Enterprise IoT Security and Scalability: How Unikernels can Improve the Status Quo

Bob Duncan
Computing Science
University of Aberdeen
Aberdeen, UK
bobduncan@abdn.ac.uk

Andreas Happe
Dept. Digital Safety & Security
Austrian Inst. of Tech. GmbH
Vienna, Austria
andreas.happe@ait.ac.at

Alfred Bratterud
Dept. of Computer Science
Oslo and Akershus University
Oslo, Norway
alfred.bratterud@hioa.no

ABSTRACT

Cloud computing has been a great enabler for both the Internet of Things and Big Data. However, as with all new computing developments, development of the technology is usually much faster than consideration for, and development of, solutions for security and privacy. In a previous paper, we proposed that a unikernel solution could be used to improve security and privacy in a cloud scenario. In this paper, we outline how we might apply this approach to the Internet of Things, which can demonstrate an improvement over existing approaches.

CCS Concepts

•Information systems → Enterprise information systems;

Keywords

Cloud Security and Privacy; attack surface; compliance

1. INTRODUCTION

The Internet of Things (IoT) has been around for quite a while, but it was not until cloud computing and big data arrived that the IoT really started to take off. In 2007, Gantz et al [13], suggested that global data collection would double every 18 months, and Cisco noted that the IoT had really come of age in 2008, as there were now more things connected to the internet than people [11]. Now, there is no longer any limitation on what we can do with it. The importance of this technology should not be underestimated. It can be used for varied, and immensely important uses such as: defence, domestic and home automation, eHealth, industrial control, logistics, retail, security and emergencies, smart airports, smart agriculture, smart animal farming, smart cars, smart cities, smart environment, smart metering, smart parking, smart roads, smart trains, smart transport, smart water, to name but a few.

UCC '16 December 6–9, 2016, Shanghai, China

DOI: 10.1145/1235

The general consensus on predictions for the number of connected devices by 2020 is reckoned to be between 20 billion to 35 billion [14][1][26][10][24], connected devices. Those devices will certainly be capable of generating a huge volume of data. We can look at the IoT as a potentially huge producer of data, which is why the use of cloud is so important. As a first stage in the flow of data, the Cloud/BigData can be seen as the data consumer, although, ultimately, there will usually be an onward flow of this data for other commercial reasons. The data producers then are "mobile" and geographically dispersed, and the first stage data consumer, the cloud, can also be geographically dispersed, meaning the enterprise could benefit from the consumers being near to the data producers in order to minimise transport load. Dead capacity is not sustainable, which might lead to some kind of on-demand computing that should be co-located near the consumers.

But, of course, while this all sounds really useful and interesting, it also raises other issues and concerns. It is well known that IoT technology is particularly vulnerable to attack, and the IoT and Big Data is no exception. Indeed, this area is poorly regulated, with few proper standards yet in place, which suggests it might be potentially more vulnerable than existing systems which have been around for some time now. Issues of security, privacy and accountability have yet to be properly resolved.

Traditionally a very important part of the enterprise architecture was the central enterprise firewall through which all traffic was routed. This was a fundamental element of achieving enterprise security, which evolved in the late 80's [19]. The adoption of distributed computing architecture would lead to improvements in the central firewall to provide a distributed firewall. Subsequent adoption of mobile technology, followed by cloud technology, started to affect the efficacy of this approach, and many enterprises have yet to adequately adapt their systems to the additional pressures these new technologies bring to the goal of achieving good enterprise security. But now, with IoT devices the possible routes in to the enterprise's network have exploded. If we need cloud computing resources for scale-out, we are fully outside the typical enterprise firewall. Fog computing makes this even tougher, as now additional processing is added outside of the traditional security scope. As [28], suggests, technology tends to get used in unexpected ways. This makes enterprise security tough, as these unintended ways are often not forseen. In addition, IoT devices are too resource-constrained to employ traditional security tools (virus scanner, etc.), so that leaves the enterprise network

far more exposed to attack.

It is this concern that drives us to consider how the use of a unikernel solution might be deployed to improve the status quo. In this paper, we outline how we might approach developing a solution to these issues and concerns. In Section 2, we look at some typical deployment architectures currently in use, and note how they fail to address the security and privacy challenges we will mention in Section 3, where we outline the security and privacy challenges faced by any enterprise wishing to safely use this new technology. In Section 4, we outline the unikernel angle for both client and server. In Section 5, we talk about some of the challenges, problems and limitations IoT brings to the enterprise, and in Section 6, we discuss our conclusions.

