**DROZER DYNAMIC ANALYSIS**

**Table of Contents**

* Installation of Drozer in windows and Drozer agent on the device
* Help menu
* Shell commands
* Information gathering of device
* Information gathering of packages installed
* View debuggable packages!
* Dumping AndroidManifest.xml file
* Exploring the attack surface of a package
* Exploiting activities
* Exploiting content providers
* Exploiting services
* Exploiting broadcast receivers

**1. Installation of Drozer in windows and Drozer agent on the device**

1)First we need to install drozer client on your PC and drozer agent on the mobile device on emulator(in this case it is genymotion).

2)After the above step is done we need to connect our drozer client to the drozer agent.

3) To connect the drozer client running on the PC and the agent running on the emulator/device, we need to do port forward on port 31415 to connect the drozer client and drozer agent.

A screenshot of a computer screen

Description automatically generated with medium confidence

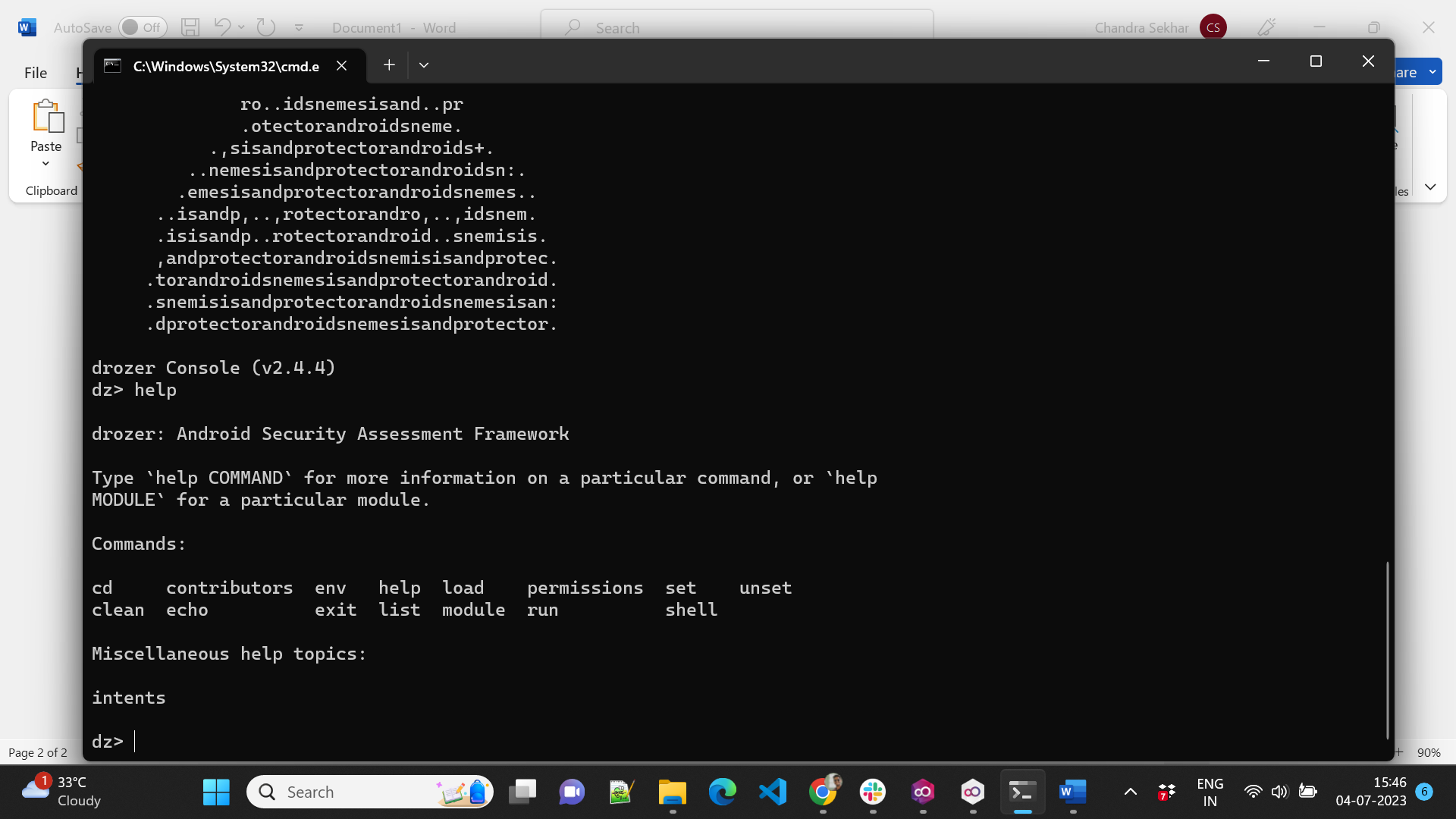
4) In order to run the drozer console in the terminal type command: drozer console connect.

