fenômeno dessa época, o projeto e a prática dos sistemas de UbiComp foram discutidos um quarto de século atrás. Neste capítulo, revisamos a noção, que permeia os mais variados níveis de nossa sociedade, sob um ponto de vista de segurança e privacidade. Nos próximos anos, esses dois tópicos ocuparão muito do tempo de pesquisadores e engenheiros. Em nossa opinião, o uso deste tempo deve ser guiado por algumas observações.

Conforme discutido na Seção 1.2, o software voltado para o contexto da de UbiComp é muitas vezes produzido como a combinação de diferentes linguagens de programação, compartilhando um núcleo comum implementado em um tipo inseguro Linguagem como C, C++ ou montagem.

A Seção 1.3 apresenta uma discussão sobre segurança a longo prazo. As infraestruturas críticas como redes elétricas são projetadas para durar muitas décadas e não podem ser atualizadas com frequência. Não sabemos se problemas matemáticos usados em técnicas criptográficas modernas podem ser resolvidos em um tempo que comprometa a técnica, ou se serão resolvidos algum dia. A vida útil de tais sistemas, juntamente com a difícil atualização e reimplantação, os torna vulneráveis ao inexorável progresso da tecnologia, que traz novos jogadores, como a criptografia pós-quântica.

Na Seção 1.4, é apresentado que o acesso físico e os recursos restritos compliquem o projeto de algoritmos criptográficos eficientes e seguros, que são frequentemente acessíveis aos ataques de canais laterais.

A Seção 1.5 discute sobre resiliência cibernética, ou seja, a capacidade que um sistema de UbiComp tem de recuperar-se e reduzir danos de eventuais falhas dos diversos equipamentos tecnológicos e oriundas de ataques cibernéticos.

Conforme discutido na Seção 1.6, para que a computação ubíqua seja de fato invisível para usuários, uma abordagem também pervasiva de gestão de identidade federada deve ser provida. Um dos desafios da gestão de identidade (GId) em sistemas de UbiComp

é garantir a autenticidade de dispositivos e de usuários em cenários com múltiplos e heterogêneos domínios administrativos de segurança. Diante da necessidade de controlar o acesso a recursos e serviços nos dispositivos de computação ubíqua, as infraestruturas de autorização devem ser flexíveis para que diferentes modelos de controles de acesso e abordagens de tomadas de decisão possam ser implantados de acordo com os requisitos de cada aplicação. Logo, o desafio é projetar ou aprimorar sistemas GId considerando os cenários de aplicação. Requisitos, muitas vezes conflitantes, precisam ser avaliados em cada cenário, tais como: flexibilidade, usabilidade, escalabilidade, interoperabilidade, segurança forte, mas leve e estabelecimento dinâmico de relações de confiança.

Na Seção 1.7, vemos que a disponibilidade e o volume de dados frequentemente manipulados pelos sistemas de UbiComp dificulta inda mais a proteção da privacidade dos usuários. Sistemas de UbiComp que levam em conta os problemas de privacidade têm todos os desafios encontrados em segurança, porém, com mais os desafios provenientes da proteção à privacidade.

Dadas essas observações e a importância da computação ubíqua, é fácil concluir que o futuro tem desafios fascinantes à espera da atenção da academia e da indústria.

Referências

[ddo] 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: 2017-02-14.

[Abowd and Mynatt 2000] Abowd, G. D. and Mynatt, E. D. (2000). Charting past, present, and future research in ubiquitous computing. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 7(1):29–58.

[Akram and Hoffmann 2008] Akram, H. and Hoffmann, M. (2008). Supports for identity management in ambient environments-the hydra approach. In *Proceedings...*, pages 371–377. 3rd International Conference on Systems and Networks Communications, 2008. ICSNC'08.

[Al-Riyami and Paterson 2003] Al-Riyami, S. S. and Paterson, K. G. (2003). Certificateless public key cryptography. In *ASIACRYPT*, volume 2894 of *LNCS*, pages 452–473. Springer.

[Alam et al. 2011] Alam, S., Chowdhury, M. M., and Noll, J. (2011). Interoperability of security-enabled internet of things. *Wireless Personal Communications*, 61(3):567–586.

[Albrecht et al. 2014] Albrecht, M. R., Driessen, B., Kavun, E. B., Leander, G., Paar, C., and Yalçin, T. (2014). Block ciphers - focus on the linear layer (feat. PRIDE). In *CRYPTO (1)*, volume 8616 of *LNCS*, pages 57–76. Springer.

[Almeida et al. 2016] Almeida, J. B., Barbosa, M., Barthe, G., Dupressoir, F., and Emmi, M. (2016). Verifying constant-time implementations. In *USENIX Security Symposium*, pages 53–70. USENIX Association.

[Aranha et al. 2014] Aranha, D. F., Karam, M. M., Miranda, A., and Scarel, F. (2014). *Software vulnerabilities in the Brazilian voting machine*, pages 149–175. IGI Global.

[Arias-Cabarcos et al. 2015] Arias-Cabarcos, P., Almenárez, F., Trapero, R., Díaz-Sánchez, D., and Marín, A. (2015). Blended identity: Pervasive idm for continuous authentication. *IEEE Security Privacy*, 13(3):32–39.

[Ashton 2009] Ashton, K. (2009). That 'Internet of Things' Thing. *RFiD Journal*, 22:97–114.

[Atzori et al. 2010] Atzori, L., Iera, A., and Morabito, G. (2010). The Internet of Things: A survey. *Computer Networks*, 54(15):2787–2805.

[Aumasson and Bernstein 2012] Aumasson, J. and Bernstein, D. J. (2012). Siphash: A fast short-input PRF. In *INDOCRYPT*, volume 7668 of *LNCS*, pages 489–508. Springer.

[Aumasson et al. 2013] Aumasson, J., Neves, S., Wilcox-O'Hearn, Z., and Winnerlein, C. (2013). BLAKE2: simpler, smaller, fast as MD5. In *ACNS*, volume 7954 of *LNCS*, pages 119–135. Springer.

