International Journal of Research
Available at https://edupediapublications.org/journals
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 02 Issue 08
August 2015

# Threats and Vulnerabilities of BYOD and Android

**Sandip Yadav**
**SCIT, India**
sandip.yadav@associates.scit.edu

**Udyan Ganguly**
**SCIT, India**
udyan.ganguly@associates.scit.edu

**Saurav Suman**
**SCIT, India**
saurav.suman@associates.scit.edu

**Dr. Priti Puri, Phd Assistant Professor**
**SCIT, India**
**puri_priti@rediffmail.com**

**Keywords**

Bring Your Own Device (BYOD), Mobile Device Management (MDM), Control Objective, Android, Container, Context related access control.

## Abstract

*This paper discusses about the vulnerabilities related to Bring-Your-Own-Device (BYOD) wherein we have explored the concept of containerization and the vulnerability related to BYOD Container. Some mitigation process has also been suggested for this vulnerability. The Second vulnerability - what we have tried to highlight is- the vulnerability arising as the Application Installation Permission cannot be cancelled at the runtime. One solution for this vulnerability dealt in the literature is Context-related access control. Paper has discussed some limitation to that solution also and proposed the solution for the same. We have limited the discussion to Mobile usage of BYOD wherein we have taken Android phones as the base.*

## Introduction

*Bring-Your-Own-Device (BYOD)* assists the users / developers in the businesses to use their own devices to use the company resources for both the professional and their personal usage purpose. It may also include accessing official documents, applications and company network. In 2009[7], this idea was first time came into the business environment when Intel accepted the significance of workforce using their personal devices for using industry network and resources. In 2011[8], service providers of Information Technology as Unisys and software vendor like Citrix shared their observations about this rising inclination and businesses has started considering it [1].

By considering the idea of employees having mobile devices, the benefits are incontestable and comprehensive. For organizations, operational and business velocity is not much dependent on place and time. By giving these unshackling benefits of mobility to their employees, industry is trying to provide balance between employee's work output and their personal lives. [2]

In present time, when companies are searching ways to handle the space crunch and making the work/office culture more comfortable to the employee's nature, BYOD is a good solution to resolve these things. BYOD allows employees to carry their own devices like Tablets, laptops, and their Smart phones in their office for office use. Each device has multiple flavors in the computer world e.g. laptops have Windows, MAC OS, LINUX, etc and similarly Smart phones have Android, iOS and Windows as major providers of OS. In this paper, we have focused on vulnerabilities and security control of Android related to Smart phones as a part of BYOD.

2. **Organizations need Security Practices**

Conventional practices [3] in the industry are the following:

1. Organization's security policies should be compulsory on the mobile devices,
2. For data communications and storage, strong encryption is required, wiping and checking the device before issuing the same device to different user, and also wiping the device should be possible remotely when it is missing or stolen
3. Device& user authentication should be there, remotely reset option for forgotten passwords, automatically as well as remotely locking the devices if left unlocked in an unsecured place.
4. Some kind of restrictions should be there for which app stores may be used. Not only limiting the permissions for installing and updating each application but also restricting the use of synchronization services.

Applications digital signatures verifications and distributing the company's applications from dedicated application store are also essential features. [3]

Security is the important feature of the mobile device implementation; for this, updating the patch of all the components should be there and best security practices should be followed. Mobile devices should be automatically detected if they are Jail broken or rooted mobile devices.

If mobile device has given by previous organization with some security settings, yet the new organization should provide their own best security practices to that device with supplemental security controls as antivirus, risk merits, and data loss prevention (DLP) technologies [3].

Mobile device management (MDM) offers collection of potential controls such as pushing VPN connection settings, providing Wi-Fi credentials, installing selected apps for use(application white listing), limitations on camera use,

prohibiting screenshots, and preventing access to device settings within the environment. A fully managed (usually company-owned) device within security controlled environment set to be locked down significantly with a few warnings.[4]

## 3. Containerization

By using this concept of containerization [9], organizations have the aptitude to securely arrange and control their company data in an encrypted place at the device, isolating the personal data from company's own. This container which is not only password protected but also gives users to access all organization data via single sign on (SSO ). The containerization concept also controls applications that can access data and how that information is to be shared. If the some data is compromised, the entire container or that particular app can be removed remotely [9].

Containerization permits the company's security policy implementation on the fixed secure containers without touching the data functionality plus data itself on the device's confidential area, [11] [12]. Containerization allocates the employment of several security systems, as [13] directly renewing or updating Content to safe container, policies of Access control for the time (when used) and location (where used) and Strong content Encryption, etc.

