FRIDA

Dynamic Android App Instrumentation

Team: Abdic Eldin, Rajic Djordje

What is Frida?

- instrumentation toolkit
- Great for:
  - reverse engineering
  - researching
  - development
- Highly scriptable and portable
- Works in black-box environments

Taken from: https://twitter.com/fridadotre?lang=en

Our project

- Get familiar with Frida
- Explore analysis capabilities of Frida
- Write scripts

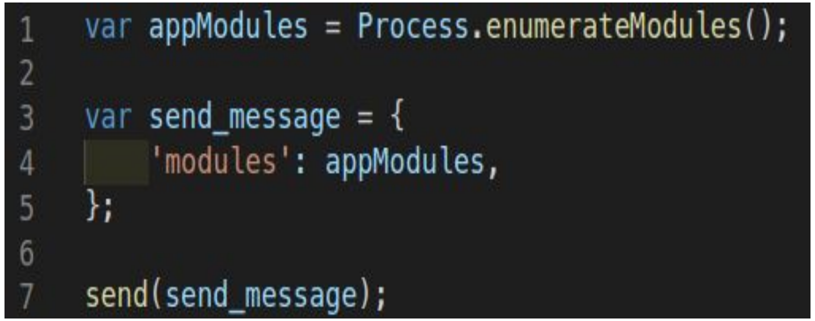
Tasks:

- Get familiar with Frida
- Explore analysis capabilities of Frida
- Write scripts

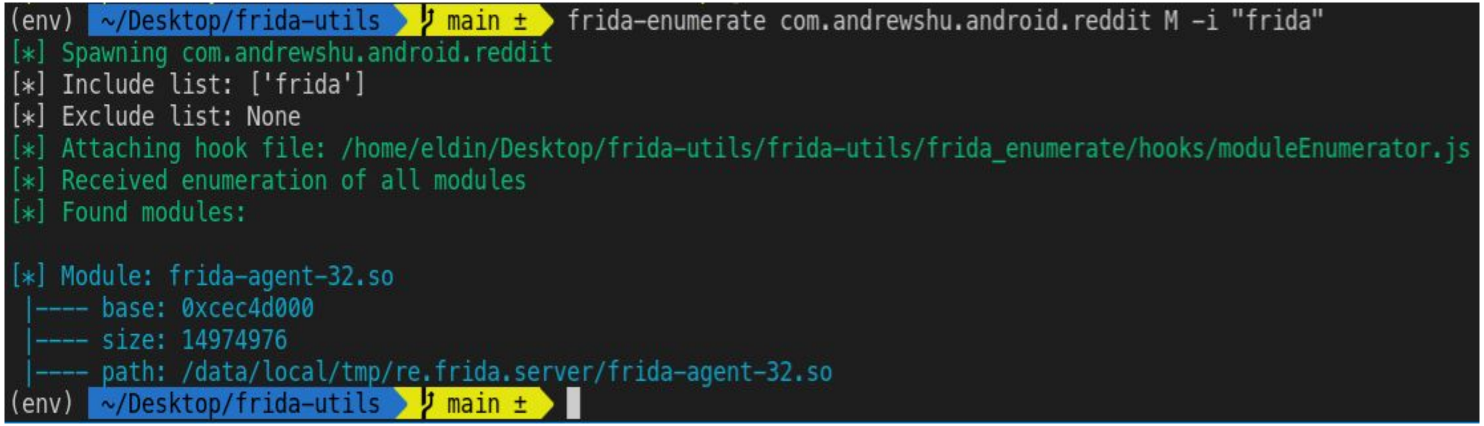
Ideas:

- We wanted to:
  - Write scripts under one framework
  - Allow modularity
  - Utilise Python bindings
- frida-utils - Python package for our scripts

frida-enumerate



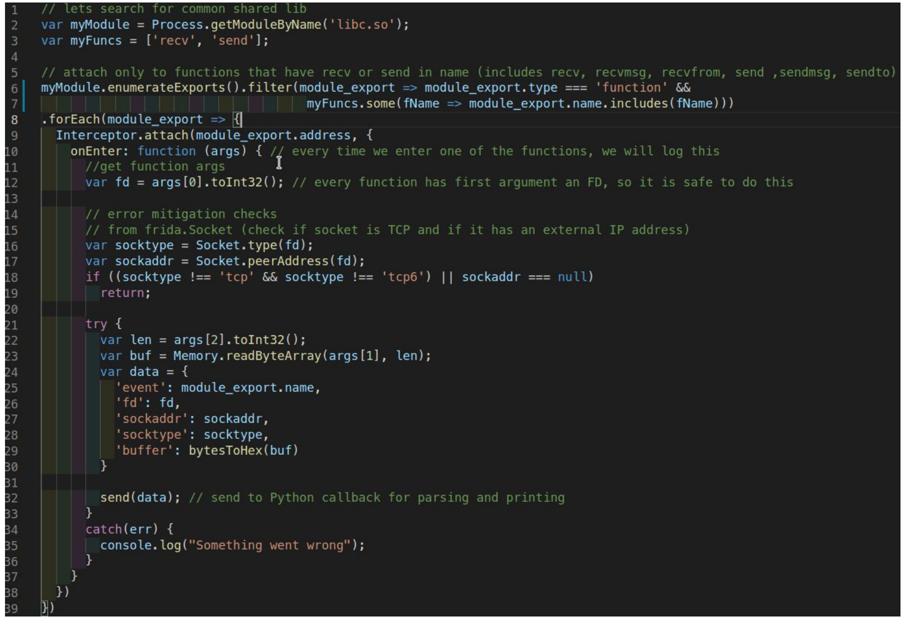
```
var appModules = Process.enumerateModules();

var send_message = {
    'modules': appModules,
};

send(send_message);
```



