Pinned

Description

- This app has stored my credentials and I can only login automatically. I tried to intercept the login request and restore my password, but this seems to be a secure connection. Can you help bypass this security restriction and get back my password?

Objective

- `Perform SSL Pinning Bypass to intercept the login request and get the flag.`

Difficulty

- `Medium`

Flag

- `HTB{trust_n0_1_n0t_3v3n_@_c3rt!}`

Release:

- /release/pinned.zip (`f75b6eb5295e50b994bd360d7fa9fbe9ab01c07a1d69de6b7832a6cf6013d2d8`)

Prerequisites

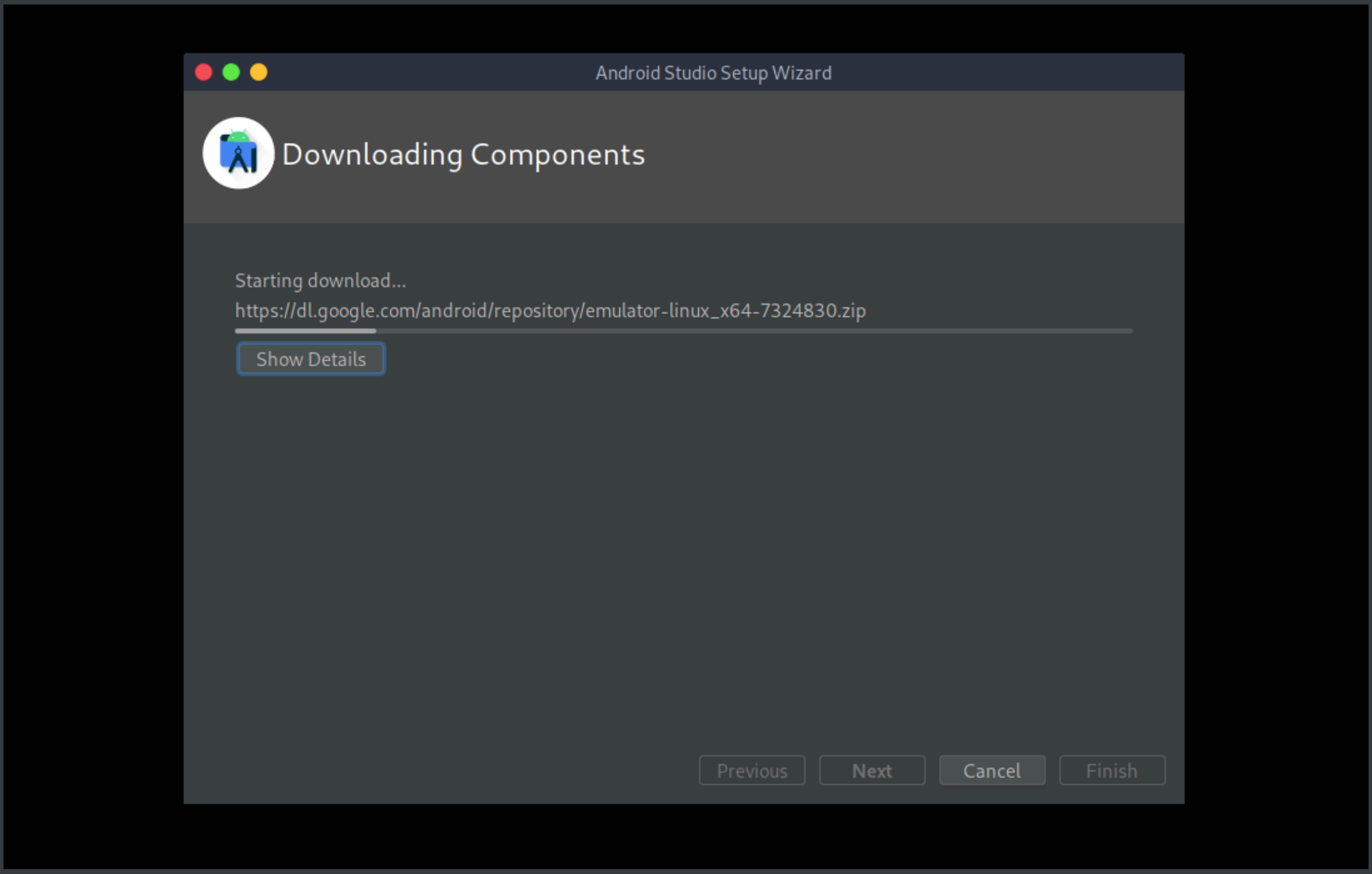
- A virtual Android image like Android-x86 running on Virtual Box or VMware or another Android emulator, running with the Developer mode on.
- Alternatively, a real android device connected via USB, running with the Developer mode on.

Challenge:

Unzipping the `pinned.zip` file reveals the file `pinned.apk`. In order to run the `pinned.apk` file, we have to set up an Android emulator. To achieve this, we are going to use Android Studio IDE.

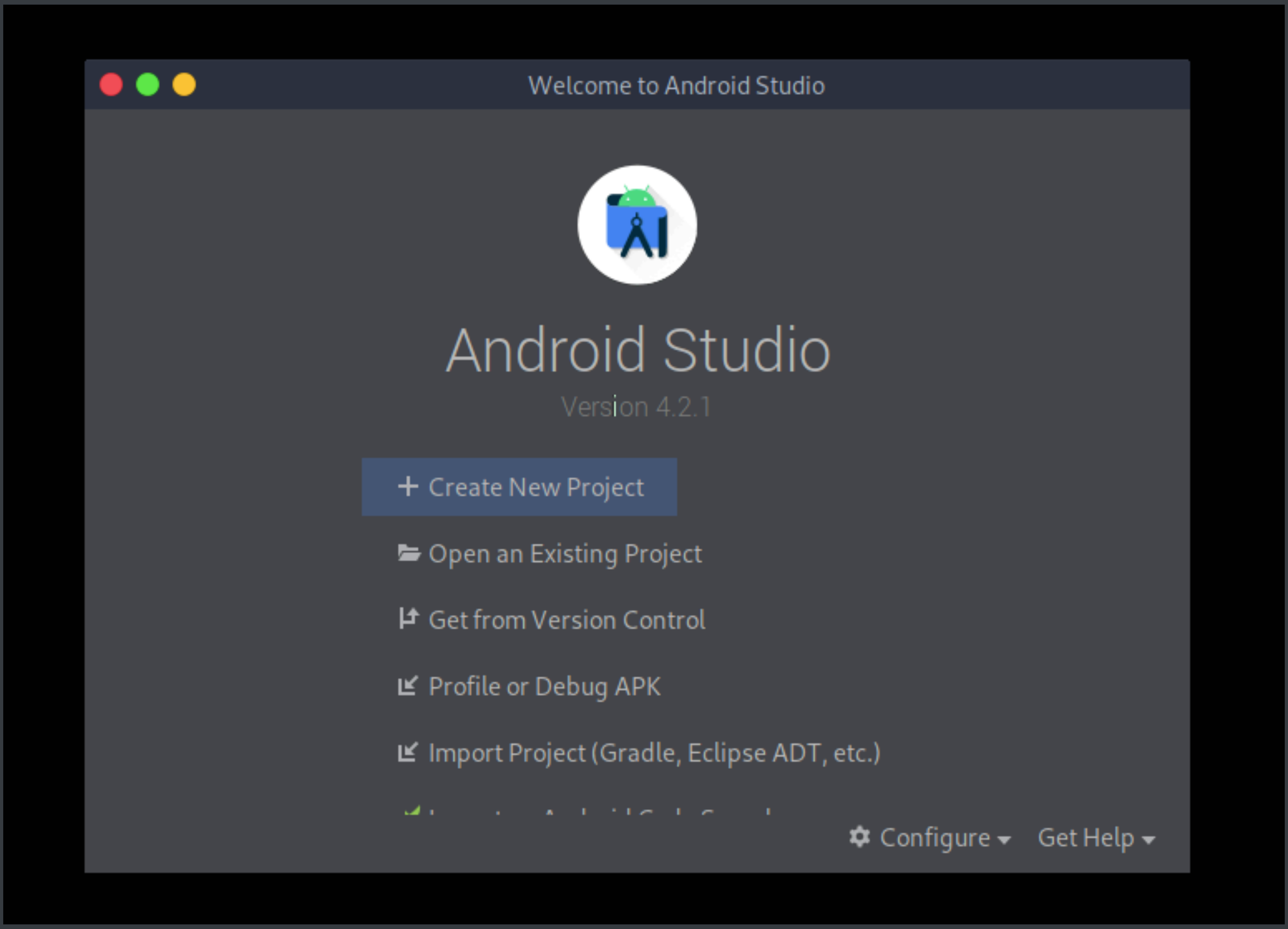
```
wget https://redirector.gvt1.com/edgedl/android/studio/ide-zips/4.2.1.0/android-studio-ide-202.7351085-linux.tar.gz
tar xvzf android-studio-ide-202.7351085-linux.tar.gz
sh android-studio/bin/studio.sh
```

On the setup wizard we click `OK`, then we click on `Next`, and finally click on `Finish`. Next, we wait for the Android Studio to download the components.

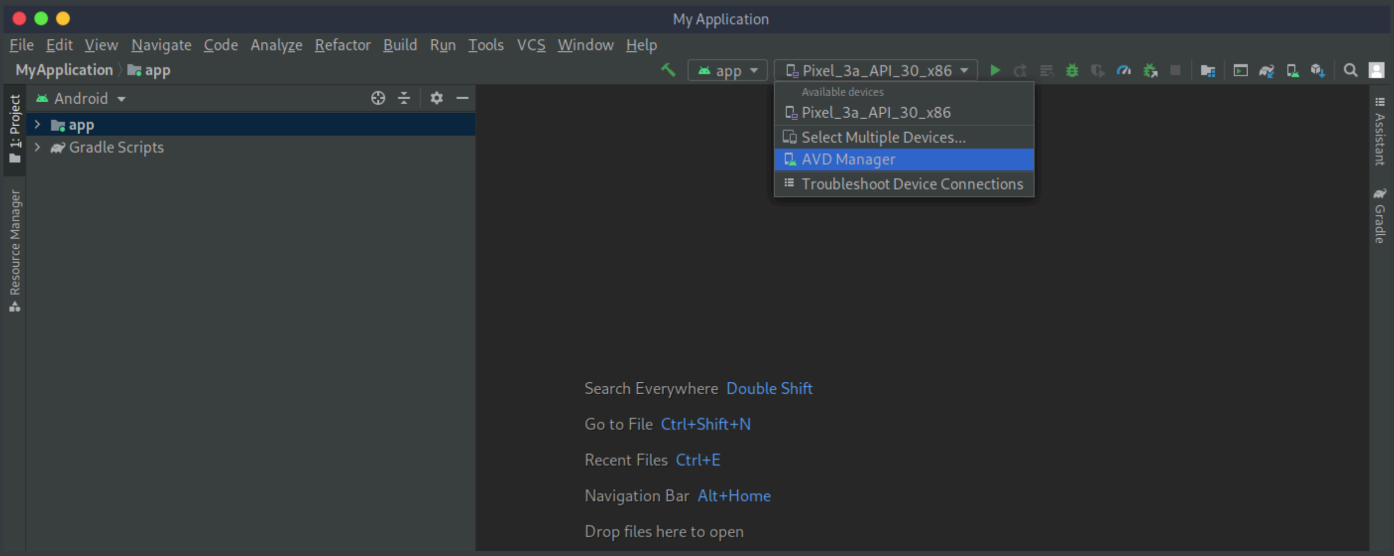


Starting download...
https://dl.google.com/android/repository/emulator-linux_x64-7324830.zip

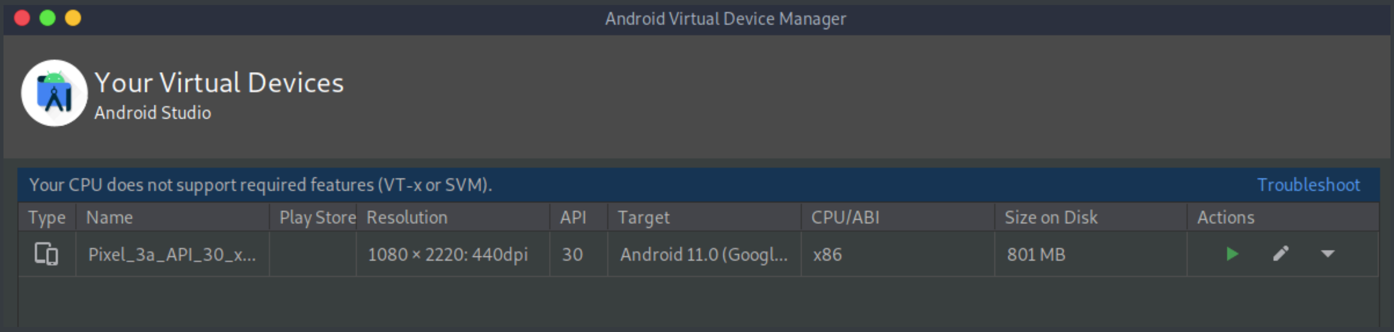
Once it's done, we click `Finish` once again.



