SMART II: Android apps, cloud computing and mobile device management as enablers for efficient operations

Frank T. Johnsen[a] and Ida M. Frøseth[b]

[a]Norwegian Defence Research Establishment (FFI), Kjeller, Norway
[b]The Norwegian Defence University College, Lillehammer, Norway

ABSTRACT

"SMART" – pervasive situational awareness at the individual soldier level – was a Concept Development and Experimentation (CD&E) project carried out in Norway during 2016. The concept being tested was the use of smart technology as a cheap and low-complexity platform for collaboration and situational awareness for the Norwegian home guard (HV). SMART included building a prototype based on the Android platform. The prototype was tested in several field trials. In general, the results from the activity strongly indicated that using civilian smart technology yields an operational value. The SMART prototype and concept can provide cheap and low-complexity communications to HV and others who may need this capability. The prototype has since 2016 further evolved, and was used in another CD&E project in 2018, the so-called "SMART II". Here, we focused on necessary communication and computing infrastructure support for forces using smart devices to establish their common operational picture. As part of the experiments, we evaluated approaches such as cloud computing and mobile device management (MDM) when deploying and using the software. HV tested the prototype extensively in 2018, with the culmination being its use during the Trident Juncture exercise by one of HV's rapid response forces. In this paper, we give an overview of our prototype, with emphasis on technology and our experiences with the supporting infrastructure tools.

Keywords: Cloud computing, Mobile device management, Android, Situational awareness

Track 8: Information and Knowledge Systems

Paper ID 17

Point of contact

Frank T. Johnsen
Norwegian Defence Research Establishment (FFI)
P.O. Box 25, 2027 Kjeller, Norway
E-mail: Frank-Trethan.Johnsen@ffi.no

1. INTRODUCTION

At the Norwegian Defence Research Establishment (FFI) we are researching the *mobile complex*[1] for military use. In this paper, we explore the mobile complex from a technology perspective, as opposed to the *socio-cognitive perspective*.[2] In short, the *technology perspective* of the mobile complex is the eco-system arising around smart devices and the networks such devices utilize (e.g., the mobile Internet).

Several initiatives have examined the use of smart technology in military operations, one of which was the Concept Development and Experimentation (CD&E) "SMART" project in 2016.[3] It differs from other military smart experiments by focusing on users with limited resources and time for training, primarily the Norwegian home guard (HV). The objective of the SMART project was to evaluate the operational value of using smart technology as a tool for improving situational awareness at the squad level.

1.1 The Norwegian home guard

HV is one the branches of the Norwegian Armed Forces. Their main responsibility is territorial protection. HV consist of 40,000 soldiers distributed in 4 regions, 11 districts and more than 200 areas covering all of Norway.[4] HV constitutes a mobilization force, meaning that that the soldiers are not employed by the military, but train on a regular basis. HV is organized as 12 Rapid-reaction Intervention Forces (ISTY) and 220 Area forces. While the ISTY are capable of rapid response and consist of highly trained and equipped personnel, the Area forces have longer reaction time, are less equipped and the personnel have less training. The SMART project aimed at providing a secondary data-channel communication system for the Area forces, hence the concept had to be cheap and require as little training and management as possible.[3]

As for any military unit, the command structure within an Area force is hierarchical, see Figure 1. The smallest unit is a squad. Squads usually consist of eight squad members. Each squad member has a designated task, and the squad is led by a squad leader (LF), who is supported by a second-in command (NLF). Three to five squads are organized in one platoon, and three two five platoons are organized in one Area. Each organizational level has one commander and one second in command. Additional roles like intelligence, operations, logistics, plans etc. are found at the Area level and up. The typical HV soldier is a person who is 18-55 years old, and has at a minimum fulfilled one year of mandatory conscription. This applies to both soldiers on the squad and platoon level. The commanders at each level usually have more training than the average soldier.

In the SMART prototype, we aimed to support the Area force command structure. So, we designed a system to separate concerns such that each area would have its own server. The area command would have ownership of the operational picture, and full control over the server via a designated administrator's interface. Soldiers

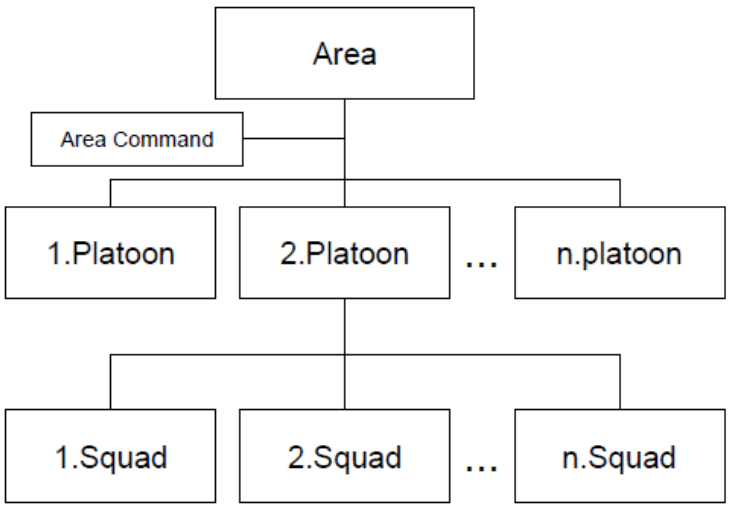


Figure 1: Area force command structure

in the field would be equipped with Android phones with a selection of applications (apps) to support their communication needs. It should be noted that while we in SMART in 2016 experimented with several different deployment approaches, we ended up recommending that phones should be made available down to the LF and NLF, but not the remaining squad members. This way of equipping soldiers (down to and including the NLF) was also used in "SMART II" in 2018, the follow-on CD&E project to "SMART".

1.2 SMART concept prototype

Through the SMART CD&E, a prototype system that we at FFI referred to as *Titans*, was developed. Titans included a back-end called *Athena* providing a Representational State Transfer (REST)[5] service interface, a web-based management user interface called *Metis* (to be used by the area command) and an Android app called *CAGED*, short for *Communication Application with Geographical Element Data*. For persistence, PostgreSQL was used on the server, and SQLite was used on Android. Since CAGED was the main development of the project that was visible to the soldiers, the prototype came to be referred to as *CAGED* by HV personnel, whereas software developers continued using the Titans code name. Figure 2 gives an overview of the prototype architecture.