2. SOME EXAMPLES OF CURRENTLY DEPLOYED IOT ARCHITECTURE

There are a number of goals for the IoT, namely cost-effectiveness, efficiency, quality of service, mobility, manageability and of course security and privacy. The first of these, has lead to a proliferation of cheap sensor-like devices being used. While they are certainly cheap, they are also generally dumb, in the sense that they have very limited resources, meaning it is unlikely that these devices can be made to be smart. They merely collect data, which has to be passed on down the line. In this context, efficiency refers to low power consumption, since it may be necessary to rely on battery or solar power. Quality of service in this context means the ability to prioritise data streams from different devices. Mobility, obviously refers to the fact that some devices need to be mobile, where these devices move physically from one place to another. Manageability means there needs to be some way of intelligently managing the architecture, including both centralised and distributed control. Security and privacy are vital aspects of the architecture to ensure the archtecture is both resilient to attack, and able to withstand leakage of personally identifiable data.

However, since there are no agreed standards currently in existence, and there are many different ways of approaching this task, we need to consider each deployment very carefully based on the architecture specified, or actually in use. This makes the task of ensuring the security and privacy aspects of the architecture very difficult indeed.

The overall IoT architecture consists of uncountable minimal devices — distributed throughout the world — exchanging data with fewer centralized servers. Given the amount of data produced, the impact upon communication and processing infrastructure is very high. Another problem is the potentially high cyclic traffic pattern that creates short-term traffic and utilization spikes.

The local devices themselves are resource-constrained. Neither processing nor memory is abundant. Compared to a desktop-class computer, updating an IoT device is tough. As there is no user interface, a botched update may shut down the device. Special care must be taken to make sure that even a corrupt update can be recovered from. In addition, all means that prevent updates in the first place, e.g., hardened systems that reduce the amount of needed security updates, are important investments in an enterprise's long-term business prospects.

Many of these devices use different hardware architectures and configurations, and often software needs to be written on a per-device basis, meaning there is often little consistency from one device to another.

On the server-side, the cloud, with its limitless scaling capabilities, would solve the dynamic processing requirements. To allow for scale-out, new paradigms have to be embraced. In the database world, the rise of NoSQL databases was partially backed by their good scaling capabilities — even accepting their reduced consistency models when compared to traditional SQL databases.

The natural way of reducing transported data is to compact and reduce the data on-site. To achieve this, parts of the server-side code must be moved towards local processing stations. We assume that those are more powerful than sensors, but powerless compared to full cloud offerings. When data is processed on-site, special attention must be paid to the processing operation's integrity itself. If the on-site processing can modify all incoming data, it will be a prime target for offensive security attackers.

It is very important to realise that medical and power grid data will become a fundamental cornerstone of future societies. It will therefore be extremely important to introduce some form of verifiable computing in order to safeguard this vital data. This would allow the final data-centre to verify that the computations carried out along the data trail by other machines have been performed correctly.

Other concerns arise for so called Smart Home setups. Future homes will be augmented with a multitude of sensors and control mechanisms, which in addition to forwarding data to the cloud, might be controlled by the latter. Current setups are, for example, feedback-loops between window sensors and the heating system/climate control, automatic door openers as well as integrated kitchen systems where pans can automatically control stoves and ovens. All of which might be interesting for a malicious actor that can utilize this to create additional costs (energy consumption) or open otherwise closed doors. Infrastructure-wise we see a combination of dumb sensors, a local Smart Home hub with limited processing capabilities and a direct uplink to the cloud for remote control capabilities. This Smart Hub will also be a prime target for attackers as it is in scope of the home user and thus seldom maintained by professional IT administration.

The Smart Home area can be seen as the natural commercial extension of the original Industry 4.0 area. Here sensors are integrated into the manufacturing process to improve efficiency. The direct monetary incentive has lead to fast adoption within this area. Too fast, if you look at the various security incidents against industrial control systems. This area is also very vulnerable as traditional systems assume the local network to be secure, i.e. there are very few defensive security mechanisms in place. Now those areas are accessible through a network, maybe even the Internet. As already seen, those networks are a target for professional advanced persistent threat (APT) groups as well as for state-owned actors. What could go wrong?