Then we click `Next` and finally we click on `Finish`. Now that we have create a new project, we wait for some more files to get downloaded automatically from the IDE. When that's done, click on the top centre of the IDE and select `AVD Manager`.



On the AVD Manager menu, click on the green "play" button to start the emulator.

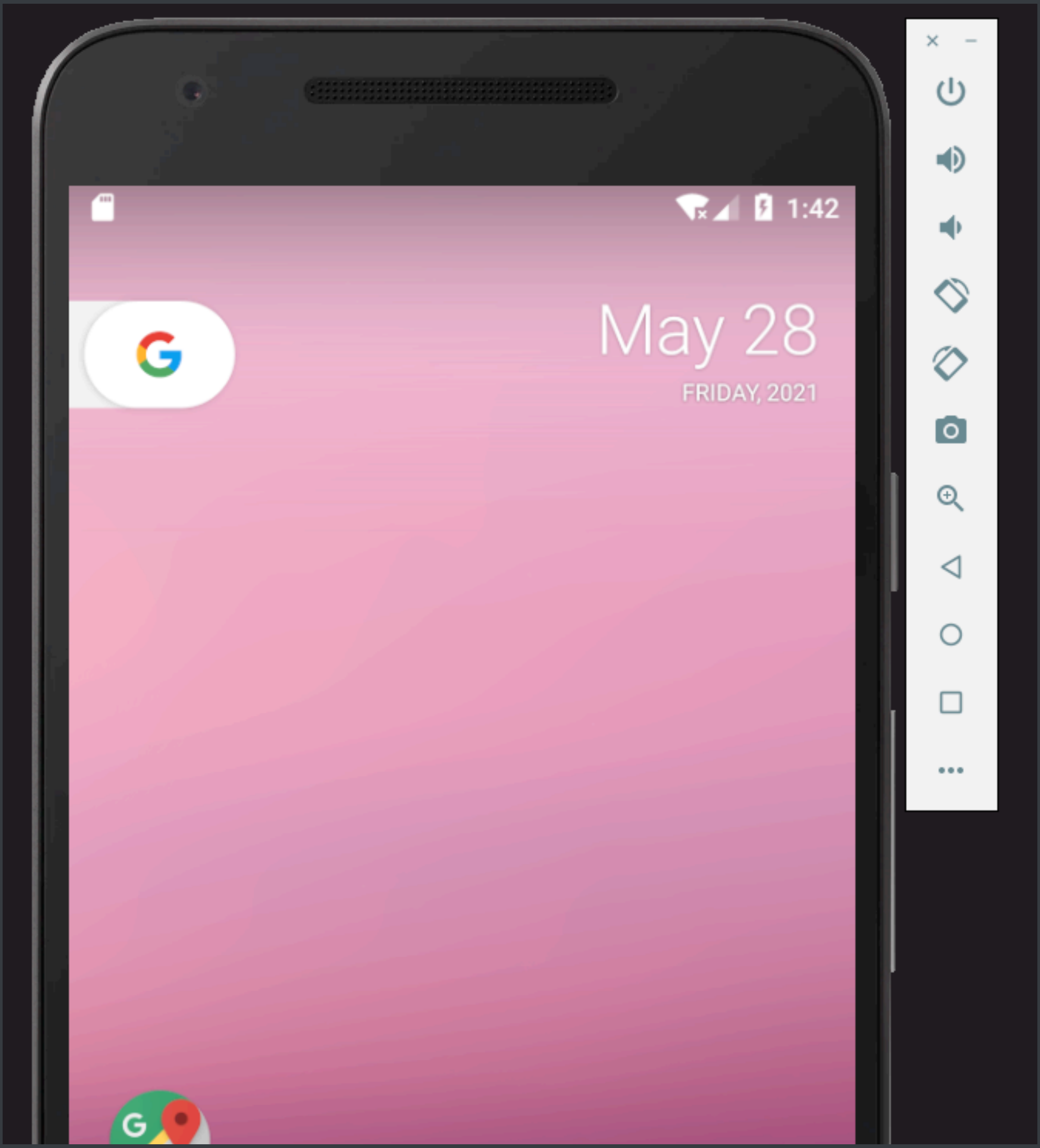

Android Virtual Device Manager

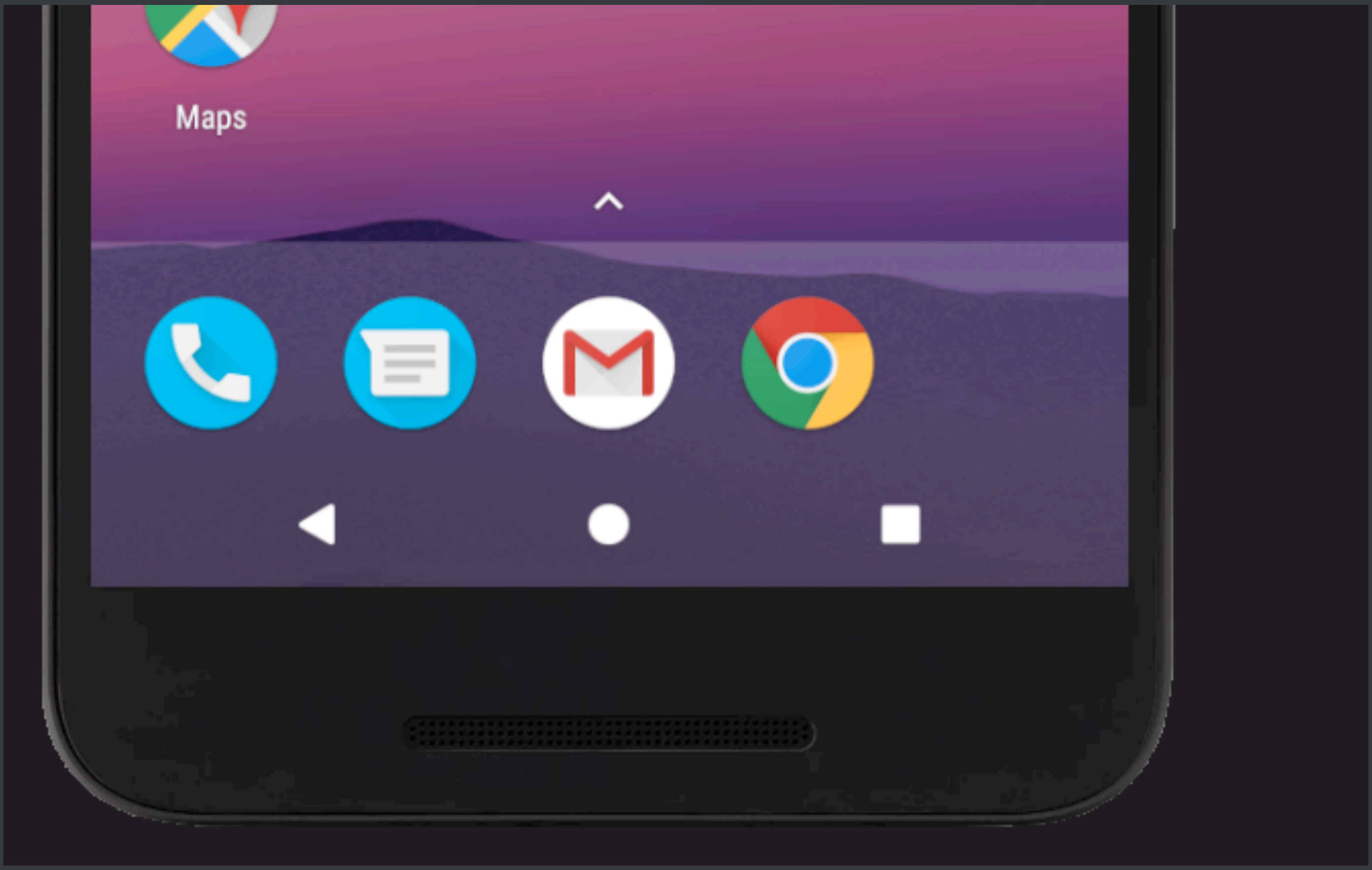
Your Virtual Devices
Android Studio

Your CPU does not support required features (VT-x or SVM). Troubleshoot

Type	Name	Play Store	Resolution	API	Target	CPU/ABI	Size on Disk	Actions
	Pixel_3a_API_30_x...		1080 × 2220: 440dpi	30	Android 11.0 (Googl...	x86	801 MB	

Once the device is started, It should be looking like this.





Then, we install `adb` so we can communicate with it.

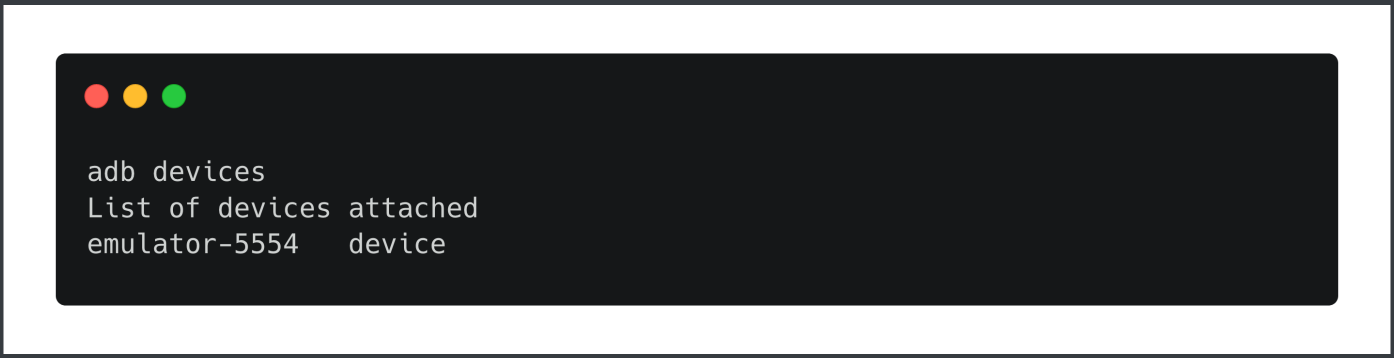
```
sudo apt install adb
```

In our Android device terminal we type `ifconfig` to get the device's IP. Once we get the IP, we type the following to establish the connection.

```
adb connect 192.168.232.2:5555
```

Next, we can list the devices that are connected by typing the following command.