[Balasch et al. 2014] Balasch, J., Gierlichs, B., Grosso, V., Reparaz, O., and Standaert, F. (2014). On the cost of lazy engineering for masked software implementations. In *CARDIS*, volume 8968 of *LNCS*, pages 64–81. Springer.

[Banik et al. 2015] Banik, S., Bogdanov, A., and Regazzoni, F. (2015). Exploring energy efficiency of lightweight block ciphers. In *SAC*, volume 9566 of *LNCS*, pages 178–194. Springer.

[Barbulescu et al. 2014] Barbulescu, R., Gaudry, P., Joux, A., and Thomé, E. (2014). A heuristic quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic. In *EUROCRYPT 2014*, pages 1–16. Springer.

[Barker et al. 2012] Barker, E., Barker, W., Burr, W., Polk, W., and Smid, M. (2012). Recommendation for key management part 1: General (revision 3). *NIST special publication*, 800(57):1–147.

[Beierle et al. 2016] Beierle, C., Jean, J., Kölbl, S., Leander, G., Moradi, A., Peyrin, T., Sasaki, Y., Sasdrich, P., and Sim, S. M. (2016). The SKINNY family of block ciphers and its low-latency variant MANTIS. In *CRYPTO (2)*, volume 9815 of *LNCS*, pages 123–153. Springer.

[Bennett and Brassard 1984] Bennett, C. H. and Brassard, G. (1984). Quantum Cryptography: Public Key Distribution and Coin Tossing. In *Proceedings of IEEE ICCSSP'84*, pages 175–179, New York. IEEE Press.

[Bernstein 2006] Bernstein, D. J. (2006). Curve25519: New diffie-hellman speed records. In *Public Key Cryptography*, volume 3958 of *LNCS*, pages 207–228. Springer.

[Bernstein et al. 2012a] Bernstein, D. J., Duif, N., Lange, T., Schwabe, P., and Yang, B. (2012a). High-speed high-security signatures. *J. Cryptographic Engineering*, 2(2):77–89.

[Bernstein et al. 2015] Bernstein, D. J., Hopwood, D., Hülsing, A., Lange, T., Niederhagen, R., Papachristodoulou, L., Schneider, M., Schwabe, P., and Wilcox-O'Hearn, Z. (2015). *SPHINCS: Practical Stateless Hash-Based Signatures*, pages 368–397. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Bernstein et al. 2012b] Bernstein, D. J., Lange, T., and Schwabe, P. (2012b). The security impact of a new cryptographic library. In *LATINCRYPT*, volume 7533 of *Lecture Notes in Computer Science*, pages 159–176. Springer.

[Bhargav-Spantzel et al. 2007] Bhargav-Spantzel, A., Camenisch, J., Gross, T., and Sommer, D. (2007). User centricity: a taxonomy and open issues. *Journal of Computer Security*, 15(5):493–527.

[Biham and Shamir 1997] Biham, E. and Shamir, A. (1997). Differential fault analysis of secret key cryptosystems. In *CRYPTO*, volume 1294 of *LNCS*, pages 513–525. Springer.

[Biryukov et al. 2016] Biryukov, A., Dinu, D., and Khovratovich, D. (2016). Argon2: New generation of memory-hard functions for password hashing and other applications. In *EuroS&P*, pages 292–302. IEEE.

[Bogdanov et al. 2007] Bogdanov, A., Knudsen, L. R., Leander, G., Paar, C., Poschmann, A., Robshaw, M. J. B., Seurin, Y., and Vikkelsoe, C. (2007). PRESENT: an ultra-lightweight block cipher. In *CHES*, volume 4727 of *LNCS*, pages 450–466. Springer.

[Boldyreva et al. 2012] Boldyreva, A., Goyal, V., and Kumar, V. (2012). Identity-based encryption with efficient revocation. *IACR Cryptology ePrint Archive*, 2012:52.

[Boneh and Franklin 2003] Boneh, D. and Franklin, M. K. (2003). Identity-based encryption from the weil pairing. *SIAM J. Comput.*, 32(3):586–615.

[Borges 2016a] Borges, F. (2016a). *Introdução à Privacidade: Uma Abordagem Computacional*, pages 1–43. SBC.

[Borges 2016b] Borges, F. (2016b). *Privacy-Preserving Data Aggregation in Smart Metering Systems*. Energy Engineering Series. Institution of Engineering & Technology.

[Borges et al. 2014] Borges, F., Demirel, D., Bock, L., Buchmann, J. A., and Mühlhäuser, M. (2014). A privacy-enhancing protocol that provides in-network data aggregation and verifiable smart meter billing. In *ISCC*, pages 1–6. IEEE.

[Borges de Oliveira 2017a] Borges de Oliveira, F. (2017a). *Background and Models*, pages 13–23. Springer International Publishing, Cham.

[Borges de Oliveira 2017b] Borges de Oliveira, F. (2017b). *Introduction*, pages 3–12. Springer International Publishing, Cham.

[Borges de Oliveira 2017c] Borges de Oliveira, F. (2017c). *Quantifying the Aggregation Size*, pages 49–60. Springer International Publishing, Cham.

[Borges de Oliveira 2017d] Borges de Oliveira, F. (2017d). *Reasons to Measure Frequently and Their Requirements*, pages 39–47. Springer International Publishing, Cham.

[Borges de Oliveira 2017e] Borges de Oliveira, F. (2017e). *Selected Privacy-Preserving Protocols*, pages 61–100. Springer International Publishing, Cham.

[Borges de Oliveira 2017f] Borges de Oliveira, F. (2017f). *A Selective Review*, pages 25–36. Springer International Publishing, Cham.

[Brainard et al. 2006] Brainard, J., Juels, A., Rivest, R. L., Szydlo, M., and Yung, M. (2006). Fourth-factor authentication: somebody you know. In *Proceedings of the 13th ACM conference on Computer and communications security*, pages 168–178. ACM.