According to some literature [10], there are three important types of containers are

1) Application specific containers: .This is also called bolt-on Software Development Kit (SDK) containers. These containers make changes in the user interface exterior, and that sometimes do not like by customers.

2) Application neutral containers: They are using an application wrapping process for providing security means which are not the part of the application source code. Applications wrapping can be implemented in a short period because this type of containers does not need the original application code change. This gives users layout of original application and consistency for personal and industry space.

3) Integrated containers: it permits security and business productivity optimization by using tools and applications designed, because of integration into the operating system.

According to the comparative study [14], the most suitable method is the application neutral container. Because of uncomplicated application containerization process does not change the exterior of the applications user interface and at the same time does not change in source code.

4. **A Container Provider-AirWatch**, This product uses containerization approach among MDM solutions to protect sensitive data. The containerization approach helps directly protecting against some threats and indirectly against others. It also prevents data leakage resulting from the physical device loss. For logically compromised devices, AirWatch gives additional controls beyond data encryption at rest within the container [4].

For wiping the data, the simplest mechanism is to destroy the encryption

key so that, data will be there in container but with no use. For this mechanism, strong encryption algorithm is required to resist long term offline cracking attempts.

AirWatch and other MDM controls are based upon a base of prevent and detect root-level access to the file system. A user with root-level privileges might be able to bypass the MDM policies and possibly access the data stored on the device [4].

**5. Container Vulnerability- Air Watch products**

According to Tim Collyer's study [4], one email container functions had a vulnerability which could expose the data. AirWatch was notified through a responsible disclosure process and fixed the vulnerability in a subsequent version of the AirWatch Inbox app. We independently verified that the issue was addressed [4].

AirWatchstores attachments with a plaintext title, but with the contents encrypted. However, when a user needs to view an attachment on the mobile device, it must be decrypted. The email client accomplishes this by storing a decrypted copy of the attachment

There are two issues which lead to the data leakage vulnerability. The first issue has already been mentioned – the method for data wipe. When a mobile device needs to have its data wiped, one consideration is velocity. If the data wipe took a longtime, an attacker might be able to challenge or stop the process and get access to sensitive data. One method to destroy the data quickly is to delete the encryption key. By this way, encrypted data still will be there.

The presence of the data after a device wipe becomes important in terms of the contents of the Decrypt directory.

The files stored in Decrypt side are in unencrypted state. The data leakage vulnerability exists due to data contents are not removed after reading it. Even the contents of the Decrypt folder may be there after several months of the attachments when they were downloaded and viewed, and notably, after the device wipe command was executed.

When a remote wipe is issued, the end user is given the option to remove the

AirWatch email application from the device. If selected 'No', nothing will removed. The device appears in the AirWatch management console to have been successfully wiped. Then any attachments which had been opened on the device still are there in unencrypted state.

By describing the vulnerability is to emphasize some of the limitations around containerized data stored on an Android device which is owned by the end user. [4].

**6. Solution:** In this paper, we have tried to propose some solutions for container vulnerability which has discussed above.

1. Temp files from the Decrypt folder should be deleted once the attachment is closed. This can be achieved by setting up the Cache memory on timer so that once the timer is complete; the message will get deleted from the device storage itself.

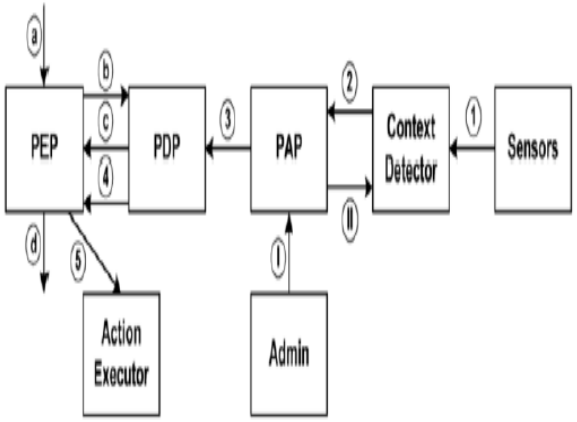
Benefits: This measure will help in making optimum use of the memory and on the same side avoiding any storage of organization's data in the user's mobile.

2. In case of device loss, when the user informs the network administrator ,at that time the flag will be set OFF which controls the activation /deactivation of the keys(encryption/decryption key present in the device.) If both keys flags are ON only then the decrypt folder should be opened. In the case of device loss, this key can be wiped out from the device remotely so that the attacker cannot open the folder itself.