An area especially sensitive is Smart Health. It ranges from local on-site medical monitoring to the highly sensitive network within hospitals. Especially after the EU's General Data Protection Act (GDPA) with its increased fines for personal data breaches, industry cannot ignore this area. Documented attacks (ransom-ware) as well as penetration-tests within prototype test hospital settings have shown the current low security level that medical devices currently have.

IoT will increase the amount of devices used within this setting, if they are not secure data leaks or invasive malicious attacks against medical devices (such as pacemakers and drug dispensers) will become more common.

All of those scenarios have the common theme of massive deployments. While the initial roll-out will be highly driven (and sometimes performed) by customers buying new devices, subsequent update needs might introduce new problems. Companies producing the IoT devices (e.g., Smart Light Bulbs, Heat Sensors) need ways of updating the millions of deployed devices. Compared to software, this has an additional problem: if a security sensor mandates the update of a million of loosely connected devices, and lives are in danger, there is a much higher level of urgency.

3. IOT SECURITY AND PRIVACY CHALLENGES

For any enterprise, there are many security challenges which must first be addressed. For any application using cloud, [9], have developed a useful list of ten key security goals that must be addressed, which we see in Table 1 below.

-5pt

Number	Key Security Challenges
1	The definition of security goals
2	Compliance with standards
3	Audit issues
4	Management approach
5	Technical complexity of cloud
6	Lack of responsibility and accountability
7	Measurement and monitoring
8	Management attitude to security
9	Security culture in the company
10	The threat environment

Table 1: Duncan and Whittington 10 Key Security Issues — 2016 [9]

However, when we want to incorporate IoT into a cloud setting, we must do more. For this purpose, we can start by looking at the work done by the Open Web Application Security Project (OWASP), who publish a number of relevant lists we can use to help us deal with the additional issues we will face in using the IoT. The first of these is their top ten list of web security vulnerabilities , which they publish every three years. These lists are derived from real world intrusions reported globally, which are reflective of what the attackers are actually doing successfully. Since these are live and successful attacks, it makes sound sense to close these loopholes first. The latest list is provided in the table below.

This list is based on the result of analysis of successful security breaches across the globe, which seeks to highlight the worst areas of impact of weaknesses in web based computing systems. However, thanks to the innovative techniques in use for IoT, that is still not enough. OWASP now produce a list of the worst 10 vulnerabilities in the use of mobile technology, which we show in the list below.

But, of course, it is not quite as simple as that. The IoT mechanics extend beyond traditional web technology and mobile technology. In 2014, OWASP developed a provisional top ten list of IoT vulnerabilities, which we outline below in Table 4.

2013 Code	Threat
A1	Injection Attacks
A2	Broken Authentication and Session Management
A3	Cross Site Scripting (XSS)
A4	Insecure Direct Object References
A5	Security Misconfiguration
A6	Sensitive Data Exposure
A7	Missing Function Level Access Control
A8	Cross Site Request Forgery (CSRF)
A9	Using Components with Known Vulnerabilities
A10	Unvalidated Redirects and Forwards

Table 2: OWASP Top Ten Web Vulnerabilities — 2013 [20]

2013 Code	Threat
M1	Insecure Data Storage
M2	Weak Server Side Controls
M3	Insufficient Transport Layer Protection
M4	Client Side Injection
M5	Poor Authorization and Authentication
M6	Improper Session Handling
M7	Security Decisions via Untrusted Inputs
M8	Side Channel Data Leakage
M9	Broken Cryptography
M10	Sensitive Information Disclosure

Table 3: OWASP Top Ten Mobile Vulnerabilities — 2013 [20]

An important point to bear in mind is that the above table represents just the OWASP top ten vulnerability list. OWASP are currently working on a full list of 130 possible IoT vulnerabilities which might need to be taken into account. While all this at first might seem like a huge task, OWASP do provide good suggestions as to how to mitigate these issues.