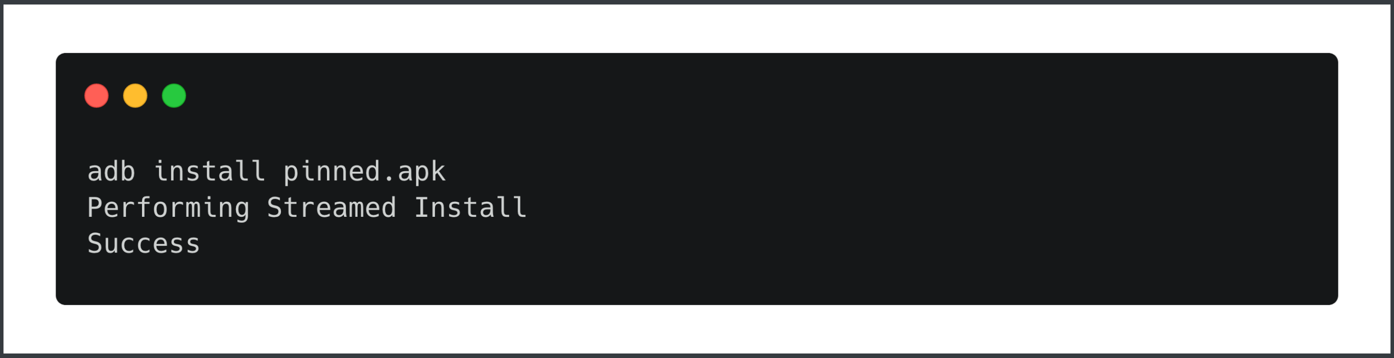
```
adb devices
```



```
adb devices
List of devices attached
emulator-5554	device
```

The connected devices might be displayed either with the format of `name-port` or `ip-port`. In this case the device is displayed as `emulator-5554`. Now that we are connected to the device, let's go on and install the `pinned.apk` file.

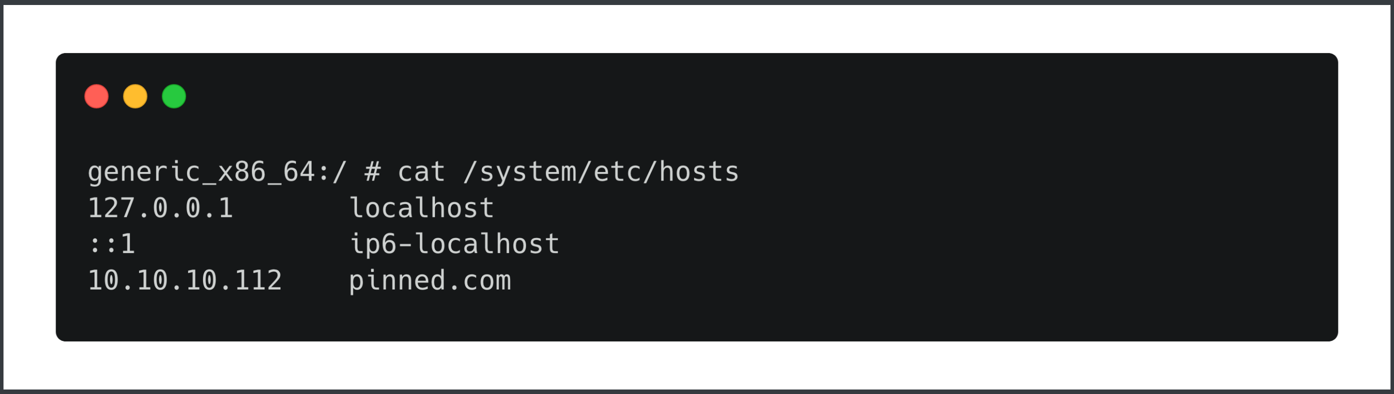
```
adb install pinned.apk
```



```
adb install pinned.apk
Performing Streamed Install
Success
```

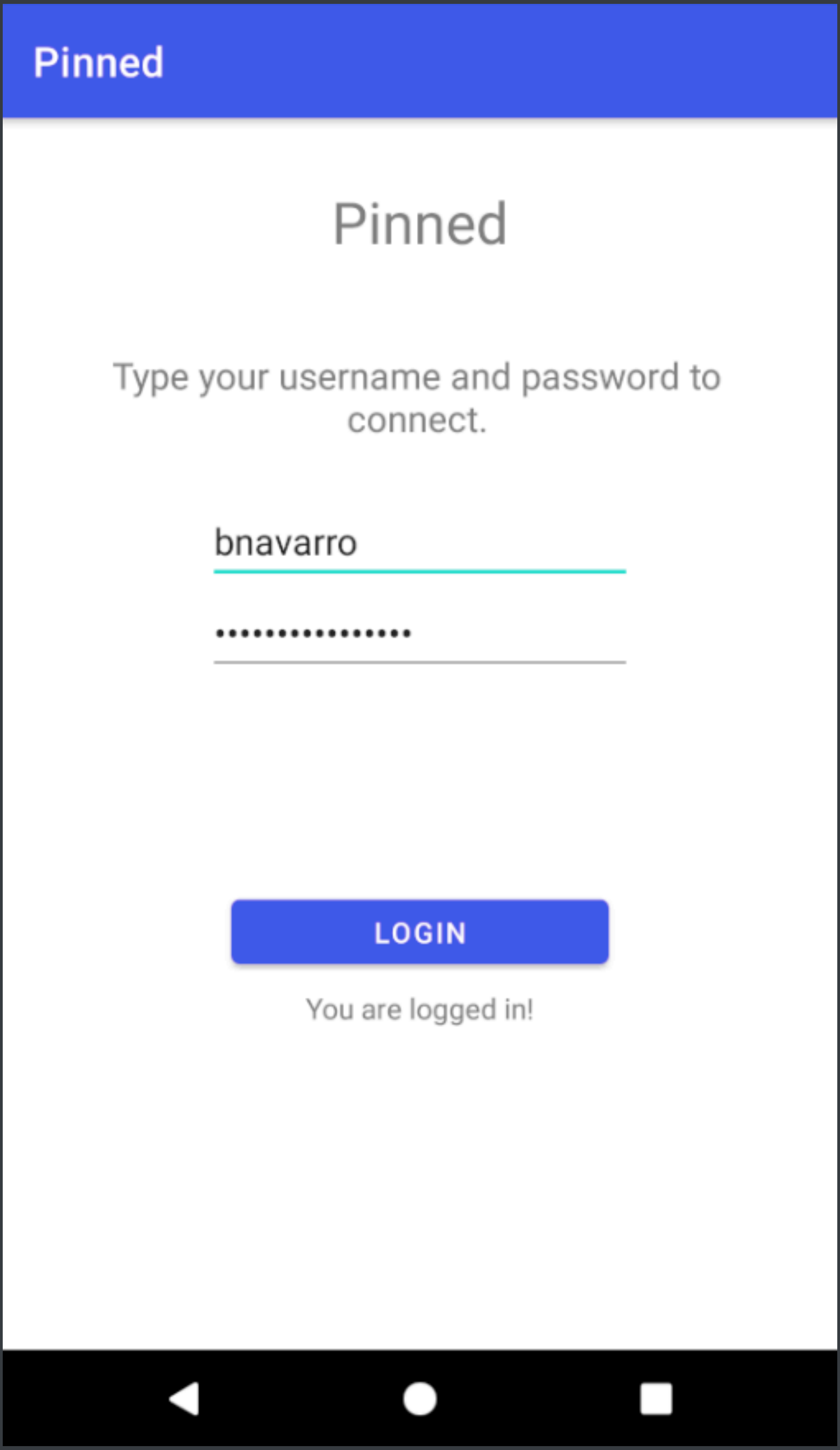
According to the challenge description, before we go on and run the application, we have to add the following domain name into the `/system/etc/hosts` file.

```
adb root
adb shell
mount -o rw,remount /system
echo "10.10.10.112    pinned.com" >> /system/etc/hosts
mount -o ro,remount /system
cat /system/etc/hosts
reboot
```



```
generic_x86_64:/ # cat /system/etc/hosts
127.0.0.1       localhost
::1             ip6-localhost
10.10.10.112    pinned.com
```

We can now run the application on the Android device.


Pinned

Pinned

Type your username and password to connect.

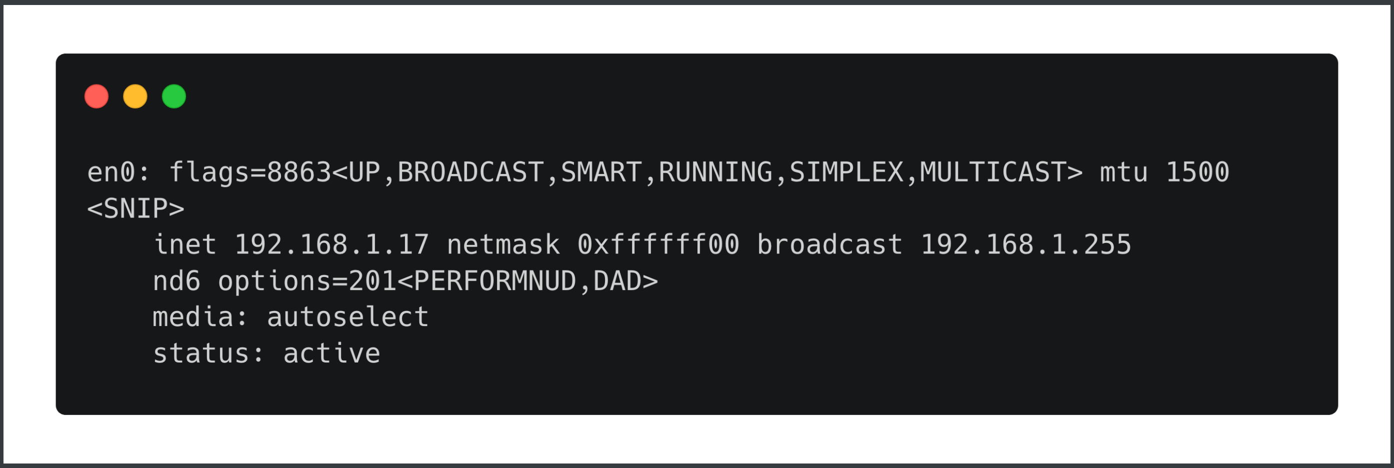
bnavarro

••••••••••••••••

You are logged in!

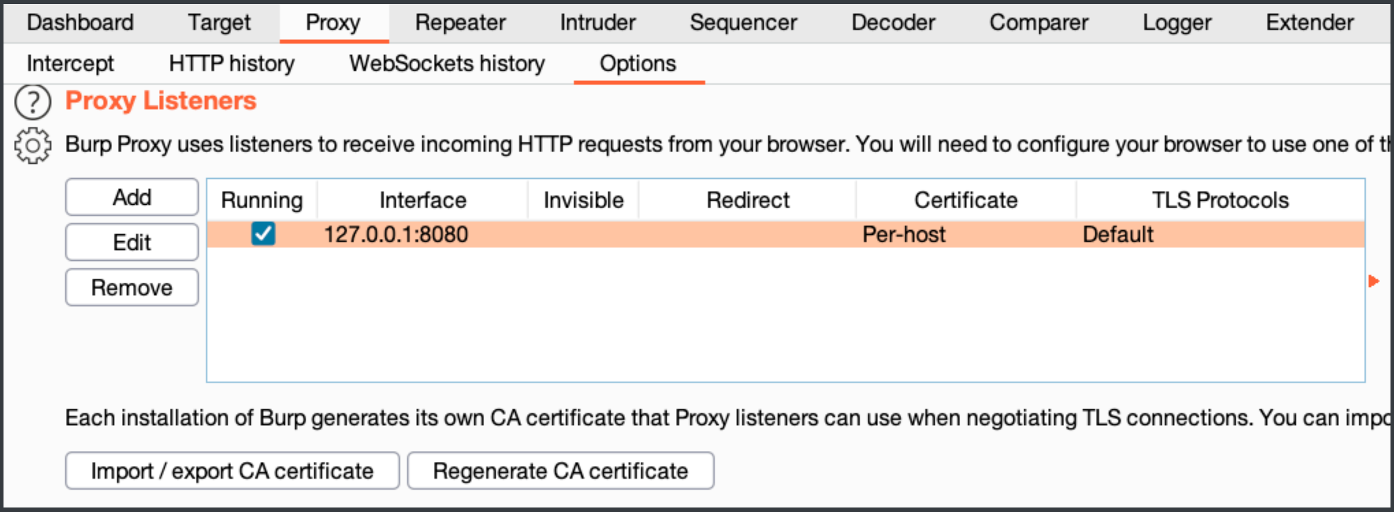
This is a login screen asking for username and password. The credentials are stored and as the message indicates, we can successfully login. Let's try to intercept the request and see if we can get the password in plaintext. In order to setup Burp, type the following on the host machine to get the ip address.

```
ifconfig
```



```
en0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
<SNIP>
    inet 192.168.1.17 netmask 0xffffff00 broadcast 192.168.1.255
    nd6 options=201<PERFORMNUD,DAD>
    media: autoselect
    status: active
```

On the Burp Suit `Proxy` tab, go on `Options` and press the `Add` button under the `Proxy Listeners` section.