[Bronevetsky 2009] Bronevetsky, G. (2009). Communication-sensitive static dataflow for parallel message passing applications. In *CGO*, pages 1–12, Washington, DC, USA. IEEE.

[BTS 2017] BTS, U. B. o. T. S. (2017). Average age of automobiles and trucks in operation in the united states. Accessed: 2017-09-14.

[Buchmann et al. 2011] Buchmann, J., Dahmen, E., and Hülsing, A. (2011). Xmss - a practical forward secure signature scheme based on minimal security assumptions. In Yang, B.-Y., editor, *PQCrypto*, pages 117–129. Springer.

[Cadar et al. 2008] Cadar, C., Dunbar, D., and Engler, D. (2008). KLEE: Unassisted and automatic generation of high-coverage tests for complex systems programs. In *OSDI*, pages 209–224. USENIX.

[Carlini et al. 2015] Carlini, N., Barresi, A., Payer, M., Wagner, D., and Gross, T. R. (2015). Control-flow bending: On the effectiveness of control-flow integrity. In *SEC*, pages 161–176, Berkeley, CA, USA. USENIX.

[Committee et al. 2012] Committee, O. S. S. T. et al. (2012). Security assertion markup language (saml) 2.0.

[Conti 2006] Conti, J. P. (2006). The internet of things. *Communications Engineer*, 4(6):20–25.

[Coppa et al. 2012] Coppa, E., Demetrescu, C., and Finocchi, I. (2012). Input-sensitive profiling. In *PLDI*, pages 89–98, New York, NY, USA. ACM.

[Costello and Longa 2015] Costello, C. and Longa, P. (2015). Four$\mathbb{Q}$: Four-dimensional decompositions on a $\mathbb{Q}$-curve over the mersenne prime. In *ASIACRYPT (1)*, volume 9452 of *LNCS*, pages 214–235. Springer.

[Cousot and Cousot 1977] Cousot, P. and Cousot, R. (1977). Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *POPL*, pages 238–252, New York, NY, USA. ACM.

[Cousot et al. 2011] Cousot, P., Cousot, R., and Logozzo, F. (2011). A parametric segmentation functor for fully automatic and scalable array content analysis. In *POPL*, pages 105–118, New York, NY, USA. ACM.

[Cremonezi et al. 2017] Cremonezi, B. M., Vieira, A. B., Nacif, J. A. M., and Nogueira, M. (2017). A dynamic channel allocation protocol for medical environment under multiple base stations. In *IEEE Wireless Communications and Networking Conference, WCNC*, pages 1–6.

[Daemen and Rijmen 2002] Daemen, J. and Rijmen, V. (2002). *The Design of Rijndael: AES - The Advanced Encryption Standard*. Information Security and Cryptography. Springer.

[Dagenais and Hendren 2008] Dagenais, B. and Hendren, L. (2008). Enabling static analysis for partial java programs. In *OOPSLA*, pages 313–328, New York, NY, USA. ACM.

[Dai et al. 2009] Dai, C., Ghinita, G., Bertino, E., Byun, J.-W., and Li, N. (2009). Tiamat: A tool for interactive analysis of microdata anonymization techniques. *Proc. VLDB Endow.*, 2(2):1618–1621.

[De Souza et al. 2008] De Souza, L. M. S., Spiess, P., Guinard, D., Köhler, M., Karnouskos, S., and Savio, D. (2008). Socrades: A web service based shop floor integration infrastructure. In *The internet of things*, pages 50–67. Springer.

[Delic 2016] Delic, K. A. (2016). On resilience of iot systems: The internet of things (ubiquity symposium). *Ubiquity*, 2016(February):1–7.

[Diffie and Hellman 2006] Diffie, W. and Hellman, M. (2006). New directions in cryptography. *IEEE Trans. Inf. Theor.*, 22(6):644–654.

[Dinu et al. 2015] Dinu, D., Corre, Y. L., Khovratovich, D., Perrin, L., Großschädl, J., and Biryukov, A. (2015). Triathlon of Lightweight Block Ciphers for the Internet of Things. NIST Workshop on Lightweight Cryptography.

[Dinu et al. 2016] Dinu, D., Perrin, L., Udovenko, A., Velichkov, V., Großschädl, J., and Biryukov, A. (2016). Design strategies for ARX with provable bounds: Sparx and LAX. In *ASIACRYPT (1)*, volume 10031 of *LNCS*, pages 484–513.

[Domenech et al. 2016] Domenech, M. C., Boukerche, A., and Wangham, M. S. (2016). 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, pages 39–46, New York, NY, USA. ACM.

[DoT 2013] DoT, U. D. T. (2013). IEEE 1609 - Family of Standards for Wireless Access in Vehicular Environments WAVE.

[Ellison et al. 1997] Ellison, R., Fisher, D., Linger, R., Lipson, H., Longstaff, T., and Mead, N. (1997). Survivable network systems: an emerging discipline – CMU/SEI-97-TR-013. Technical report, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA.

[Estrin et al. 1999] Estrin, D., Govindan, R., Heidemann, J. S., and Kumar, S. (1999). Next century challenges: Scalable coordination in sensor networks. In *MobiCom'99*, pages 263–270.

[Faz-Hernández et al. 2015] Faz-Hernández, A., Cabral, R., Aranha, D. F., and López, J. (2015). Implementação Eficiente e Segura de Algoritmos Criptográficos. In *Minicursos do XV Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (SBSEG)*, pages 93–140. Sociedade Brasileira de Computação.

[Fedrecheski et al. 2016] Fedrecheski, G., Costa, L. C. P., and Zuffo, M. K. (2016). Elixir programming language evaluation for iot. In *ISCE*, page Online, Washington, DC, USA. IEEE.