A screenshot of a computer

Description automatically generated

**2. Help Menu**

Now that drozer is up and running, we’ll first look at all the modules that drozer has. Below, we can see all the various operations you can perform on activities, services, content providers, and broadcast receivers as well as some other scanners, information-gathering modules, and exploits



A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

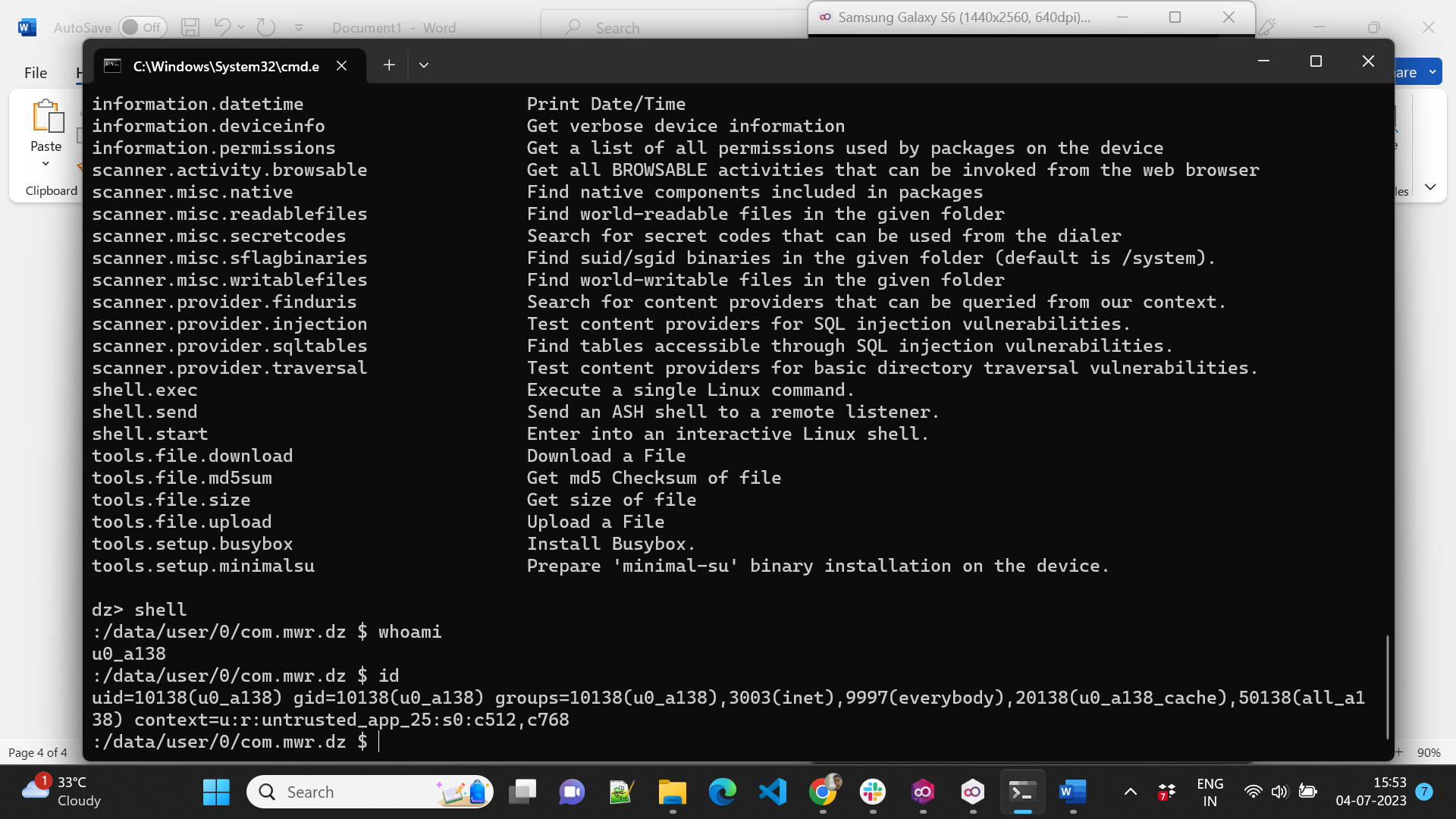
**3. Shell Command**

We can launch a shell on the device from within drozer console by:

“shell”

“whoami”

“id”

