Remote Mobile Screen (RMS): an approach for secure BYOD environments

Santiago Gimenez Ocano*, Byrav Ramamurthy*
*Department of Computer Science & Engineering,
University of Nebraska-Lincoln, Lincoln, NE 68588 USA
Email: {gimenez,byrav}@cse.unl.edu

Yong Wang†
† College of Business and Information Systems,
Dakota State University, Madison, SD 57042 USA
Email: yong.wang@dsu.edu

***Abstract*—The introduction of bring your own device (BYOD) policy in the corporate world creates benefits for companies as well as job satisfaction for the employee. However, it also creates challenges in terms of security as new vulnerabilities arise. In particular, these challenges include space isolation, data confidentiality, and policy compliance as well as handling the resource constraints of mobile devices and the intrusiveness created by installed applications seeking to perform BYOD functions. We present Remote Mobile Screen (RMS), an approach for secure BYOD environments that addresses all these challenges. In order to achieve this, the enterprise provides the employee with a trusted virtual machine running a mobile operating system, which is located in the enterprise network and to which the employee connects using the mobile BYOD device. We describe our proposed solution and discuss our experimental results. Finally, we discuss advantages and disadvantages of RMS and possible future work.**

***Index Terms*—Bring your own device (BYOD), data confidentiality, policy enforcement, security, space isolation, virtualization.**

I. Introduction

Smartphones, tablets and phablets have become key elements in the workplace, as they increase the productivity of the employees. However, the use of these devices has created inconveniences for companies as they must take care of the costs associated with obtaining and maintaining such devices. On top of that, the speed with which new technologies are introduced make these devices obsolete in a short period of time, and employees often want to choose their devices themselves. As a result, companies adapted to this new scenario by using a policy called Bring Your Own Device (BYOD), where each employee provides his or her own mobile device as needed. The advantages of BYOD, also known as dual-use devices, are clear. On the one hand, employees increase their productivity and job satisfaction. On the other hand, companies save the costs of purchasing and maintaining such assets.

However, there is concern related to employee's privacy, as companies might monitor the personal data saved in the devices. From the company perspective, the worries are bigger as mobile devices connect to the corporate network, increasing the chances of cyber attacks. As Wang et al. [1] mention, as employees own the BYOD devices, they are vulnerable to data leakage, unauthorized sharing of data space between employee and corporation, and lack of security policy compliance.

In order to prevent these vulnerabilities, a BYOD solution must achieve the following goals:

- Space isolation, by separating the corporate's space from the employee's space so that different security policies can be enforced.
- Corporate data protection, by employing encryption and rejecting unauthorized access.
- Security policy enforcement, where the mobile device complies with the corporation's security policies.
- True isolation, where the corporate's data is not located on the BYOD device.
- Non-intrusive, meaning that any software installed in the mobile device must not need any special privilege that might allow it to monitor the behavior of the user on his or her device.
- Non-resource-intensive, as mobile devices are resource-constrained and do not have much spare resources for demanding applications.

The first three requirements were identified by Wang et al. [1]. They are necessary, but they are not sufficient. Encryption does not guarantee corporate data protection, as the data is still located in the device and the protection mechanism can be foiled. Moreover, these goals only consider the enterprise perspective, without contemplating the employee's needs. To overcome these flaws we introduce the last three goals.

In this paper we introduce Remote Mobile Screen (RMS), an approach that addresses the desired goals in order to provide a secure BYOD environment.

II. Related Work

Recent research studies have covered BYOD, providing classifications and solutions. We categorize these solutions into agent-based, cloud-based, Mobile Virtual Machine (MVM), framework and Trusted Execution Environment (TEE).

Leavitt [2] elaborates on Mobile Device Management (MDM) and Mobile Application Management (MAM). They are agent-based because an application must be installed in the employee's device to allow the enterprise to lock down, control, encrypt and enforce policies. With an MDM, the enterprise can control the behavior, data, and applications installed in the BYOD device; while an MAM only focuses on managing applications. Companies such as VMware, IBM, MacAfee and SAP offer agent-based solutions. This type

of solutions provides policy enforcement, and might provide isolation, but the corporate's data is stored in the device. The agent installed is intrusive as it requires special permissions.