[Foundation 2014] Foundation, T. O. (2014). Openid connect core 1.0. `http://openid.net/specs/openid-connect-core-1\_0.html`.

[Fremantle et al. 2014] Fremantle, P., Aziz, B., Kopecký, J., and Scott, P. (2014). Federated identity and access management for the internet of things. In *2014 International Workshop on Secure Internet of Things*, pages 10–17.

[Furr and Foster 2008] Furr, M. and Foster, J. S. (2008). Checking type safety of foreign function calls. *ACM Trans. Program. Lang. Syst.*, 30(4):18:1–18:63.

[Garcia-Morchon et al. 2017] Garcia-Morchon, O., Kumar, S., and Sethi, M. (2017). State-of-the-Art and Challenges for the Internet of Things Security. Internet-Draft draft-irtf-t2trg-iot-seccons-04, Internet Engineering Task Force. Work in Progress.

[Gardel et al. 2013] Gardel, T., Andrade, N., Farias, F., and Prazeres, C. (2013). Autenticação e autorização para acesso a aplicações em um barramento de serviços para a web das coisas. In *Anais do*. 13 Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (SBSeg), 2013, SBC.

[Ge et al. 2016] Ge, Y., Deng, B., Sun, Y., Tang, L., Sheng, D., Zhao, Y., Xie, G., and Salamatian, K. (2016). A comprehensive investigation of user privacy leakage to android applications. In *2016 25th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–6.

[Gentry 2009] Gentry, C. (2009). *A Fully Homomorphic Encryption Scheme*. PhD thesis, Stanford University, Stanford, CA, USA.

[Godefroid 2014] Godefroid, P. (2014). Micro execution. In *ICSE*, pages 539–549, New York, NY, USA. ACM.

[Godefroid et al. 2005] Godefroid, P., Klarlund, N., and Sen, K. (2005). Dart: directed automated random testing. In *PLDI*, pages 213–223, New York, NY, USA. ACM.

[Graham et al. 1982] Graham, S. L., Kessler, P. B., and McKusick, M. K. (1982). gprof: a call graph execution profiler (with retrospective). In *Best of PLDI*, pages 49–57, New York, NY, USA. ACM.

[Grosso et al. 2014] Grosso, V., Leurent, G., Standaert, F., and Varici, K. (2014). Ls-designs: Bitslice encryption for efficient masked software implementations. In *FSE*, volume 8540 of *LNCS*, pages 18–37. Springer.

[Großschädl et al. 2009] Großschädl, J., Oswald, E., Page, D., and Tunstall, M. (2009). Side-channel analysis of cryptographic software via early-terminating multiplications. In *ICISC*, volume 5984 of *Lecture Notes in Computer Science*, pages 176–192. Springer.

[Grover 1996] Grover, L. K. (1996). A fast quantum mechanical algorithm for database search. In *Proceedings of ACM STOC 1996*, pages 212–219, New York, NY, USA. ACM.

[Gu et al. 2016] Gu, Y., Yao, Y., Liu, W., and Song, J. (2016). We know where you are: Home location identification in location-based social networks. In *2016 25th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–9.

[Guinard et al. 2010] Guinard, D., Fischer, M., and Trifa, V. (2010). Sharing using social networks in a composable web of things. In *Proceedings...*, pages 702–707. 8th IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010.

[Gusmeroli et al. 2013] Gusmeroli, S., Piccione, S., and Rotondi, D. (2013). A capability-based security approach to manage access control in the internet of things. *Mathematical and Computer Modelling*, 58(5–6):1189 – 1205.

[Han and Li 2012] Han, Q. and Li, J. (2012). An authorization management approach in the internet of things. *Journal of Information & Computational Science*, 9(6):1705–1713.

[Hanumanthappa and Singh 2012] Hanumanthappa, P. and Singh, S. (2012). Privacy preserving and ownership authentication in ubiquitous computing devices using secure three way authentication. In *Proceedings*, pages 107–112. International Conference on Innovations in Information Technology (IIT).

[Hardt 2012a] Hardt, D. (2012a). The oauth 2.0 authorization framework. RFC 6749, RFC Editor. `http://www.rfc-editor.org/rfc/rfc6749.txt`.

[Hardt 2012b] Hardt, E. D. (2012b). The oauth 2.0 authorization framework.

[Holvast 2009] Holvast, J. (2009). History of privacy. In Matyáš, V., Fischer-Hübner, S., Cvrček, D., and Švenda, P., editors, *The Future of Identity in the Information Society*, volume 298 of *IFIP Advances in Information and Communication Technology*, pages 13–42. Springer Berlin Heidelberg.

[Hu et al. 2016] Hu, J., Lin, C., and Li, X. (2016). Relationship privacy leakage in network traffics. In *2016 25th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–9.

[Hu et al. 2003] Hu, Y., Johnson, D., and Perrig, A. (2003). SEAD: secure efficient distance vector routing for mobile wireless ad hoc networks. *Journal Ad Hoc Networks*, 01:175–192.

[Ishai et al. 2003] Ishai, Y., Sahai, A., and Wagner, D. (2003). Private circuits: Securing hardware against probing attacks. In *CRYPTO*, volume 2729 of *LNCS*, pages 463–481. Springer.

[ITU 2009] ITU (2009). Ngn identity management framework. Recommendation Y.2720.

[Jacobsson et al. 2016] Jacobsson, A., Boldt, M., and Carlsson, B. (2016). A risk analysis of a smart home automation system. *Future Generation Computer Systems*, 56(Supplement C):719 – 733.

[Jindou et al. 2012] Jindou, J., Xiaofeng, Q., and Cheng, C. (2012). Access control method for web of things based on role and sns. In *Proceedings...*, pages 316–321. IEEE 12th International Conference on Computer and Information Technology (CIT), 2012.

[Kaufmann et al. 2016] Kaufmann, T., Pelletier, H., Vaudenay, S., and Villegas, K. (2016). When constant-time source yields variable-time binary: Exploiting curve25519-donna built with MSVC 2015. In *CANS*, volume 10052 of *Lecture Notes in Computer Science*, pages 573–582.