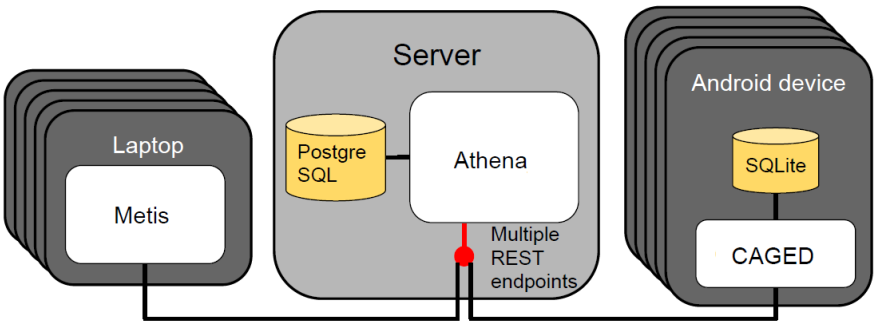
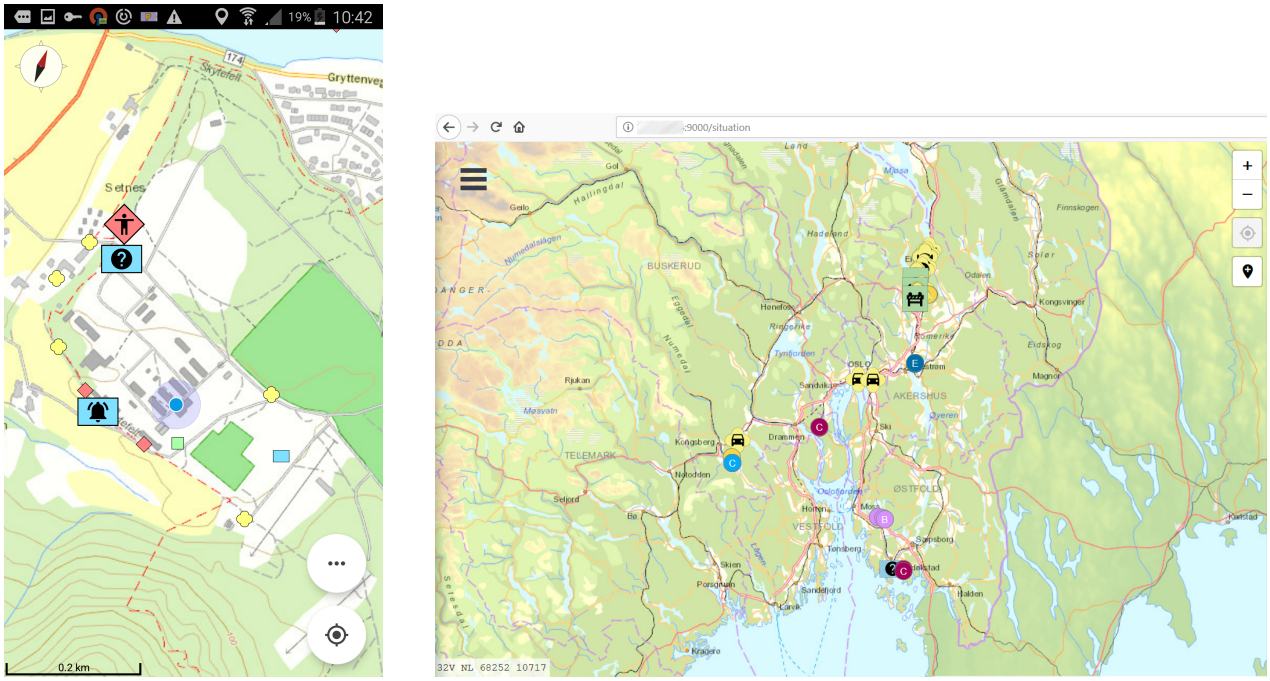


Figure 2: Titans overview

Third-party software was also used to enable text based instant messaging (aka *chat*) and to add transmission confidentiality with a Virtual Private Network (VPN). The CAGED app was the main user interface for the squad and platoon members and enabled them to share information related to the common understanding of the situation. The app provides a map where all the other users locations are displayed. Various observations added by the users are also shown. The users could communicate through observations by writing comments or adding pictures. The location of a user was logged with the smart device Global Positioning System (GPS) sensor, and the data was shared among the users through the REST server. Following SMART, Titans was updated according to user feedback, and especially CAGED was improved. In particular, the *user interface* was improved, the *map* was improved (switching from Open Streetmap to a Norwegian national map provider offering more detail in rural areas), and the GPS *positioning algorithm* was improved.[6] This improved the overall user experience of the prototype in preparation for the SMART II experiments. Further, the new positioning algorithm allowed for up to 50% less power consumption than the algorithm we had used in the SMART experiment.[7] A screenshot from this new, improved version of CAGED is shown in Figure 3. This version was the basis for the experiments in SMART II in 2018, as discussed in this paper. In SMART II, the main focus of field trials was on ease of management and software deployment, rather than on app development as it had been in SMART in 2016. Hence, SMART II reused the prototype described here, but leveraging *Cloud computing* and *Mobile Device Management* for added benefits.

1.3 Outline

The remainder of this paper is organized as follows: Section 2 discusses previous and related work. For those new to the terms Cloud computing and Mobile Device Management (MDM), Section 3 introduces these concepts. The experiments we conducted in SMART II and lessons learned from them are summarized in Section 4. Finally, Section 5 concludes the paper.



(a) CAGED main user interface (b) Metis main user interface

Figure 3: Android (CAGED) and PC/Web (Metis) screenshots.

2. RELATED WORK

The Tactical Ground Reporting (TIGR) system developed by Darpa has been in use by the U.S. Army since 2007.[8] It is a system for storing, and sharing tactical information on company level and below, and it is designed to work in disconnected, intermittent and limited (DIL) network environments. The system has properties like horizontal information flow, web technology, simple data models, and it is inspired by social media.

ATAK (sometimes referred to as the *Android Tactical Assault Kit* or the *Android Team Awareness Kit*) is an Android app providing map functionality, a user interface providing the operational picture and other functionality like a Geo-spatial infrastructure.[9,10] The list of functionality includes online and offline maps, sharing positioning information, navigation, communication and specific military functionality.[11] In addition to being a fielded system, ATAK is also used in demonstrations and experiments, for example in the context of the recently concluded NATO research task group IST-147 "Military application of Internet of Things". There, ATAK was used to show integration of military assets along with civilian sensor information as provided by smart cities and their open APIs.[12]

The Platform Independent Sensor Application (PISA),[13] was a prototype developed by FFI. It was a bi-platform (targeting Android and iOS) app, enabling position- and observation reports in DIL environments. The app provided only one-way communications (soldier to HQ) so that it could be used also in bring your own device (BYOD) deployments where there was little or no trust in the platform. This allowed everyone to share information with the HQ, but the HQ could not share information with the soldiers using this app.