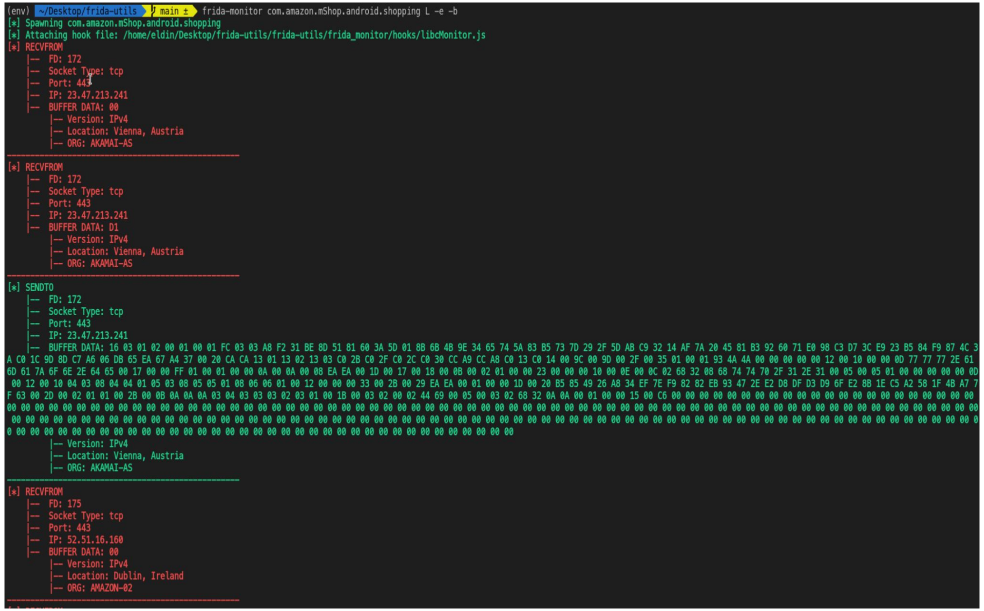
```
(env)  ~/Desktop/frida-utils   main ±  frida-enumerate com.andrewshu.android.reddit M -i "frida"
[*] Spawning com.andrewshu.android.reddit
[*] Include list: ['frida']
[*] Exclude list: None
[*] Attaching hook file: /home/eldin/Desktop/frida-utils/frida-utils/frida_enumerate/hooks/moduleEnumerator.js
[*] Received enumeration of all modules
[*] Found modules:

[*] Module: frida-agent-32.so
 |---- base: 0xcec4d000
 |---- size: 14974976
 |---- path: /data/local/tmp/re.frida.server/frida-agent-32.so
(env)  ~/Desktop/frida-utils   main ± 
```

frida-monitor (libc.so)

- Continuous monitoring using Interceptor
- address accesses interception and function overloading
- Monitor libc for network traffic



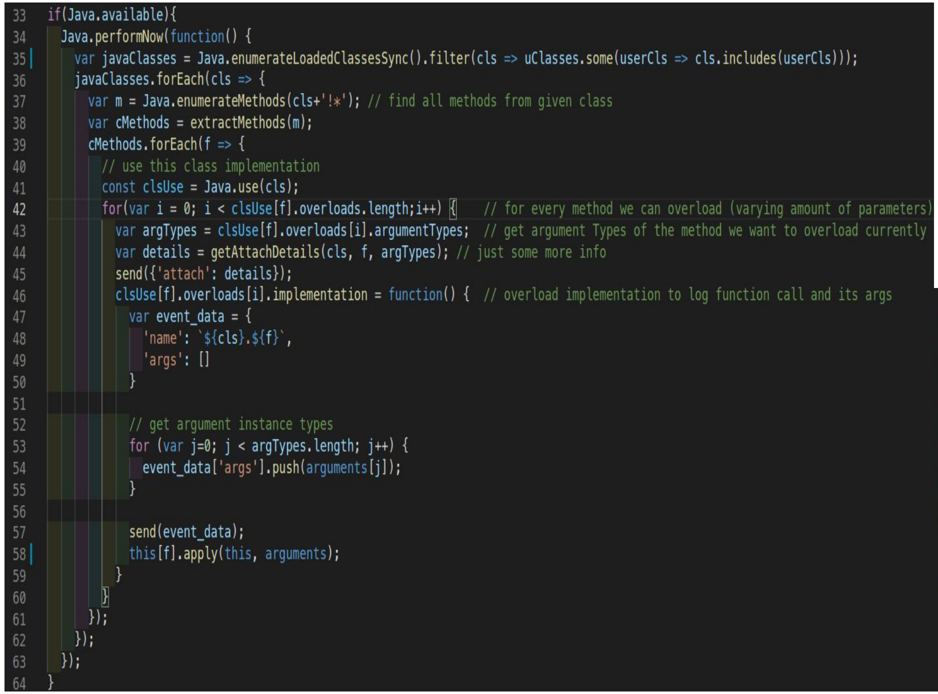
```javascript
// lets search for common shared lib
var myModule = Process.getModuleByName('libc.so');
var myFuncs = ['recv', 'send'];

// attach only to functions that have recv or send in name (includes recv, recvmsg, recvfrom, send ,sendmsg, sendto)
myModule.enumerateExports().filter(module_export => module_export.type === 'function' &&
                                    myFuncs.some(fName => module_export.name.includes(fName)))
.forEach(module_export => {
  Interceptor.attach(module_export.address, {
    onEnter: function (args) { // every time we enter one of the functions, we will log this
      //get function args
      var fd = args[0].toInt32(); // every function has first argument an FD, so it is safe to do this

      // error mitigation checks
      // from frida.Socket (check if socket is TCP and if it has an external IP address)
      var socktype = Socket.type(fd);
      var sockaddr = Socket.peerAddress(fd);
      if ((socktype !== 'tcp' && socktype !== 'tcp6') || sockaddr === null)
        return;

      try {
        var len = args[2].toInt32();
        var buf = Memory.readByteArray(args[1], len);
        var data = {
          'event': module_export.name,
          'fd': fd,
          'sockaddr': sockaddr,
          'socktype': socktype,
          'buffer': bytesToHex(buf)
        }

        send(data); // send to Python callback for parsing and printing
      }
      catch(err) {
        console.log("Something went wrong");
      }
    }
  })
})
```