[Kim and Barbulescu 2016] Kim, T. and Barbulescu, R. (2016). Extended tower number field sieve: A new complexity for the medium prime case. In *CRYPTO (1)*, volume 9814 of *LNCS*, pages 543–571. Springer.

[Kim et al. 2014] Kim, Y., Daly, R., Kim, J., Fallin, C., Lee, J., Lee, D., Wilkerson, C., Lai, K., and Mutlu, O. (2014). Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors. In *ISCA*, pages 361–372. IEEE Computer Society.

[Kim et al. 2015] Kim, Y.-P., Yoo, S., and Yoo, C. (2015). Daot: Dynamic and energy-aware authentication for smart home appliances in internet of things. In *Consumer Electronics (ICCE), 2015 IEEE International Conference on*, pages 196–197.

[Knill 2010] Knill, E. (2010). Physics: quantum computing. *Nature*, 463(7280):441–443.

[Koblitz 1987] Koblitz, N. (1987). Elliptic curve cryptosystems. *Mathematics of Computation*, 48(177):203–209.

[Koblitz 1988] Koblitz, N. (1988). A family of jacobians suitable for discrete log cryptosystems. In *CRYPTO*, volume 403 of *LNCS*, pages 94–99. Springer.

[Kocher 1996] Kocher, P. C. (1996). Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems. In *CRYPTO*, volume 1109 of *LNCS*, pages 104–113. Springer.

[Kocher et al. 1999] Kocher, P. C., Jaffe, J., and Jun, B. (1999). Differential power analysis. In *CRYPTO*, volume 1666 of *LNCS*, pages 388–397. Springer.

[Kölbl et al. 2016] Kölbl, S., Lauridsen, M. M., Mendel, F., and Rechberger, C. (2016). Haraka v2 - efficient short-input hashing for post-quantum applications. *IACR Trans. Symmetric Cryptol.*, 2016(2):1–29.

[Langley 2010] Langley, A. (2010). ImperialViolet: Checking that functions are constant time with Valgrind. `https://www.imperialviolet.org/2010/04/01/ctgrind.html`.

[Laprie et al. 2004] Laprie, J.-C., Randell, B., Avizienis, A., and Landwehr, C. (2004). Basic concepts and taxonomy of dependable and secure computing. *IEEE Transaction Dependable Security Computer*, 1(1):11–33.

[Lee 2006] Lee, E. A. (2006). Cyber-physical systems-are computing foundations adequate. In *NSF Workshop On Cyber-Physical Systems: Research Motivation, Techniques and Roadmap*, volume 2. Citeseer.

[Leroy 2009] Leroy, X. (2009). Formal verification of a realistic compiler. *Commun. ACM*, 52(7):107–115.

[Li et al. 2007] Li, N., Li, T., and Venkatasubramanian, S. (2007). t-closeness: Privacy beyond k-anonymity and l-diversity. In *2007 IEEE 23rd International Conference on Data Engineering*, pages 106–115.

[Lima et al. 2009] Lima, M. N., dos Santos, A. L., and Pujolle, G. (2009). A survey of survivability in mobile ad hoc networks. *IEEE Communications Surveys Tutorials*, 11(1):66–77.

[Linger et al. 1998] Linger, R. C., Mead, N. R., and Lipson, H. F. (1998). Requirements definition for survivable network systems. In *Proceedings of the 3rd International Conference on Requirements Engineering*, page 0014, Washington, DC, USA. IEEE Computer Society.

[Lipa et al. 2015] Lipa, N., Mannes, E., Santos, A., and Nogueira, M. (2015). Firefly-inspired and robust time synchronization for cognitive radio ad hoc networks. *Computer Communications*, 66:36–44.

[Liu et al. 2012] Liu, J., Xiao, Y., and Chen, C. P. (2012). Authentication and access control in the internet of things. In *Proceedings...*, pages 588–592. 32nd International Conference on Distributed Computing Systems Workshops (ICDCSW), 2012.

[Liu et al. 2014] Liu, W., Park, E. K., and Zhu, S. S. (2014). e-health pst (privacy, security and trust) mobile networking infrastructure. In *2014 23rd International Conference on Computer Communication and Networks (ICCCN)*, pages 1–6.

[Liu and Wong 2016] Liu, Z. and Wong, D. S. (2016). Practical attribute-based encryption: Traitor tracing, revocation and large universe. *Comput. J.*, 59(7):983–1004.

[López et al. 2015] López, H. A., Marques, E. R. B., Martins, F., Ng, N., Santos, C., Vasconcelos, V. T., and Yoshida, N. (2015). Protocol-based verification of message-passing parallel programs. In *OOPSLA*, pages 280–298, New York, NY, USA. ACM.

[Lopez et al. 2004] Lopez, J., Oppliger, R., and Pernul, G. (2004). Authentication and authorization infrastructures (aais): a comparative survey. *Computers & Security*, 23(7):578–590.

[Lou et al. 2004] Lou, W., Liu, W., and Fang, Y. (2004). SPREAD: enhancing data confidentiality in mobile ad hoc networks. In *Proceedings of IEEE INFOCOM*, volume 4, pages 2404–2413, Washington, DC, USA. IEEE Computer Society.

[Luk et al. 2005] Luk, C.-K., Cohn, R., Muth, R., Patil, H., Klauser, A., Lowney, G., Wallace, S., Reddi, V. J., and Hazelwood, K. (2005). Pin: Building customized program analysis tools with dynamic instrumentation. In *PLDI*, pages 190–200, New York, NY, USA. ACM.

[Lyytinen and Yoo 2002] Lyytinen, K. and Yoo, Y. (2002). Ubiquitous computing. *Communications of the ACM*, 45(12):63–96.