Leavitt [2] also elaborates on cloud-based solutions, which rely on cloud storage to provide mobile access to data and applications. They create space isolation but once the data is downloaded, data isolation is lost. Additionally, they do not offer policy enforcement.

Wang et al. [1] discuss the use of Mobile Virtual Machine (MVM). They separate spaces in the mobile device, as they can use one Operating System (OS) for the user and another for the enterprise. There are two types of virtual machines for mobile phones: heavy duty and simplified lightweight. The former allows the user to install multiple OSs, while the latter needs less resources but it does not allow the user to install multiple OSs. Russello et al. [3] developed MOSES, a policy-based framework for enforcing software and data isolation on the Android platform, using a lightweight approach. Andrus et al. [4] present Cells, a lightweight Virtual Machine (VM) architecture for mobile phones based on Android. MVMs are resource-intensive for constraint-based mobile devices and do not allow an automatic policy enforcement. Further, they cannot be implemented in a BYOD environment because it is impossible to modify the device without being intrusive to the employee.

Frameworks are solutions that include components in the device and the enterprise side. Titze et al. [5] propose a Security Service Architecture (SSA) for security checks of smartphones that replicates the employee's smartphone on the enterprise side and analyzes it to find security flaws. Chung et al. [6] developed 2TAC, which addresses access issues by using a double layer access control (one at the device and the other in the cloud) along with device security profiles, anti-virus/malware scanners, and social networking. Wang et al. [1] propose a BYOD Security Framework (BSF), which we describe in detail at the end of this section. Cisco offers BYOD Smart Solution [7], which focuses on the infrastructure of the network and provides policy management, mobility and applications, while supporting MDMs from third parties. The disadvantages with all these frameworks are that they install an intrusive application in the employee's device, and they allow corporate data to be stored in the devices.

Ekberg et al. [8] discuss TEEs, which separates the execution of an application into secure and insecure parts by using a protected area in the processor. Zhao and Osono [9] developed TrustDroid, an Android application that analyzes other applications, to prevent them from accessing the corporate data. TEE provides space isolation but data can be stored in the mobile device and it does not offer policy enforcement.

A. BYOD Security Framework

In this framework, proposed by Wang et al. [1], there is an enterprise side that includes the corporate's resources, a Security Policy Database (SPD), an MDM, as well as a Network Access Control (NAC) that separates requests coming from the personal space or corporate space based on the policies from the SPD. Additionally, there is a BYOD side where isolation is present by implementing an MVM. In the corporate space, there is an MDM agent that enforces the security policies in the corporate data. The corporate space includes cryptographic primitives that make the enterprise's data confidential to non-enterprise actions. This framework provides data protection, isolation and policy enforcement, but at the expense of installing an MVM and an MDM agent in the BYOD device.

III. Remote Mobile Screen (RMS): Description

Below, we describe our solution, Remote Mobile Screen (RMS) which addresses the challenges of BYOD security.

A. Architecture

RMS modifies BSF [1] by moving the corporate space inside the enterprise network, adding a new element called Corporate Space Manager (CSM) and using the VNC protocol (based on the RFB protocol [10]) to allow the user to access the enterprise space. Just as in BSF [1], RMS presents a BYOD side and an enterprise side, which are described as follows.

1) BYOD side: Compared to the BSF [1], this part of the architecture is simpler. The mobile device only contains the personal data and applications of the user. The only requirement is that one of these applications must be a VNC client used to access the enterprise space, located in the corporate network. Part (A) of Figure 1 shows this side.

The novelty of our architecture relies on the way the employee access the corporate resources. In order to access the resources, the employee must use a VNC client found in a app store. But then the user does not access a desktop OS (i.e. Windows, Linux, Mac OS X) but a Mobile OS (MOS) (i.e. Android, iOS). This addresses the poor level of usability that desktop OSs have when they are accessed from a BYOD device, as they are not designed to be scaled for small screens nor implement gestures customary to an MOS. Consequently, when the employee accesses the enterprise side, he or she is presented with an interface designed for mobile devices. To our understanding, the concept of a mobile device that access a MOS has not been used in BYOD environments, or for any other kind of purpose.