Kings Eye[14] was the evolution of PISA to a bi-directional app, allowing soldiers to share with the HQ and vice versa. It was a prototype of a data centric social tactical reporting system, never put in operational use, but tested with BYOD in an FFI experiment. The system was inspired by TIGR and how TIGR is designed to build situational awareness through horizontal sharing of information about experiences in an operational area. Experiments with PISA and Kings Eye have drawn attention to platform independence, intuitive user interfaces, appropriate security and robust communication as critical BYOD enablers to military uses. However, platform independence was shown also to come at a great cost when developing the app: One needed to maintain a platform-independent basis, but still add on and maintain certain platform specific modules to leverage platform

specific capacities. These experiences led us to adopting more of an ATAK inspired approach for the SMART prototype, where we concentrated our efforts on the Android platform. Hence, we ended up with the platform-specific approach of CAGED described earlier in this paper.

FACNAV Mobile is a hand held command and control information system.[15] It is a recent development by the Norwegian company Teleplan Globe, replacing their old Dina system that was specific to Windows mobile. FACNAV mobile is implemented as a multi-platform system. It was not available on the market when we started the SMART series of experiments, otherwise we would have considered using this app in our trials rather than building CAGED ourselves. CAGED and FACNAV Mobile have similar feature sets, but where CAGED is geared more towards ease of use, FACNAV Mobile targets professional soldiers and promotes interoperability to a larger extent. It supports NATO Friendly Force Identification (NFFI) (STANAG 5527) and also Variable Message Format (VMF) (MIL-STD-2045-47001 and MIL-STD-6017). It should be mentioned that there exist many different commercial alternatives on the market these days, both by domestic and international vendors, so we present FACNAV Mobile here only as an example and do not intend to promote it in any way, as we do not have any hands-on experiences with this system.

3. A BRIEF INTRODUCTION TO CLOUD AND MDM

Cloud computing is a trend in the time, often provided as a service by companies selling computing resources to consumers. One of the major reasons for the popularity of cloud computing services is that you do not need to own your own server, you can just buy the resources you need from someone else. This may seem like good old server park hosting or renting access to large mainframe computers, but cloud computing also involves other aspects that together make this a new, cost-efficient and very powerful way of deploying your software. An overview of these aspects is given below in Section 3.1.

While cloud computing addresses the server side of a system, there is also a need to gain tight control over the client devices owned by an enterprise. The main approach to such control is using one of the many available MDM solutions on the market. MDM, among other things, enrolls smart devices (e.g., Android phones) under a set of policies governed centrally by the enterprise. An overview of the features offered by MDM solutions is provided in Section 3.2.

3.1 Cloud computing

Cloud computing arose more than a decade ago as a more rational approach to server deployment than self-hosting, which requires a lot of expertise and capital expenses to buy hardware. By renting the resources from someone else, you merely pay as you go and shift your need from capital expenses to operational expenses. This is beneficial for new, small company startups, since it allows them to get up and running on the Internet without having much capital to get started. As cloud computing has matured over the years, there are now many flavors, depending on the needs of the customer. Figure 4 is adapted from the National Institute of Standards (NIST) definition of cloud computing,[16] with some highlighting added to show the properties of cloud computing we found to be particularly useful to us.

As the figure shows, Cloud computing can be divided into several categories of aspects: *Deployment models*, *Service models*, *Essential characteristics*, and *Common characteristics*. Let us explore each of these aspects in turn.

3.1.1 Deployment models

The Deployment models illustrate different approaches to the way the Cloud computing infrastructure is deployed and hosted. In the case of a *Public Cloud*, the resources are available to anyone who can pay for access. Public Cloud services are offered by major vendors like Amazon, Microsoft and Google, and also a lot of national and multi-national smaller companies. Of the three deployment models, it is usually the cheapest alternative for the customer to choose, and also the most readily available.

A *Community Cloud* is Cloud computing restricted to a particular community. For example, a university may have Cloud computing resources made available to staff and students for use in computer science teaching. So, it is available to a certain community (those affiliated with the university), but none outside the community may

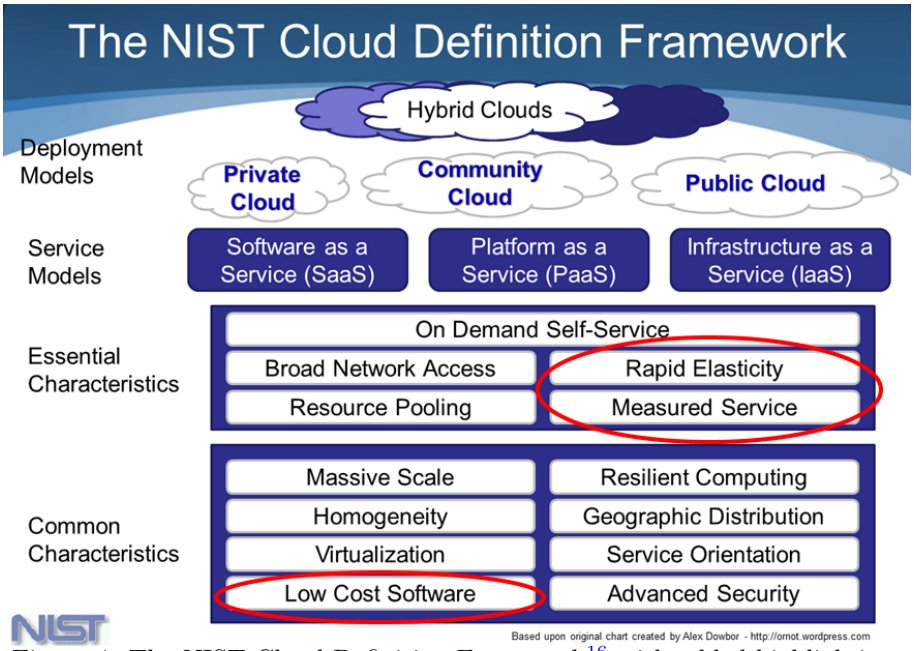


Figure 4: The NIST Cloud Definition Framework[16] with added highlighting.

deploy their services there. Another example is considering NATO member nations as a community, and offering Cloud computing resources to them. This approach is taken by the Joint Force Training Centre (JFTC), which hosts a virtualized infrastructure. It is possible for nations participating in the Coalition Warrior Interoperability eXploration, eXperimentation, eXamination eXercise (CWIX) to bring virtual machines, load these on the JFTC infrastructure and leverage these virtualized computing resources during the exercise.