And the above just covers security issues. We also have to consider the challenges of privacy issues. With the increase in punitive legislation and regulation surrounding issues of privacy, we must necessarily concern ourselves with providing the ability to ensure the goal of privacy can be achieved. The good news is that if we can achieve a high level of security, then it will be much easier to achieve a good level of privacy [8]. Good privacy depends on having a high level of security. We can have security without privacy, but we can't have privacy without security.

While the IoT has progressed significantly in technical terms in recent years, it has very much done so at the expense of security and privacy, for example accessing utility companes, including nuclear in the US [31], damage caused to German steel mill by hackers [32], drug dispensing machines hacked in US [27], plane taken over by security expert mid-air [5], and a hack that switched off smart fridges if it detected ice cream [3]. While enterprises often might not care too much about these issues, they should. If nothing else, legislators and regulators are unlikely to forget, and will be keen to pursue enterprises for security and privacy breaches. In previous years, it was often the case that legislators and regulators had little teeth, but consider how punitive fines have become in recent years following the banking crisis in 2008. In the UK in 2014, the Financial Conduct Authority

2014 Code	Threat
I1	Insecure Web Interface
I2	Insufficient Authentication/Authorization
I3	Insecure Network Services
I4	Lack of Transport Encryption
I5	Privacy Concerns
I6	Insecure Cloud Interface
I7	Insecure Mobile Interface
I8	Insufficient Security Configure-ability
I9	Insecure Software/Firmware
I10	Poor Physical Security

Table 4: OWASP Top Ten IoT Vulnerabilities — 2014 [21]

(FCA) fined a total of £1,427,943,800 [12], during the year.

4. THE UNIKERNEL ANGLE

Why use unikernels for the IoT [22]? Unikernels are uniquely suited to benefit all areas (sensor, middleman, servers) within the IoT chain. They allow for unified development utilizing the same software infrastructure for all layers. This may sound petty, but who would have thought JavaScript could be used on servers (think node.js) a couple of years ago?

4.1 On the Client

Unikernels are a form of virtualisation and thus offer all of its benefits: they provide a unified interface to diverse hardware platforms to application developers. This allows the latter to focus on application development. They allow the ability to mask changes of the underlying hardware platform behind the hypervisor. This also allows for application code to be reused between different hardware revisions. In addition, system and application development is often performed by disjunct groups within an enterprise. Using a unikernel decouples both groups and thus allows parallel-alized development. Application developers can utilize a virtualized testing environment on their workstations during development, but can assume that the same environment will be available within the production environment.

Unikernels can produce leaner virtual machines when compared to traditional virtualization solutions. This minimalism yields a much reduced attack surface which in turn creates more secure applications. Using a resource efficient unikernel such as IncludeOS will minimize the computational and memory overhead that otherwise would prevent virtualization from being used. While the small memory and processing overhead enables the usage of virtualisation on low-powered IoT devices in the first place, it also aids higher capacity devices. Lower resource utilization allows for either better utilization (i.e., running more services on the same hardware) or higher usage of low power modes (thus reducing energy consumption). Both increase the sustainability of IoT deployments.

Another feature that is in high demand by embedded systems is atomic updates: a system supporting atomic updates either installs a system update or reverts back to a known (working) system state. For example, Google's Chrome OS [16], achieves this by using two system partitions. A new system upgrade is installed onto the currently unused partition. On the next boot the newly installed system is used, but the old system is pre-selected as a backup boot option if the initial boot does not work. If the new system boots, the new system is marked as the new default operating system and the (now) old partition will be used for the next system upgrade. This allows high resilience in the face of potentially disrupting Chrome OS updates. A similar scheme is set to be introduced for the upcoming Android Version 7. This scheme would be greatly aided by unikernels: they already provide a clear separation of data and control logic. A system upgrade would thus start a new unikernel and forward new requests to it. Of course, if the underlying hypervisor has to be upgraded (which due to its minimal size should be a very rare event) the whole system might incorporate the dual boot-partition approach.

4.2 On the Server

Given the large estimated number of IoT devices to be deployed in the near future, computational demand on data centres (or nowadays the cloud) can be immense. While IoT amplifies the amount of incoming traffic, it has some characteristics that should favour unikernel-like architectures.

