RFID Platform as a Service, Containerized Ecosystem, Feasibility and Security Impact Analysis

Lukas KYPUS, Lukas VOJTECH, Tomas ZITTA, Marek NERUDA

Department of Telecommunication Engineering, Faculty of Electrical Engineering,
Czech Technical University in Prague, Technicka 2, 166 27 Prague, Czech Republic

kypusluk@fel.cvut.cz, vojtecl@fel.cvut.cz, zittatom@fel.cvut.cz, nerudmar@fel.cvut.cz

DOI: 10.15598/aeee.v13i5.1499

Abstract. *This paper presents a new concept as a special type of virtualization of particular event based communication components in RFID ecosystems. The new approach is containers based virtualization, and it is applied and tested on the container of Object name service. The results of the experiment allowed us to do the preliminary analysis of security consequences on the isolated containerized DNS-based RFID subservice. We confirmed feasibility with this sandboxing technology represented by the special container. They bring the benefits in terms of efficient software component life-cycle management and integrity improvements. Experiments results of the containerization are discussed to show the possible isolation ways of other components like EPCis and middleware. There is present evaluation towards external threats and vulnerabilities. The result is a higher level of integrity, availability of whole ecosystem and resiliency against external threats. This gives a new opportunity to build robust RFID as Platform as a service, and it proves the ability to achieve a positive impact on the end to end service Quality of service.*

Keywords

Application virtualization, information security, PaaS, RFID.

1. Introduction

Before we faced the era of virtualization and containerization, as proposed in [1], developers and administrators have had the only possibility to develop and maintain components of any software based ecosystem on premises or within data-centres. All those installations required physical hardware to be designed, sized, shipped, tested, operated and maintained. Software operated on it required a similar approach. The usual life-cycle management consists of many phases like building, configuration, versions, shipping, deployment and running of applications and services discussed in [2] and we know it brings a lot of follow up problems. Same principles are valid for a general software solution in Radio Frequency Identification (RFID) ecosystems.

To solve it there are proposals for the modern RFID application development to move upper described process towards Platform as a Service (PaaS) direction, as proposed in [3]. We realized there is necessary to allow the creation of utilization models, which help to align systems to new multi-tenant RFID environment with all in previous researched summarized benefits possibly achieved by the virtualization.

Nowadays more and more projects are being stressed to shorten the time from design, through delivery up to hand-over to the operations. RFID ecosystems can be considered as platforms of basic and complementary, well-known components. Tagged objects are basic components, which are read by readers via radio based environment. Each read consists of several events, which belong to accordingly mapped and processes record within systems like RFID middleware. RFID middleware, complex events based units, are responsible for the processing as well as for granting appropriate level of quality how proposed in [4] and [5].

Each and every RFID ecosystem usually consist of complementary processing components like: secured network infrastructure, object name service (ONS), middlewares, databases and certification authorities, as researched in the architecture published in [6] with its cloud variant.

Containerization problems as briefly introduced in [7] we enhanced, researched and analyzed can be considered as a new, specific way towards PaaS. It is crucial to emphasize that there exist system components

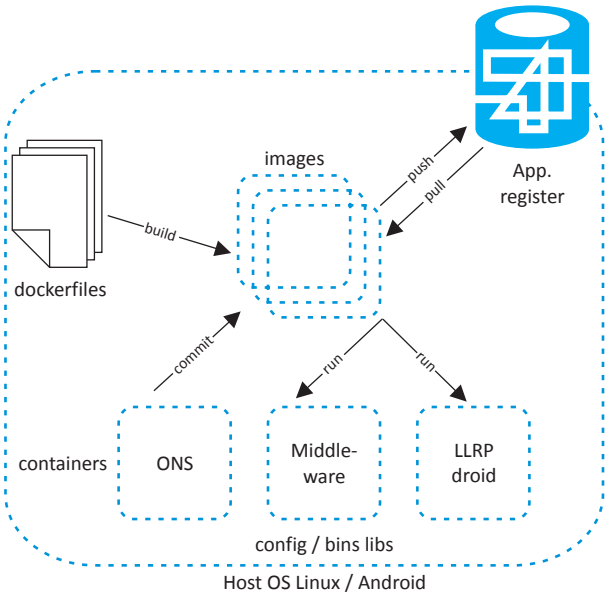

Fig. 1: Configure, build, ship and operate flows.