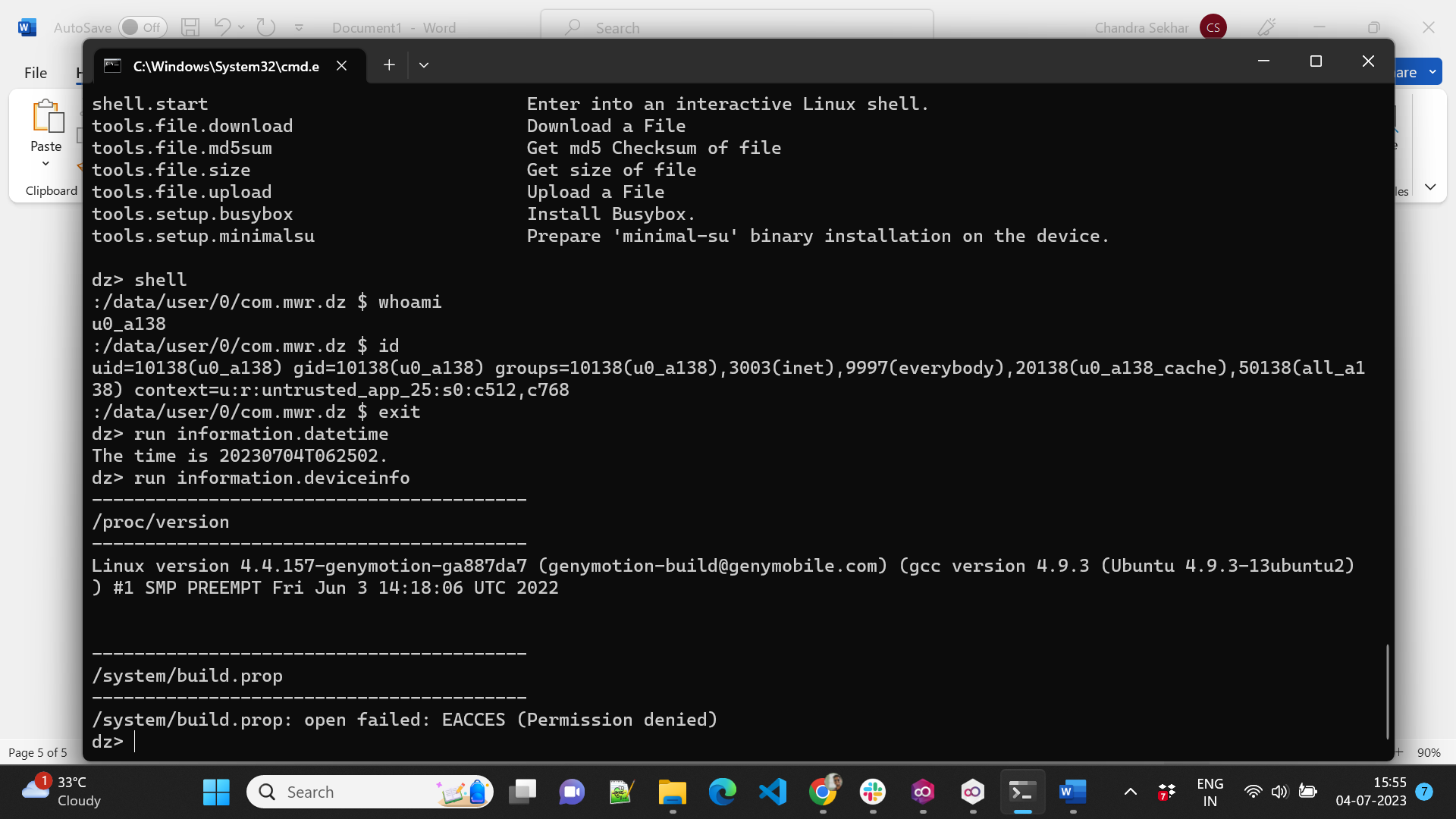


**4. Information Gathering on Device**

Drozer has a couple of modules to display date/time of the device and some other information on the device as well

“run information.datetime “

“run information.deviceinfo”



**5. Information Gathering on Packages**

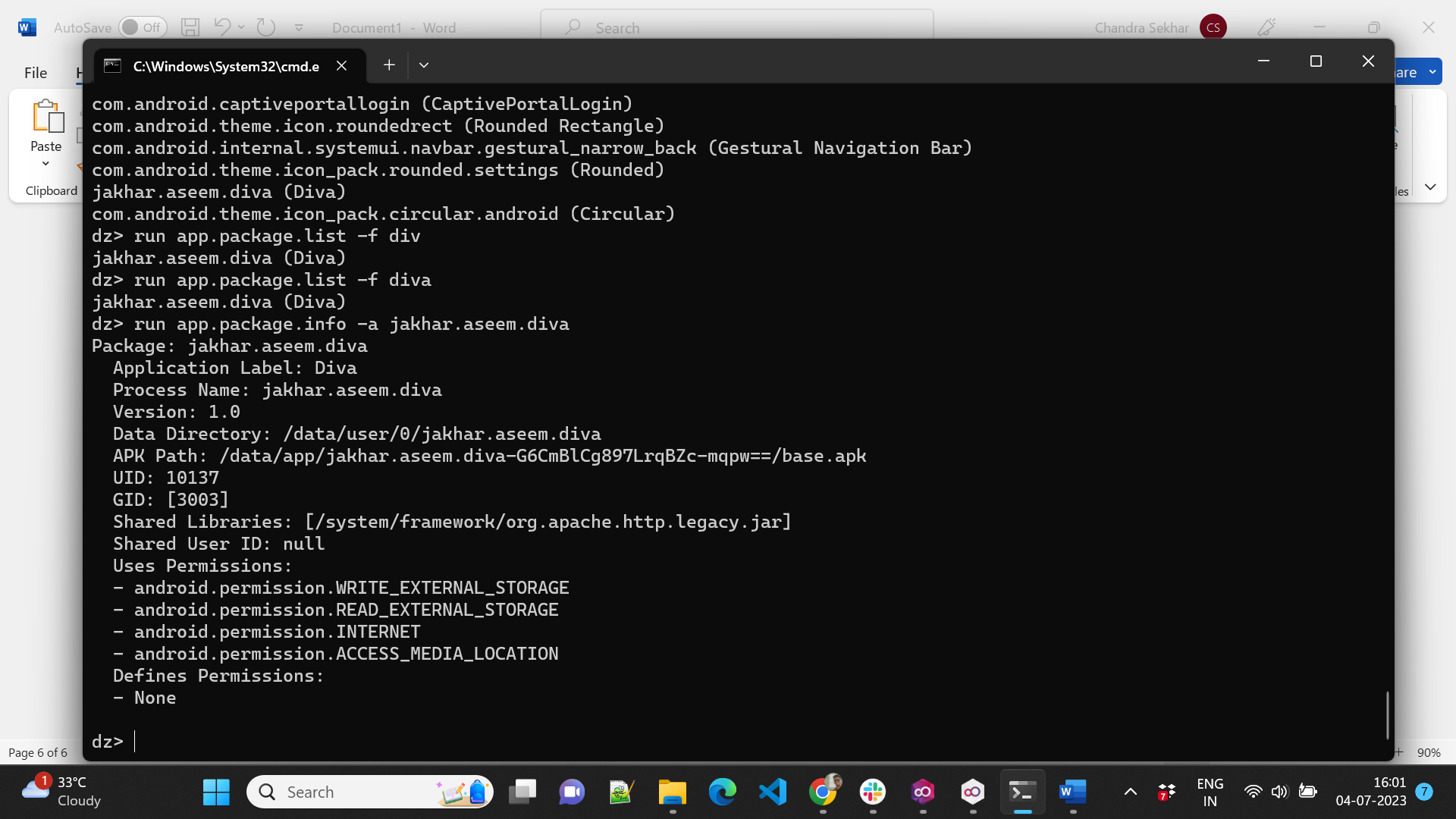
To list all the packages installed on the device, we run the following command:

“run app.package.list”

Further, to filter out the certain package we can apply the -f flag

“run app.package.list -f diva”

“run app.package.info -a jakhar.aseem.diva”



**6. Debuggable Packages**

If a certain package is marked debuggable, we can inject our custom code in it while run-time and modify its behaviour. For this we can manually check the manifest file for the string “android\_debuggable=”true”” or we can run the following drozer module:

“run app.package.debuggable”

A screenshot of a computer

Description automatically generated

**Mitigation:** One possible mitigation of this is to set “android\_debuggable=”false”” in the AndroidManifest.xml file.

**7. Dumping AndroidManifest.xml File**

To dump the manifest file of a package, we run the following command:

“run app.package.manifest jakhar.aseem.diva”

A screenshot of a computer

Description automatically generated

**8. Exploring Attack Surface of an Application**

One of the handiest features of Drozer is to identify the attack surface of an application. This module will give us information on the attack surface of an android application. Android applications have 4 essential components that can be exploited along with the debuggable flag. Run the following command to get the attack surface of any android application with a package name.

“run app.package.attacksurface jakhar.aseem.diva “

A screenshot of a computer

Description automatically generated

**9. Exploiting Activities**

An application may have exported activities that can be launched remotely and bypass various kinds of authentication mechanisms that the developer may have put on the class calling that activity. To check for all the exported activity, we have the following command:

“run app.activity.info -a jakhar.aseem.diva”

Now to launch an exported activity we can do this:

“run app.activity.start --component jakhar.aseem.diva jakhar.aseem.diva.APICredsActivity”

A screenshot of a computer

Description automatically generated

As we can see below, APICredsActivity has now been launched without any authentication

A screenshot of a computer

Description automatically generated

**10. Exploiting the exported content provider:**

To get more information about the exported content provider run the following command.

“run app.provider.info -a jakhar.aseem.diva”