For one, our envisioned unikernels utilize a non-mutable state and are event-based. This combination allows for simplified scale-out, i.e. it allows for dynamically starting more unikernels if incoming requests demand it. We do believe that many processing steps during an IoT dataflow's lifetime will be parallelizable, e.g. data collected from one household will not interact with data gathered by a different household from another continent during the initial processing steps, or possibly never at all. As they do not interact, there is no chance of side effects, thus the incoming data can instantly be processed by a newly spawned unikernel.

Two recent trends in cloud computing are cloudlets and fog computing. The former describes a small-scale data centre located near the internet's edge (i.e. co-located near many sensors and acting as upstream for the incoming IoT sensors) while the latter describes the overall technique of placing storage or computational capabilities near the network edges. To allow for easy usage of this paradigm, a unified execution environment is needed: when the same environment is employed, application code can easily be moved from the cloud towards the networks' edge, i.e. into the cloudlets. Unikernels offer closure over the application's code, so the same unikernel can be deployed at a cloudlet or within a central data centre. Of course, the unikernel itself might place requirements upon external facilities such as storage, which would need to be provided by the current execution environment. A consumer-grade version of this trend can already be seen: many high-powered NAS devices allow for local deployment of virtual machines or containers. This moves functionality from the cloud to a smallest-scale local processing environment. A good use-case for this would be Smart Homes: here a local NAS can perform most of the computations and then forward the compressed data towards a central data centre. In addition, this local preprocessing can apply various cryptographic means to improve the uploaded data's integrity or confidentiality.

5. CHALLENGES, PROBLEMS, LIMITATIONS

5.1 Unikernels are Only a Part of the Solution

While they offer benefits for deployment and security — mostly through their compactness as well as their closure guarantees — large-scale deployments place high stress on

infrastructure for handling the roll-out, monitoring and logging. The software for this infrastructure has yet to be written.

5.2 Production-Level Debugging

First, in a perfect world, no production-level debugging would ever occur, as all bugs would be detected and fixed within the development or staging environment. But reality begs to differ. A common complaint is that unikernels lack debugging facilities: there is just no shell to log-in and visualize the environment. The more root dependent the user becomes, the more frequent this complaint arises. Unikernels (as well as Function-as-a-Service architectures) target DevOp outfits where development and administration has been integrated. The person in charge of debugging is a software developer herself and thus can utilize developer-centric debugging facilities. If debugging is to be performed by developers, groundwork is needed: there must be infrastructure in place that allows securely connecting the debugging tools to the running unikernel within the virtual machine. While this might be feasible for unikernels running within an enterprise's private cloud, the security impact of connecting to deployed IoT devices is massive. We believe that this infrastructure will remove most of the debugging complaints.

5.3 Impact upon Software Development

We assume a unikernel to have a single execution flow, i.e. to have a single execution thread or process, as well as to offer no mutable state within the unikernel itself [2]. This prevents quick adoption of many existing software packages. Where others see limitations, we see opportunities. The single execution-flow paradigm almost enforces the usage of an event-based software programming style, i.e. all processing is triggered by internal (e.g., timers) or external (e.g., new incoming data) events. If no new event is available or processed, the device can safely enter a deeper power-saving state. Minimizing energy consumption is paramount for IoT devices so we assume that this is a much wanted feature. On the server-side this will reduce the overall power-consumption: while not being an essential requirement, as for IoT devices, the reduced power bill will be a nice benefit for enterprises.

This stateless-ness allows us to start new unikernels on-demand. Together with an event-driven architecture and rapid boot-up times, this allows us to minimize the number of running unikernels. This reduces the memory consumption (thus energy impact) of deployed services. Upgrading a unikernel also becomes easier: while the old version of unikernels are still processing their current request, new requests will be forwarded to the new version of the unikernel.

This feature also allows for improved resilience in the face of errors. When functionality is split up between multiple unikernels, faults are automatically contained within a single executing unikernel [17]. Together with monitoring and automatic life-cycle management, this leads to error-resilient services that have limited self-healing capabilities. Contrast this with monolithic applications where often the whole application — or even worse, the application server containing multiple applications — has to be restarted in the case of errors. Due to resource constraints, we are initially limiting resilience research to server-side unikernels, but the same techniques can be applied on IoT devices as well. Arguably, self-healing capabilities are of even higher importance on devices with only limited means of user interaction.