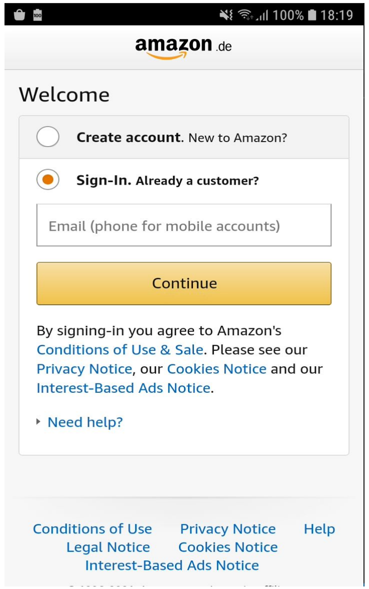
```
(env) ~/Desktop/frida-utils  main ±  frida-monitor com.amazon.mShop.android.shopping L -e -b
[*] Spawning com.amazon.mShop.android.shopping
[*] Attaching hook file: /home/eldin/Desktop/frida-utils/frida-utils/frida_monitor/hooks/libcMonitor.js
[*] RECVFROM
    |--  FD: 172
    |--  Socket Type: tcp
    |--  Port: 443
    |--  IP: 23.47.213.241
    |--  BUFFER DATA: 00
         |-- Version: IPv4
         |-- Location: Vienna, Austria
         |-- ORG: AKAMAI-AS
--------------------------------------------------
[*] RECVFROM
    |--  FD: 172
    |--  Socket Type: tcp
    |--  Port: 443
    |--  IP: 23.47.213.241
    |--  BUFFER DATA: D1
         |-- Version: IPv4
         |-- Location: Vienna, Austria
         |-- ORG: AKAMAI-AS
--------------------------------------------------
[*] SENDTO
    |--  FD: 172
    |--  Socket Type: tcp
    |--  Port: 443
    |--  IP: 23.47.213.241
    |--  BUFFER DATA: 16 03 01 02 00 01 00 01 FC 03 03 A8 F2 31 BE 8D 51 81 60 3A 5D 01 8B 6B 4B 9E 34 65 74 5A 83 B5 73 7D 29 2F 5D AB C9 32 14 AF 7A 20 45 81 B3 92 60 71 E0 98 C3 D7 3C E9 23 B5 84 F9 87 4C 3
A C0 1C 9D 8D C7 A6 06 DB 65 EA 67 A4 37 00 20 CA CA 13 01 13 02 13 03 C0 2B C0 2F C0 2C C0 30 CC A9 CC A8 C0 13 C0 14 00 9C 00 9D 00 2F 00 35 01 00 01 93 4A 4A 00 00 00 00 00 12 00 10 00 00 0D 77 77 77 2E 61
6D 61 7A 6F 6E 2E 64 65 00 17 00 00 FF 01 00 01 00 00 0A 00 0A 00 08 EA EA 00 1D 00 17 00 18 00 0B 00 02 01 00 00 23 00 00 00 10 00 0E 00 0C 02 68 32 08 68 74 74 70 2F 31 2E 31 00 05 00 05 01 00 00 00 00 00 0D
 00 12 00 10 04 03 08 04 04 01 05 03 08 05 05 01 08 06 06 01 00 12 00 00 00 33 00 2B 00 29 EA EA 00 01 00 00 1D 00 20 B5 85 49 26 A8 34 EF 7E F9 82 82 EB 93 47 2E E2 D8 DF D3 D9 6F E2 8B 1E C5 A2 58 1F 4B A7 7
F 63 00 2D 00 02 01 01 00 2B 00 0B 0A 0A 0A 03 04 03 03 03 02 03 01 00 1B 00 03 02 00 02 44 69 00 05 00 03 02 68 32 0A 0A 00 01 00 00 15 00 C6 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0
0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
         |-- Version: IPv4
         |-- Location: Vienna, Austria
         |-- ORG: AKAMAI-AS
--------------------------------------------------
[*] RECVFROM
    |--  FD: 175
    |--  Socket Type: tcp
    |--  Port: 443
    |--  IP: 52.51.16.160
    |--  BUFFER DATA: 00
         |-- Version: IPv4
         |-- Location: Dublin, Ireland
         |-- ORG: AMAZON-02
--------------------------------------------------
```

frida-monitor (javax.crypto.Cipher)

- Highlight Java crypto.Cipher calls



```javascript
if(Java.available){
  Java.performNow(function() {
    var javaClasses = Java.enumerateLoadedClassesSync().filter(cls => uClasses.some(userCls => cls.includes(userCls)));
    javaClasses.forEach(cls => {
      var m = Java.enumerateMethods(cls+'!*'); // find all methods from given class
      var cMethods = extractMethods(m);
      cMethods.forEach(f => {
        // use this class implementation
        const clsUse = Java.use(cls);
        for(var i = 0; i < clsUse[f].overloads.length;i++) {    // for every method we can overload (varying amount of parameters)
          var argTypes = clsUse[f].overloads[i].argumentTypes;  // get argument Types of the method we want to overload currently
          var details = getAttachDetails(cls, f, argTypes); // just some more info
          send({'attach': details});
          clsUse[f].overloads[i].implementation = function() {  // overload implementation to log function call and its args
            var event_data = {
              'name': `${cls}.${f}`,
              'args': []
            }

            // get argument instance types
            for (var j=0; j < argTypes.length; j++) {
              event_data['args'].push(arguments[j]);
            }

            send(event_data);
            this[f].apply(this, arguments);
          }
        }
      });
    });
  });
}
```



amazon.de

Welcome

Create account. New to Amazon?

[✓ Sign-In. Already a customer?]