A *Private Cloud* is private to the entity using it. It may be a self-hosted Cloud computing approach for use by an enterprise, or it can be a so-called *Virtual Private Cloud* hosted externally by a Cloud providing company, but with dedicated resources to the customer and additional security measures like e.g., running on a dedicated server (so no multi-tenancy, which is often considered a security issue[17]). Then, offering VPN access along with a leased line with guaranteed capacity to the resources, making the Private cloud a trusted part of the enterprise network. Private cloud is usually the most expensive option of Cloud computing, but also gives the best control over the data, processing resources and network communication.

Finally, there is the possibility of *Hybrid Clouds*. Basically, this means using Cloud deployments that span more than one of the above categories. For example, a government may choose to use a Private cloud approach to contain sensitive data, while information to the general public can be offered from a Public cloud.

3.1.2 Service models

The choice of service model is orthogonal to the choice of deployment model that we discussed above, meaning that any service model in principle can be combined with any deployment model.

The most basic service is *Infrastructure as a Service* (IaaS), where the provider hosts the infrastructure (so all necessary hardware for processing, storage and networking, and provides you with a virtualized set of resources to use). The provider may offer you a choice of different operating systems (OSs) to use on the IaaS, for example Windows or one or more flavors of Linux. The customer will have to maintain the OS (security patches, etc) and also any software installed along with the OS. IaaS can be had as a low cost service, but the customer must be ready to maintain the installation on the infrastructure that is offered.

A more elaborate approach is *Platform as a Service* (PaaS), where the provider hosts the infrastructure and the OS, hence providing a maintained platform to the customer. The provider is responsible for providing a stable OS and a stable infrastructure. The customer can then use this platform to install own software, and

is limited to only having to maintain the software and not other components. Since this service includes more hosted management than IaaS, PaaS is usually a more costly option to choose for the customer. On the other hand, less security expertise and management is needed on the part of the customer.

Finally, *Software as a Service* (SaaS) is the most expensive but possibly also the most convenient service model for a customer. Here, the provider does the same as in PaaS, but in addition also provides specific, maintained software on the provided platform. The customer only needs to specify the software that is needed, and the provider will handle the rest. An example of a potential customer class here is web bloggers, who are interested in using Wordpress[18] to drive their blogs but do not want to be concerned with maintaining servers, OSs or software, only writing their articles and publishing them online. That said, examples exist of SaaS being offered for free to consumer, like the many online e-mail services currently being offered (e.g., Gmail and Hotmail). Instead of charging the user directly, these are financed indirectly through indexing the users' e-mails, building user profiles and providing them with targeted commercials.

3.1.3 Essential characteristics

The essential characteristics constitute the set of properties that is considered essential to calling a particular system "Cloud". These characteristics are what distinguish today's Cloud computing from the leased servers and mainframes of previous times.

On-Demand Self-Service allows the customer to self-provision resources in the Cloud. An example here is an online Cloud provider that provides a Web interface where a customer may log in and assign resources for use without needing more than registering a PayPal account or credit card information. This approach is common to many Public Cloud providers, where the low overhead of starting to use their services is considered a business advantage. Other service models usually require a more elaborate and less automated approach to self-service, in that one usually needs to get a legally binding Service Level Agreement (SLA) in place for such services. Nevertheless, when the SLA is in place, the customer may either provision through customer service, or a self-provisioning web portal or other, automated means (e.g., Terraform[19]). That said, a SLA may also be needed for Public Cloud, depending on the Cloud provider in question.

The characteristics *Rapid Elasticity* and *Measured Service* go hand in hand, in that you scale up to as much resources as you need and pay for them as you go. A Cloud provider can scale your resources rapidly, since it is a matter of assigning more virtual resources to you out of the pool they have. This can be done much faster than having to add on physical hardware resources in an actual on-premise server room as in previous times. The price you have to pay is proportional to the amount of resources you consume, and can be measured using metrics like processing (number of CPU cores used), memory (number of Gigabytes of RAM used), storage (Gigabytes of virtual storage used), and network traffic (either total use, sustained bandwidth, or both).

From the provider's viewpoint, offering *Broad Network Access* and using *Resource Pooling* is important. Accessibility allows customers to use services from all over the world, whereas resource pooling makes the most out of the hardware resources the company has on hand.

3.1.4 Common characteristics

The common characteristics are often found across Cloud providers, as these are either characteristics that are in-demand from customers (like low cost and resilient computing) or help the providers offer a competitive and profitable service (like virtualization and service orientation). Of importance to both provider and customer is the advanced security measures a provider can employ to protect the Cloud computing facilities. The physical and cyber security measures put in place by certain providers are quite elaborate, and better than an average small company would be able to employ themselves.[20]

3.2 MDM

MDM allows for centralized management of a fleet of mobile devices. The MDM software solution (server side) can be installed either on-premise, or offered as SaaS. Regardless of how the server side is deployed, the MDM imposes several restrictions on the devices that are enrolled. This means potentially less freedom of use for the consumer, but necessarily more control (and hence trust in the end-devices) for the enterprise. Concerning mobile devices, there exist several different ownership models. Each such model may, or may not, work with

MDM. To clarify this aspect, we first look into ownership models, before discussing what MDM can and cannot do.

3.2.1 Ownership models

There are different ownership models that all have benefits and drawbacks. These models can be explored from different perspectives, and, as can be seen in Figure 5, more secure approaches are also more expensive.

OVERVIEW OF DIFFERENT OWNERSHIP MODELS				
	USER OWNERSHIP	SHARED OWNERSHIP	ENTERPRISE OWNERSHIP	CUSTOM SOLUTION
TRUST IN SECURITY MECHANISMS	Low	Low-Medium	Medium-High	High
SCOPE OF ENFORCED SECURITY	Single Application	Work Partition	Application Layer And Platform Configuration	Kernel And Above (possibly pre-boot)
ACQUISITION COST	Low	Low-Medium	Medium-High	High
MAINTENANCE COST	Low	Low-Medium	Medium-High	High
PORTABILITY	High	Medium-Low	Medium	Low

Figure 5: Summary of the pros and cons of the different ownership models.[21]

As the figure shows, ownership models where the user either owns or shares ownership with the enterprise are overall the cheapest approaches, both in terms of acquisition costs and also maintenance costs. These can be categorized as BYOD. While desirable from a cost perspective, such approaches usually leave a high degree of control to the user, with little means to employ advanced security measures. This again means that one can have no (or very little) trust in the platform under such ownership models. In SMART, we explored the BYOD approach as discussed in the experiment report.[3] Conversely, in SMART II, we explored the enterprise ownership approach to be able to enforce security policies not only for a single application, but for all applications and also the platform itself. The latter category in the figure, that of a *custom solution* was considered too costly and thus outside the scope of solutions that are viable for HV. For further discussions about security aspects within each ownership model, please refer to the full report on modern mobile platforms from a security perspective.[21]

3.2.2 Managing mobile devices

MDM can be defined as follows:[22]

> Mobile Device Management software are tools that are used for deploying, monitoring, securing and managing the smartphones, tablets and laptops in the workplace that are involved in the distribution of application data and configuration settings.