[Maas et al. 2016] Maas, A. J., Nazaré, H., and Liblit, B. (2016). Array length inference for c library bindings. In *ASE*, pages 461–471, New York, NY, USA. ACM.

[Macedo et al. 2016] Macedo, R., de Castro, R., Santos, A., Ghamri-Doudane, Y., and Nogueira, M. (2016). 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*, pages 1–6.

[Mahalle et al. 2013] Mahalle, P. N., Anggorojati, B., Prasad, N. R., and Prasad, R. (2013). Identity authentication and capability based access control (iacac) for the internet of things. *Journal of Cyber Security and Mobility*, 1(4):309–348.

[Maler and Reed 2008] Maler, E. and Reed, D. (2008). The venn of identity: Options and issues in federated identity management. *IEEE Security Privacy*, 6(2):16–23.

[Mann 1997] Mann, S. (1997). Wearable computing: A first step toward personal imaging. *Computer*, 30(2):25–32.

[Manna and Waldinger 1971] Manna, Z. and Waldinger, R. J. (1971). Toward automatic program synthesis. *Commun. ACM*, 14(3):151–165.

[Markmann et al. 2015] Markmann, T., Schmidt, T. C., and Wählisch, M. (2015). Federated end-to-end authentication for the constrained internet of things using ibc and ecc. *SIGCOMM Comput. Commun. Rev.*, 45(4):603–604.

[Marti et al. 2000] Marti, S., Giuli, T. J., Lai, K., and Baker, M. (2000). Mitigating routing misbehavior in mobile ad hoc networks. In *Proceedings of the ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*, pages 255–265, New York, NY, USA. ACM Press.

[Martin and Healey 2007] Martin, T. and Healey, J. (2007). 2006's wearable computing advances and fashions. *IEEE Pervasive Computing*, 6(1).

[Maurer et al. 2016] Maurer, M., Gerdes, J. C., Lenz, B., and Winner, H. (2016). *Autonomous driving: technical, legal and social aspects*. Springer Publishing Company, Incorporated.

[McEliece 1978] McEliece, R. J. (1978). A public-key cryptosystem based on algebraic coding theory. *Deep Space Network*, 44:114–116.

[McGrew et al. 2016] McGrew, D., Kampanakis, P., Fluhrer, S., Gazdag, S.-L., Butin, D., and Buchmann, J. (2016). State management for hash based signatures. *IACR Cryptology ePrint Archive*, 2016/357.

[McGrew et al. 2017] McGrew, D. A., Curcio, M., and Fluhrer, S. (2017). Hash-Based Signatures. Internet-draft, IETF.

[McGrew and Viega 2004] McGrew, D. A. and Viega, J. (2004). The security and performance of the galois/counter mode (GCM) of operation. In *INDOCRYPT*, volume 3348 of *LNCS*, pages 343–355. Springer.

[McMillan 1993] McMillan, K. L. (1993). *Symbolic Model Checking*. Kluwer Academic Publishers, Norwell, MA, USA.

[Merkle 1979] Merkle, R. C. (1979). *Secrecy, authentication and public key systems / A certified digital signature*. PhD thesis, Stanford.

[Miller 1985] Miller, V. S. (1985). Use of elliptic curves in cryptography. In *CRYPTO*, volume 218 of *Lecture Notes in Computer Science*, pages 417–426. Springer.

[Mouha 2015] Mouha, N. (2015). The design space of lightweight cryptography. *IACR Cryptology ePrint Archive*, 2015:303.

[Nazaré et al. 2014] Nazaré, H., Maffra, I., Santos, W., Barbosa, L., Gonnord, L., and Quintão Pereira, F. M. (2014). Validation of memory accesses through symbolic analyses. In *OOPSLA*, New York, NY, USA. ACM.

[Ndibanje et al. 2014] Ndibanje, B., Lee, H.-J., and Lee, S.-G. (2014). Security analysis and improvements of authentication and access control in the internet of things. *Sensors*, 14(8):14786–14805.

[Nethercote and Seward 2007] Nethercote, N. and Seward, J. (2007). Valgrind: a framework for heavyweight dynamic binary instrumentation. In *PLDI*, pages 89–100, New York, NY, USA. ACM.

[Neto et al. 2016] Neto, A. L. M., Souza, A. L. F., Cunha, Í. S., Nogueira, M., Nunes, I. O., Cotta, L., Gentille, N., Loureiro, A. A. F., Aranha, D. F., Patil, H. K., and Oliveira, L. B. (2016). Aot: Authentication and access control for the entire iot device life-cycle. In *SenSys*, pages 1–15. ACM.

[Nguyen et al. 2010] Nguyen, T., Al-Saffar, A., and Huh, E. (2010). A dynamic id-based authentication scheme. In *Proceedings...*, pages 248–253. Sixth International Conference on Networked Computing and Advanced Information Management (NCM), 2010.

[NIST 2006] NIST (2006). Sp 800-56A recommendation for pair-wise key establishment schemes using discrete logarithm cryptography.

[NIST 2013] NIST (2013). FIPS PUB 186: Digital signature standard (DSS).

[NIST 2016] NIST (2016). Post-quantum crypto project.

[NIST 2017] NIST (2017). Digital Identity Guidelines. *NIST Special Publication 800-63-3*. `https://doi.org/10.6028/NIST.SP.800-63-3`.

[Nogueira 2009] Nogueira, M. (2009). *SAMNAR: A survivable architecture for wireless self-organizing networks*. PhD thesis, Université Pierre et Marie Curie - LIP6.

[OASIS 2005] OASIS (2005). Authentication context for the oasis security assertion markup language (saml) v2.0.

[OASIS 2013] OASIS (2013). extensible access control markup language (xacml) version 3.0.

[Oliveira et al. 2017] Oliveira, L. B., ao Pereira, F. M. Q., Misoczki, R., Aranha, D. F., Borges, F., and Liu, J. (2017). The computer for the 21st century: Security & privacy challenges after 25 years. In *2017 26th International Conference on Computer Communication and Networks (ICCCN)*.