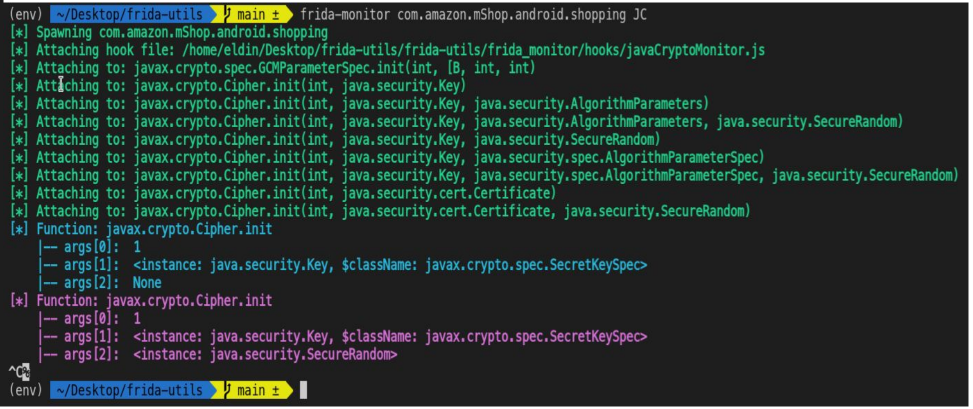
Email (phone for mobile accounts)

Continue

By signing-in you agree to Amazon's Conditions of Use & Sale. Please see our Privacy Notice, our Cookies Notice and our Interest-Based Ads Notice.

Need help?

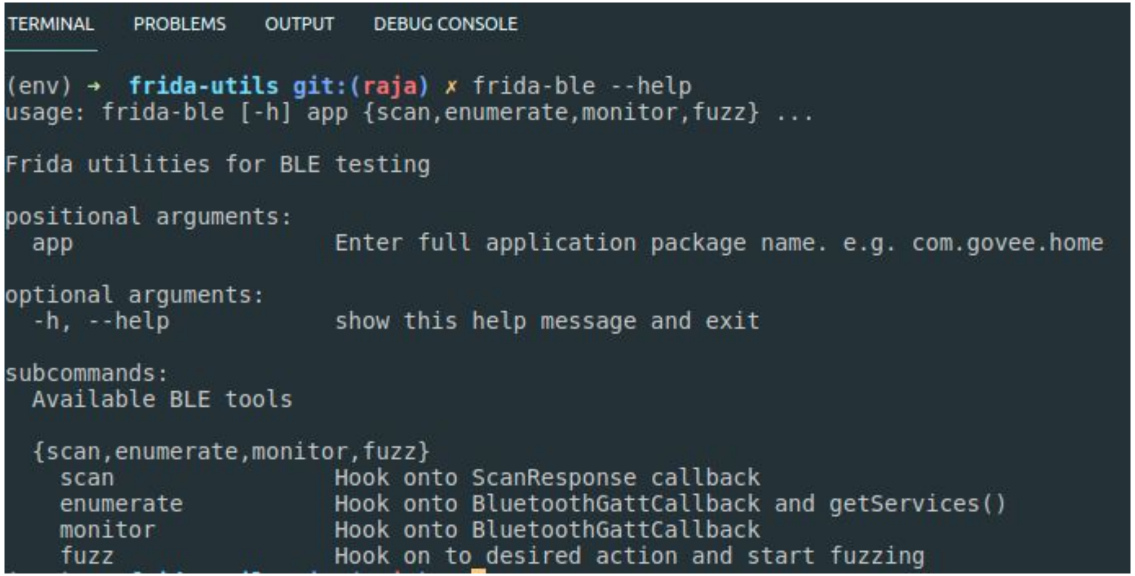
Conditions of Use Privacy Notice Help
Legal Notice Cookies Notice
Interest-Based Ads Notice



```
(env) ~/Desktop/frida-utils  main ±  frida-monitor com.amazon.mShop.android.shopping JC
[*] Spawning com.amazon.mShop.android.shopping
[*] Attaching hook file: /home/eldin/Desktop/frida-utils/frida-utils/frida_monitor/hooks/javaCryptoMonitor.js
[*] Attaching to: javax.crypto.spec.GCMParameterSpec.init(int, [B, int, int)
[*] Attaching to: javax.crypto.Cipher.init(int, java.security.Key)
[*] Attaching to: javax.crypto.Cipher.init(int, java.security.Key, java.security.AlgorithmParameters)
[*] Attaching to: javax.crypto.Cipher.init(int, java.security.Key, java.security.AlgorithmParameters, java.security.SecureRandom)
[*] Attaching to: javax.crypto.Cipher.init(int, java.security.Key, java.security.SecureRandom)
[*] Attaching to: javax.crypto.Cipher.init(int, java.security.Key, java.security.spec.AlgorithmParameterSpec)
[*] Attaching to: javax.crypto.Cipher.init(int, java.security.Key, java.security.spec.AlgorithmParameterSpec, java.security.SecureRandom)
[*] Attaching to: javax.crypto.Cipher.init(int, java.security.cert.Certificate)
[*] Attaching to: javax.crypto.Cipher.init(int, java.security.cert.Certificate, java.security.SecureRandom)
[*] Function: javax.crypto.Cipher.init
    |-- args[0]:  1
    |-- args[1]:  <instance: java.security.Key, $className: javax.crypto.spec.SecretKeySpec>
    |-- args[2]:  None
[*] Function: javax.crypto.Cipher.init
    |-- args[0]:  1
    |-- args[1]:  <instance: java.security.Key, $className: javax.crypto.spec.SecretKeySpec>
    |-- args[2]:  <instance: java.security.SecureRandom>
^C%
(env) ~/Desktop/frida-utils  main ± 
```