MDM can, in essence, be combined with any of the ownership models that we discussed in the previous section. It is, however, important to note that depending on the ownership model, not all features may be fully supported. For example, in the case of *shared ownership*, it is typical that the MDM software takes partial control of the device, enforcing security in a so-called *work partition*. Conversely, under *enterprise ownership*, the MDM software may take full control of the device, severely limiting how the user may interact with it. Examples here

include limiting applications that may be installed, enforcing VPN to always be enabled, etc. Under enterprise ownership the MDM solution may also remotely wipe the entire phone if necessary (for example if the employee loses it). Conversely, under shared ownership, the MDM software may in such circumstances only wipe the work partition. Figure 6 gives an overview of the feature set typical MDM software provide. The figure was fetched from an AlignMinds post.[22]

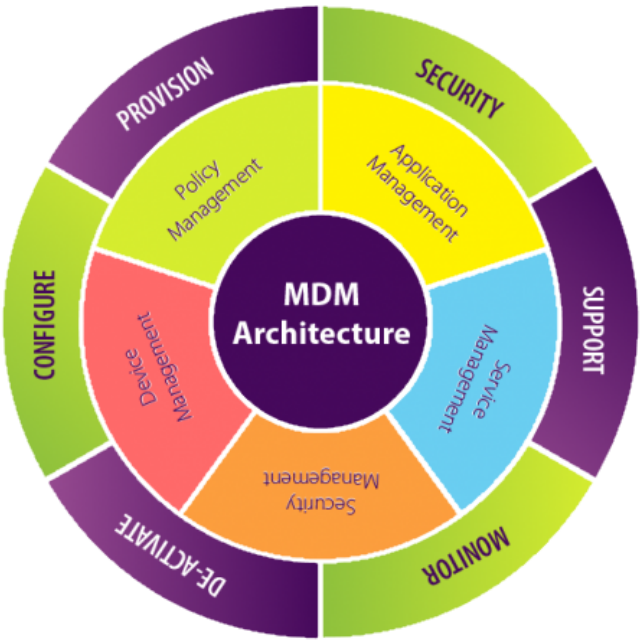


Figure 6: Typical MDM feature set.[22]

Figure 7 gives an overview of currently available MDM software. The overview was fetched from a PC Quest article.[23] All the solutions shown in the figure are readily available, and the MDM solution we used in SMART II is included in this overview. As the figure shows, the feature sets of these solutions are similar (almost equal, one might say), so the product we actually used shall remain nameless in the interest of not promoting any specific vendor or solution.

The way we employed MDM in SMART II was to use the MDM software (provided as SaaS) to enroll all the enterprise owned phones. This was done prior to deployment in the exercise, in that we factory reset each phone and then immediately enrolled it as an Android enterprise device. Once enrolled, we were able to enforce the security measures we deemed necessary: VPN always on and allowing only white-listed apps (which basically meant that only apps pre-approved by us were available in the app store). In the following section we discuss further details of the SMART II experiments.

4. SMART II EXPERIMENTS AND RESULTS

Both SMART and the follow-on SMART II projects were supported by the HV Training Centre ("Heimevernets skole- og kompetansesenter (HVSKS)"). During SMART in 2016, we established that using Titans, the SMART concept prototype, yielded an operational value:[3] The soldiers were able to get a better understanding of the situation, which again led to quicker and better decision-making.

The main drawback was the limited battery capacity of the mobile devices used. Hence, for SMART II we ensured that every phone came with a wall-plug charger. Naturally, not all deployments will be able to make use of a wall-plug charger, so we also issued pre-charged powerbanks together with the phones as well as cigarette-plug type chargers for vehicles. For the main exercise, i.e., Trident Juncture (TRJE), we were also able to obtain a few vehicle chargers that fitted the military vehicles (the normal cigarette-plug style charger cannot be used

Vendor		WSO2	Airwatch B2	BlackBerry	Citrix	Fiberlink (an IBM company)	MobileIron	SAP	Soti	Symantec
Product Name(s)		WSO2 Enterprise Mobility Manager	AirWatch Enterprise Mobility Management	BlackBerry Enterprise Service 10	XenMobile	MaaS360 by Fiberlink	MobileIron	SAP Afaria, SAP Mobile Documents, SAP Enterprise Store	MobiControl	ymantec Mobile Management Suite
Type		Open source	Commercial	Commercial	Commercial	Commercial	Commercial	Commercial	Commercial	Commercial
Deployement Options	On Premises	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes
	Cloud/SaaS	Yes	Yes	Version in Development	Yes	Yes	Yes	Yes	Yes	Yes
Mobile Platform Supported	Android	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	iOS	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	Windows Mobile	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	Blackberry	No	Yes	No	Yes	Yes	Yes	Yes	No	Yes
Mobile Device Management	Password Protection	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	Password Reset	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	Remote Device Wipe	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	Remote Lock	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	Disable Wi-Fi	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	Disable Camera	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	Disable Bluetooth	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Security	Secure Web Browser	N/A	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
	Application Blacklisting/ Whitelisting	N/A	Yes	Blacklisting Only	Yes	Yes	Yes	Yes	Yes	Yes
	Data Loss Prevention (Dlp)	N/A	Yes, Via Advanced Device Security Features And Airwatch Secure Content Locker	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	Encrypted Email Attachments	N/A	Yes	Yes	Yes	Yes	Yes	Yes; Also Supports Nitrodesk Touchdown Secure Mail Client for Android	Yes	Yes
Network Management	Data Usage Restriction WiFI or Celluluar	Yes	Yes	For Blackbery Devices Only	Yes	Yes	Yes	No	Yes	No
	Device Diagnostics	Yes	Yes	For Blackbery Devices Only	Yes	Yes	Yes	Yes	Yes	No
	Block Device from Accessing Email	N/A	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Reporting	Real-time Dashboards	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Figure 7: Comparison of nine MDM tools, both open source and commercial.[23]

there) – this allowed recharging phones and powerbanks in the field for the soldiers who were operating out of vehicles.