5.4 Virtualization in the Embedded Space

As we already stated in Section 1, there are no standards when it comes to components for the IoT. This means there is a huge range of different architectures vying for a place in this potentially massive market space. Obviously, from a technical standpoint, greater flexibility and power can be obtained through good use of virtualization. Virtualisation is not new, and has been around since 1973 [23]. Bearing in mind that dumb sensors do not have enough resources or lack hardware support for virtualisation (or at least Linux-based virtualisation), we will have a quick look at some of the most popular hardware in use in this space.

ARM [15], presented the ARM capabilities at this workshop in 2009. ARM is one of the most used platforms in the IoT. and has virtualization extensions. Columbia University have developed KVM/ARM, an Open-Source ARM Virtualization System [4]. Dall and Nieh [6], have written an article on this work for LWN.net, and for a conference [7]. There has been paravirtualization support in ARM Coretex A8 since 2006, and ARM Coretex A9 since 2008, with full virtualization since approx. 2009. Virtualisation is also in Linux Kernel 3.8. There are also MMU-less ARMs, although it unlikely that these could be used, unless we were to forfeit the unikernel's protection.

Most smart devices can generally handle virtualization — devices such as smart phones, smart automotive systems, video boxes, play stations, and smart TVs too, although this may not necessarily be the case for small embedded components, such as wear-ables, sensors and other IoT components. MIPS also supports virtualization [18][30]. Some Intel Atom processors support virtualization (the atom range is huge). However, the low-power Intel Quark has no support for virtualization whatsoever. The new Open-Source RISC-V architecture [25], also supports virtualization.

As we can see, many of the current IoT systems in use do have the capability to handle virtualization. For example most high-powered NAS systems now have virtualization (and app) support. Thus we could potentially utilize NAS or other low-powered devices (which are mostly ARM, MIPS or x86) to aggregate data on-site and then transport the reduced data to the "real" cloud.

At the moment, we should carefully consider the current state of security and privacy in a massively connected world. Now we can really see that "big brother" is watching you. Not just through the use of massive CCTV networks, but also through IoT enabled devices which will become embedded in every smart city. It is estimated that in smart cities of the future there will be approximately 5000 sensors watching as you move through the city at all times. What could possibly go wrong? How much personal information could leak as you walk? How much of your money could NFC technology in the wrong hands steal from you, without you being aware of it happening? Do you trust the current technology? We can read about more of these issues in [29].

6. CONCLUSIONS

We have taken a look at the exciting new paradigm of the IoT. While the possibilities are indeed exciting, the consequences of getting it wrong are likely to be catastrophic. We cannot afford to carry blindly on. Instead, we must recognise

that if the issues we have outlined on security and privacy are not tackled properly, and soon, we will all be sleep-walking into a disaster. However, if we realise that we need to take some appropriate actions now, then we will be much better placed to feel comfortable in living in an IoT world. There are considerable potential benefits for everyone to be offered from using our unikernel based approach. While we see security and confidentiality of data as paramount — and given the EU's GDPA, we believe the EU agrees — security and privacy do not directly translate into a monetary benefit for companies and thus are seldom enough for change to gain traction. To better convince enterprises, we offer the added benefit of increasing developer efficiency. Experienced and talented developer resources are scarce at hand, so making the most of it is within an enterprise's best interest. The broad application of a virtualisation solution allows to better reuse existing knowledge and tools as developers gain a virtual long-term environment that they can work in.

Virtualisation in combination with the special state-less nature of many unikernels provide a solution for short-term processing spikes. Processing can be scaled-out to in-company or public clouds by deploying unikernels—as they do not require external dependencies and do not contain state, deployments are simplified. After their usage they can be discarded (no state also means that no compromising information is stored at the cloud provider). In case of sensitive information special means, e.g., homomorphic encryption or verifiable computing technologies need to be employed to protect data integrity or confidentiality.

Unikernels offer a high energy efficiency. This allows companies to claim higher sustainability for their solutions while reducing their energy costs. We view our proposed solution as taking a smart approach to solving smart technology issues. It does not have to be exorbitantly expensive to do what we need, but by taking a simple approach, sensibly applied, we can all have much better faith in the consequences of using this technology (as well as having the comfort of being able to walk through a smart city without having your bank account emptied.