A screenshot of a computer

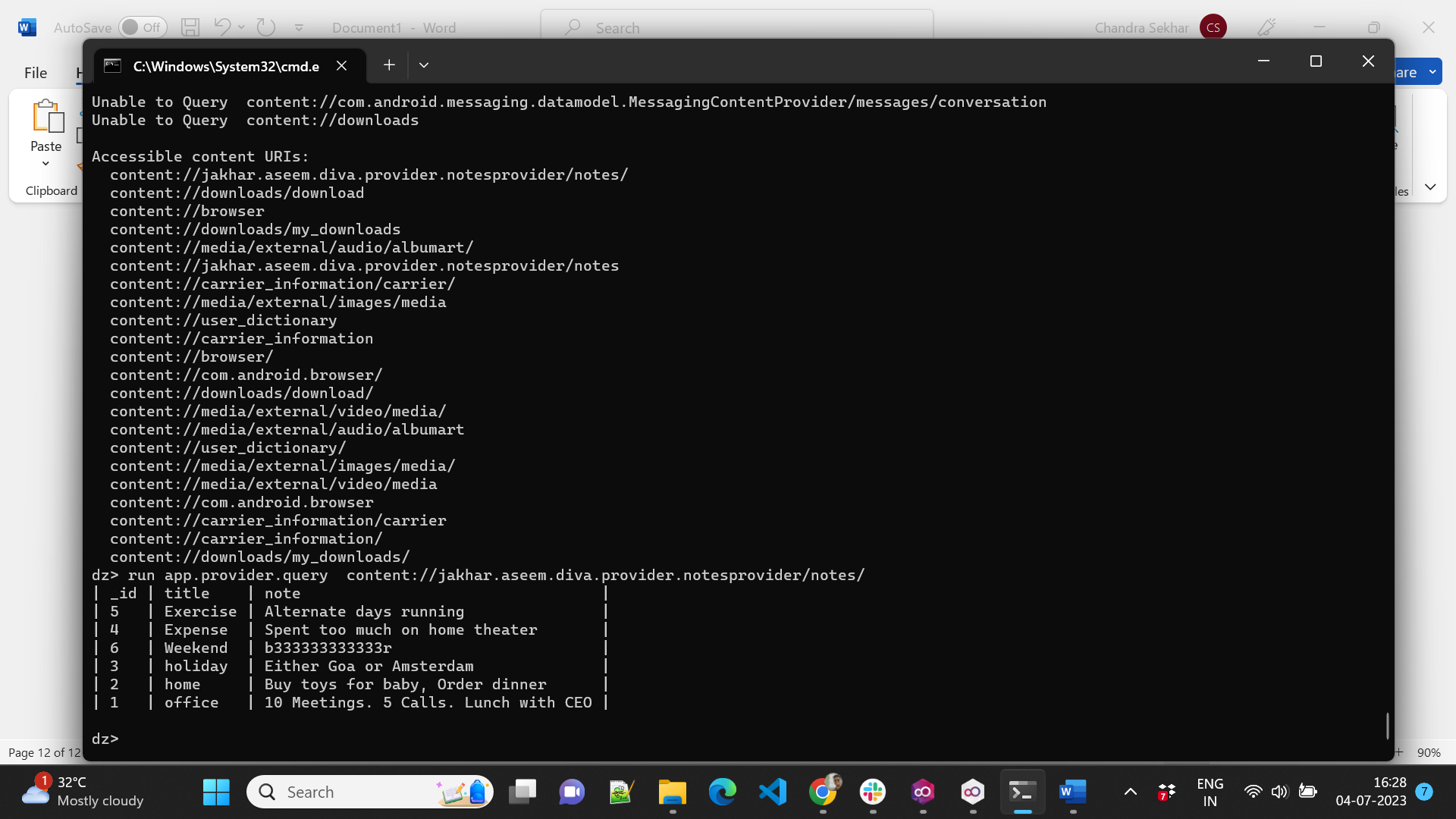
Description automatically generated

To fetch the data from the application’s content provider. Use the drozer scanner module “run scanner.provider.finduris” which gives the accessible URI’s.

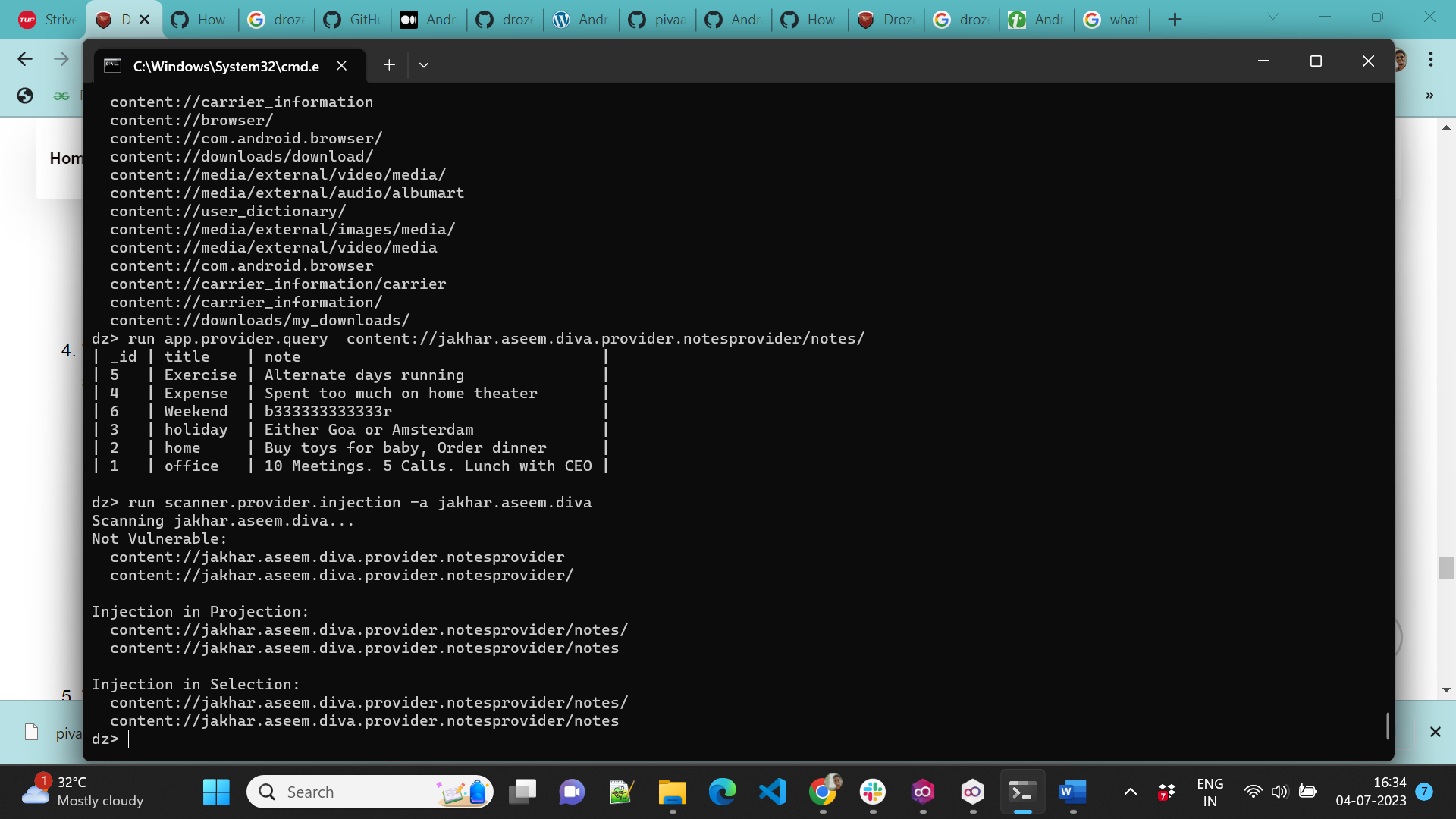
A screenshot of a computer

Description automatically generated

To query the URI’ from the content provider run the following command “run app.provider.query” with the accessible query.