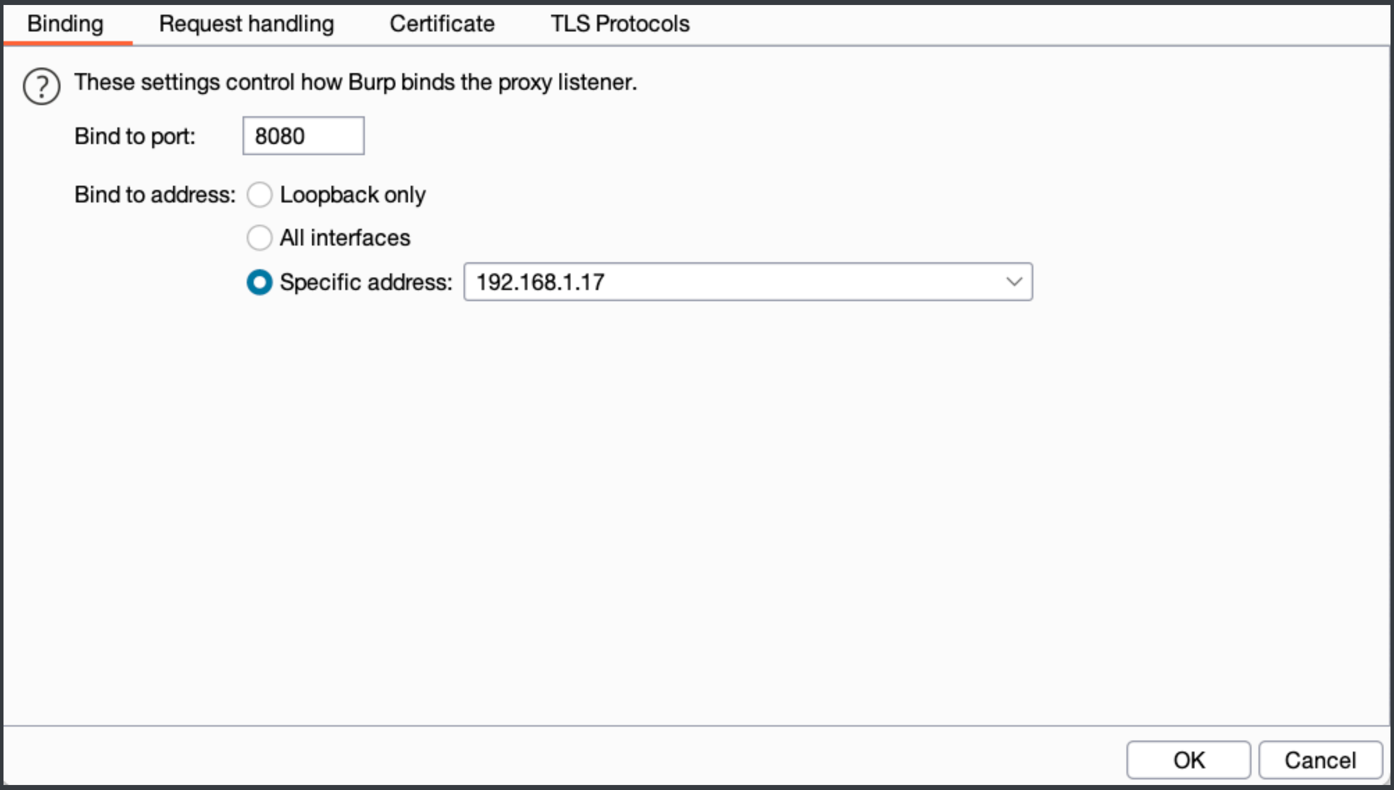
Proxy Listeners

Burp Proxy uses listeners to receive incoming HTTP requests from your browser. You will need to configure your browser to use one of t

Running	Interface	Invisible	Redirect	Certificate	TLS Protocols
✓	127.0.0.1:8080			Per-host	Default

Each installation of Burp generates its own CA certificate that Proxy listeners can use when negotiating TLS connections. You can impo

On the pop up window type the port `8090` and select the host's IP from the drop down menu.

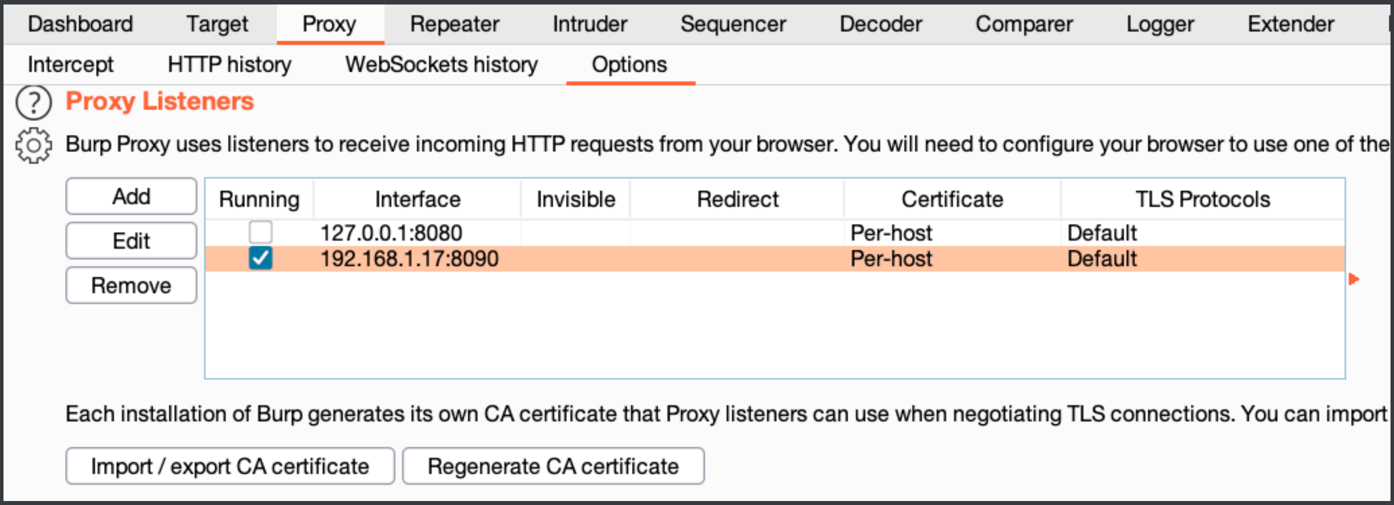


These settings control how Burp binds the proxy listener.

Bind to port: 8080

Bind to address: Loopback only
All interfaces
[✓ Specific address: 192.168.1.17]

Click `OK` and make sure the new proxy listener is selected.



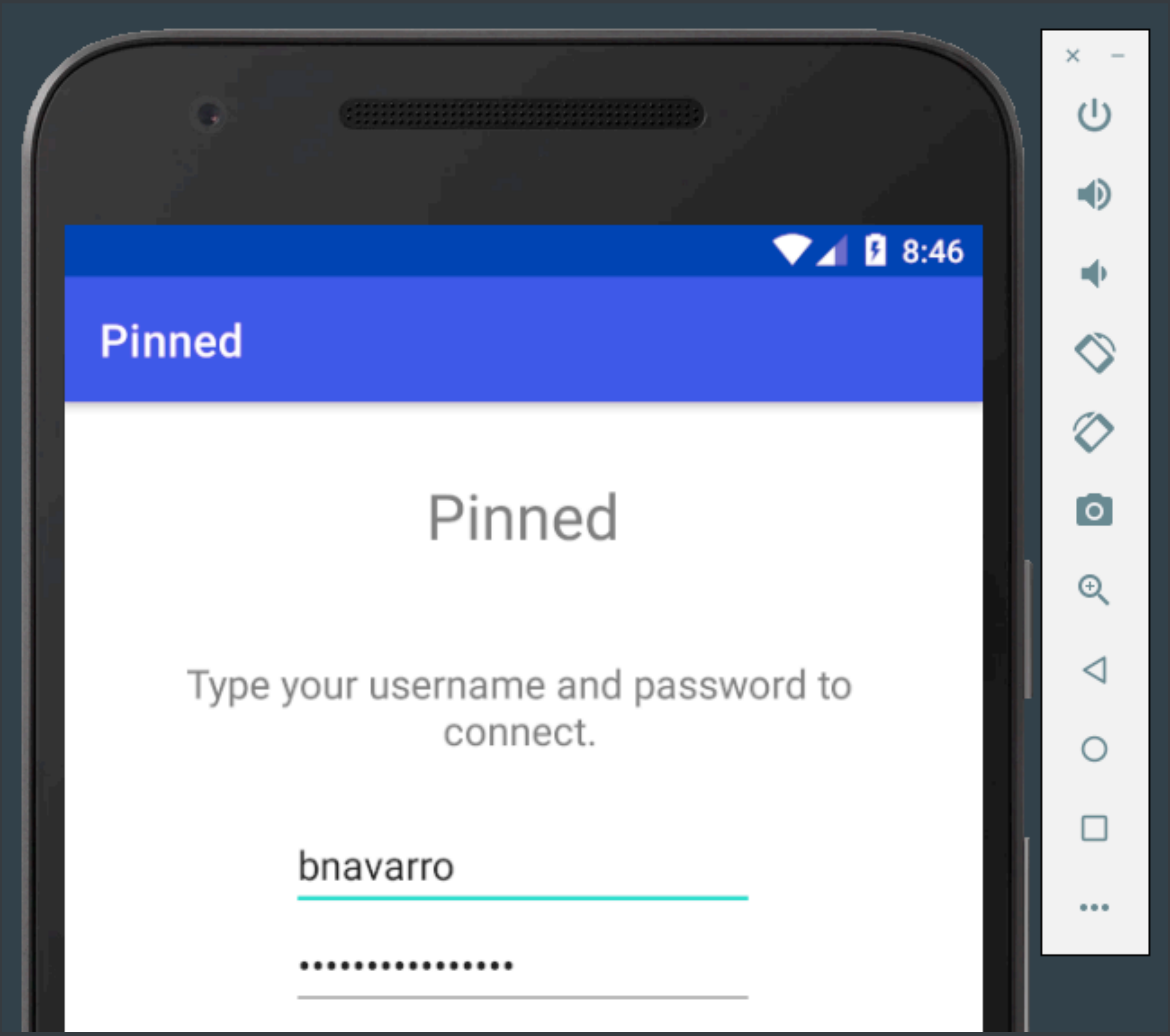
Proxy Listeners

Burp Proxy uses listeners to receive incoming HTTP requests from your browser. You will need to configure your browser to use one of the