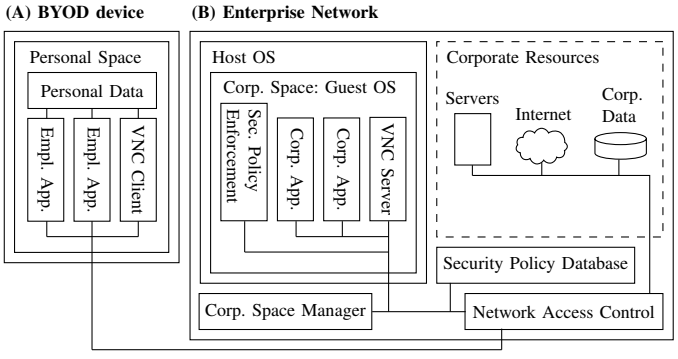

Fig. 1: Diagram of the architecture of RMS.

2) Enterprise Side: Since this side does not suffer from limitations in terms of resources as a mobile device does, it is composed of most of the elements in the RMS architecture. Corporate resources, NAC and SPD perform the same roles as the ones described in BSF. The first one is the group of resources that the company has, such as e-mail servers, web servers, gateways to the Internet, etc., while NAC is in charge of authorizing and rejecting access to the resources from inside and outside of the corporate network by relying on the policies located in the SPD.

On the enterprise side, depicted in Part (B) of Figure 1, we find the corporate space. This space is an MOS running as the guest OS of a VM. In this OS, a VNC server is installed and configured in such a way that the user can access the enterprise space using a VNC client in his or her BYOD device. In addition, to comply with the enterprise policies, the corporate space has a security policy enforcement entity.

Our Corporate Space Manager (CSM) is in charge of managing all the enterprise spaces, by creating, starting and stopping MOSs in the VM, providing elasticity to the architecture. CSM operates as a proxy server, since it inspects the content of VNC packets and performs actions based on such content. In order to decide to create or start an existing MOS, it must keep track of which MOS is assigned to each user. Further, it must add and remove NAT entries in the VM so that messages get forwarded to the corresponding MOS.

Consequently, by installing the VNC application, the user can access the corporate resources just as if he or she were using an MVM installed in the BYOD device, but without affecting its resources. The client application is in charge of the visualization of the of the guest OS screen, and of taking the gestures from the client and sending them to the guest OS.

B. Features of RMS

RMS achieves all the desired goals for a secure BYOD environment. It provides separation of spaces by implementing the corporate space in the enterprise network while leaving the personal space at the mobile device. This produces true isolation, since both spaces are never shared either in the BYOD device nor in the MOS. Data confidentiality is guaranteed since, even if the employee loses the device, the corporate data remains always in the enterprise network. This means that there is no data leakage and no unauthorized access. Further, the use of NAC and CSM provides a strict control of the information's flow between both spaces.

Security policy enforcement and data protection are easily achieved since the user does not control the MOS. The enterprise can enforce the use of policies by using an MDM agent in the MOS. For example, instead of installing applications from the Internet in the guest OS, the user is offered a curated, authorized list of applications allowed by the company. Furthermore, the enterprise can decide which permissions an employee can have in the corporate space. There is an additional advantage; instead of focusing the security efforts on every mobile device, RMS concentrates all the security efforts on a single point in the corporate's network.

TABLE I: Comparison between the different type of solutions.

		Desired Goals					
Solution	Type	Space Isolation	Security Policies	Corporate Data Protection	Non-intrusive	Non Resource Intensive	True Isolation
MDM/MAM [2]	agent-based		✓	✓		✓	
cloud-based [2]	cloud-based			✓	✓	✓	✓
MOSES [3]	MVM	✓		✓			
Cell [4]	MVM	✓		✓			
SSA [5]	framework		✓				
2TAC [6]	framework		✓				
Cisco [7]	framework		✓		✓	✓	
TrustDroid [9]	TEE	✓	✓				
BSF [1]	framework	✓	✓	✓			
RMS	framework	✓	✓	✓	✓	✓	✓

RMS is a non-intrusive solution that does not require the amount of resources that an MVM needs. The employee only needs to install a VNC client available in the app store of the mobile device's platform. Such clients do not require administrative resources as MDM agents do. In addition, the enterprise does not need to provide support to the BYOD at all, since there is no software installed by the corporation.

Another benefit is the fact that RMS is platform-agnostic, as it can provide service to every mobile platform that has a VNC client application. BYOD devices can be changed by the employees without consulting the enterprise, providing flexibility and exploiting one of the main reasons why BYOD policies were implemented. Further, the employee does not need to allocate storage space on his or her device for the corporate data and work-related applications.