of RFID ecosystem, which are not suitable or not recommended to be containerized from different reasons or by their nature, even though there could be foreseen benefits from cloud perspective. Those reasons could be dependencies to physical hardware like USB, Ethernet or other peripherals. Followed by a system, which requires extremely low latency operations up to fulfilling extreme performance demand and finally by a system which by design does not support a mission critical applications which were not secured for virtualization.

Other reasons preventing containerization are a local presence and physical and logical security limitations. For the feasibility targeted experiment, we had chosen those which are capable of being containerized as middleware and ONS. These components were suitable for our tests. We applied containerization methods, and it is important to mention that sand-boxing is more about the secure execution of untrusted code [8] while containerization is about secure execute of trusted applications as depicted in Fig. 1.

Although containerization is the new virtualization as presented in [9] we decided to build this experiments based on Docker which allowed us to do sand-boxing (focus on integrity), and by containerizing (focus on build, ship, run everywhere and anytime). We had chosen containerization over pure virtualization with Docker technology mainly, supported by research in [10] to uncover the performance impact and get better insights into QoS.

Technological principle of Docker isolates applications by packing them with their dependencies, configuration files, libraries, interfaces into one consistent image based environment like atomic unit, called containers according to [11]. Those containers can be seamlessly moved between hosts or within cloud providers and we combined different approaches to prove competitiveness with equipped.

The rest of the article focuses on the analysis. A study is conducted by experiments which were compiled based of two problem questions. As first we studied if it is possible to incorporate containerization into RFID ecosystems. Secondly we analysed what will be the level of overhead/impact in case we focus on achieving higher level of security by application containerization.

2. Methods of Containerization

Docker as the chosen containerization technology heavily relies on two components of Linux kernel like **Cgroups** and **Namespaces**. Other components could be combination of SElinux, AppArmor, Netlin, Netfilter, lxc, libcontainer and libvirt development libraries.

The environment was recently extended with execution driver API, very similar in principle to the one which is used in software defined networks. Execution environment is surrounding each virtualized container. Docker engine replaces hypervisor and guest OS in case of common containerization as visible in Fig. 2, where we introduced RFID components into this concept. Each container is combination of Cgroups and Namespaces. Namespaces and the Cgroups are capable to be combined to build final container configuration.

Cgroup is the way to combine and isolate resources like processes into process trees groups. It helps to differentiate which processes belong to which container. The main security feature here is that same processes in different process trees cannot see, interact, influence,

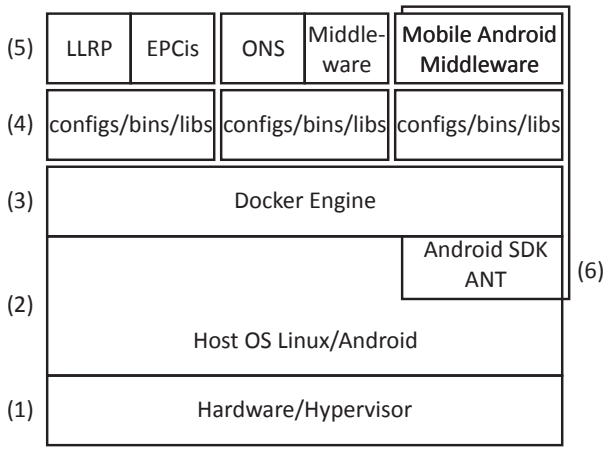

Fig. 2: RFID containerized layered architecture.

and threaten the others. It is even possible to encapsulate main process INIT which is in charge of all other subprocesses. Cgroups like major components are responsible for management of subprocesses to get the resources which were allocated available on time and within required configuration.