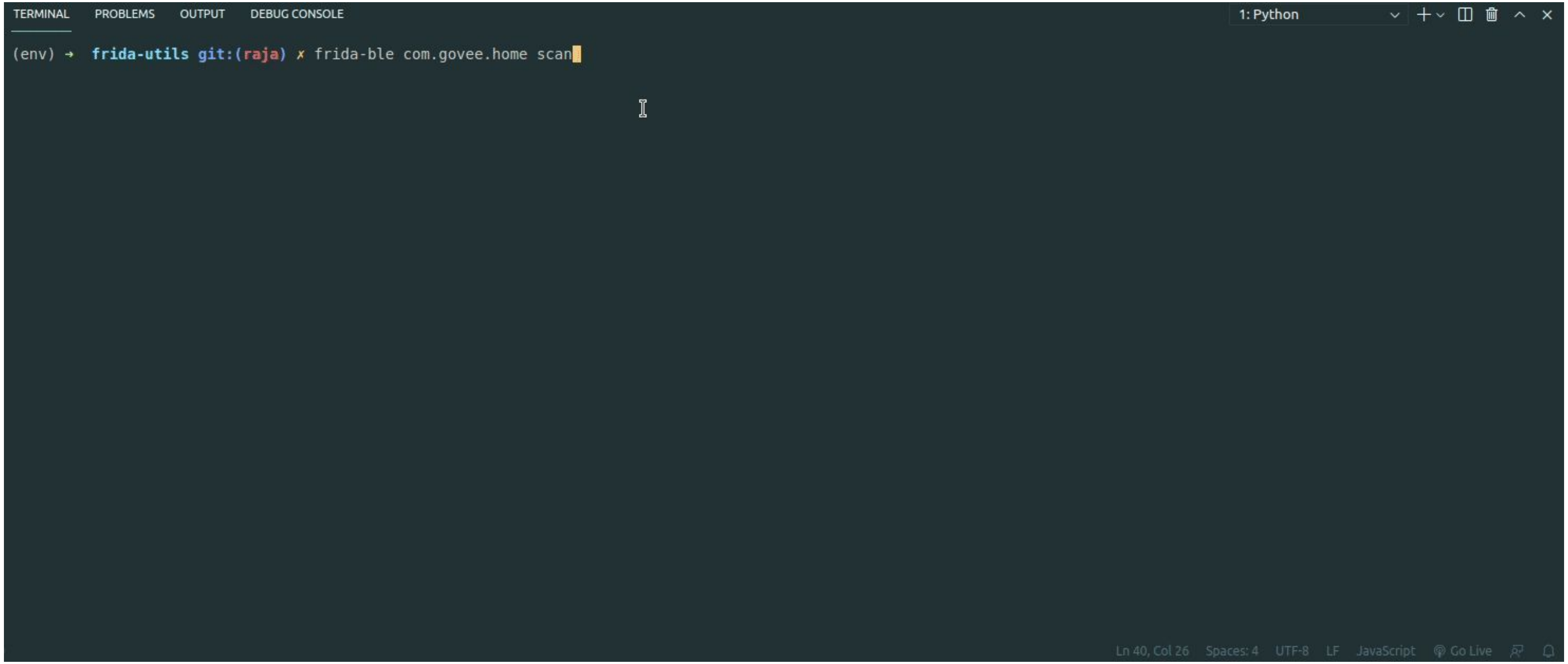
BLE Tool

- Analyse BLE applications and BLE devices
- Check scan data
- Enumerate device - get complete GATT profile
- Monitor the communication
- Fuzz application and possibly BLE device
- WHY?
  - Cheap
  - Fast
  - Does not need additional HW
  - Flexible



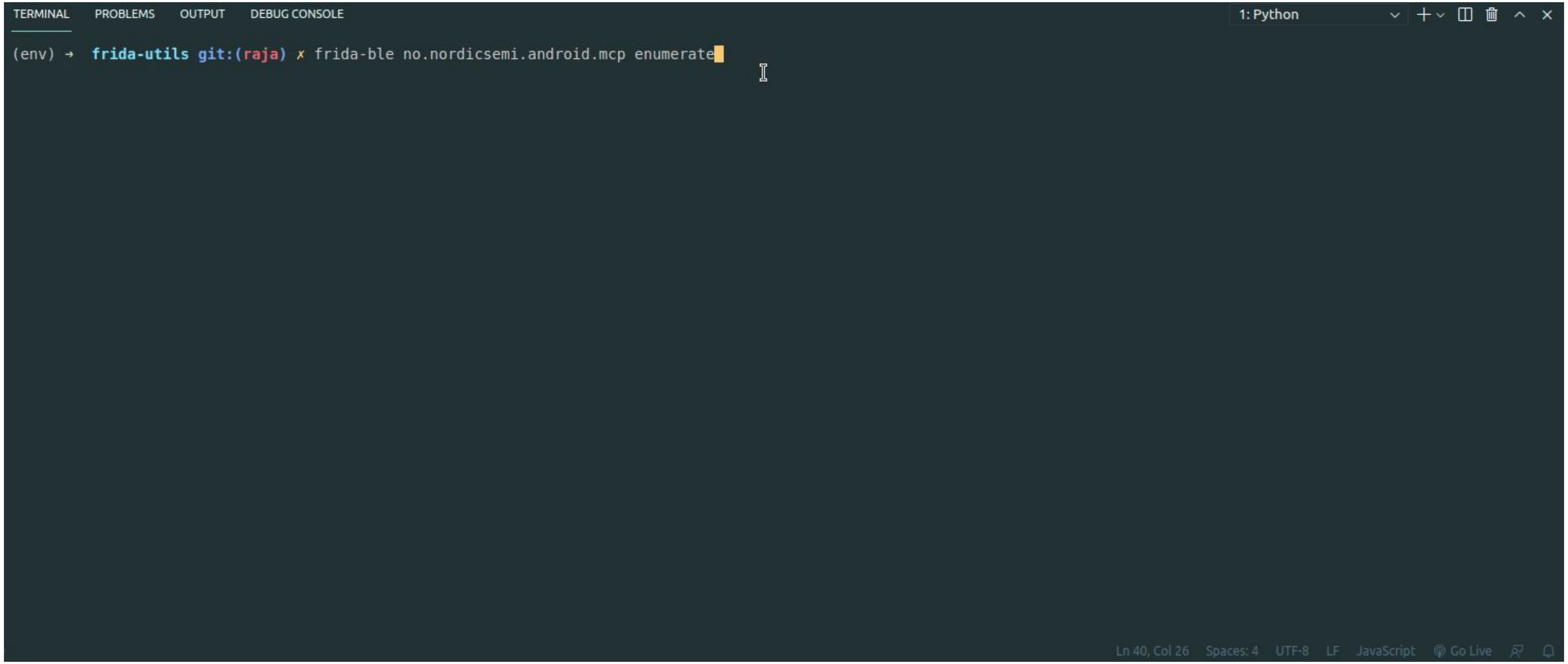
```
(env) ➜  frida-utils git:(raja) ✗ frida-ble --help
usage: frida-ble [-h] app {scan,enumerate,monitor,fuzz} ...

Frida utilities for BLE testing

positional arguments:
  app                   Enter full application package name. e.g. com.govee.home

optional arguments:
  -h, --help            show this help message and exit

subcommands:
  Available BLE tools

  {scan,enumerate,monitor,fuzz}
    scan                Hook onto ScanResponse callback
    enumerate           Hook onto BluetoothGattCallback and getServices()
    monitor             Hook onto BluetoothGattCallback
    fuzz                Hook on to desired action and start fuzzing
```