The goal of the SMART II experiments was to investigate Cloud computing and MDM in context of the mobile complex. Ideally, using these techniques should ease the load on the administrative part of deploying and maintaining Titans. In addition, they should ideally have no negative impact on the use of the system.

4.1 Experiments

Table 1 gives an overview of the experiments performed during SMART II. The experiments prior to TRJE should be considered technical trials where we tested Titans in different settings as preparation for TRJE, which was the main exercise and evaluation arena. Experiments were geared towards testing aspects of Cloud deployment models (Private cloud, Public cloud), and management aspects like a private app store (we used F-droid* for this) and MDM (with MDM, we used the built-in Android Enterprise app store rather than F-droid).

The three first experiments were performed with support from HVSKS. The fourth was performed with support from the Norwegian Cyber Defence and the Norwegian Defence University College as part of a thesis[24] for the Bachelor degree. In the first test, we set up and evaluated a private app store together with Titans. The setup was as illustrated in Figure 8.

*F-Droid is a software repository for Android system, similar to the Google Play store. We did not use the openly hosted repository for our experiments, but set up our own F-Droid repository in our cloud deployment, along with Openfire (open source XMPP chat server) and our own Athena. In doing this, we were able to provide a private app store for HV with pre-approved apps like CAGED and others. We also compiled our own F-droid app set to communicate with our F-Droid server rather than the open repository. For more on F-Droid, see: https://www.f-droid.org/

Test venue	Timeframe	Purpose
Course at HVSKS, Dombås, Norway	April 2018	Private cloud, F-droid app store
Exercise, Heistadmoen, Norway	August 2018	Public cloud, F-droid app store
TRJE with ISTY, Oppdal, Norway	November 2018	Private cloud, MDM
TRJE with Norwegan Cyber Defence, Northern Norway	November 2018	Public cloud and Docker

Table 1: Experiment activity in context of SMART II.

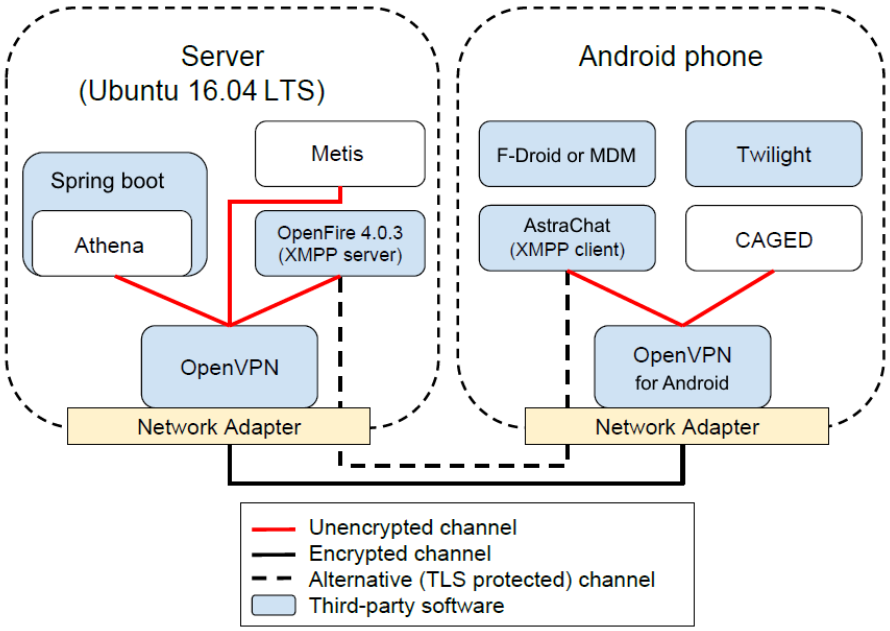


Figure 8: Server and Android phone deployment for experiments.

As the figure shows, we used a combination of third-party software and in-house developed solutions. Where possible, we used Free and Open-Source Software (FOSS) as the third-party solutions. Athena was built on the Java *Spring Boot*[25] framework. We used the FOSS *OpenFire*[26] as our chat server, and the FOSS *OpenVPN*[27] as our VPN solution in all experiments except the one including MDM. There, due to tight integration and ease of configuration, a commercial solution (offering the same features as OpenVPN) was used together with the MDM solution. During the course of the SMART experiment we tested several chat apps, and found that our users preferred AstraChat out of the freely available clients we tested from Google play store. Hence, we used AstraChat in all the SMART II experiments. That said, basically any XMPP-capable client can be used, since OpenFire implements the XMPP standard. *Twilight* was also freely available in Google Play store, and preserves night vision by applying a red filter on the screen and reducing screen intensity. In addition to these, either the F-Droid client (or the MDM software) was installed on the Android phones, depending on the specific experiment being performed. F-Droid was not needed together with MDM, since the MDM solution allowed us to control which apps were available via the built-in app store.

Security considerations for SMART II were as for the predecessor SMART, which can be summarized as follows:

- Communication must be protected to ensure data integrity and confidentiality.
- Data can be aggregated on the server, but should be disseminated to phones on a need-to-know basis.
- Data can be aggregated only on a per-deployment basis.
- Data should remain in Norway or in a NATO member country.

To accommodate these security requirements we protected communication using a combination of VPN and Transport Layer Security (TLS) as shown in Figure 8. The reason for exposing chat outside VPN protected only with TLS was to ensure that we could provide tech-support over chat also supporting any VPN issues. That said, with VPN active, also chat would be routed through the established VPN tunnel, benefiting from this added security measure. We set up OpenVPN with individual client certificates, enabling us to revoke the certificate of a certain device if it were lost. The server implemented fine grained access control, which allowed us to distribute data to users on a per squad level. This meant that if a device was lost, only data relevant to that particular squad would be on the phone and not the entire operational picture of the HV area. For the experiment with MDM, we also had the opportunity of remotely wiping any phone that was lost. As for the per-deployment aggregation, we adhered to that by wiping the database between experiments (if we reused the server) or installing a new server from scratch. As for the hosting of data, this was done in various ways in our experiments (see Table 1) For the Private cloud experiments we used the Information Infrastructure virtualized laboratory (INI lab) at FFI. This meant full control over the Cloud infrastructure, and ensuring that data stayed on our servers in Norway. For the Public cloud experiment at Heistadmoen we used a Norwegian cloud provider to host data. This provider offered several different deployment and service models, from which we chose Public cloud IaaS with the added restriction that data had to remain in Norway. For the Public cloud experiment in November, an international provider of Public cloud IaaS was chosen. This provider is multi-national and offers Cloud computing resources in many countries. For that experiment we used resources hosted in London, UK since it was a location that is both relatively close to Norway (considering latency over 4G Internet) and to fulfill the requirement of having data in a NATO member country.