Finally, in Table I we provide a comparison table of all types of solutions for BYOD and the goals that they achieve. It can be seen that RMS achieve all desired goals.

IV. Implementation and results

A. Implementation

We developed a proof-of-concept implementation. For supporting the MOS we used VirtualBox [11] that hosts AndroVM [12] as a guest. Since virtual machine platforms use x86 architecture, the MOS must use the same architecture. As an alternative to AndroVM, we have also tested Android-x86 [13], a port of Android for the desired architecture. The VNC server used is Droid VNC server [14] modified for the x86 architecture. Finally, for VNC clients, we have employed VNC Lite, VNC Viewer and TinyVNC. Figure 2 shows our implementation, where AndroVM is displayed on the screen of an Apple iPhone 5s running iOS 7 and a Samsung Focus with Windows Phone 7.

B. Scalability

The framework we propose permits one employee per guest OS. The resources needed allow for a single server with standard configuration to run several guest OSs, providing service to several employees. Moreover, the VM software has

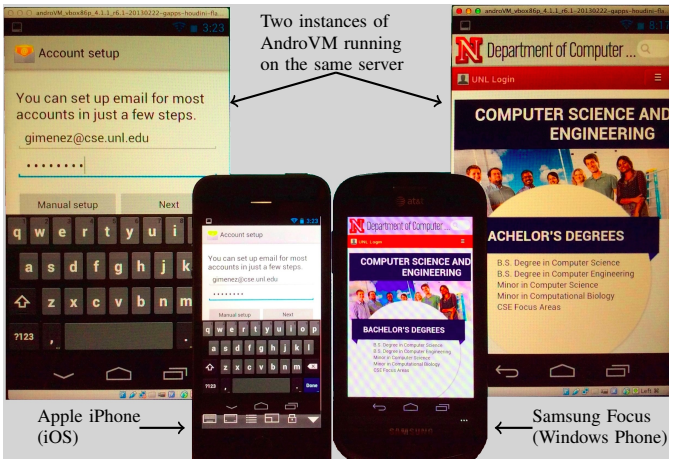

Fig. 2: Implementation of RMS using different devices.

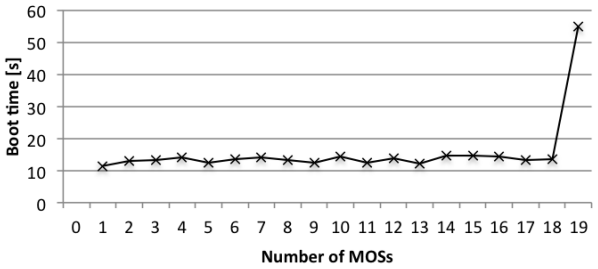

Fig. 3: Boot time vs. number of MOSs

the feature of saving a machine state instead of turning it off, while releasing resources at the same time. This saves unused resources for other employees and provides a fast start after the employee has disconnected and re-connects.

C. Experimental results

To test the scalability and performance of RMS, our experiment consisted of finding the maximum amount of MOSs that the server can support before crashing. The server hardware includes a 1.3GHz dual-core Intel Core i5 CPU, 8GB of 1600MHz LPDDR3 RAM and a 128 GB flash drive. The experiment consisted of starting one MOS after the other until such a limit was achieved. We measured the time taken for each of these OSs to boot up, as well as checked the performance by measuring the CPU load, the number of threads, and different parameters related to memory usage.

1) Boot time: Even though AndroVM requests 1GB of RAM memory, the MOS needed 540 MB. This allowed us to run 18 MOSs without affecting the booting time. Such a behavior can be seen in Figure 3, where the booting time is on average 13 seconds until the 19th instance of MOS, when the boot time reached around 55 seconds. The system continued working but after 2 minutes and 40 seconds, the server crashed.