Cgroups allow controllers: a) to limit cpu - split not according percentage but according the predefined units, which are supposed to quantify fragment of cpu time the same is valid for the memory allocation, b) to provide cpuset - example: 4 cores, it is not possible to provide 1.25 of core, and c) to limit block devices input/output performance by blkio to limit virtual drives access, write, read, input/output operations.

We performed set of feasibility tests in two different set-ups with virtual NAT; option for container to run is `docker run -P portIN:portOUT` and with native network TCP/IP stack; option for container is `docker run -net=host`. Usual Docker based container with network stack brings a slight decrease of network performance measurable already on by simple ICMP ping tools towards system gateway: **Native:** 1000 packets transmitted, 1000 received, packet loss 0 %, time 9008 ms. rtt min /avg/ max /mdev = 0.542/0.703/0.912/0.094 ms. **Containerized:** 1000 packets transmitted, 1000 packets received, packet loss 0 %, round-trip min /avg/ max /stddev = 0.704/0.833/1.103/0.134 ms.

Namespaces provide the most straightforward way of isolation. By this we achieve behaviour when one process cannot see neighbour or parent processes. Examples of namespaces are: mnt - mount, particular container, can see only root fs, not devices mounted by the underlying system, UTS - Unix time sharing namespace, PID namespace – assures INIT not capable without PID 1 and user namespace - root in container is not root high privilege admin account on hardware node.

For all the experiments we relied on Docker engine version 1.6.0. The principle is to bring all necessary entities (files, objects, services, stacks) into an isolated environment presented like sandbox or container. We let the external broker entity, in our case it will be Docker engine, to manage and operate separated applications domains for each RFID component. Separation of resources dictates to objects and subjects to achieve certain predefined functionality and security goals.

2.1. Mobile Middleware

As the first application to containerization, we have chosen, is RFID mobile middleware called LLRP.droid which was developed recently on our department. Mobile application was assessed and assured ability to be containerized by following industry best practices. To be able to make LLRP.droid containerized, there was created an application image. To build an image, it is necessary to create Dockerfile prior and put it into the root of a project directory. Inside Dockerfile there is a mandatory field with the base image details. Then we built the containerized application LLRP.droid by the command: `docker build -t <android.LLRP.droid>`.

In details we followed this process:

- installed prerequisites, tools and libraries,
- installed Android SDK development environment and its update,
- set SDK environment variables,
- prepared WORKDIR for Android (assets, bin, gen, libs, res, src),
- copied Android application over into the Docker container for compilation,
- built the Android application within the Docker container.

2.2. Containerized Object Name Service

Another RFID component after middleware and supposed to be containerized/tested was the Object name service (ONS) as described in GS1 standard [12]. ONS is one of the key heterogeneous RFID ecosystems components. In our experiment the ONS is a combination of Knot 2.0 DNS, which is responsible for a name authority pointer (NAPTR) responses. DNS design specification and recommendation for containerization are available in [13]. Knot Domain name service (DNS) is an authoritative DNS server which supports key features of the domain name system including DNSSEC.

Containerized DNS daemon is in interaction with containerized Nginx implementation (responsible to route client via http protocol within NAPTR response) of Electronic product code information system (EPCis). EPCis is web based information system for RFID/IoT ecosystems.

Table 1 shows a comparison of response time of ONS running in native and containerized environment.

Tab. 1: ONS responsiveness.

	Load time [s]	**First byte [s]**	**Speed index**
Native			
first	0.498	0.463	594
repeated	0.07	0.317	71
Container			
first	0.546	0.481	696
repeated	0.138	0.307	140

```
CONTAINER      IMAGE            COMMAND
789cbdbd0dd4   lk/ONS           "knotd"

CREATED    STATUS    NAMES
2min ago   Up 2min   stoic_payne

CONTAINER      IMAGE            COMMAND
8ba00375bbb4   lk/ONS-resolver  "kresolved"

CREATED    STATUS    NAMES
2min ago   Up 2min   focused_carson
```

Fig. 3: Containerized DNS components of ONS service.