4.2 Results

In the first experiment (April) several users reported on trouble using the system. They had problems logging on CAGED, and when logged on reports from others were spurious and updates were slow. These results did not inspire confidence in our Private cloud setup, which led us to investigate closer. These results were puzzling at first, since we had performed pre-trials in 2017 where the system functioned as it should using the same infrastructure. However, we found that prior to this experiment our INI lab had received upgrades and security patches. Among the patches were means to mitigate the *Spectre* and *Meltdown* vulnerabilities[28] that recently had been discovered. The means necessary to mitigate these attacks are known to reduce performance of the software running on the server. So, the means necessary to increase the security of our lab had negatively impacted the user experience, due to having a performance impact on our virtualized infrastructure. Following this experiment we were able to get the system running as it had before by assigning more CPU cores (in essence, more computing power) to our VMs.

In the second experiment (August) the server was hosted by a Norwegian cloud provider. Here, we experienced no ill effects on the server side, and the users reported no significant technical problems (so no problems that could not be solved during the experiment via our tech-support chat channel). The app store we had set up worked well, and our users were able to install and update apps from it. Some did report, however, that they found it a bit confusing to have two different app stores on the phone (i.e., they *should* use F-Droid, but they still had and could use Google Play store).

The third experiment (November) had yet again the INI lab as the Private cloud with IaaS. This time the server side had enough processing power to accommodate the server load, and Titans performed admirably. The users were overall very satisfied with using the phones as a communications aid, and reported that the tool provided them with timely and relevant information. Also, the users found that using the Android Enterprise app store (essentially Google Play store, but only with our white-listed apps in it) was easy. Other than imposing a restriction on the apps they could install, the users did not notice any other impact of having the devices enrolled in MDM. From the administrative point of view, MDM made it significantly easier and also less time-consuming (see Table 2) to set up and manage the smart phones.

In the fourth experiment (November) two bachelor degree students at the Norwegian Defence University College, supervised by Frøseth and Johnsen, investigated applying Titans as a means to keep track of Norwegian Cyber Defence personnel during TRJE.[24] Specifically, they used the Titans demonstrator as-is, and deployed it in a IaaS Public cloud for use. In addition, the students performed comparative evaluation of setting up the

Experiment	Number of phones	MDM in use	Time spent on phone admin.
HVSKS	60	NO	Two days
Heistadmoen	25	NO	One day
Oppdal	20	*YES*	*Two hours*
Northern Norway	20	NO	One day

Table 2: Time spent setting up and administering phones for the experiments

demonstrator on IaaS from scratch, and setting up on IaaS from pre-made *Docker* images. Simply put, Docker is a software containerization platform, adding an extra abstraction layer to virualization when used together with Cloud computing.[29] As Table 3 shows, employing Docker significantly reduced the time it took to set up a new server.

Personnel	Activity	Time
FFI	New, complete server setup: Athena, Metis, OpenFire, OpenVPN and F-Droid	3–4 days
Reference group 1	New, partial server setup: Athena and Metis only.	*None* completed during the allotted time.
Reference group 2	New, partial server setup: Athena and Metis from Docker images.	*All* completed in less than a day.

Table 3: Time spent setting up a new server instance.

In the table, setting up a new server involves doing a new server setup once the IaaS is provided. Here, Ubuntu 16.04 LTS was used as the OS. The personnel categories were *FFI*, which included researchers and developers directly involved in the SMART II experiments. This group knows the software very well, but still needs a few days to set it up correctly. More time is needed for larger setups, since the server setup also includes configuring the software once the components are installed (e.g., adding users, setting up user groups, etc), hence the timespan of 3–4 days. As for the reference groups, these consisted of technically competent students from the Norwegian Defence University College. So, they had basic knowlegde of server setup, but no prior knowledge of the Titans prototype and collection of third-party applications we used. These groups were tasked with setting up a new server (on IaaS) with only the Athena and Metis components. For the reference groups (two groups in each category, so four in total), we allowed them one week (7 days) to attempt to fulfill their tasks. For the first reference group category, this meant installing Java, Maven (build tool), PostgreSQL (database), then building Athena, followed by installing node.js (JavaScript server), npm (Node package manager), and using these to fetch necessary libraries and building Metis. When this was done, the group should use Metis to log on to and administer Athena. The details of installing these components were left to the groups to find out, but the necessary components and a step-by-step high-level installation guide was provided along with the prototype software. However, none of the two groups assigned to this task were able to finish setting up the new server within the allotted time. For the second reference group category, the setup involved installing Docker on the Ubuntu server, then fetching, installing and running the Athena and Metis images. No instructions on using Docker were provided, so the groups had to read up on this tool as part of the assignment. They were provided with the address and login details to fetch the images once they had installed Docker. In this category, both the reference groups were able to finish, in fact both of them finished this task in less than a day. So, we can conclude that setting up the Titans software is a complex task, but using Docker significantly reduces the complexity (and hence time needed) to set up a new server.

5. CONCLUSION

In SMART in 2016, the main goal was to determine whether commercial smart technology, namely the Android platform, was a viable tool for soldiers to get a better understanding of their surroundings, which again could help improve their situational awareness. The SMART project specifically targeted the needs of the HV Area forces. Due to the large number of the Area forces, and the limited time these forces have for training, Titans was

developed as the technical solution that aimed to accommodate their communications needs. Titans targeted only data exchange, as voice was left for traditional radio communication. The result was a simple and intuitive communications concept, that came at a low cost due to the empasis on consumer electronics. During the SMART project, experiments were performed using Titans where we gathered information from the users about their experiences using questionnaires. The respondents not only expected, but actually *experienced a more rapid and efficient execution of their missions* using Titans. Additionally, the respondents offered a lot of suggestions for improvements, several of which we did include in an updated version of Titans.

In SMART II in 2018, we used the updated Titans prototype we had as an example blue force tracking and collaboration tool in a set of experiments. These experiments were mainly devised to evaluate the use of Cloud computing and MDM for military applications. Our conclusions were that Cloud computing and MDM offered significant benefits from a management perspective, while employing these techniques did not negatively impact the users' experiences of the system.

ACKNOWLEDGMENTS

The authors would especially like to thank Trude H. Bloebaum, Marianne R. Brannsten, Federico Mancini and Ann-Kristin Elstad, and also all the other contributors (who are too many to name individually here) to the SMART and SMART II projects for their efforts.

