 

# Airdrop 2.0 Vulnerability Detection System

Oreoluwa Adesegun* Olutimi Onilede Aaron Izang Raymond Okoro
School of Computing and Engineering Sciences, Babcock University, P.M.B. 4005 Ilishan-Remo, Nigeria

**Abstract**
In the world of today cyber security has become of upmost importance, it provides a means for protecting vital information that is uploaded to the internet. A vulnerability detection system seeks to detect possible exploits in systems that can be used to harm or steal information. On mobile platforms security is also an issue and various means have been created to combat these security flaws. One of the commonest vulnerabilities know in the world is that of the Denial of Service (DoS) attack, even though they are many ways in which this can be conquered there is no true solution that may work for every platform .This study focuses on the effect of Denial of Service attack on a mobile platform, along with a viable and easily deployable solution. This will give researchers an insight into the security threats that mobile platforms have and also give a solution to be built upon for development of secure applications in the future.
**Keywords:** Denial of Service, Detection System, Vulnerabilities, Mobile platform.
**DOI:** 10.7176/NCS/11-03
**Publication date:**July 31st 2020

## 1.0 INTRODUCTION

From the birth of the Advanced Research Project Agency Network (ARPANET) in the 1970's (Murphey, 2019), cybersecurity has been an ever growing field that protects our digital devices. It is a field that deals with but not limited to activities such as security policy governance and compliance, penetration testing and vulnerability assessment among others. A vulnerability is a loophole or defect in a network or an application. In order to detect a vulnerability in a system or an application an activity called vulnerability scanning must take place. A vulnerability scanner is a computer program designed to assess computer networks or applications for known weaknesses (Cedar101, 2020). A common vulnerability that networks and applications deal with is that of undergoing a denial of service (DoS) attack. According to the Common vulnerability and exposures (CVE) website airdrop 2.0 application has this vulnerability (CVE-2019-9832), but what exactly can be done to correct such vulnerability? It was the purpose of this research, to answer such a question.

## 2.0 LITERATURE REVIEW

As part of the processes taken in order to find a solution the review of related works was done, and literature is as follows:

### 2.1 Airdrop

Airdrop is an application that enables file sharing between devices over a Wi-Fi network. It uses Bluetooth to create a peer-to-peer Wi-Fi network between devices. Each device creates a firewall around the connection and encrypted files are sent between the devices (Nations, 2020)

### 2.2 Denial Of Service (DoS) and Distributed Denial Of Service (DDoS)

A denial of service attack is a cyber-attack in which the transgressor looks to make a network resource unreachable to its users by briefly or perpetually obstructing services of a host connected to the internet. It is done so by flooding the target machine or resource with redundant request in an attempt to overload the system and prevent some or all request from being answered. (Pbsouthwood, 2020)
A distributed denial of service attack in essence is the same as a DoS attack except for the fact that the use of botnets. The attacker will install command and control software on several user's system to create a botnet. Once the attacker has the botnet ready he will send a start command to all the nodes of the botnet and they in turn will send their programmed request to the target server. (Petters, 2020)

### 2.3 Security In Android And Its Architecture

Android security consist of various systems put in place to establish a solid architecture some of which include:

#### 2.3.1 Permission Mechanism

The primary objective of a permission is to protect the privacy of an android user. Certain application must request permission to access certain data and features on the system, it is dependent on the setting of the mechanism whether to automatically grant permission or to as the user to approve the request.
Application Sandbox prevents one android application from accessing the resources of another, this is often called a "sandbox". The android system does this by assigning every application a user identification (UID). The

 

UID's are assigned unambiguously therefore making sure that no two apps have the same UID and therefore apps with constant UID's will share resources. (Dhankar, 2017)

Access Control enables an android application to control access to its data through its settings. These settings and change in permissions establish an order whereby collection level permission determines who can access the collection as a whole. (Access Control, n.d.)

Application Signing Android Application signing allows software developers to identify the maker of the application and to update the application without causing complicated interfaces and permissions. (Application Signing, 2020)

2.4 Other Vulnerabilities in Android Operating System

2.4.1 Binary Protection

Inadequate Jailbreak/ Root Detection: Rooting or jailbreaking a device bypasses data protection and encryption program on the system, which compromises the device and allows exploits to be run on the system. (Codersera, 2019)

2.4.2 Insufficient Transport Layer Protection

Android applications usually are unable to encrypt network traffic when it is necessary to protect sensitive internet communications. (Codersera, 2019)

2.4.3 Insufficient Authorization/Authentication

Insufficient Authorization results once associate application doesn't perform adequate authorization checks to make sure that the user is playacting to access information in an exceeding manner outside the protection policy. (Codersera, 2019)

2.4.4 Brute Force – User Enumeration

A brute force attack is a strategy used to get a hidden value by utilizing a mechanized procedure to attempt a various number of potential values. The programmer utilizes the way that the entropy of the values is littler than seen. For instance, while an 8-character alphanumeric secret phrase can have 2.8 trillion potential values, numerous individuals will choose their passwords from an a lot littler subset comprising of basic words and terms. (Codersera, 2019)

2.4.5 Insufficient Session Expiration

When a user signs out of an application the identifiers used during the session are meant be made invalid. If the application does not invalidate the identifiers then there is a possibility that another user can use those identifiers to impersonate them. (Codersera, 2019)

## 3.0 METHODOLOGY & DESIGN OF SYSTEM

The process model used to develop the system is the waterfall model, it is a common process model used in the development of software. Each phase of the process model is given crucial attention before moving on to the next phase. In order to build a system that would combat the vulnerability of the airdrop 2.0 application we had to first exam how the vulnerability was executed

.