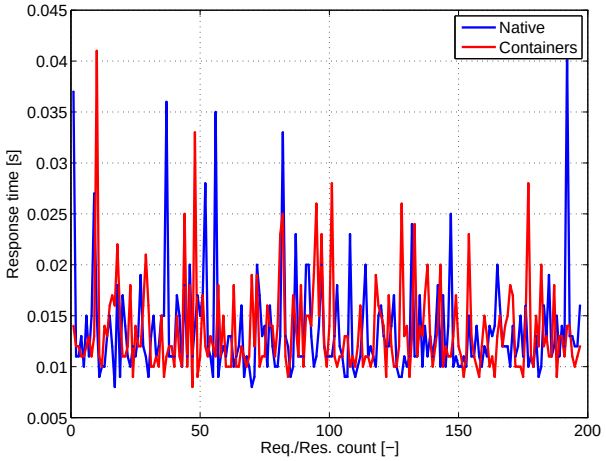

Fig. 4: EPCis responses.

In the Fig. 3 you can see DNS components running containerized service on the target platform.

Figure 4 depicts a graphical representation of response time visualization over count of 200 requests/responses processed by combined native and containerized EPCis service. We can see very little difference in latencies which could mean minimum network performance negative impact.

Figure 5 shows the main impact by the containerization brought by DNS container, one component of whole ONS solution. Native DNS responses are without major deviations, although the containerized DNS show the performance decrease. NAPTR query count was set to 50.

Benchmarking utilities used for measuring web server performance are: **httperf and ab suites:**

```
httperf –server native/container –port
80 –uri /page.html –rate 150 –num-conn
10000 –num-call 1

ab –kc 20 –t 60 http://server
native/container /page.html.
```

Containerized EPCis is operated by: `docker run –d –p 80:80 ngingx epcis.`

Containerized ONS is operated by: `docker run –p 53:53 knot knotd.`

3. Benefits

Containers main benefits are:

- speed - container can start in less than one second, development and deployment time is shortened to minimum,
- consistency and integrity - each container encapsulates only those components which are needed and protect them from altering, it minimizes risk of bug and malicious code,
- density and resource efficiency - minimizing of environment allows focus on resource efficiency),
- flexibility in delivery and operations - portability achieved by container pull approach instead of huge complex installation processes.

4. Security Impacts

Based on the previous experiments we can do particular conclusion describing the security impact of tested technology. Each Docker-based container can be equipped with measures which will prevent to launch unsigned (assured integrity) software content. The environment is protected and isolated as the whole unit.

There is a minimum set of interfaces towards outside world of the underlying Docker engine and operating system. We can achieve prevention of the attacks as discussed in [14] by its combination with SElinux and AppArmor. Containers do not prevent to use advanced comprehensive authentication techniques which

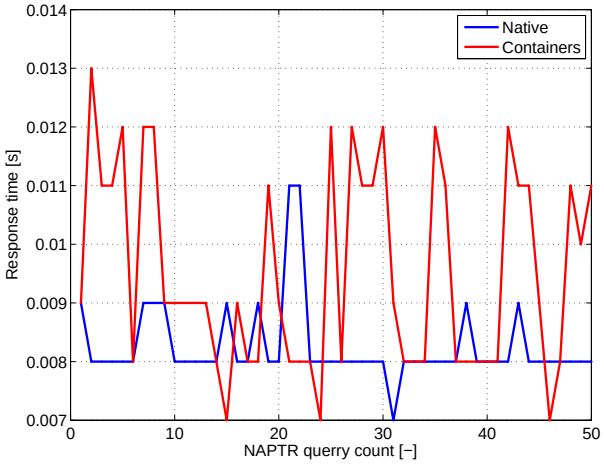

Fig. 5: ONS responses.

is the advantage for future enhancement with certification authority functionalities. By the containerization we do not lose contextual awareness of security and privacy in RFID systems how it is described in [15].

