

Damn Vulnerable iOS Application Solutions

<http://damnvulnerableiosapp.com/>

Security Decisions via untrusted input

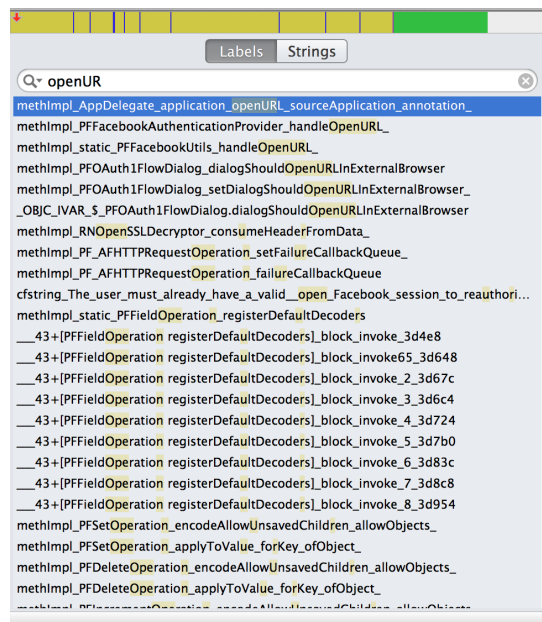
First we need to find out the url scheme for which this application is registered to. By looking at the info.plist file in the application sandbox using a file explorer utility like iExplorer, we can see that this application is registered for the url scheme *dvia*

CFBundleExecutable	String	DamnVulnerableIOSApp
▶ UILaunchImages	Array	(2 items)
▼ CFBundleURLTypes	Array	(1 item)
▼ Item 0	Dictionary	(2 items)
CFBundleTypeRole	String	None
▼ CFBundleURLSchemes	Array	(1 item)
Item 0	String	dvia
CFBundleIdentifier	String	com.highaltitudehacks.dvia
DTCompiler	String	com.apple.compilers.llvm.clang.1_0
DTPlatformVersion	String	7.0
CFBundleSignature	String	???
LSRequiresiPhoneOS	Boolean	YES

If you are familiar with a bit of iOS development, you will also know that the method that is called to handle an incoming url is

```
-(BOOL)application:(UIApplication *)application openURL:(NSURL *)url sourceApplication:(NSString *)sourceApplication  
annotation:(id)annotation
```

Hence, let's search for this method's implementation using Hopper.



And tap on the Pseudo code section on the top-right to look for this method's pseudo code.

You can find the pdf containing pseudo code in the same folder as this solution.

From the pseudo code, after doing a bit of analysis we can deduce that this method checks for the string *call_number* in the url and then looks for the *phone* parameter in the url.

```
loc_d0f4:
    var_16 = [r10 rangeOfString:@"call_number"];
    r4 = 0x0;
    if (var_16 == (0x80000000 ^ 0xffffffff)) goto loc_d23e;
    goto loc_d128;
}
```

```
loc_d128:
    var_12 = r5;
    [r6 getParameters:r5];
    r7 = r7;
    r0 = objc_retainAutoreleasedReturnValue();
    [r0 objectForKey:@"phone"];
    r7 = r7;
    r0 = objc_retainAutoreleasedReturnValue();
    r5 = r0;
```

So let's go to safari and type in the url the application is looking for. Let the url be

`dvia://highaltitudehacks.com/call_number/?phone=1234567890`

And as you can see, there was no validation on the application side and the call went through.