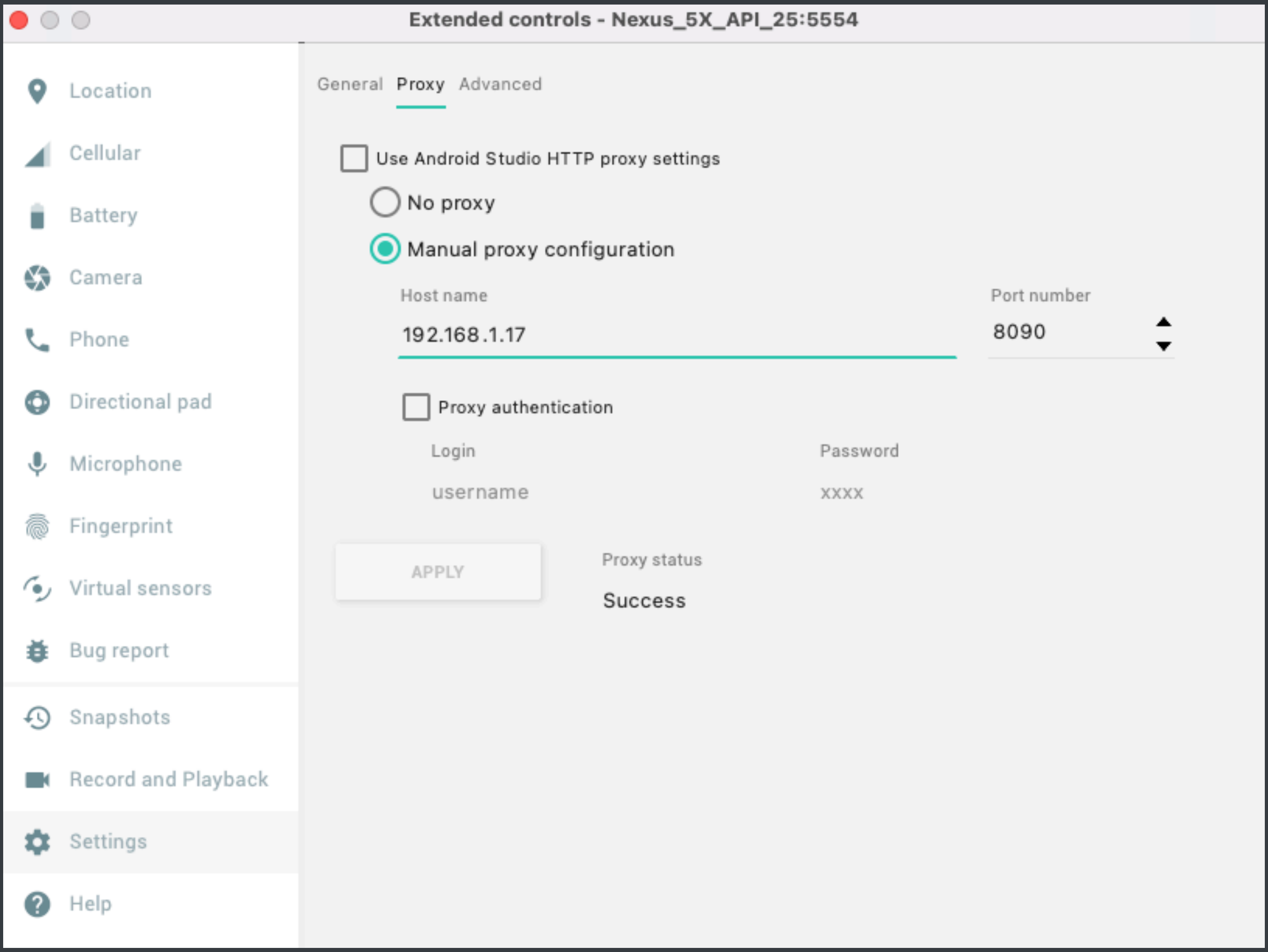
Running	Interface	Invisible	Redirect	Certificate	TLS Protocols
☐	127.0.0.1:8080			Per-host	Default
✓	192.168.1.17:8090			Per-host	Default

Each installation of Burp generates its own CA certificate that Proxy listeners can use when negotiating TLS connections. You can import

Back to the Android emulator, click the three dots from the vertical menu near the device.



On the `Extended Controls` pop up window, click on settings. Then, uncheck the `Use Android Studio HTTP proxy settings`, and check the `Manual proxy configuration`. On the `Host name` field, add the host IP address, and on the `Port number` add `8090`. Finally click `APPLY`.


Extended controls - Nexus_5X_API_25:5554

☐ Use Android Studio HTTP proxy settings

No proxy [✓ Manual proxy configuration]

Host name: 192.168.1.17

Port number: 8090

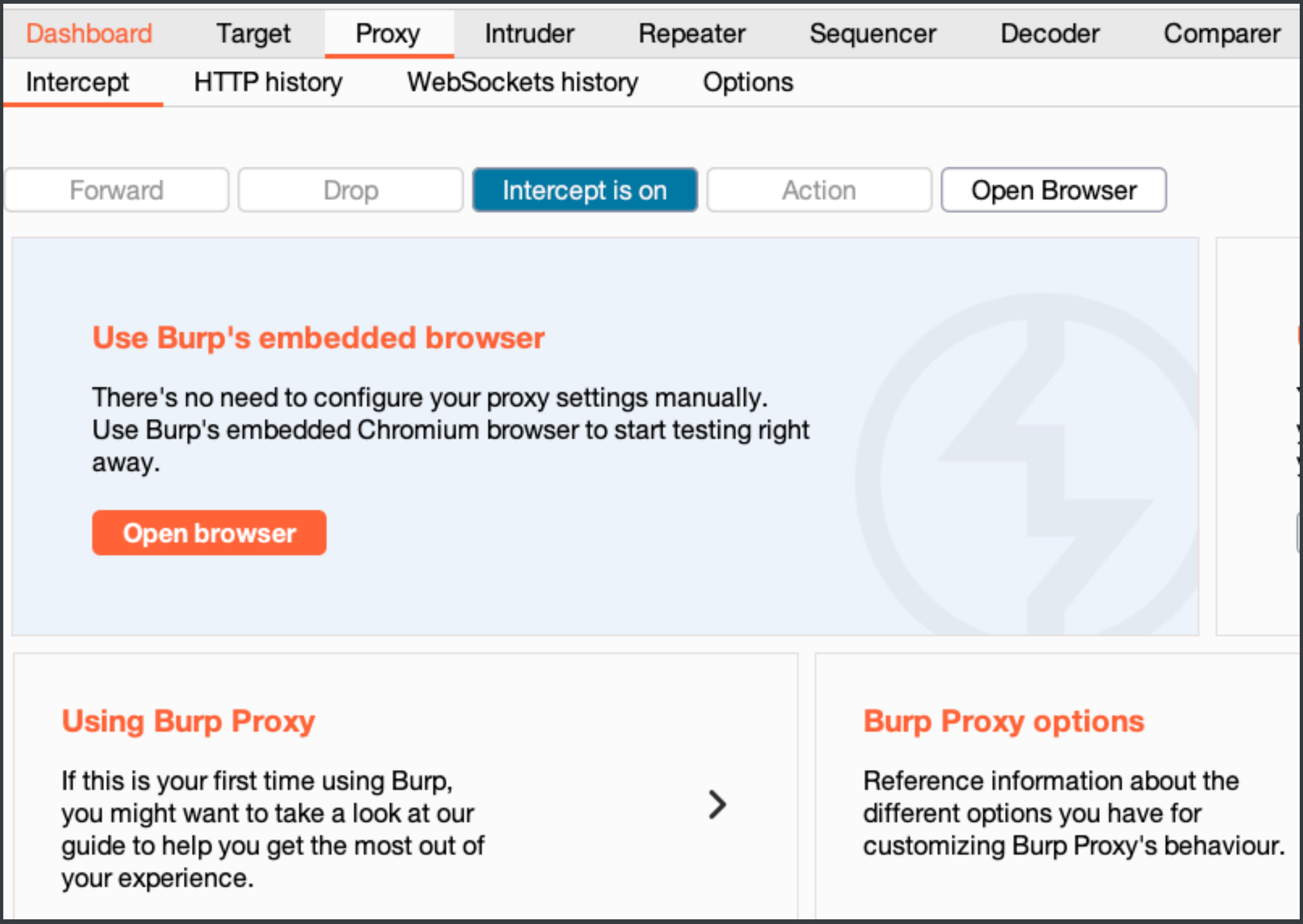
☐ Proxy authentication

Login: username

Password: xxxx

Proxy status: Success

The `Proxy status` should say `Success`. Then, go back on the `Intercept` tab on Burp, and make sure the `Intercept is on` button is toggled. Finally, on the Android device click the `LOGIN`. The request should now be intercepted on Burp.



Use Burp's embedded browser

There's no need to configure your proxy settings manually. Use Burp's embedded Chromium browser to start testing right away.

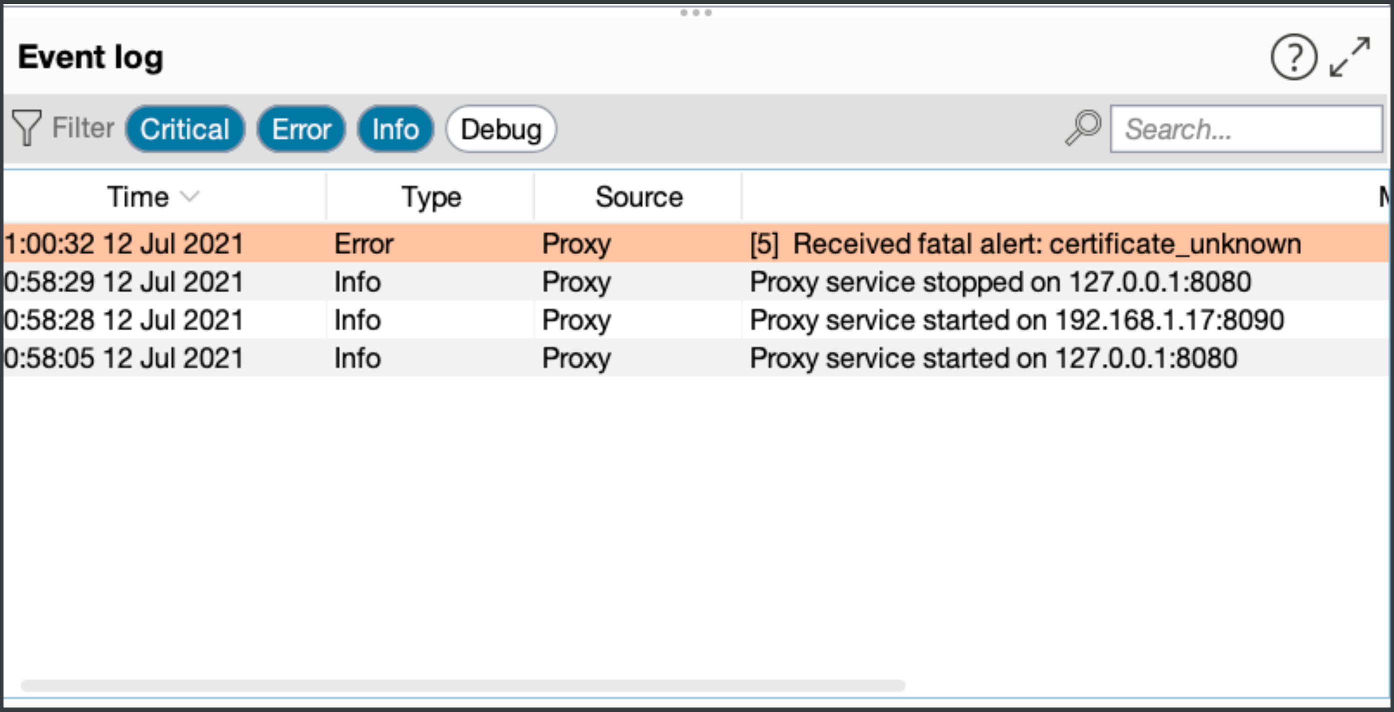
Using Burp Proxy

If this is your first time using Burp, you might want to take a look at our guide to help you get the most out of your experience.

Burp Proxy options

Reference information about the different options you have for customizing Burp Proxy's behaviour.

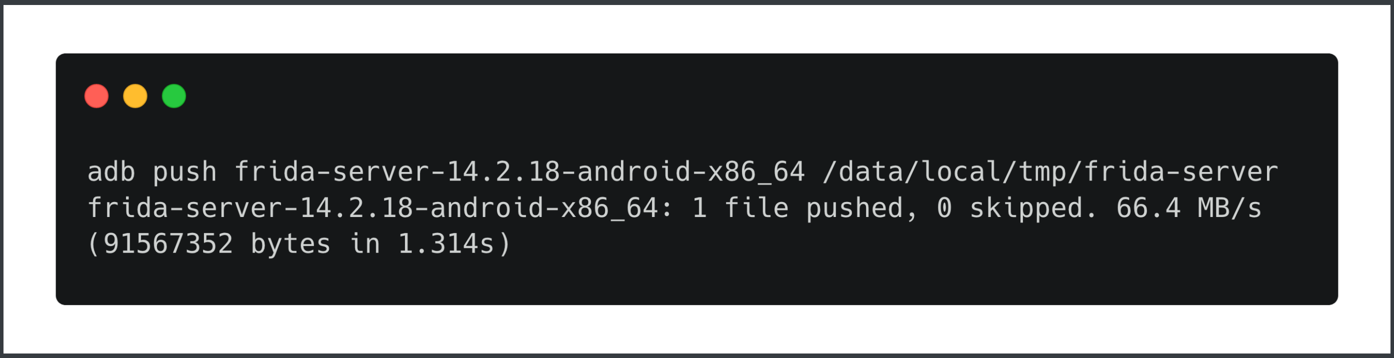
As we can see, it failed to intercept the request and the tab `Dashboard` is now toggled on. Navigating to the `Dashboard` tab, we can see that the alert `Received fatal alert: certificate_uknown`. has been raised.