**Second Vulnerability:** This is a situation that arises because the App Installation Permission cannot be cancelled at runtime.

Sometimes Android users don't have the privilege to stop the app installation once its installation started. An intruder can use this vulnerability in order to attach a file or malware to one of the apps and then wait for someone to install the app. [5]

To tackle this issue one solution that is **Context-related access control** which is described below by CRêPE [6] i.e. context related policy enforcement system. This system permits context-related policies to be set (even at runtime) by both the authorized third parties and the user remotely or locally which can be installed, updated and applied system-wide at runtime.




In the figure, points a, b, c, and d are the development flow of a runtime way in request: the interruption of request is shown (arrow) by the Policy Enforcement Point (PEP); which in sequence request for (b) the Policy Decision Point (PDP) if the request should be approved based on the Answer (c), the request will be approved (d) or not. The progression flow is shown by I and II due to executive commands. Context can be set by certified parties and related policies in the system (arrow 1). Then, the Context Detector informed by the Policy Administration Point (PAP) that the newly set contexts need to be observed [6].

If there is any malware or keystroke file attached to the app then with the help of Context-related access control the installation can be stopped in between. In this approach, policies are linked to contexts, and the run time activation/deactivation of contexts is acknowledged automatically by the sensors present in each mobile device. [6]

**7. Our findings**:

In the above mentioned approach, identification of threat profile is not clearly visible i.e. when user wish to stop downloading app in middle, there is no way to find if that app is carrying any threat/virus/trozen horse with that app.
In that scenario, threat detection should be followed by this context related access control or during the process inputs for context related access control should also be taken from threat detector. There should be some check or reconfirmation for downloading that particular app by users or not.
Other limitation is time i.e. how much time this process should take for stopping the app.

## Conclusion

In this paper, Containerization have been discussed. After that, we have been tried to explore the two main problems-one is container vulnerability and other is 'second vulnerability' that is App Installation Permission cannot be cancelled at runtime of Android phones. For such vulnerabilities, we have proposed some solutions. The logical findings mentioned so far implies that if implemented these steps then the BYOD and Android can become more reliable and more secure way to access the data and information.

## References

[1] Prashant Kumar Gajar, Arnab Ghosh and Shashikant Rai., BRING YOUR OWN DEVICE (BYOD): SECURITY RISKS AND MITIGATING STRATEGIES. Volume 4, No. 4, April 2013,Journal of Global Research in Computer Science.

[2] A Frost & Sullivan., BYOD Done Right is a Win-Win for Workforce Mobility,A Frost & Sullivan White Paper,2015

[3] Guidelines for Managing the Security of Mobile Devices in the Enterprise, NIST Special Publication 800-124 Revision 1, June 2013

[4] SANS Institute InfoSec Reading RoomTim Collyer. Airwatch MDM and Android: a policy and technical Review

[5] Pritam R. Tarle, Comparative Study of Smart Phone Security Techniques,International Journal of Emerging Technology and Advanced Engineering, Volume 5, Issue 2, February 2015

[6] M. Conti, B. Crispo, E. Fernandes, and Y. Zhauniarovich, "CRePE: A System for Enforcing Fine-Grained Context-Related Policies on Android," IEEE Trans.

[7] Mobile: Learn from Intel's CISO on Securing Employee-Owned Devices

http://www.govinfosecurity.com/webinars/mobile-learn-from-intels-ciso-on-securing-employee-owned-devices-w-264

[8] Bring your own device http://en.wikipedia.org/wiki/Bring_your_own_device#cite_no te-6 (as accessed on 1st February 2013)

[9] Enterprise Mobility Best Practices: MDM, Containerization or Both?

http://www.airwatch.com/downloads/resources/white-papers/airwatch-whitepaper-mdm-containerization-or-both.pdf

[10] BlackBerry: Finding the right mobile device containerization solution, BlackBerry, 2014

[11] Kaspersky Lab: Security technologies for mobile and BYOD, Kaspersky Lab, 2013

[12] Prabhu, N., Rendell, J.: Smart containerization TM, CA Technologies, 2014

[13] Mobile device management technologies. (2014, Jun 23) Online Document. Available: http://ukblog.immobility.com/book/export/html/7

[14] Dragan Peraković, Ivan Cvitić, Siniša Husnjak, Comparative analysis of enterprise mobility management systems in BYOD environment, Nov,2014.

https://www.researchgate.net/profile/Dragan_Perakovic/publication/269337883_Comparative_analysis_of_enterprise_mobility_management_systems_in_BYOD_environment/links/54f8a4950cf28d6deca2bc5b.pdf