# Security Analysis of the gas application and the API

Target System : Gas-app.apk and related web API

Date : 2020/02/02

Time : 20.00 PM

Tested by : Hashan Eranga (E/16/275)
: Thilini Deshika (E/16/078)
: Sudam Kalpage (E/16/168)

**(Please Note this Security analysis do not include gaining access, maintaining access, clearing tracks steps only vulnerability analysis is done)**

## Penetration Testing

Here are the basic steps to be followed by an attacker to exploit the system .

### Reconnaissance

Reverse Engineered the **gas-app.apk** using **apktool.** Using this decompiling method the attacker can find the access points(**ip address** and **port**) to the API. It can be found

**gas-app/sources/com/example/gasapp4/MainActivity.java**

Upon decoding the url. Using PHP decode exploited Url will be,

**http://18.206.64.63**

Packet sniffing is also a better method for information gathering

### Scanning

Once gaining that information. Using **nmap** available ports can be determined

**Nmap -Pn -sS -A  18.206.64.63**

Scanning results will be as follows.

Host discovery disabled (-Pn). All addresses will be marked 'up' and scan times will be slower.
Starting Nmap 7.91 ( https://nmap.org ) at 2021-02-03 04:46 EST
Nmap scan report for ec2-18-206-64-63.compute-1.amazonaws.com (18.206.64.63)
Host is up (0.27s latency).
Not shown: 996 filtered ports
PORT  STATE  SERVICE VERSION
22/tcp  open  ssh  OpenSSH 8.2p1 Ubuntu 4ubuntu0.1 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|  3072 a4:f0:c2:8c:47:40:45:66:21:ac:c6:d8:9e:ed:d5:dd (RSA)
|_  256 b9:f9:66:5b:91:8c:b4:1b:02:d3:e6:2a:93:20:26:28 (ECDSA)
80/tcp  closed http
9000/tcp open  http  Node.js (Express middleware)
|_http-title: Site doesn't have a title (text/html; charset=utf-8).
9001/tcp open  http  Node.js (Express middleware)
|_http-title: Site doesn't have a title (text/html; charset=utf-8).
Aggressive OS guesses: Linux 2.6.32 (90%), HP P2000 G3 NAS device (90%), Infomir MAG-250 set-top box (89%), Ubiquiti AirMax NanoStation WAP (Linux 2.6.32) (89%), Linux 5.0 - 5.4 (89%), Ubiquiti AirOS 5.5.9 (89%), Linux 2.6.32 - 3.13 (88%), Linux 3.3 (88%), Linux 2.6.32 - 3.1 (87%), Linux 3.7 (87%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 19 hops
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE (using port 80/tcp)
HOP RTT  ADDRESS
1  3.03 ms  192.168.8.1
2  ...
3  25.11 ms  10.174.66.65
4  ...
5  24.53 ms  125.214.190.223
6  20.26 ms  125.214.162.64
7  113.96 ms 195.229.27.81
8  237.99 ms 195.229.3.203
9  249.47 ms equinix02-iad2.amazon.com (206.126.236.35)
10  ... 12
13  263.52 ms 52.93.28.242
14  ... 18
19  267.44 ms ec2-18-206-64-63.compute-1.amazonaws.com (18.206.64.63)

OS and Service detection performed. Please report any incorrect results at ttps://nmap.org/submit/ .Nmap done: 1 IP address (1 host up) scanned in 110.08 seconds

## Conclusions on Scanning Process

1. SSH port is Open therefore an attacker can gain access through SSH. But the only way to gain access with a provided private key.
2. Port 9000 and Port 9001 tcp ports are opened. These ports are dedicated to the API. Also the server backend is running on a Node server.
3. Using traceroute the logical distance to the server can be determined as well as final ip with the service provider will reveal that this server is running on a cloud based system and running an EC2 instance.

## Major Security Flows in the system

1. Communication happens in both two ports as HTTP. Therefore if there will be an Man in the middle attack the attacker could be able to identify the information the user shares. But considering the relevance of the data they can ignore this matter. But changing the protocol to HTTPS is more convenient.
2. API route exploitation can be a threat. Packet sniffing is a better way to exploit routes to the API. Therefore HTTPS protocol is more convenient.

## Best security practices are followed in following implementations

1. The API gives no responses for ICMP ping command. This configuration is done when creating the inbound traffic rules to the system.
2. When developing the mysql database,
    a. Remove Users Without Password
    b. Tight Remote Access
    c. Remove Test Database
    d. Obfuscate Access to MySQL are followed.

End of the document