Time	Type	Source	
1:00:32 12 Jul 2021	Error	Proxy	[5] Received fatal alert: certificate_unknown
0:58:29 12 Jul 2021	Info	Proxy	Proxy service stopped on 127.0.0.1:8080
0:58:28 12 Jul 2021	Info	Proxy	Proxy service started on 192.168.1.17:8090
0:58:05 12 Jul 2021	Info	Proxy	Proxy service started on 127.0.0.1:8080

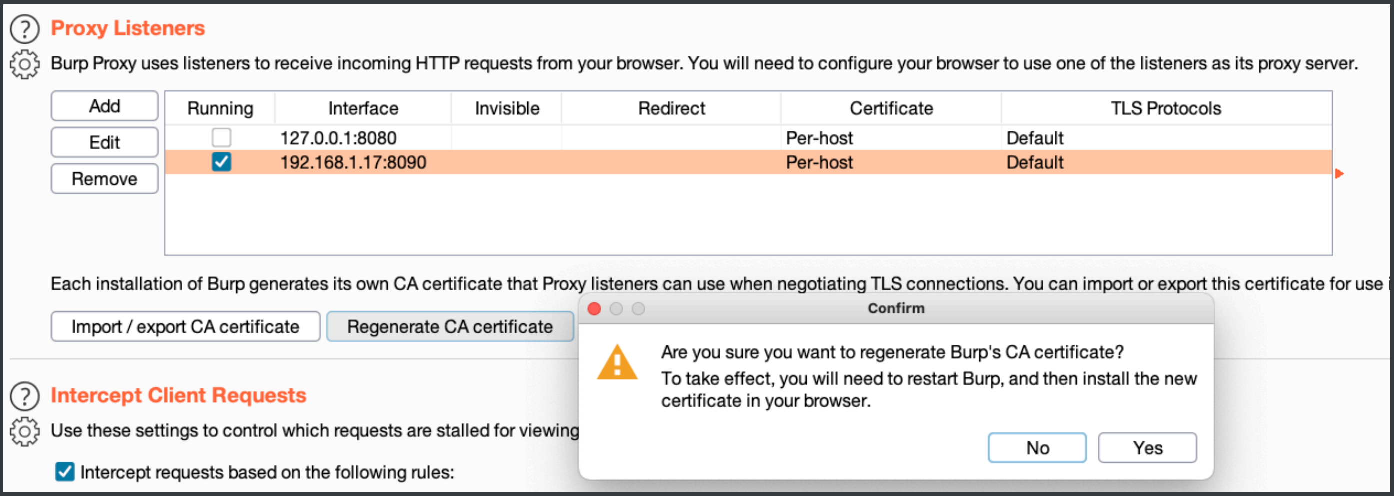
This means that SSL Pinning might be used. In order to bypass SSL Pinning, we first need to run frida-server on the Android emulator device. Once we download the file, we type the following to upload it to the device.

```
adb root
adb push frida-server-14.2.18-android-x86_64 /data/local/tmp/frida-server
```



```
adb push frida-server-14.2.18-android-x86_64 /data/local/tmp/frida-server
frida-server-14.2.18-android-x86_64: 1 file pushed, 0 skipped. 66.4 MB/s
(91567352 bytes in 1.314s)
```

Next, we generate a new certificate on Burp, rename it to `cert-der.crt`, and we place it to the same directory with `frida-server` on the emulator so `frida-server` can read it. On the `Proxy` -> `Options` tab on Burp, we click `Regenerate CA Certificate`.



Proxy Listeners

Burp Proxy uses listeners to receive incoming HTTP requests from your browser. You will need to configure your browser to use one of the listeners as its proxy server.

Running	Interface	Invisible	Redirect	Certificate	TLS Protocols
☐	127.0.0.1:8080			Per-host	Default
☑	192.168.1.17:8090			Per-host	Default

Each installation of Burp generates its own CA certificate that Proxy listeners can use when negotiating TLS connections. You can import or export this certificate for use

Confirm

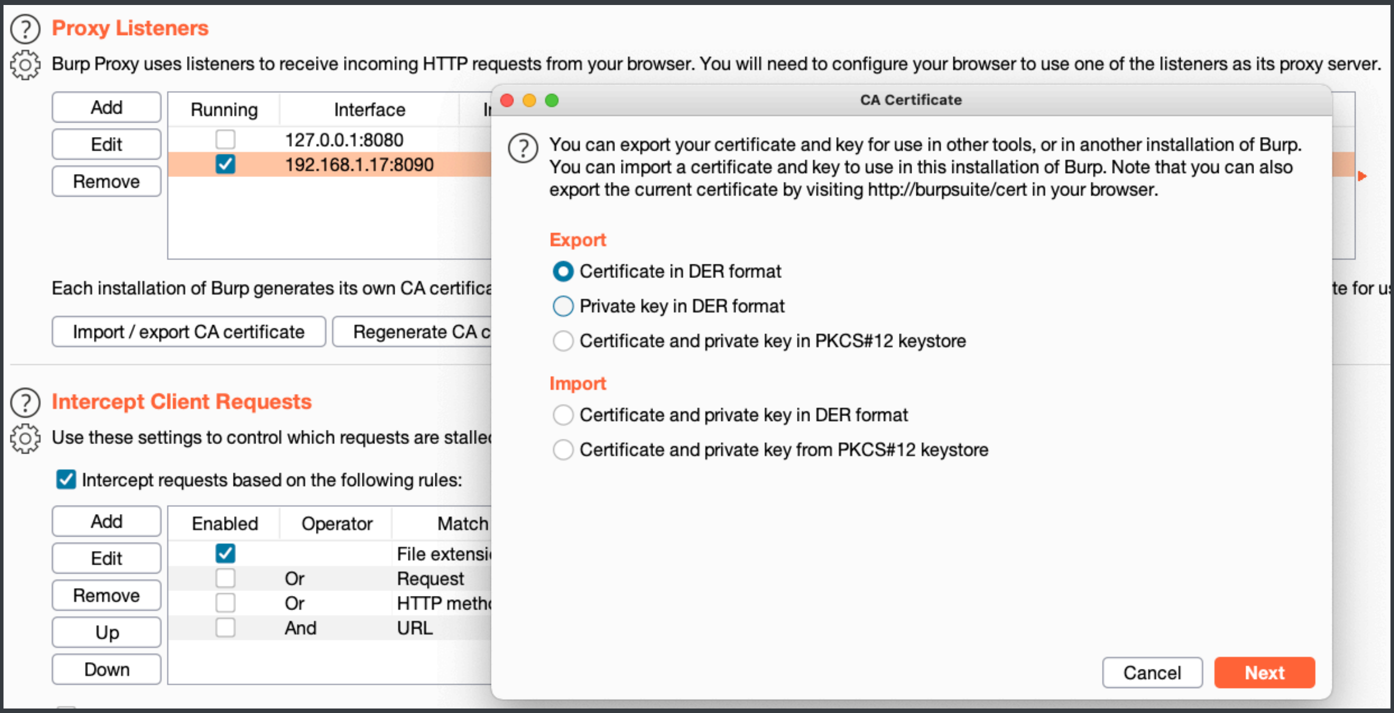
Are you sure you want to regenerate Burp's CA certificate?
To take effect, you will need to restart Burp, and then install the new certificate in your browser.

Intercept Client Requests

Use these settings to control which requests are stalled for viewing

☑ Intercept requests based on the following rules:

On the confirmation dialog, press `Yes`. Then, we click on `Import / export CA certificate`. On `export` section on the popup window, select `Certificate in DER format` and click `Next`.



Proxy Listeners

Burp Proxy uses listeners to receive incoming HTTP requests from your browser. You will need to configure your browser to use one of the listeners as its proxy server.

	Running	Interface
	☐	127.0.0.1:8080
	✓	192.168.1.17:8090

Each installation of Burp generates its own CA certifica

Import / export CA certificate | Regenerate CA c

Intercept Client Requests

Use these settings to control which requests are stalle

✓ Intercept requests based on the following rules:

Enabled	Operator	Match
✓		File extensi
☐	Or	Request
☐	Or	HTTP meth
☐	And	URL

CA Certificate

You can export your certificate and key for use in other tools, or in another installation of Burp. You can import a certificate and key to use in this installation of Burp. Note that you can also export the current certificate by visiting http://burpsuite/cert in your browser.

Export

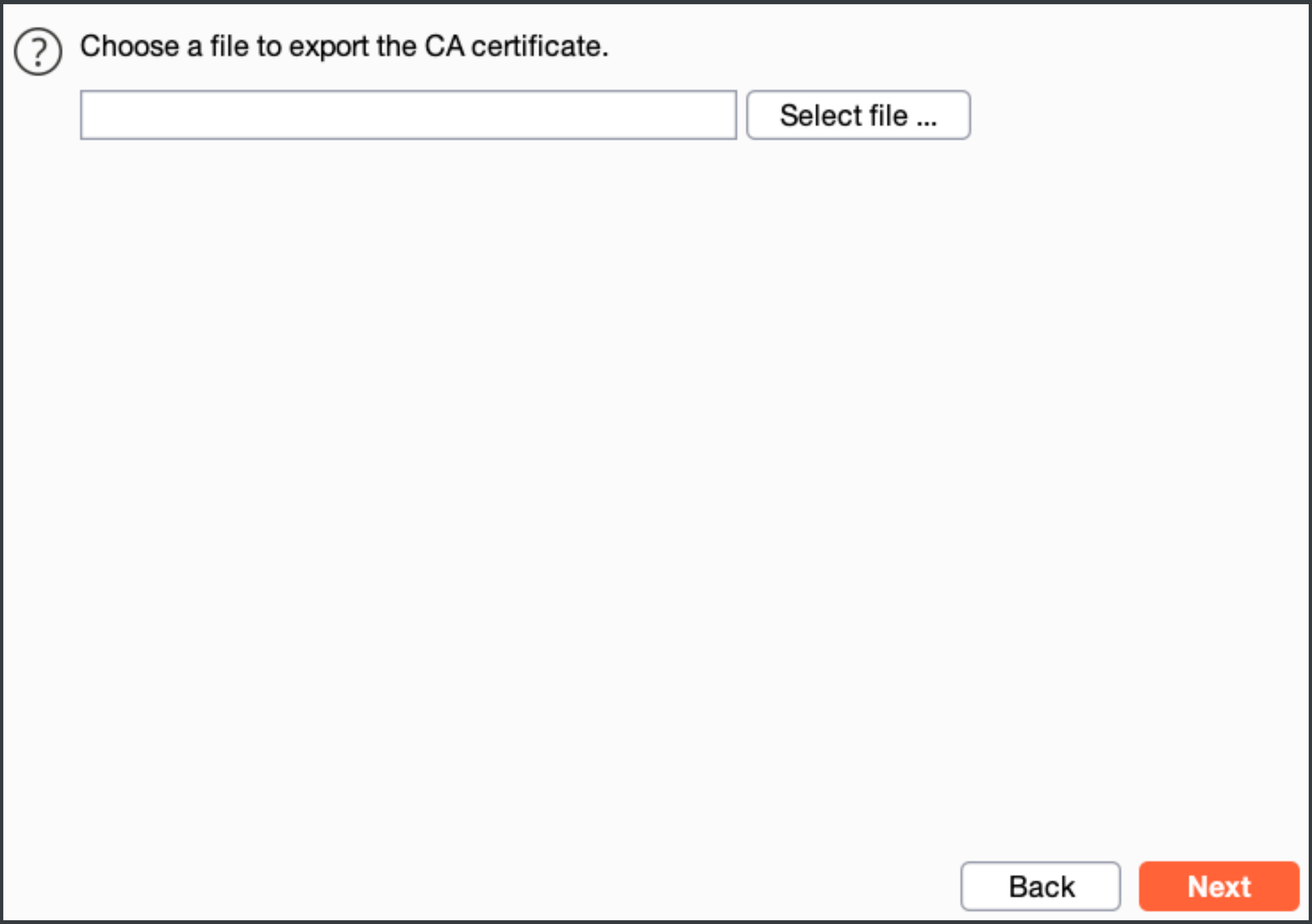
- [✓ Certificate in DER format]
- Private key in DER format
- Certificate and private key in PKCS#12 keystore

Import

- Certificate and private key in DER format
- Certificate and private key from PKCS#12 keystore

Cancel | Next

Next, click on `select file`.