2) CPU usage: Figure 4 shows the response of the CPU load and the number of threads as a function of the number of MOSs. As expected both indicators increase as the number of MOSs increases. On the one hand, the number of threads

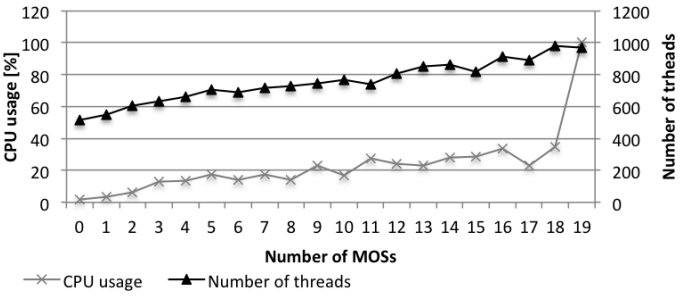

Fig. 4: CPU usage and number of threads vs. number of MOSs

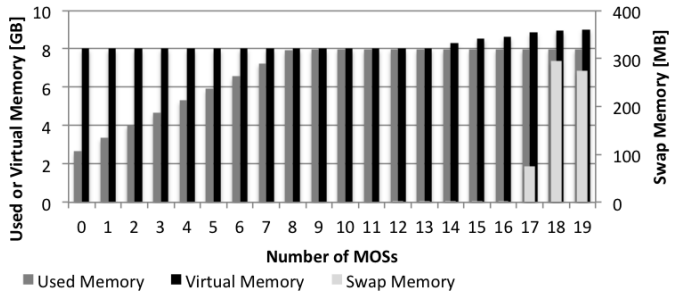

Fig. 5: Virtual memory, used memory and swap memory vs. number of MOSs

presents a linear behavior, as 24 of them are added by each MOS. On the other, the CPU load presents a linear behavior until the maximum number of MOS is reached, when it reaches 100% of use. This effect is because there is not enough RAM memory in the system to support all the MOSs and the CPU must perform paging more frequently.

3) Memory usage: To analyze memory usage, we gathered information of virtual memory, used memory and swap memory. Figure 5 illustrates these indicators. Virtual memory remains constant until the used memory reaches 8GB. At this point, the virtual memory increases its value by incrementing the size of the swap memory.

Consequently, based on the discussion related to scalability and the results of our experiments, we can conclude that RMS scales very well with the number of users.

V. Discussion

In this section we discuss how RMS can benefit from the migrating it to the Cloud, provide a security analysis and list the disadvantages of RMS.

A. The Cloud

The natural evolution in RMS is to migrate the enterprise side to the cloud, as it addresses concerns related to latency, scalability, and availability.

First, since the Cloud is composed of several datacenters distributed across the world, the employee will likely be close to one, making the latency minimal. Further, architectures like the ones presented in MobilityFirst [15] propose a content-aware network, where the first-hop router stores all the data

needed by the user and, when the user roams to other networks, the data is migrated to remain close to the user.

Second, RMS benefits from elastic computing, as the Cloud adapts to the needs of the employees dynamically by activating and disabling guest OSs on-demand.

Finally, the use of distributed datacenters provides redudancy against failure that guarantees continuous service to the employee. Even if the entire datacenter is unavailable, another datacenter can be used at the expense of higher latency.

B. Security Analysis

We have identified a set of threats that affect the security of our framework. First, the VNC protocol lacks an acceptable level of security, and it must run on top a VPN protocol [10].

Second, this protocol allows sharing the clipboard and files between the client and the server. Disabling the features from the server side can prevent both situations, isolating the clipboard at each space and preventing file-sharing between the spaces. Another approach is to allow the flow of information between spaces only in one direction, meaning that an employee can only upload files to the server, or send information from the user's clipboard to the server's clipboard.

Third, there might be situations where the content of the screen of the BYOD device is captured, i.e. by taking screenshots of the device's screen or shoulder surfing. There is no solution to prevent this since company does not have any intrusive software installed in the mobile device. The alternative against this threat is user education.

Finally, Denial of Service is an important threat. If employees cannot access the corporate space, they will suffer from work disruption. These problems are addressed by using redundancy of single points of failures and backing up data. For RMS, the NAC and the CSM must have a fail-over system and the MOSs must be frequently backed up.

C. Disadvantages of RMS

In order to access the corporate network, the employee must have connectivity, with an acceptable amount of delay. These two issues are part of the trade-off that RMS creates in order to achieve all the goals needed in a secure BYOD environment. These drawbacks are discussed as follows.

1) Latency: We define latency as the time between the user's input and the response obtained from the client application. Just like another real-time application, RMS will suffer from latency-related issues. In general, the farther away the user is from the corporate resources, the bigger the latency.