BLE Scan



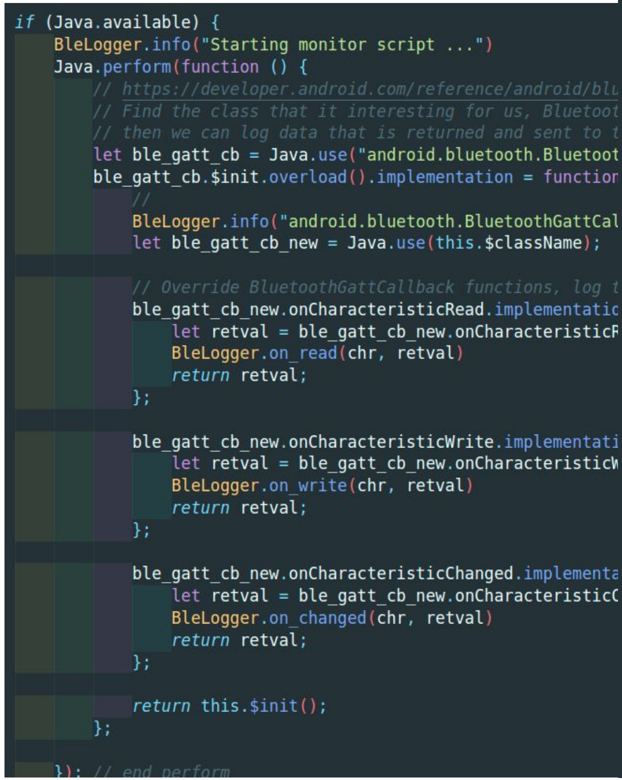
```
(env) ➜  frida-utils git:(raja) ✗ frida-ble com.govee.home scan
```

BLE Enumerate

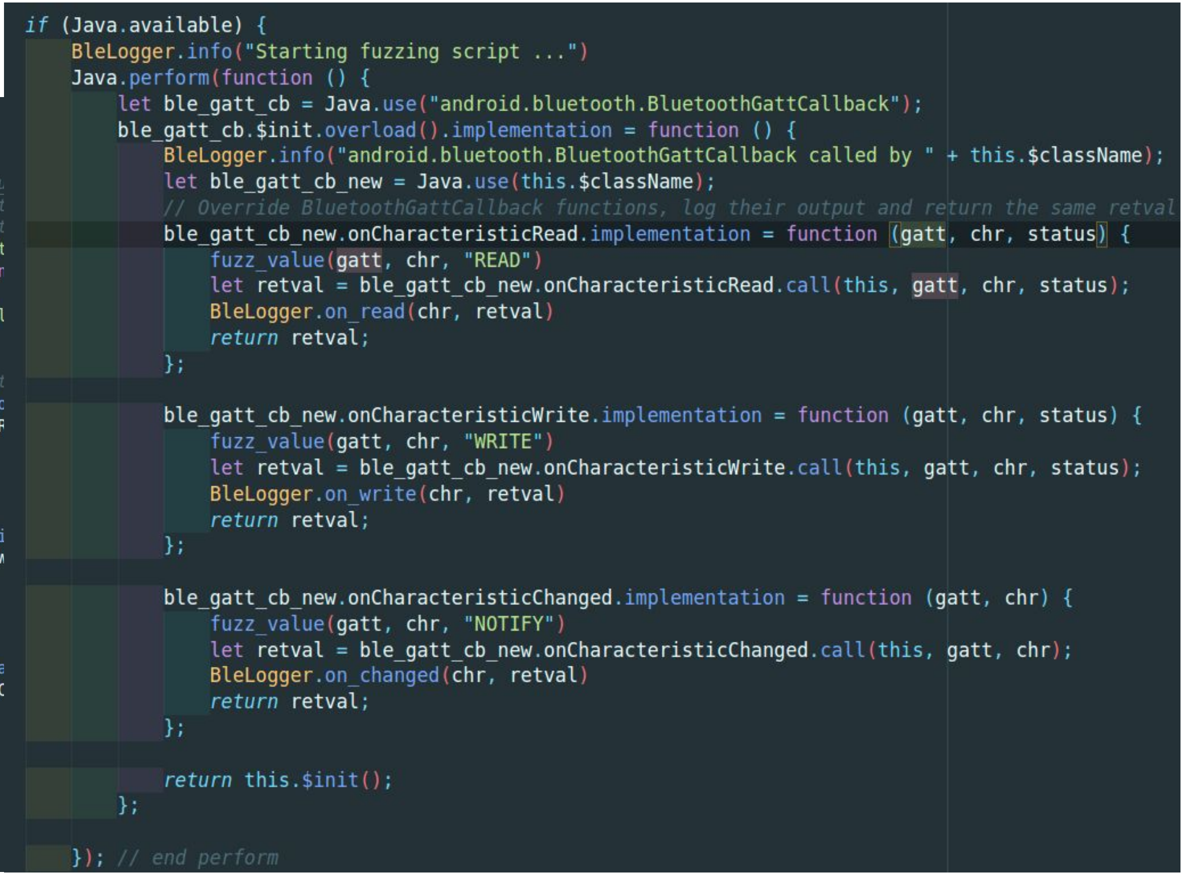


```
(env) ➜  frida-utils git:(raja) ✗ frida-ble no.nordicsemi.android.mcp enumerate
```