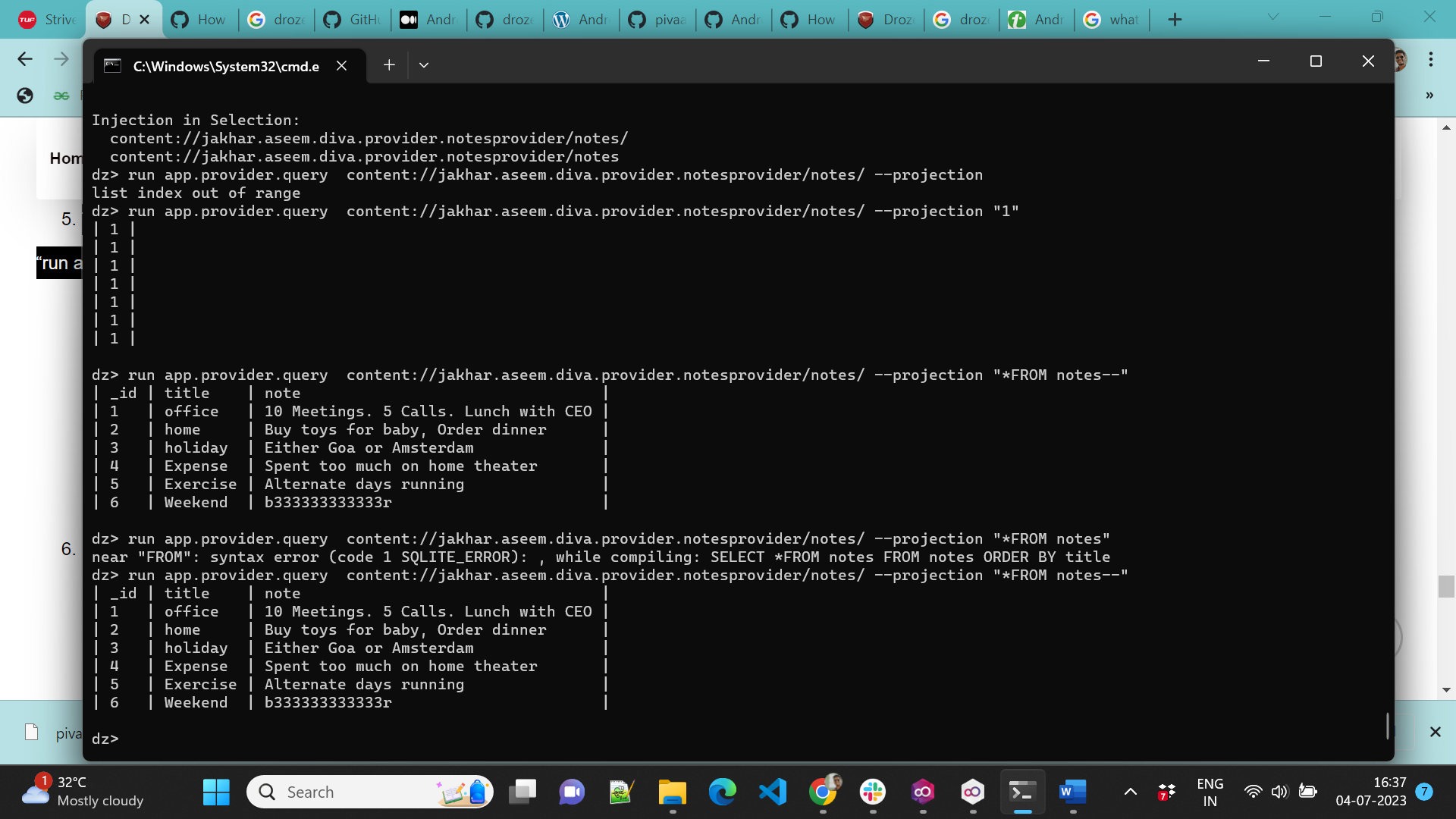


We can also use drozer to scan for SQL injection in content provider using the scanner module for injection scanner.provider.injection as shown in the below screenshot.