REFERENCES

[1] Reitan, B. K., Fidjeland, M., Hafnor, H., and Darisiro, R., "Approaching the mobile complex – in search of new ways of doing things." 17th International Command and Control Research and Technology Symposium (ICCRTS), Fairfax, VA, USA (2012).

[2] Karud, C. R., Johnsen, F. T., and Bloebaum, T. H., "A socio-cognitive perspective on implementation and use of smart technology in the Norwegian home guard." 24th International Command and Control Research and Technology Symposium (ICCRTS), Laurel, MD, USA (2019).

[3] Johnsen, F. T., Brannsten, M. R., Elstad, A.-K., Bloebaum, T. H., and Mancini, F., "SMART: Situational awareness experiments with the Norwegian home guard using Android." FFI-Report 17/00735, https://www.ffi.no/no/Rapporter/17-00735.pdf (2017).

[4] Forsvaret, "The Home Guard." Retrieved 5 June 2019, https://forsvaret.no/en/organisation/home-guard.

[5] Fielding, R., "REST: Architectural Styles and the Design of Network-based Software Architectures." University of California, Irvine, Doctoral dissertation (2000).

[6] Frøseth, I. M., "CAGED 2.0: Know you enemy." Master thesis, University of Oslo, Norway, http://urn.nb.no/URN:NBN:no-60823 (2017).

[7] Frøseth, I. M., Johnsen, F. T., and Bloebaum, T. H., "Battery Efficient Location Strategy on Android." IEEE International Conference on Military Communications and Information Systems (ICMCIS) 14–15th May 2019, Budva, Montenegro (2019).

[8] Evans, J. B., Ewy, B. J., Swink, M. T., Pennington, S. G., Siquieros, D. J., and Earp, S. L., "TIGR: the tactical ground reporting system." IEEE Communications Magazine, vol. 51, pp. 42-49, 2013 (2013).

[9] Homeland Security, "Snapshot: ATAK increases situational awareness, communication and alters understanding of actions across agencies." Retrieved 5 June 2019, https://www.dhs.gov/science-and-technology/news/2017/11/17/snapshot-atak-increases-situational-awareness-communication, published 17 November 2017 (2017).

[10] Stonewall Defense, "ATAK." Retrieved 5 June 2019, https://www.stonewall-defense.com/atak.

[11] King Jr., S., "Defenders go mobile with new comm system." Retrieved 7 June 2019, https://www.af.mil/News/Article-Display/Article/1575959/defenders-go-mobile-with-new-comm-system/, published 16 July 2018 (2018).

[12] Johnsen, F. T., Zielinski, Z., Wrona, K., Suri, N., Fuchs, C., Pradhan, M., Furtak, J., Vasilache, B., Pellegrini, V., Dyk, M., Marks, M., and Krzyszton, M., "Application of IoT in Military Operations in a Smart City." IEEE International Conference on Military Communications and Information Systems (ICMCIS) 22–23 May 2018, Warsaw, Poland (2018).

[13] Krog, M. A., Johnsen, F. T., Bloebaum, T. H., Brannsten, M. R., and Reitan, B. K., "PISA: Platform Independent Sensor Application." International Command and Control Research and Technology Symposium (ICCRTS), CCRP publication, USA, June, 2015 (2015).

[14] Brannsten, M. R., Bloebaum, T. H., Johnsen, F. T., and Reitan, B. K., "Kings Eye: Platform Independent Situational Awareness." IEEE International Conference on Military Communications and Information Systems (ICMCIS) 15–16th May 2017, Oulu, Finland (2017).

[15] Teleplan Globe, "FACNAV MOBILE – Handheld C2IS." Retrieved 5 June 2019, https://www.teleplanglobe.no/defence/facnavmobile.

[16] NIST, "The NIST Definition of Cloud Computing." NIST Special Publication 800-145, Final 9/28/2011, https://doi.org/10.6028/NIST.SP.800-145 (2011).

[17] Wainewright, P., "Security risks of multi-tenancy." Retrieved 6 June 2019, https://www.zdnet.com/article/security-risks-of-multi-tenancy/, published 18 March 2010 (2010).

[18] Prot, A., "Is SaaS the Future of WordPress?." Retrieved 6 June 2019, https://torquemag.io/2017/08/saas-future-wordpress/, published 11 August 2017 (2017).

[19] HashiCorp, "HashiCorp Terraform – Write, Plan, and Create Infrastructure as Code." Retrieved 6 June 2019, https://www.terraform.io/.

[20] Sensei Enterprises Inc., "Google Clouds Defense In Depth Includes Physical Security." Retrieved 6 June 2019, https://senseient.com/ridethelightning/google-clouds-defense-in-depth-includes-physical-security/, published 28 August 2018 (2018).

[21] Mancini, F., "Modern mobile platforms from a security perspective." Retrieved 7 June 2019, FFI-Report 16/00319, ISBN P: 978-82-464-2752-2 E: 978-82-464-2753-9, https://www.ffi.no/no/Rapporter/16-00319.pdf, published 2 May 2016 (2016).

[22] Sebastian, E. J., "What is key to an effective Mobile Device Management (MDM) strategy?." Retrieved 7 June 2019, http://www.alignminds.com/blog/what-is-key-to-an-effective-mobile-device-management-strategy/, published 30 March 2015 (2015).

[23] Maurya, R. K., "Some Commercial and Open Source MDM Software." Retrieved 7 June 2019, https://www.pcquest.com/some-commercial-and-open-source-mdm-software/, published 5 March 2015 (2015).

[24] Aasen, O. J. and Paulsen, B.-A., "Skybasert tjenesteinfrastruktur." (in Norwegian) Thesis for the Bachelor Degree, The Norwegian Defence University College, Lillehammer, Norway (2018).

[25] Pivotal, "Spring Boot." Retrieved 11 June 2019, https://spring.io/projects/spring-boot (2019).

[26] Ignite Realtime, "Openfire." Retrieved 11 June 2019, https://www.igniterealtime.org/projects/openfire/, published 31 January 2019 (2019).

[27] OpenVPN Inc, "A business vpn to access network resources securely." Retrieved 11 June 2019, https://openvpn.net/ (2019).

[28] Graz University of Technology, "Meltdown and Spectre." Retrieved 11 June 2019, https://meltdownattack.com/ (2018).

[29] "Docker: The Modern Platform for High-Velocity Innovation." Retrieved 11 June 2019, https://www.docker.com/why-docker (2019).