BLE Code

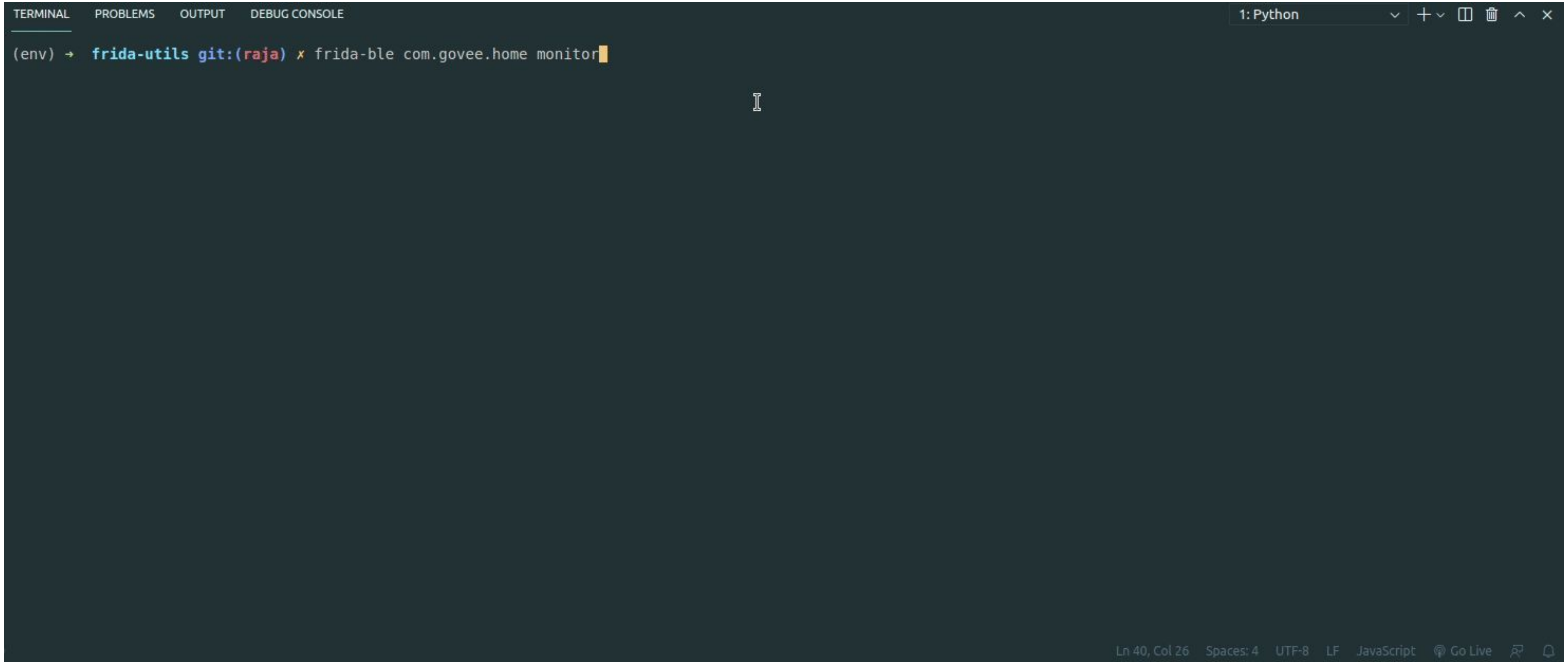


```
if (Java.available) {
    BleLogger.info("Starting monitor script ...")
    Java.perform(function () {
        // https://developer.android.com/reference/android/blu
        // Find the class that it interesting for us, Bluetoot
        // then we can log data that is returned and sent to t
        let ble_gatt_cb = Java.use("android.bluetooth.Bluetoot
        ble_gatt_cb.$init.overload().implementation = functior
            //
            BleLogger.info("android.bluetooth.BluetoothGattCal
            let ble_gatt_cb_new = Java.use(this.$className);

            // Override BluetoothGattCallback functions, log t
            ble_gatt_cb_new.onCharacteristicRead.implementatic
                let retval = ble_gatt_cb_new.onCharacteristicF
                BleLogger.on_read(chr, retval)
                return retval;
            };

            ble_gatt_cb_new.onCharacteristicWrite.implementati
                let retval = ble_gatt_cb_new.onCharacteristicW
                BleLogger.on_write(chr, retval)
                return retval;
            };

            ble_gatt_cb_new.onCharacteristicChanged.implementa
                let retval = ble_gatt_cb_new.onCharacteristicC
                BleLogger.on_changed(chr, retval)
                return retval;
            };

            return this.$init();
        };

    }); // end perform
```