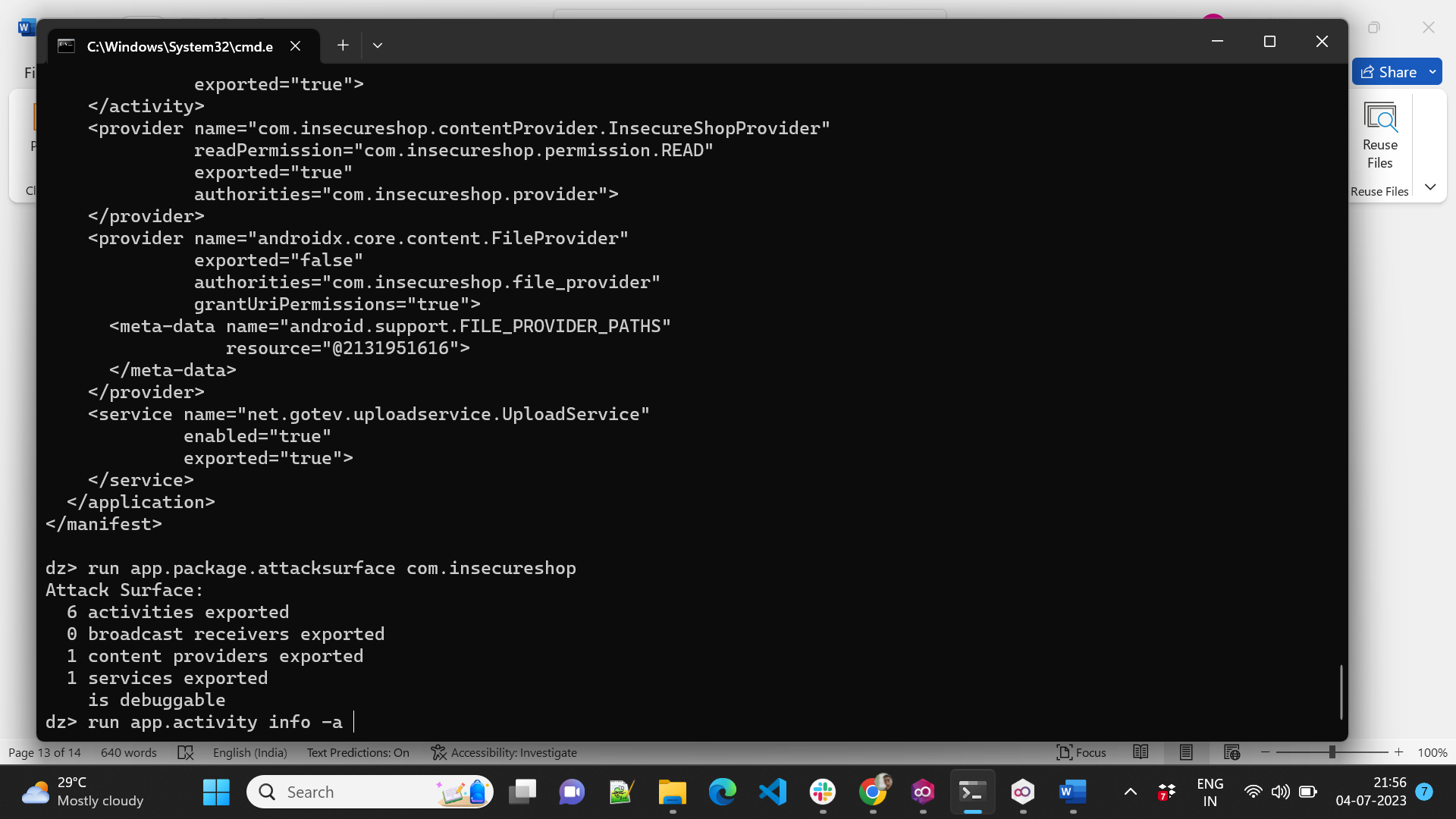


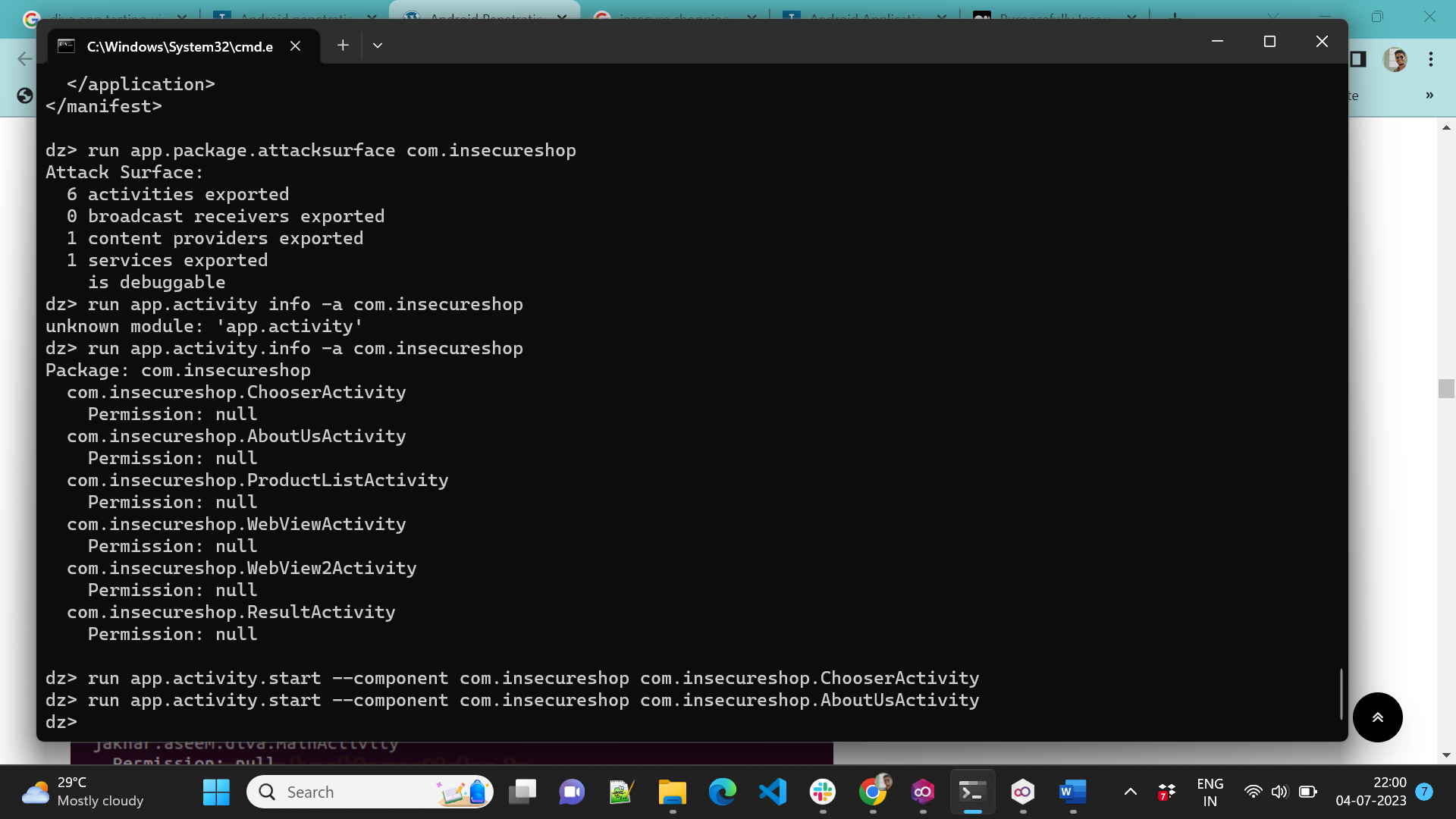
To exploit this vulnerability, we can use the following command.