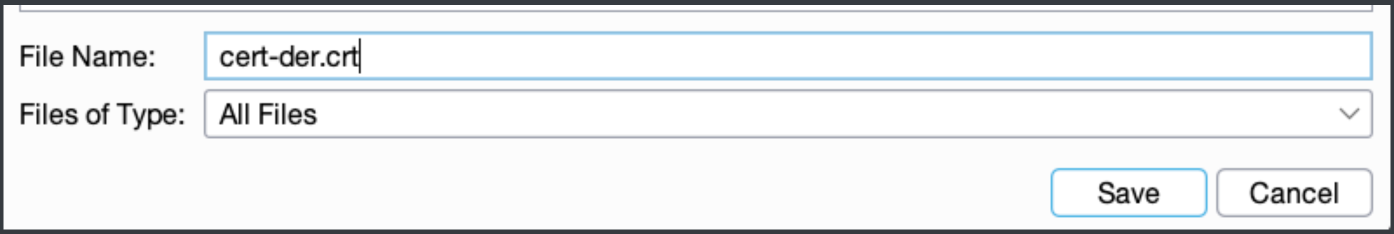


Choose a file to export the CA certificate.

Select file ...

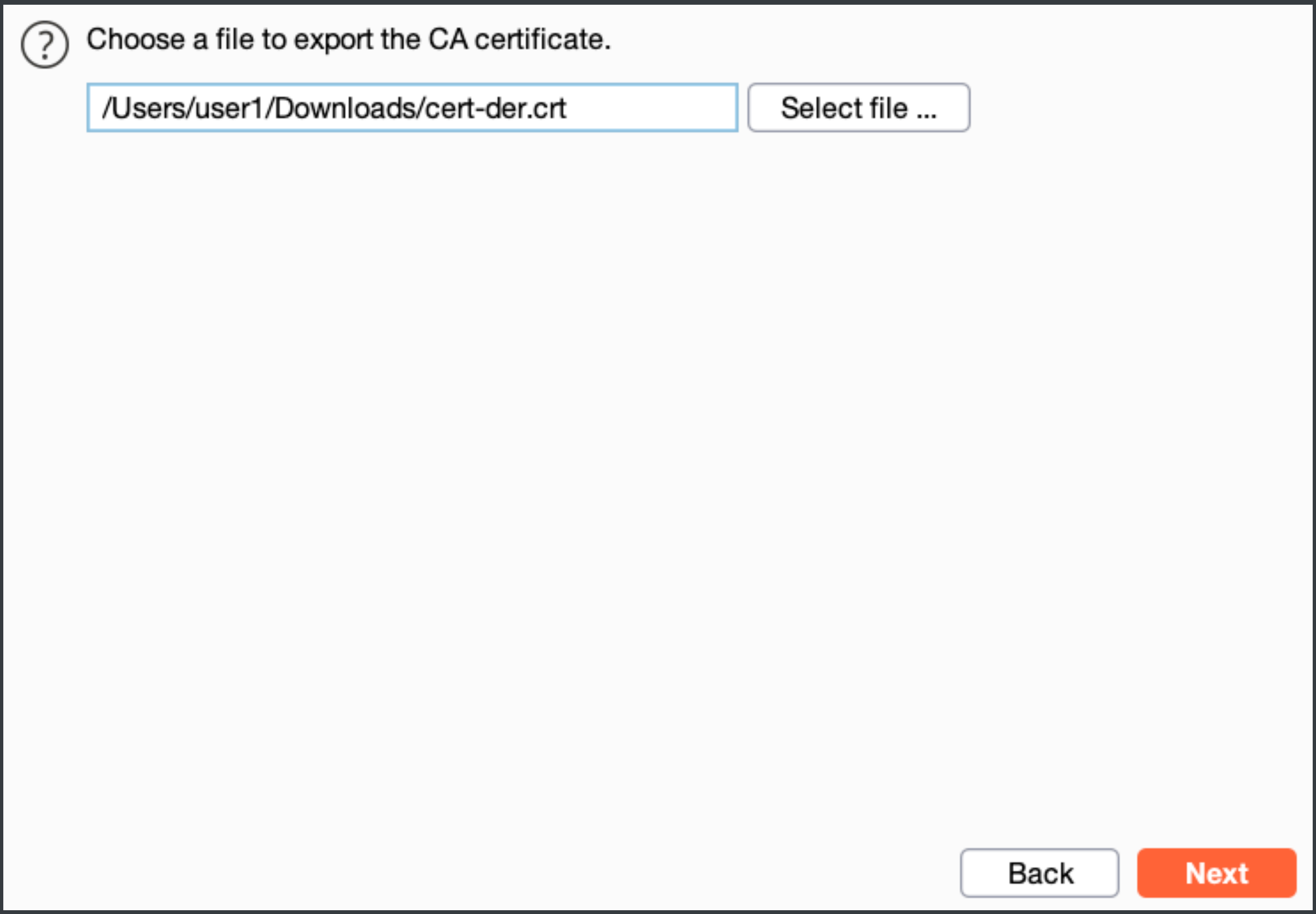
Back | Next

Name it `cert-der.crt` and save it to `Downloads` directory.


File Name: cert-der.crt

Files of Type: All Files

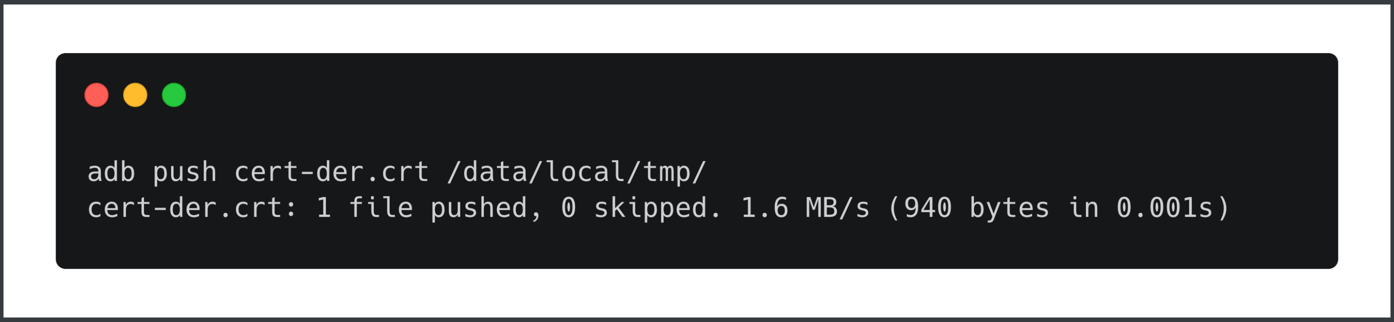
Then click on `Next` once again and then `Close`.


Choose a file to export the CA certificate.

/Users/user1/Downloads/cert-der.crt

To upload the certificate to the emulator, type the following.

```
adb push cert-der.crt /data/local/tmp/
```



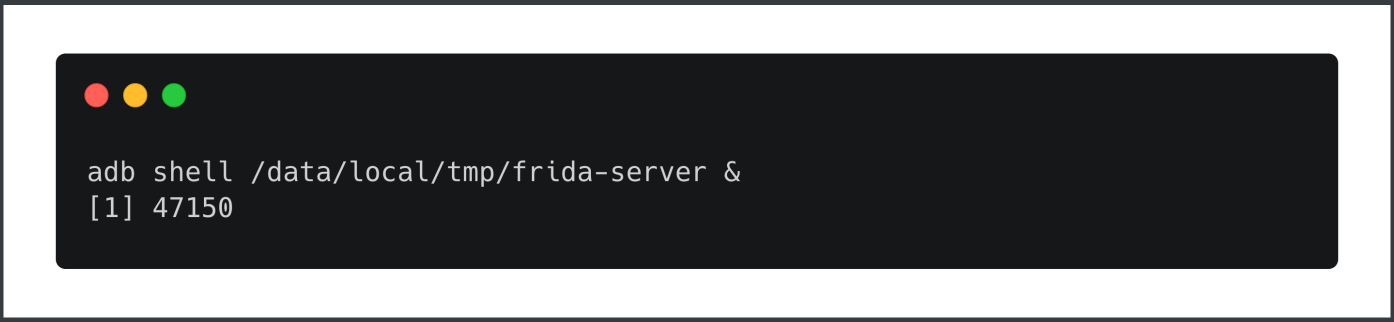
```
adb push cert-der.crt /data/local/tmp/
cert-der.crt: 1 file pushed, 0 skipped. 1.6 MB/s (940 bytes in 0.001s)
```

Once both the certificate and `frida server` have been pushed to the emulator, we also need to setup some more things to the host machine. Type the following to install `frida-tool`.

```
pip install frida-tools
```

In order to bypass SSL Pinning using `frida`, we also need to download the script frida-android-repinning. Once we copy and paste the code into a file and name it `frida-android-repinning.js`, we start the `frida-server` on the emulator by executing the following command. First, make sure you are in `root` mode by executing the `adb root` command.

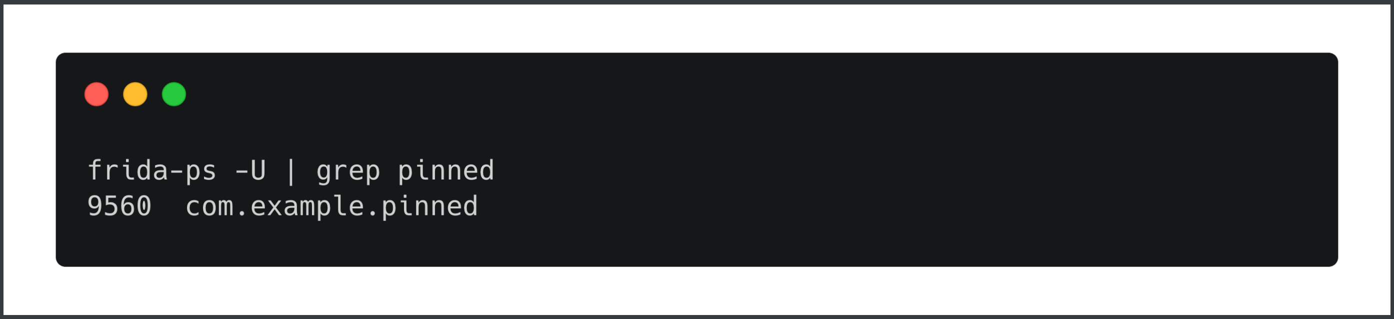
```
adb root
adb shell chmod 755 /data/local/tmp/frida-server
adb shell /data/local/tmp/frida-server &
```



```
adb shell /data/local/tmp/frida-server &
[1] 47150
```

Then, locate the app Pinned in the emulator and tap on it to start. Once the app is started type the following on the terminal to locate the full name of the application.