Research communities share a plenty of security issues and problems which are being continuously addressed by developers of containerized technologies like Docker although the RFID implementation is very specific and we had to consider security aspects during re-design and proposals of feasibility tests.

By simple mistakes there could be compromised systems or data processed by any component of RFID system. The execution development API offers many advantages and introduced tools like chroot and jails which bring higher level of security, but were not utilized here.

To use cryptography, we should always consider possible entropy problem inside containers. Others are the problems regarding logging and monitoring due to isolation limitations. In general containers helps to prevent and secure against:

- denial of service attacks,
- privilege escalations in and out of containers,
- not authorized integrity changes,
- processed data leakages.

Container adds layer based security, blocks intention to get out of the container, and increases privileges safety inside the container. Containers/layer are new barriers which the attacker has to overcome. Images we were pulling from the registry have been verified with hash algorithm processed on both ends of the communication channel with used digest of SHA256.

5. Future Work

The future work should be split into two parts. The first part will be focused on the unification of containerization method of on-premises approach based software solution components, plus identification of other containerization capable RFID ecosystem components.

Second part is supposed to tune the way of simulation containerized virtualization with focus on improvement of measurable performance indicators.

With this knowledge and experience based on this proof of concept to build whole RFID ecosystem and get it ready for future built, ship and operate quick approach. This approach enables efficient testing and rapid system development. We proved that these experiments are doable to be followed up with respecting best practices and described limitations.

6. Conclusion

We presented this feasibility study, where we did an assessment of proposed approach of containerization of usual RFID components. This feasibility study consisted of an evaluation of impact plus analysis, and it delivers confirmation as the results of a correspondence with proposed initially described problems.

Our impact analysis included these steps: preparation and identification of main impact areas like enumeration as impact on event processing efficiency, security (availability, confidentiality and data/records integrity), assessment and evaluation impact.

First question was fulfilled by the feasibility study extended with detailed description of realization. Second one was covered by the impact analysis, with result as a proof that containerization has no serious negative efficiency, negative performance and negative security impact.

Containerization is opening new space for advanced threat analytic to be implemented into heterogeneous containerized ecosystems.

Acknowledgment

This work was supported by grant in EUREKA Cluster program with EUREKA7592 AutoEpcis project and by Student grant at Czech Technical University in Prague SGS14/141/OHK3/2T/13.

References

[1] DUA, R., A. R. RAJA and D. KAKADIA. Virtualization vs Containerization to Support PaaS. In: *IEEE International Conference on Cloud Engineering (IC2E)*. Boston: IEEE, 2014, pp. 610–614. ISBN 978-1-4799-3766-0. DOI: 10.1109/IC2E.2014.41.

[2] Docker on AWS Running Containers in the Cloud. *Amazon: web services* [online]. Available at: `http://docs.aws.amazon.com/AmazonECS/latest/developerguide/Welcome.html`.

[3] SLOMINSKI, A., V. MUTHUSAMY and R. KHALAF. Building a Multi-tenant Cloud Service from Legacy Code with Docker Containers. In: *IEEE International Conference on Cloud Engineering (IC2E)*. Tempe: IEEE, 2015, pp. 394–396. ISBN 978-1-4799-8218-9 DOI: 10.1109/IC2E.2015.66.

[4] SONGYAN, J., D. WANG and B. DU. Research and Development of QoS for RFID Middleware. In: *Second International Conference on Computer Modeling and Simulation*. Kathmandu: IEEE, 2010, pp. 397–402. ISBN 978-1-4244-4569-1. DOI: 10.1109/AHICI.2009.5340325.

[5] DING, G.-J., F.-G. LIU, Y.-X. RUAN and Y.-D. LIN. AN RFID-middleware-based RFID router architecture. *International Conference on Machine Learning and Cybernetics (ICMLC)*. Xian: IEEE, 2012, pp. 591–595. ISBN 978-1-4673-1484-8. DOI: 10.1109/ICMLC.2012.6358989.