“run app.provider.query” with the content URI as shown in the below screenshot.



**InsecureShop**





A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

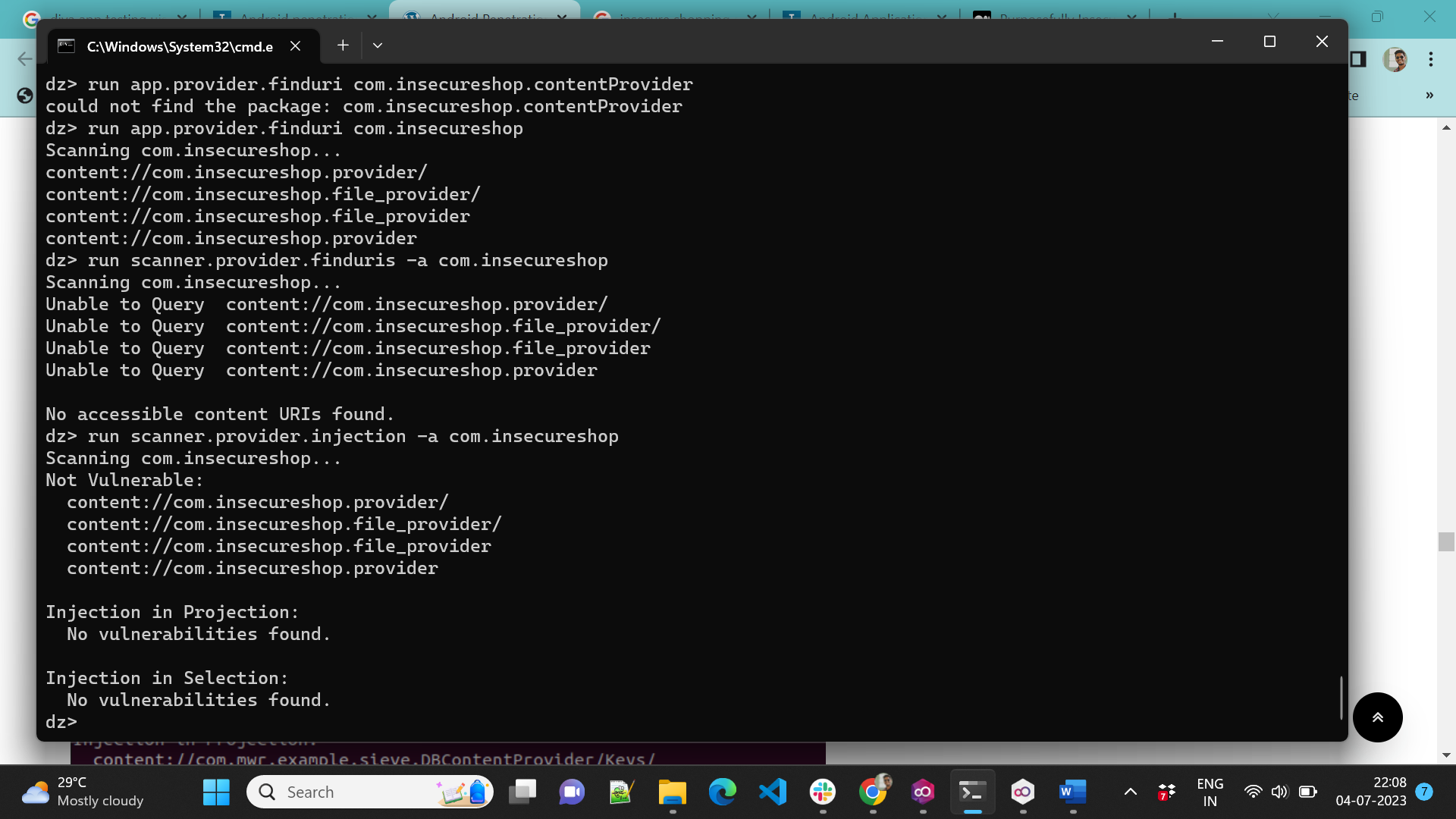
Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated



**AndroGoat**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