3.1 Examining The Airdrop 2.O Application Vulnerability

In the process of carrying out our research there was documentation on the CVE website that shows how to carry out the DoS attack. The process were:

Creating A Local Area Network(LAN) From The App To Allow File Sharing

In order for the application to allow files to be shared, a server must be created in order for two users to connect with one another. After the creation of server the port number and the internet protocol (IP) address of the host is displayed on their screen.

The Use Of Nmap To Scan For The LAN

Nmap is then used to scan the server using the IP address shown on the phone of the host. With the use of certain commands. These commands are:

–p (used to specify the port number)
–T (sets the timing template of the scan)
–open(shows only open ports)
–v(verbose mode prints extra information about the scan)
–n(no resolution of domain name server)

3.2 Executing The Dos Attack On The App

From the information gotten from the process stated above, a DOS script gotten from the Common Vulnerabilities and Exposures (CVE) site written in C++ is executed (Marcelo Vázquez, 2019). The script then proceeds to make multiple socket connections to the server which uses all the available resources, hindering the sharing of files between two phones.

 

From this analysis the system was designed in the following manner:

3.3 Reading The IP Address Of The Client
The system will read the IP address of each client that makes a connection request to the server. This will be done so that the client who wants to connect to the server can be identified in order for the connection request count to be monitored.

3.4 Monitoring The Number Of Connection Request Made By Each IP Address
There will be a counter in place in order to keep track of each successive connection request made by each IP address. There will also be a limit to which each IP address can request for a connection successively.

3.5 Stop Connection Of IP Address If Multiple Connection Request Are Made
Once this limit is reached a flag will be raised indicating that a DoS attack is being executed on the application. The system will automatically stop the connection request automatically.
A detailed diagram of how the system should work is provided below in figure 1

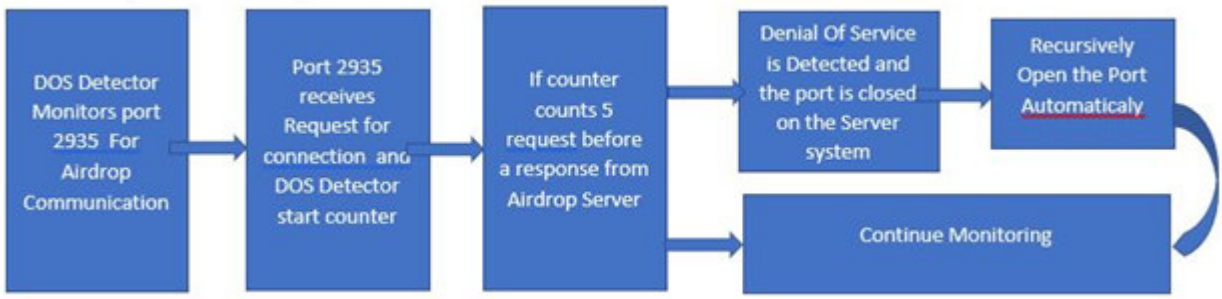


Figure 1 System Architecture

The following tools were used to implement the system:

Java was the programming language used for this project, the reason for this is that Java is a general-purpose programming language and can be used for virtually any task. Java also is a common language used for android application development and contains extensive modules for networking. Android Studio was the integrated development environment used for the implementation of the detection system.

**4.0 TEST AND RESULTS**

This research implemented a solution in java as an android application to solve the DoS or DDoS vulnerability in Airdrop 2.0. The system was built to monitor the airdrop app on android operating sytem on tcp port 2935 which is the port on which airdrop listens for request as server.

Whenever there is a request for connection and it is granted but if such request persist for more than four times, our proposed app detects this situation as a Denial of Service attack on the airdrop app and will close connection on the client requesting.

However, at the time of development, we realized that as a result of closing the connection, the airdrop app closes abruptly as well. Further testing revealed the error was an unhandled exception.
This error was handled by creating an exception, that recursively opens up the airdrop app by starting the intent for the app.

**5.0 SUMMARY AND CONCLUSION**

Mobile applications vulnerability assessment is an aspect of vulnerability assessment that is not so common. However if this vulnerabilities are exploited they can have varying impact for users. From breach in confidentiality of file to remote code execution and even to denial of service and its distributed variant.