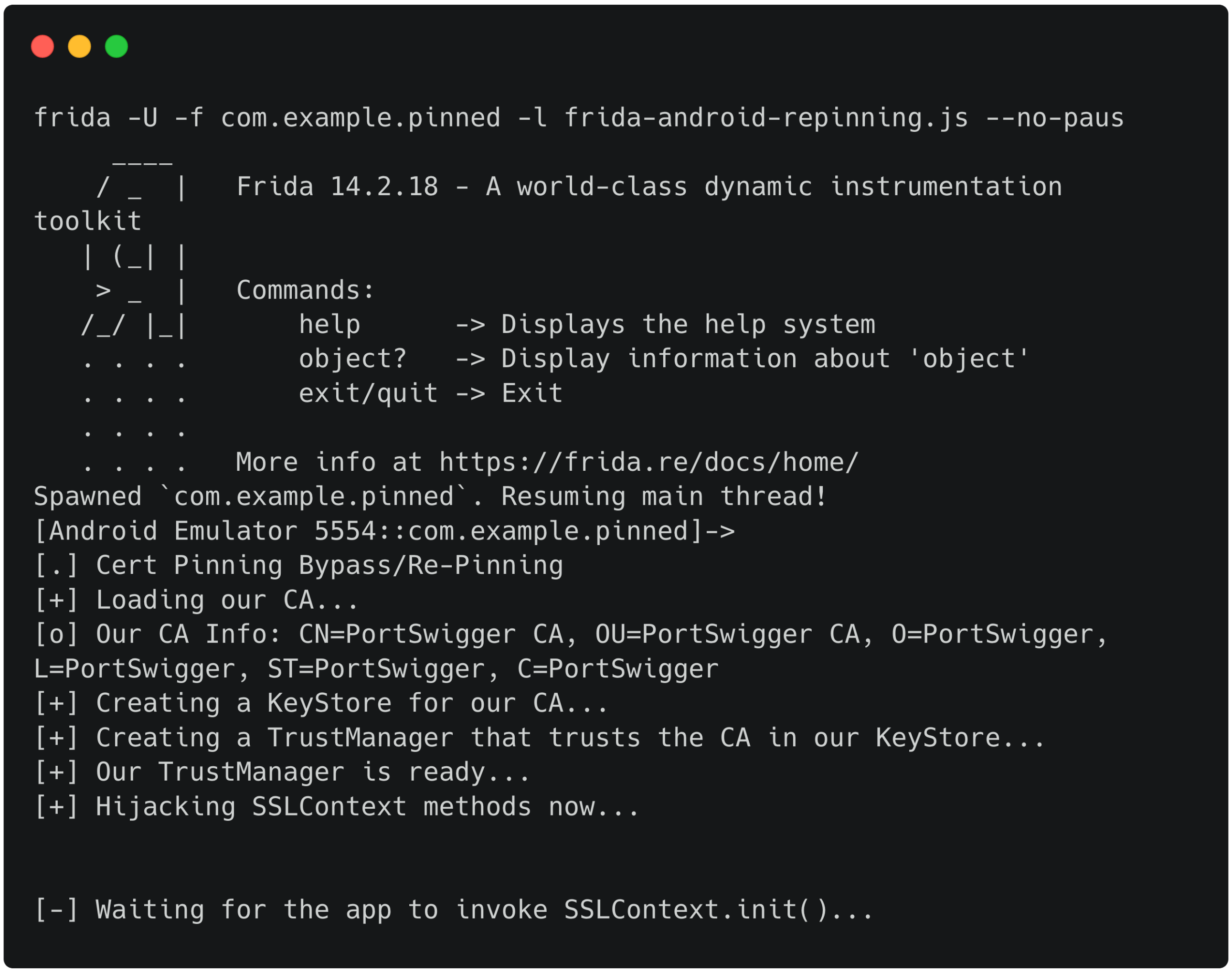
```
frida-ps -U | grep pinned
```



```
frida-ps -U | grep pinned
9560  com.example.pinned
```

Now that we have the full name of the app, we can go on and start it using `frida`. This way, we will be able to intercept the request with Burp. The following command starts the application.

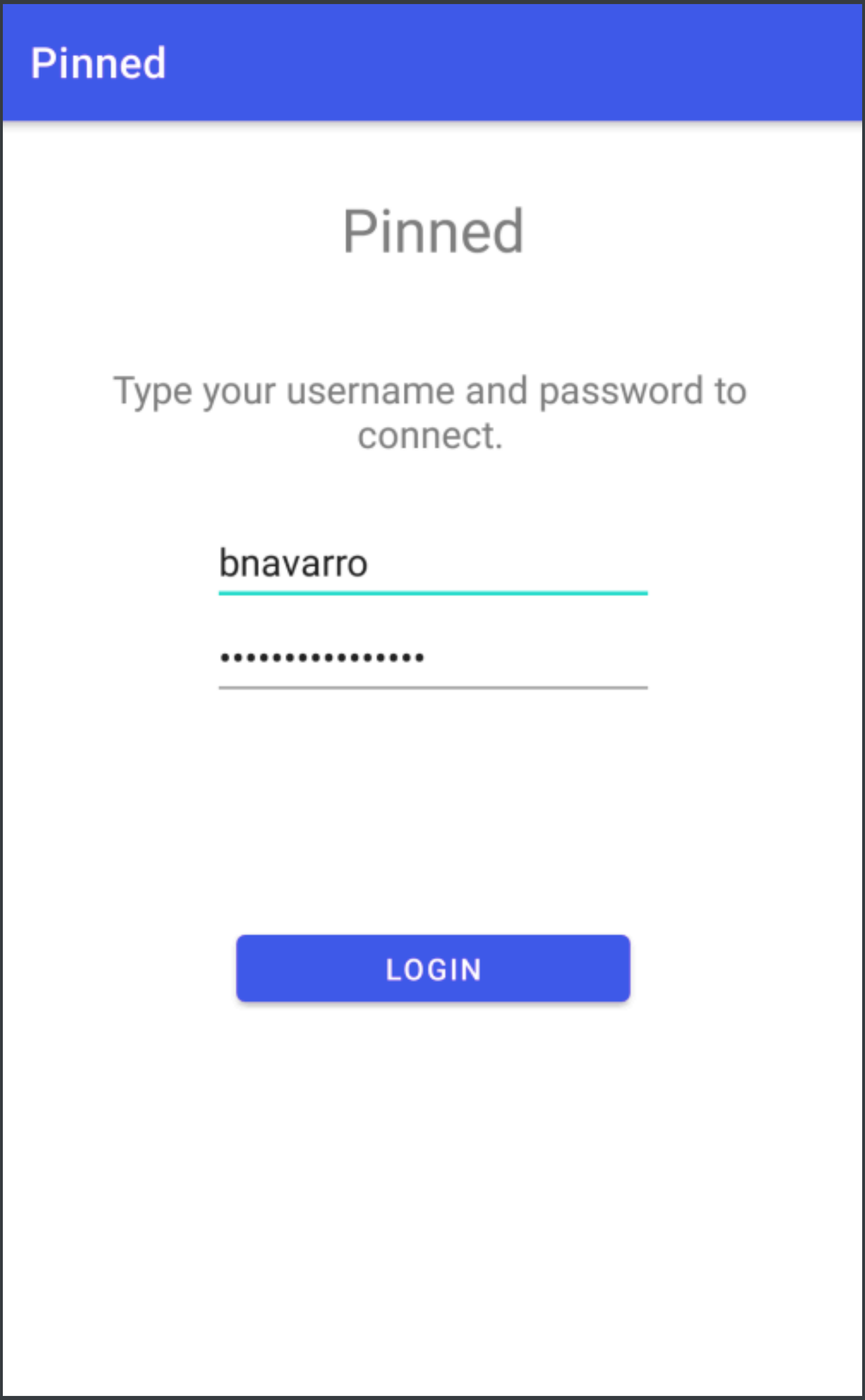
```
frida -U -f com.example.pinned -l frida-android-repinning.js --no-paus
```



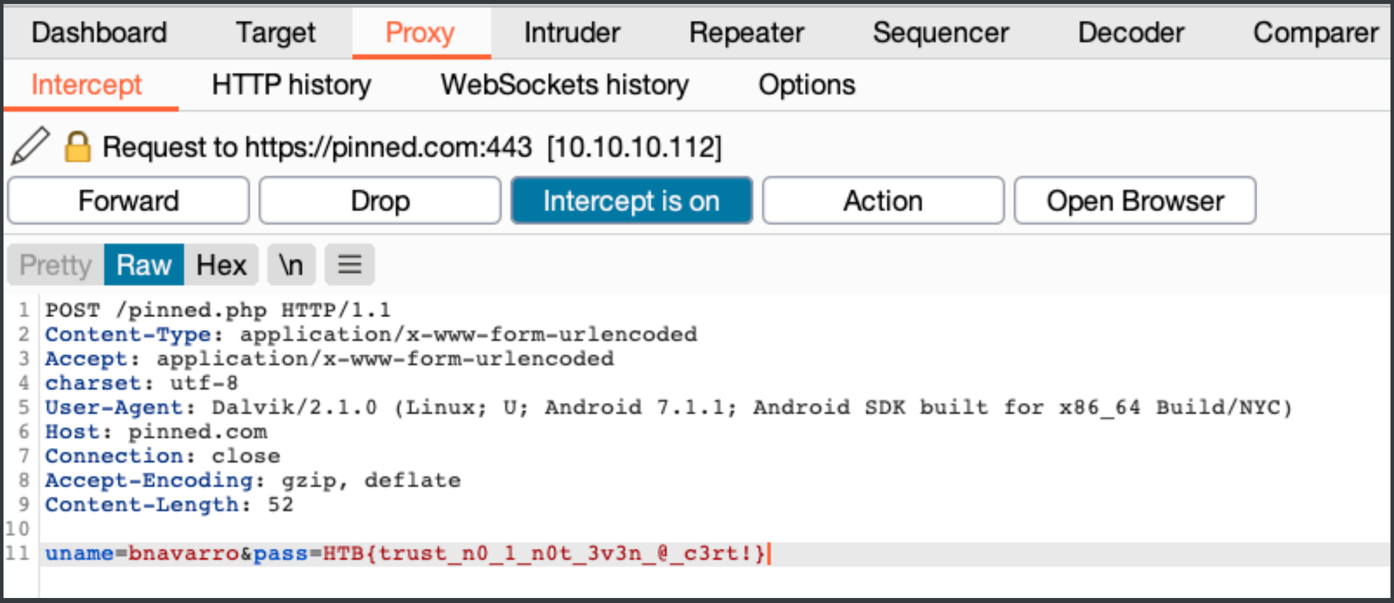
```
frida -U -f com.example.pinned -l frida-android-repinning.js --no-paus
     ____
    / _  |   Frida 14.2.18 - A world-class dynamic instrumentation
toolkit
   | (_| |
    > _  |   Commands:
   /_/ |_|       help      -> Displays the help system
   . . . .       object?   -> Display information about 'object'
   . . . .       exit/quit -> Exit
   . . . .
   . . . .   More info at https://frida.re/docs/home/
Spawned `com.example.pinned`. Resuming main thread!
[Android Emulator 5554::com.example.pinned]->
[.] Cert Pinning Bypass/Re-Pinning
[+] Loading our CA...
[o] Our CA Info: CN=PortSwigger CA, OU=PortSwigger CA, O=PortSwigger,
L=PortSwigger, ST=PortSwigger, C=PortSwigger
[+] Creating a KeyStore for our CA...
[+] Creating a TrustManager that trusts the CA in our KeyStore...
[+] Our TrustManager is ready...
[+] Hijacking SSLContext methods now...


[-] Waiting for the app to invoke SSLContext.init()...
```

The app is now started via `frida`.



Now let's try to catch the request. Make sure the `Intercept is on` button is toggled on, on the `Proxy` -> `Intercept` tab in Burp. Finally we tap the `LOGIN` button in the app, and check Burp.



```
POST /pinned.php HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Accept: application/x-www-form-urlencoded
charset: utf-8
User-Agent: Dalvik/2.1.0 (Linux; U; Android 7.1.1; Android SDK built for x86_64 Build/NYC)
Host: pinned.com
Connection: close
Accept-Encoding: gzip, deflate
Content-Length: 52

uname=bnavarro&pass=HTB{trust_n0_1_n0t_3v3n_@_c3rt!}
```

The request is intercepted successfully and the flag is revealed in the `pass` parameter.