[Oliveira et al. 2011] Oliveira, L. B., Aranha, D. F., Gouvêa, C. P. L., Scott, M., Câmara, D. F., López, J., and Dahab, R. (2011). Tinypbc: Pairings for authenticated identity-based non-interactive key distribution in sensor networks. *Computer Communications*, 34(3):485–493.

[Paci et al. 2009] Paci, F., Ferrini, R., Musci, A., Jr., K. S., and Bertino, E. (2009). An interoperable approach to multifactor identity verification. *Computer*, 42(5):50–57.

[Paisante et al. 2016] Paisante, V., Maalej, M., Barbosa, L., Gonnord, L., and Quintão Pereira, F. M. (2016). Symbolic range analysis of pointers. In *CGO*, pages 171–181, New York, NY, USA. ACM.

[Papadimitratos and Haas 2003] Papadimitratos, P. and Haas, Z. J. (2003). Secure data transmission in mobile ad hoc networks. In *Proceedings of the ACM Workshop on Wireless Security (WiSe)*, pages 41–50, New York, NY, USA. ACM Press.

[Patel 2015] Patel, N. (2015). 90% of startups fail: Here is what you need to know about the 10%.

[Pierce 2002] Pierce, B. C. (2002). *Types and Programming Languages*. The MIT Press, 1st edition.

[Poovendran 2010] Poovendran, R. (2010). Cyber-physical systems: Close encounters between two parallel worlds [point of view]. *Proceedings of the IEEE*, 98(8):1363–1366.

[Pornin 2017] Pornin, T. (Accessed in July 2017). BearSSL - Constant-Time Mul. `https://www.bearssl.org/ctmul.html`.

[Pottie and Kaiser 2000] Pottie, G. J. and Kaiser, W. J. (2000). Wireless Integrated Network Sensors. *Communications ACM*, 43(5):51–58.

[Poulis et al. 2015] Poulis, G., Gkoulalas-Divanis, A., Loukides, G., Skiadopoulos, S., and Tryfonopoulos, C. (2015). *SECRETA: A Tool for Anonymizing Relational, Transaction and RT-Datasets*, pages 83–109. Springer International Publishing, Cham.

[Prasser and Kohlmayer 2015] Prasser, F. and Kohlmayer, F. (2015). *Putting Statistical Disclosure Control into Practice: The ARX Data Anonymization Tool*, pages 111–148. Springer International Publishing, Cham.

[Rajkumar et al. 2010] Rajkumar, R. R., Lee, I., Sha, L., and Stankovic, J. (2010). Cyber-physical systems: the next computing revolution. In *47th Design Automation Conference*. ACM.

[Refaei et al. 2005] Refaei, M. T., Srivastava, V., DaSilva, L., and Eltoweissy, M. (2005). A reputation-based mechanism for isolating selfish nodes in ad hoc networks. In *Proceedings of the Annual International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services (MOBIQUITOUS)*, pages 3–11, Washington, DC, USA. IEEE Computer Society.

[Regev 2005] Regev, O. (2005). On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of ACM STOC '05*, STOC '05, pages 84–93, New York, NY, USA. ACM.

[Reis et al. 2017] Reis, T. B. S., Aranha, D. F., and López, J. (2017). Present runs fast: Efficient and secure implementation in software. In *CHES*. Springer. To appear.

[Rellermeyer et al. 2008] Rellermeyer, J. S., Duller, M., Gilmer, K., Maragkos, D., Papageorgiou, D., and Alonso, G. (2008). The software fabric for the internet of things. In *IOT*, pages 87–104, Berlin, Heidelberg. Springer-Verlag.

[Reparaz et al. 2017] Reparaz, O., Balasch, J., and Verbauwhede, I. (2017). Dude, is my code constant time? In *DATE*, pages 1697–1702. IEEE.

[Rice 1953] Rice, H. G. (1953). Classes of recursively enumerable sets and their decision problems. *Trans. Amer. Math. Soc.*, 74(1):358–366.

[Rimsa et al. 2011] Rimsa, A. A., D'Amorim, M., and Pereira, F. M. Q. (2011). Tainted flow analysis on e-SSA-form programs. In *CC*, pages 124–143. Springer.

[Rinaldi et al. 2001] Rinaldi, S. M., Peerenboom, J., and Kelly, T. (2001). Identifying, understanding, and analyzing critical infrastructure interdependencies. *IEEE Control Systems*, 21(6):11–25.

[Rivest et al. 1978] Rivest, R. L., Shamir, A., and Adleman, L. M. (1978). A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126.

[Rodrigues et al. 2016] Rodrigues, B., Pereira, F. M. Q., and Aranha, D. F. (2016). Sparse representation of implicit flows with applications to side-channel detection. In Zaks, A. and Hermenegildo, M. V., editors, *Proceedings of the 25th International Conference on Compiler Construction, CC 2016, Barcelona, Spain, March 12-18, 2016*, pages 110–120. ACM.

[Rotondi et al. 2011] Rotondi, D., Seccia, C., and Piccione, S. (2011). Access control & iot: Capability based authorization access control system. In *Proceedings...* 1st IoT International Forum.

[Russo and Sabelfeld 2010] Russo, A. and Sabelfeld, A. (2010). Dynamic vs. static flow-sensitive security analysis. In *CSF*, pages 186–199, Washington, DC, USA. IEEE.

[Sadeghi et al. 2015] Sadeghi, A. R., Wachsmann, C., and Waidner, M. (2015). Security and privacy challenges in industrial internet of things. In *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, pages 1–6.

[Salem and Hubaux 2006] Salem, N. B. and Hubaux, J.-P. (2006). Securing wireless mesh networks. *IEEE Wireless Communications*, 13(2):50–55.

[Santos et al. 2017] Santos, A. A., Nogueira, M., and Moura, J. M. F. (2017). A stochastic adaptive model to explore mobile botnet dynamics. *IEEE Communications Letters*, 21(4):753–756.