In this paper we highlighted some the security problems of the android OS and more specifically the problem of DoS in the airdrop 2.0 and a possible solution

The system designed provides a secure solution for the combating of the vulnerability. This was deployed as a background app and tested on the android OS.

Further study can be carried out on other mobile applications to access vulnerabilities as well as on mobile operating systems such as the IOS and android.

 

**REFERENCES**

Cedar101, R. F. (2020, March 2). *Vulnerability scanner*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Vulnerability_scanner

Exposures, C. V. (2019). *CVE-2019-9832*. Retrieved from Common Vulnerabilities and Exposures: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2019-9832

Murphey, D. (2019, June 27). *A history of information security*. Retrieved from IFSEC Global: https://www.ifsecglobal.com/cyber-security/a-history-of-information-security/

Rouse, M. (2019, October). *Cybersecurity*. Retrieved from TechTarget: https://searchsecurity.techtarget.com/definition/cybersecurity

Team, M. (2019, March 21). *Cyber Security Statistics for 2019*. Retrieved from Cyberdefencemagazine: HYPERLINK "https://www.cyberdefensemagazine.com/cyber-security-statistics-for-2019/" https://www.cyberdefensemagazine.com/cyber-security-statistics-for-2019/

A. Shabtai, Y. F. (2010). *Automated static analysis for classifying android applications.*

B. Sanz, I. S.-P. (2012). *On the automatic categorisation of android applications.*

Harang, W. G. (2013). *Rapid permissions-based detection of mobile malware.*

M. Ghorbanzadeh, Y. C. (2013). *A neural network approach to category validation of android appliations.*

R. Raveendranath, V. R. (July 2014). *Android malware attacks and countermeasures.* In Control, Instrumentation, Communication.

S. Feldman, D. S. (2014). *Manilyzer: Automated malware detetion.*

S.Y. Yerima, S. S. (2018). *A new android malware detection approach.*

Shabtai, A. (2010). *Malware detection on mobile devices.*

Song, P. C.-K. (2014). *Static detection of android malware.* ICMLC.

U. Pehlivan, N. B. (2014). *The analysis of feature selection methods and classification algorithms in permission based android malware detection.*

Vuong, M. A. (2013). *Random forest classification for deteting android malware.* GreenComiThings.

Wang, H.-Y. C.-D. (August 2015). *Machine learning based hybrid behaviour models for android malware analysis.*

Zhu., N. P. (2013). *Machine learning for android malware detection.*

Android. (2020, January 6). *Application Signing*. Retrieved from Android: https://source.android.com/security/apksigning

*Android Webview Exploitation*. (n.d.). Retrieved from HackingLoops: https://www.hackingloops.com/android-webview-exploitation/

*Application Security Terminology*. (n.d.). Retrieved from WhiteHat Security: https://www.whitehatsec.com/glossary/content/information-leakage

Dhankar, A. (2017, January 31). *What is Sandbox mode in Android?* Retrieved from Quora: http://quora.com/What-is-Sandbox-mode-in-Android#

Green, D. (2017, May 16). *Top 10 Vulnerabilities in Mobile Applications*. Retrieved from WhiteHat Security: https://www.whitehatsec.com/blog/top-10-vulnerabilities-in-mobile-applications/

Kinvey. (n.d.). *Acess Control*. Retrieved from Kinvey: https://devcenter.kinvey.com/android/guides/security

*Top 7 Vulnerabilities In Android Applications 2019*. (n.d.). Retrieved from Codersera: HYPERLINK "https://codersera.com/blog/top-7-vulnerabilities-in-android-applications-2019/" https://codersera.com/blog/top-7-vulnerabilities-in-android-applications-2019/

Rubens, P. (2018, June 26). *How To Prevent DDoS Attacks: Tips To Keep Your Website Safe*. Retrieved from eSecurity Planet: HYPERLINK "https://www.google.com/amp/s/www.esecurityplanet.com/amp/network-security/how-to-prevent-ddos-attacks.html" https://www.google.com/amp/s/www.esecurityplanet.com/amp/network-security/how-to-prevent-ddos-attacks.html

Vázquez, M. (2019, February 21). *AirDrop 2.0 - Denial of Service (DoS)*. Retrieved from Exploit Database: https://www.exploit-db.com/exploits/46445

Green, D. (2017, May 16). *Top 10 Vulnerabilities in Mobile Applications*. Retrieved from White Hat Security: https://www.whitehatsec.com/blog/top-10-vulnerabilities-in-mobile-applications/

Nations, D. (2020, March 25). *What Is AirDrop? How Does It Work?* Retrieved from Lifewire: https://www.lifewire.com/what-is-airdrop-how-does-it-work-1994512

Pbsouthwood, K. e. (2020, April 2). *Denial-of-service attack*. Retrieved from Wkikipedia: https://en.wikipedia.org/wiki/Denial-of-service_attack

Petters, J. (2020, March 3). *What is a DDoS Attack? Identifying Denial-of-Service Attacks*. Retrieved from Varonis: https://www.varonis.com/blog/what-is-a-ddos-attack/