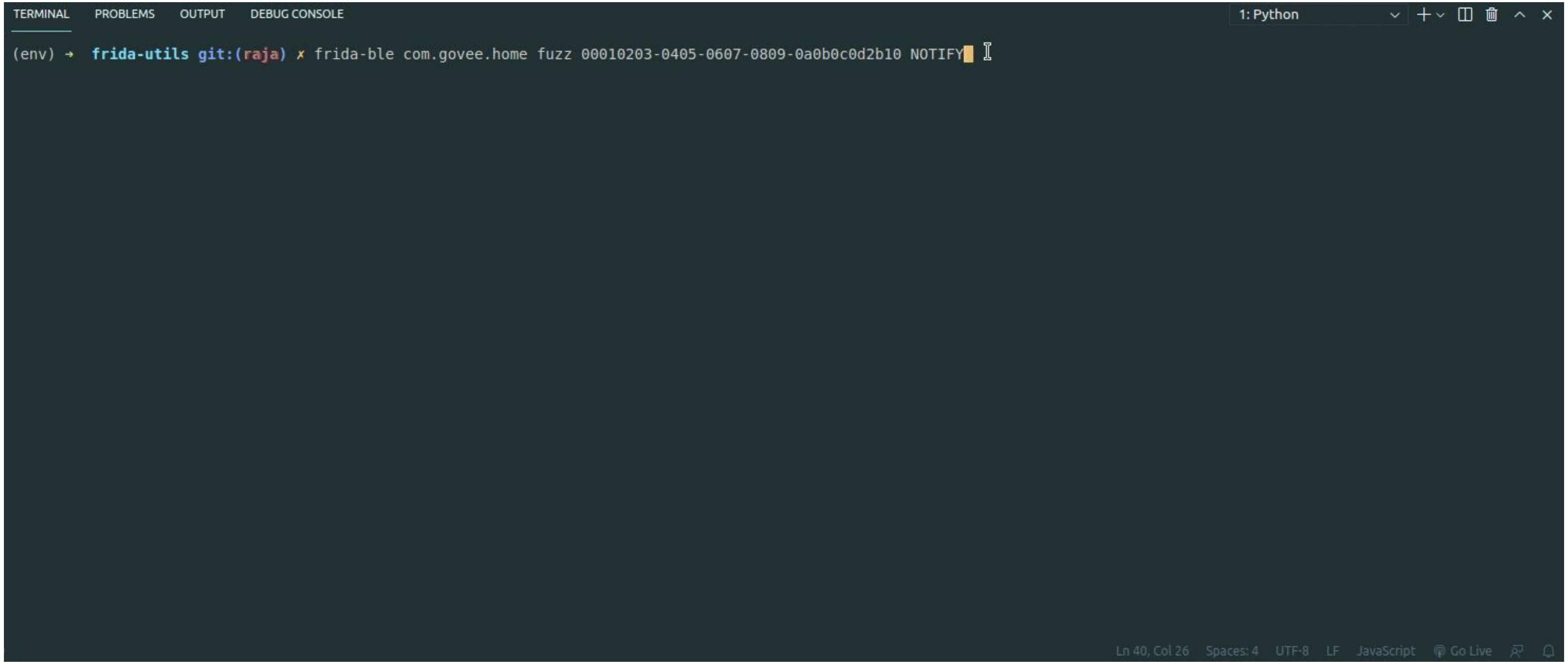
```
if (Java.available) {
    BleLogger.info("Starting fuzzing script ...")
    Java.perform(function () {
        let ble_gatt_cb = Java.use("android.bluetooth.BluetoothGattCallback");
        ble_gatt_cb.$init.overload().implementation = function () {
            BleLogger.info("android.bluetooth.BluetoothGattCallback called by " + this.$className);
            let ble_gatt_cb_new = Java.use(this.$className);
            // Override BluetoothGattCallback functions, log their output and return the same retval
            ble_gatt_cb_new.onCharacteristicRead.implementation = function (gatt, chr, status) {
                fuzz_value(gatt, chr, "READ")
                let retval = ble_gatt_cb_new.onCharacteristicRead.call(this, gatt, chr, status);
                BleLogger.on_read(chr, retval)
                return retval;
            };

            ble_gatt_cb_new.onCharacteristicWrite.implementation = function (gatt, chr, status) {
                fuzz_value(gatt, chr, "WRITE")
                let retval = ble_gatt_cb_new.onCharacteristicWrite.call(this, gatt, chr, status);
                BleLogger.on_write(chr, retval)
                return retval;
            };

            ble_gatt_cb_new.onCharacteristicChanged.implementation = function (gatt, chr) {
                fuzz_value(gatt, chr, "NOTIFY")
                let retval = ble_gatt_cb_new.onCharacteristicChanged.call(this, gatt, chr);
                BleLogger.on_changed(chr, retval)
                return retval;
            };

            return this.$init();
        };

    }); // end perform
```

BLE Monitor



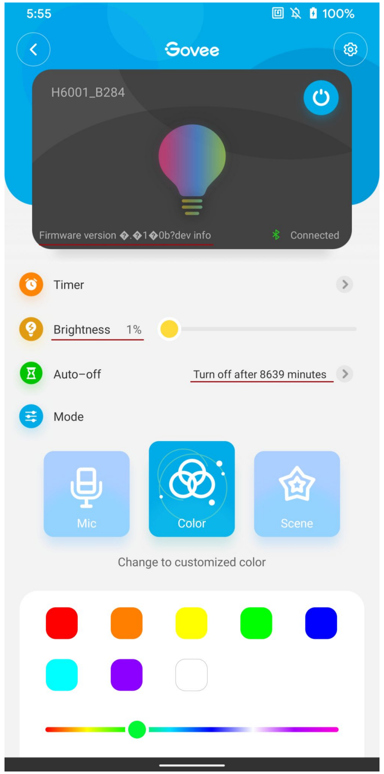
```
(env) ➜  frida-utils git:(raja) ✗ frida-ble com.govee.home monitor
```

BLE Fuzzing



```
(env) ➜  frida-utils git:(raja) ✗ frida-ble com.govee.home fuzz 00010203-0405-0607-0809-0a0b0c0d2b10 NOTIFY
```

BLE Fuzzing



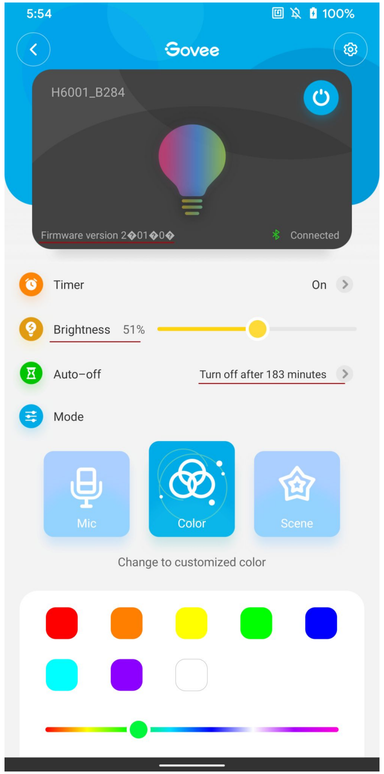
Firmware version �.�1�0b?dev info

Connected

Brightness 1%

Auto-off Turn off after 8639 minutes

Change to customized color



Firmware version 2�01�0�

Connected

Timer On

Brightness 51%

Auto-off Turn off after 183 minutes

Change to customized color

Thank you for your attention !