[Seitz et al. 2013] Seitz, L., Selander, G., and Gehrmann, C. (2013). Authorization framework for the internet-of-things. In *Proceedings...*, pages 1–6. IEEE 14th International Symposium and Workshops on a World of Wireless, Mobile and Multimedia Networks (WoWMoM).

[Serebryany et al. 2012] Serebryany, K., Bruening, D., Potapenko, A., and Vyukov, D. (2012). Addresssanitizer: a fast address sanity checker. In *ATC*, pages 28–28. USENIX.

[Shor 1997] Shor, P. W. (1997). Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509.

[Silva et al. 2015] Silva, E. F., Fernandes, N. C., and Muchaluat-Saade, D. (2015). Modelagem do across: Um arcabouço de aa baseado em políticas e atributos para organizações virtuais. In *Workshop de Gestão de Identidade (WGID), Anais do XV Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (SBSeg2015)*, pages 1–12. Sociedade Brasileira de Computação.

[Simon 1994] Simon, D. R. (1994). On the power of quantum computation. In *Symposium on Foundations of Computer Science (SFCS 94)*, pages 116–123, Washington, DC, USA. IEEE Computer Society.

[Soares et al. 2007] Soares, L. F. G., Rodrigues, R. F., and Moreno, M. F. (2007). Ginga-NCL: the declarative environment of the brazilian digital tv system. *J. Braz. Comp. Soc*, 12(4):1–10.

[Soto and Nogueira 2015] Soto, J. and Nogueira, M. (2015). A framework for resilient and secure spectrum sensing on cognitive radio networks. *Computer Networks*, 79:313–322.

[Souza et al. 2017] Souza, A., Cunha, Í., and B Oliveira, L. (2017). NomadiKey: User Authentication for Smart Devices based on Nomadic Keys. *International Journal of Network Management*, pages e1998–n/a.

[Souza and Wangham 2015] Souza, M. C. and Wangham, M. S. (2015). Estudo sobre interoperabilidade entre sistemas de gestao de identidades federadas em ambientes de pesquisas colaborativas. In *Workshop de Gestão de Identidade (WGID), Anais do XV Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (SBSeg2015)*, pages 632–643. Sociedade Brasileira de Computação.

[Stajano 2002] Stajano, F. (2002). *Security for Ubiquitous Computing*. John Wiley and Sons.

[Stevens et al. 2016] Stevens, M., Karpman, P., and Peyrin, T. (2016). Freestart collision for full SHA-1. In *EUROCRYPT (1)*, volume 9665 of *LNCS*, pages 459–483. Springer.

[Teixeira et al. 2015] Teixeira, F. A., Machado, G. V., Pereira, F. M. Q., Wong, H. C., Nogueira, J. M. S., and Oliveira, L. B. (2015). Siot: Securing the internet of things through distributed system analysis. In *IPSN*, pages 310–321, New York, NY, USA. ACM.

[Tekeoglu and Tosun 2015] Tekeoglu, A. and Tosun, A. S. (2015). Investigating security and privacy of a cloud-based wireless ip camera: Netcam. In *2015 24th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–6.

[Wangham et al. 2010] Wangham, M. S., de Mello, E. R., da Silva Böger, D., Gueiros, M., and da Silva Fraga, J. (2010). *Minicursos do X Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais -SBSeg 2010*, chapter Gerenciamento de Identidades Federadas, pages 1–52. Sociedade Brasileira de Computação.

[Wangham et al. 2013] Wangham, M. S., Domenech, M., and de Mello, E. R. (2013). *Minicursos do XIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais - SBSeg 2013*, chapter Infraestruturas de Autenticação e de Autorização para Internet das Coisas. Sociedade Brasileira de Computação.

[Waters 2011] Waters, B. (2011). Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In *Public Key Cryptography*, volume 6571 of *LNCS*, pages 53–70. Springer.

[Weiser 1991] Weiser, M. (1991). The computer for the 21st century. *Scientific american*, 265(3):94–104.

[Weiser 1993] Weiser, M. (1993). Some computer science issues in ubiquitous computing. *Communications of the ACM*, 36(7):75–84.

[Wilson and Lam 1995] Wilson, R. P. and Lam, M. S. (1995). Efficient context-sensitive pointer analysis for c programs. In *PLDI*, pages 1–12, New York, NY, USA. ACM.

[Wu et al. 2017] Wu, M., Pereira, F. M. Q., Liu, J., Ramos, H. S., Alvim, M. S., and Oliveira, L. B. (2017). Proof-Carrying Sensing: Towards a Real-World Authentication Scheme to Cyber-Physical-Human Systems. In *Conference on Embedded Networked Sensor Systems (SenSys)*.

[Yang et al. 2004] Yang, H., Luo, H., Ye, F., Lu, S., and Zhang, L. (2004). Security in mobile ad hoc networks: challenges and solutions. *IEEE Wireless Communications*, pages 38–47.

[Yi et al. 2001] Yi, S., Naldurg, P., and Kravets, R. (2001). Security-aware ad hoc routing for wireless networks. In *Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, pages 299–302, New York, NY, USA. ACM Press.

[Zapata 2002] Zapata, M. G. (2002). Secure ad hoc on-demand distance vector routing. *ACM SIGMOBILE*, 6(3):106–107.

[Zeng et al. 2011] Zeng, D., Guo, S., and Cheng, Z. (2011). The web of things: A survey. *Journal of Communications*, 6(6).

[Zhang et al. 2008] Zhang, C., Song, Y., and Fang, Y. (2008). Modeling secure connectivity of self-organized wireless ad hoc networks. In *IEEE INFOCOM*, pages 251–255.

[Zhang and Liu 2011] Zhang, G. and Liu, J. (2011). A model of workflow-oriented attributed based access control. *International Journal of Computer Network and Information Security (IJCNIS)*, 3(1):47–53.