7. REFERENCES

[1] BIIntelligence. Here's how the Internet of Things will explode by 2020, 2016.

[2] A. Bratterud, A. Happe, and B. Duncan. Enhancing Cloud Security and Privacy: The Unikernel Solution. In *Submitt. to CloudComputing 2017*, pages 1–8, 2017.

[3] CBR. IoT security breach forces kitchen devices to reject junk food, 2015.

[4] Columbia. KVM/ARM: an Open-Source ARM Virtualization System, 2016.

[5] DailyMail. Security expert who 'hacked a commercial flight and made it fly sideways' bragged that he also hacked the International Space Station, 2015.

[6] C. Dall and J. Nieh. Supporting KVM on the ARM Architecture, 2013.

[7] C. Dall and J. Nieh. KVM/ARM: the design and implementation of the linux ARM hypervisor. In *ACM SIGPLAN Not.*, volume 49, pages 333–348. ACM, 2014.

[8] B. Duncan, A. Bratterud, and A. Happe. Enhancing Cloud Security and Privacy: Time for a New Approach? In *INTECH 2016*, pages 1–6, Dublin, 2016.

[9] B. Duncan and M. Whittington. Enhancing Cloud Security and Privacy: The Power and the Weakness of the Audit Trail. In *Cloud Comput. 2016*, pages 125–130, Rome, 2016.

[10] EMC. Discover the Digital Universe of Opportunities: Rich Data and the Increasing Value of the Internet of Things, 2014.

[11] D. Evans. The Internet of Things: How the Next Evolution of the Internet is Changing Everything. Technical report, Cisco, 2011.

[12] FCA. Fines Table - 2014, 2014.

[13] J. F. Gantz, D. Reinsel, C. Chute, W. Schlichting, J. McArthur, S. Minton, I. Xheneti, A. Toncheva, and A. Manfrediz. The Expanding Digital Universe: A Forecast of Worldwide Information Growth Through 2010. In *Extern. Publ. IDC (Analyse Futur. Inf. Data*, pages 1–21. IDC, 2007.

[14] Gartner. Gartner Says 6.4 Billion Connected "Things" Will Be in Use in 2016, Up 30 Percent From 2015, 2015.

[15] J. Goodacre. No Title. In *Virtualization Euro Work. 2009*, 2009.

[16] Google. Google Chrome OS, 2015.

[17] A. Happe, B. Duncan, and A. Bratterud. An Architectural Framework for Secure, Large Unikernel Cloud Systems. In *Submitt. to Closer/Complexis 2017*, pages 1–8, 2016.

[18] Imgtech. MIPS Virtualization, 2016.

[19] K. Ingham and S. Forrest. A history and survey of network firewalls. *Univ. New Mex. Tech. Rep*, 2002.

[20] OWASP. OWASP Top Ten Vulnerabilities 2013, 2013.

[21] OWASP. OWASP Top 10 IoT Vulnerabilities (2014), 2014.

[22] R. Pavlicek. Unikernel-based microservices will transform the cloud for the IoT age, 2016.

[23] G. J. Popek and R. P. Goldberg. Formal Requirements for Virtualizable Third Generation Architectures. *ACM SIGOPS Oper. Syst. Rev.*, 7(4):112, 1973.

[24] J. Research. âĂŸInternet of Things' Connected Devices to Almost Triple to over 38 Billion Units by 2020, 2016.

[25] Riskv.org. Open-Source RISK V Architecture, 2016.

[26] G. Sachs. The Internet of Things: Making sense of the next mega-trend. Technical report, Goldman Sachs, 2014.

[27] SecurityWeek. FDA Issues Alert Over Vulnerable Hospira Drug Pumps, 2015.

[28] T. Seo. Making Sense of Enterprise Security, 2016.

[29] S. Sharma, V. Chang, U. S. Tim, J. Wong, and S. Gadia. Cloud-based Emerging Services Systems. *Int. J. Inf. Manage.*, pages 1–19, 2016.

[30] I. Technologies. The MIPS Architecture and Virtualization, 2016.

[31] U. Today. Hackers Breach US Dept of Energy Copmputers 150 Times in 4 Years, Including 19 Nuclear Breaches, 2015.

[32] Wired. German Steel Mill HackedCausing Massive Damage, 2015.