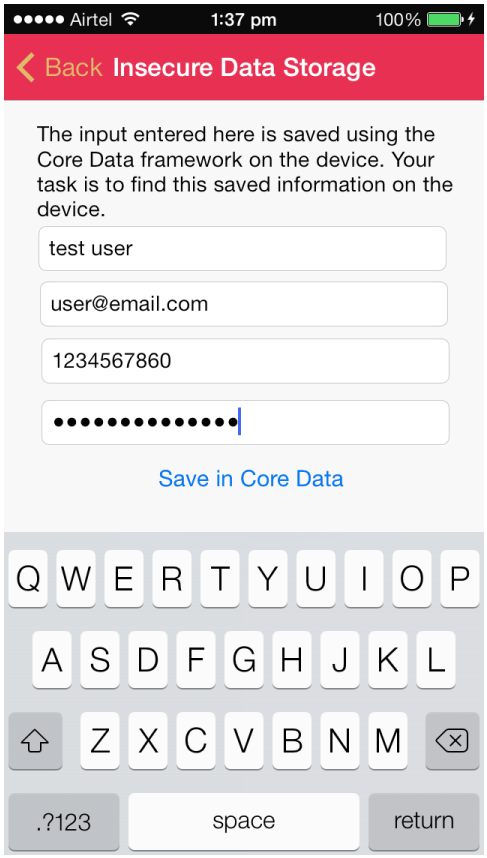
Damn Vulnerable IOS Application Solutions
http://damnvulnerableiosapp.com/

Insecure Data Storage – Core Data

Let's enter some dummy information and tap on *Save in Core Data*



Now lets ssh into our device and go to the directory */var/mobile/Applications*

```
Prateeks-MacBook-Pro-2:DVIA Prateek$ ssh root@192.168.0.104
The authenticity of host '192.168.0.104 (192.168.0.104)' can't be established.
RSA key fingerprint is 34:29:9b:88:53:4c:fe:11:03:62:4e:0b:41:8f:32:97.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.0.104' (RSA) to the list of known hosts.
root@192.168.0.104's password:
Prateeks-iPhone:~ root# cd /var/mobile/Applications/
Prateeks-iPhone:/var/mobile/Applications root#
```

Now use the command **ls *** to look for the directory of our application.

```
Prateeks-iPhone:/var/mobile/Applications root# ls *
06137015-17CD-4DEE-95C1-A773438399EB:
Documents/  Library/  Shazam.app/  StoreKit/  iTunesArtwork  iTunesMetadata.plist  tmp/

075B35FA-A928-4F20-A98C-41F4778AB542:
Documents/  Library/  StoreKit/  Zomato.app/  iTunesArtwork  iTunesMetadata.plist  tmp/

0789CB70-EDC6-4C2D-B986-9A2D49E8CABC:
Calculator.app@  Documents/  Library/  tmp/
```

After scrolling down, we find the application directory for *Damn Vulnerable IOS Application.* Let's look inside this directory.

```
A86BD515-3D23-4430-B304-E0DFABDF0EAD:
DamnVulnerableIOSApp.app/  Documents/  Library/  tmp/


Prateeks-iPhone:/var/mobile/Applications root# cd A86BD515-3D23-4430-B304-E0DFABDF0EAD
Prateeks-iPhone:/var/mobile/Applications/A86BD515-3D23-4430-B304-E0DFABDF0EAD root# ls
DamnVulnerableIOSApp.app/  Documents/  Library/  tmp/
Prateeks-iPhone:/var/mobile/Applications/A86BD515-3D23-4430-B304-E0DFABDF0EAD root#
```

On going inside the directory *Documents,* we see a sqlite file with the name CoreData.sqlite. Let's use the sqlite3 client to look into it. Use the command *`sqlite3 CoreData.sqlite`* to enter the sqlite3 interpreter with this database and then *`.tables`* to see all the tables for this database.

```
Prateeks-iPhone:/var/mobile/Applications/A86BD515-3D23-4430-B304-E0DFABDF0EAD root#
Prateeks-iPhone:/var/mobile/Applications/A86BD515-3D23-4430-B304-E0DFABDF0EAD root# cd Documents/
Prateeks-iPhone:/var/mobile/Applications/A86BD515-3D23-4430-B304-E0DFABDF0EAD/Documents root# ls
CoreData.sqlite  CoreData.sqlite-shm  CoreData.sqlite-wal
Prateeks-iPhone:/var/mobile/Applications/A86BD515-3D23-4430-B304-E0DFABDF0EAD/Documents root# sqlite3 CoreData.sqlite
SQLite version 3.7.13
Enter ".help" for instructions
sqlite> .tables
ZUSER         Z_METADATA    Z_PRIMARYKEY
sqlite>
```

Let's just dump out all the data from the table ZUSER. The other two tables are created by default in Core Data to serve a different purpose. Please note that the default tables start with **Z_** whereas custom made tables will start with just a **Z**

Make sure to turn on headers by using the command *`.headers on`*.Then, to dump all the information from the table *`ZUSER`*, use the command *`Select * from ZUSER;`*

```
sqlite> .headers on
sqlite> select * from ZUSER;
Z_PK|Z_ENT|Z_OPT|ZEMAIL|ZNAME|ZPASSWORD|ZPHONE
1|1|1|user@email.com|test user|sosecretpassword|1234567860
sqlite>
```

As we can see, the whole data stored using the CoreData framework was dumped out. It is important for developers to note that the data stored via CoreData is saved unencrypted in the application sandbox. It is therefore their responsibility to make sure they do not store confidential data using the CoreData framework locally on the device.