[6] BONIFACE, M., B. NASSER, J. PAPAY, S. C. PHILLIPS, A. SERVIN, Y. XIAOYU, Z. ZLATEV, S. V. GOGOUVIS, G. KATSAROS, K. KONSTANTELI, G. KOUSIOURIS, A. MENYCHTAS and D. KYRIAZIS. Platform-as-a-Service Architecture for Real-Time Quality of Service Management in Clouds. In: *Fifth International Conference on Internet and Web Applications and Services (ICIW)*. Barcelona: IEEE, 2010, pp. 155–160. ISBN 978-1-4244-6728-0. DOI: 10.1109/ICIW.2010.91.

[7] SANDIKKAYA, M. T. and A. E. HARMANCI. Security Problems of Platform-as-a-Service (PaaS) Clouds and Practical Solutions to the Problems. In: *IEEE 31st Symposium on Reliable Distributed Systems (SRDS)*. Irvine: IEEE, 2012, pp. 463–468. ISBN 978-1-4673-2397-0. DOI: 10.1109/SRDS.2012.84.

[8] MALAN, J. D. Secure Execution of Untrusted Code. In: *Special Interest Group on Computer Science Education (SIGCSE)*. Massachusetts: ACM, 2013, pp. 141–146. ISBN 978-1-4503-1868-6.

[9] TURNBULL, J. *The Docker Book: Containerization is the new virtualization*. New York: Amazon, 2014. ISBN 978-0-9888202-0-3.

[10] FELTER, W., A. FERREIRA, R. RAJAMONY and J. RUBIO. An Updated Performance Comparison of Virtual Machines and Linux Containers. *IBM Research report RC25482*, 2014.

[11] HANE, O. *Build Your Own PaaS with Docker*. Birmingham: Packt Publishing, 2015. ISBN 978-1-78439-394-6.

[12] Object Name Service (ONS). *GS1: The Global Language of Business* [online]. 2014. Available at: `http://www.gs1.org/epcis/epcis-ons/2-0-1`.

[13] Knot DNS. *Docker Hub* [online]. 2015. Available at: `https://registry.hub.docker.com/u/cznic/knot/`.

[14] LIN, R., B. WU, S. SU, P. XU, S. YANG and Y. ZHAO. A Security PaaS Container with a Customized JVM. In: *IEEE 7th International Conference on Cloud Computing (CLOUD)*. Anchorage: IEEE, 2014, pp. 960–961. ISBN 978-1-4799-5062-1. DOI: 10.1109/CLOUD.2014.142.

[15] PATERIYA, R. K. and S. SHARMA. The Evolution of RFID Security and Privacy: A Research Survey. In: *International Conference on Communication Systems and Network Technologies (CSNT)*. Katra, Jammu: IEEE, 2011, pp. 115–119. ISBN 978-1-4577-0543-4. DOI: 10.1109/CSNT.2011.31.

About Authors

Lukas KYPUS is a Ph.D. student since 2013 in Center of Automatic Identification and RFID lab at the Czech Technical University in Prague. He received his M.Sc. degree on faculty of electrical engineer and communication at Brno University of Technology. Currently his research interests are RFID/NFC technologies and information/cyber security. He works on thesis in area of RFID/IoT security.

Lukas VOJTECH received M.Sc. and Ph.D. at Telecommunication Engineering at the Czech Technical University in Prague, Czech Republic in 2003 and 2010. He has been actively involved in several national and international projects. He is a leader of RFID laboratory at the Czech Technical University in Prague since 2010. He is an author or co-author of more than 50 papers in international journals, conferences and results of applied research. His research interests are hardware prototyping and measurement especially in the field of RFID technology, textile antenna design and localization.

Marek NERUDA is a post doctoral researcher at Czech Technical University in Prague, Czech Republic focused on area of smart fabrics and textiles.

Tomas ZITTA is a student of master's study program at Telecommunication Engineering at the Czech Technical University in Prague, Czech Republic focused on mobile application development.