2) Connectivity: The mobile device relies heavily on continuous connectivity. Without it the employee cannot perform work actions such as saving an e-mail draft to send it later.

Our argument is that the disadvantages that RMS presents can be either overcome or accepted, as the benefits that the company obtains are greater than the drawbacks.

VI. Conclusion and Future Work

In this paper we have described the security challenges that BYOD faces. We have listed the necessary goals as well as the sufficient goals that a secure BYOD solution must achieve. A classification of the current solutions for BYOD has been presented, summarizing which goals they meet and showing that currently there is no solution that can achieve all of them. We proposed a new architecture for BYOD solutions, Remote Mobile Screen (RMS). We provided a description of RMS, discussed its features, disadvantages and describe its implementation. We have presented additional thoughts on scalability, showed the results of our experiments and discussed how to evolve RMS by migrating it to the Cloud.

Future work involves developing a CSM as well as performing a study on latency. In addition we plan to address usability issues such as support for mobile gestures that were not discussed in this paper. Further, we expect to analyze the adoption of RMS by users, by providing them access to our testbeds and by gathering feedback with a survey.

Acknowledgment

This work was supported in part by an NSF FIA-NP grant (CNS-1345277) and by NSF MRI (CNS-1337529). The authors thank the anonymous reviewers for their valuable comments on this manuscript.

References

[1] Y. Wang, J. Wei, and K. Vangury, "Bring Your Own Device Security Issues and Challenges," in *11th Annu. IEEE Consumer Communications & Net. Conf.*, January 2014.

[2] N. Leavitt, "Today's Mobile Security Requires a New Approach," *Computer*, vol. 46, no. 11, pp. 16–19, 2013.

[3] G. Russello, M. Conti, B. Crispo, and E. Fernandes, "MOSES: supporting operation modes on smartphones," in *Proc. of the 17th ACM Symp. on Access Control Models and Technologies*, 2012, pp. 3–12.

[4] J. Andrus, C. Dall, A. V. Hof, O. Laadan, and J. Nieh, "Cells: a virtual mobile smartphone architecture," in *Proc. of the 23th ACM Symp. on Operating Systems Principles*, 2011, pp. 173–187.

[5] D. Titze, P. Stephanow, and J. Schutte, "A configurable and extensible security service architecture for smartphones," in *Advanced Information Networking and Applications Workshops (WAINA), 2013 27th Int. Conf. on*, 2013, pp. 1056–1062.

[6] S. Chung, S. Chung, T. Escrig, Y. Bai, and B. Endicott-Popovsky, "2TAC: Distributed Access Control Architecture for" Bring Your Own Device" Security," in *BioMedical Computing (BioMedCom), 2012 ASE/IEEE Int. Conf. on*, 2012, pp. 123–126.

[7] "Cisco BYOD Smart Solution," Cisco, 2013. [Online]. Available: http://www.cisco.com/web/solutions/trends/byod_smart_solution/docs/byod_smart_solution_aag.pdf

[8] J.-E. Ekberg, K. Kostiainen, and N. Asokan, "Trusted execution environments on mobile devices," in *Proc. of the 2013 ACM SIGSAC Conf. on Computer & Communications Security*, 2013, pp. 1497–1498.

[9] Z. Zhao and F. C. Colon Osono, "TrustDroid: Preventing the use of Smartphones for information leaking in corporate networks through the used of static analysis taint tracking," in *Malicious and Unwanted Software (MALWARE), 2012 7th Int. Conf. on*, 2012, pp. 135–143.

[10] T. Richardson and J. Levine, "The remote framebuffer protocol," 2011.

[11] Oracle VM VirtualBox. Oracle. [Online]. Available: https://www.virtualbox.org

[12] AndroVM blog — Running Android on a Virtual Machine. [Online]. Available: http://androvm.org/blog/

[13] Android-x86 - Porting Android to x86. [Online]. Available: http://www.android-x86.org

[14] Droid VNC server — onaips.com. [Online]. Available: http://www.onaips.com/wordpress/?page_id=60

[15] D. Raychaudhuri, K. Nagaraja, and A. Venkataramani, "Mobilityfirst: a robust and trustworthy mobility-centric architecture for the future internet," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 16, no. 3, pp. 2–13, 2012.