Motivation

Mobile apps have become an essential part of our lives, providing us with a wide range of services and entertainment. However, with the increasing popularity of mobile apps comes the growing concern about the data that they collect and how it is used.

Mobile app developers collect a wide variety of data, including personal information, location data, and usage data. This data can be used for a variety of purposes, such as improving the user experience, targeted advertising, and fraud prevention. However, there is also the potential threat to mobile apps users as their collected data can be used in ways that are harmful to users, such as tracking their movements without their consent or selling their personal information to third-party advertisers.

It is important to ensure that mobile app developers collect and use data in a responsible and ethical manner. This is especially important in the context of Indian and US mobile app developers, as there are significant differences in the privacy laws and regulations of the two countries which vary in a variety of arenas. The US has a number of security standards that mobile app developers are encouraged to follow, such as the National Institute of Standards and Technology (NIST) Cybersecurity Framework and the Payment Card Industry Data Security Standard (PCI DSS). However, these standards are not mandatory.

India does not have any mandatory security standards for mobile apps. However, the Indian government has issued a number of guidelines on mobile app security, such as the "Guidelines on Mobile App Security" issued by the Indian Computer Emergency Response Team (CERT-In). Overall, the security policies for mobile apps in the US are more stringent than those in India. This is due to the fact that the US has a number of comprehensive privacy laws and security standards in place.

Feature	US	India
Privacy laws	Federal and state laws govern the collection and use of personal data.	No comprehensive privacy law at the federal level. Sector-specific laws regulate the collection and use of personal data.
Data localization requirements	No specific data localization requirements. Some states have laws that require companies to store certain types of data in the US.	A number of data localization requirements for mobile apps. For example, financial data collected by mobile apps must be stored in India.

Security standards	A number of security standards that mobile app developers are encouraged to follow, but not mandatory.	No mandatory security standards for mobile apps. However, the Indian government has issued a number of guidelines on mobile app security.

Vision

The vision of this research is to develop a framework for measuring and comparing the appropriateness of data collection purposes in mobile apps created by Indian and US developers. This framework would be used to assess the risks and benefits of different data collection practices, and to help users make informed decisions about which apps to install and use.

Objectives

The specific objectives of this research are to:

- Identify the most common data collection purposes of mobile apps created by Indian and US developers.
- Compare the stated data collection purposes of Indian and US mobile app developers.
- Assess the perceptions of Indian and US mobile app users regarding the appropriateness of different data collection purposes.
- Evaluate the compliance of Indian and US mobile app developers with their stated data collection purposes.
- Identify the factors that influence the appropriateness of data collection purposes in mobile apps.
- Develop a framework for measuring and comparing the appropriateness of data collection purposes in mobile apps created by Indian and US developers.

Research Questions

Here are some more specific research questions:

- How do Indian and US mobile app developers differ in their data collection practices for certain types of data, such as location data, personal contact information, and financial data?
- How do Indian and US mobile app users' perceptions of the appropriateness of data collection purposes vary depending on the type of data being collected, the purpose for which the data is being collected, and the app developer's country of origin?

- What are the ethical implications of data collection practices in Indian and US mobile apps?
- How can the appropriateness of data collection purposes in mobile apps be measured and compared more effectively?
- What are the best practices for ensuring that mobile app developers collect and use data in a responsible and ethical manner?

Impact

The findings of this research would have a significant impact on the following stakeholders:

1. **Mobile app developers**: The research would provide mobile app developers with guidance on how to collect and use data in a responsible and ethical manner. This would help to improve the reputation of the mobile app industry and build trust with users.
2. **Mobile app users**: The research would empower mobile app users to make informed decisions about which apps to install and use. The research would also provide users with the information they need to protect their privacy and security.
3. **Policymakers**: The research would provide policymakers with the evidence they need to develop effective privacy laws and regulations for the mobile app industry.

Detailed Methodology

The research would be conducted in three phases:

Phase 1: Data collection

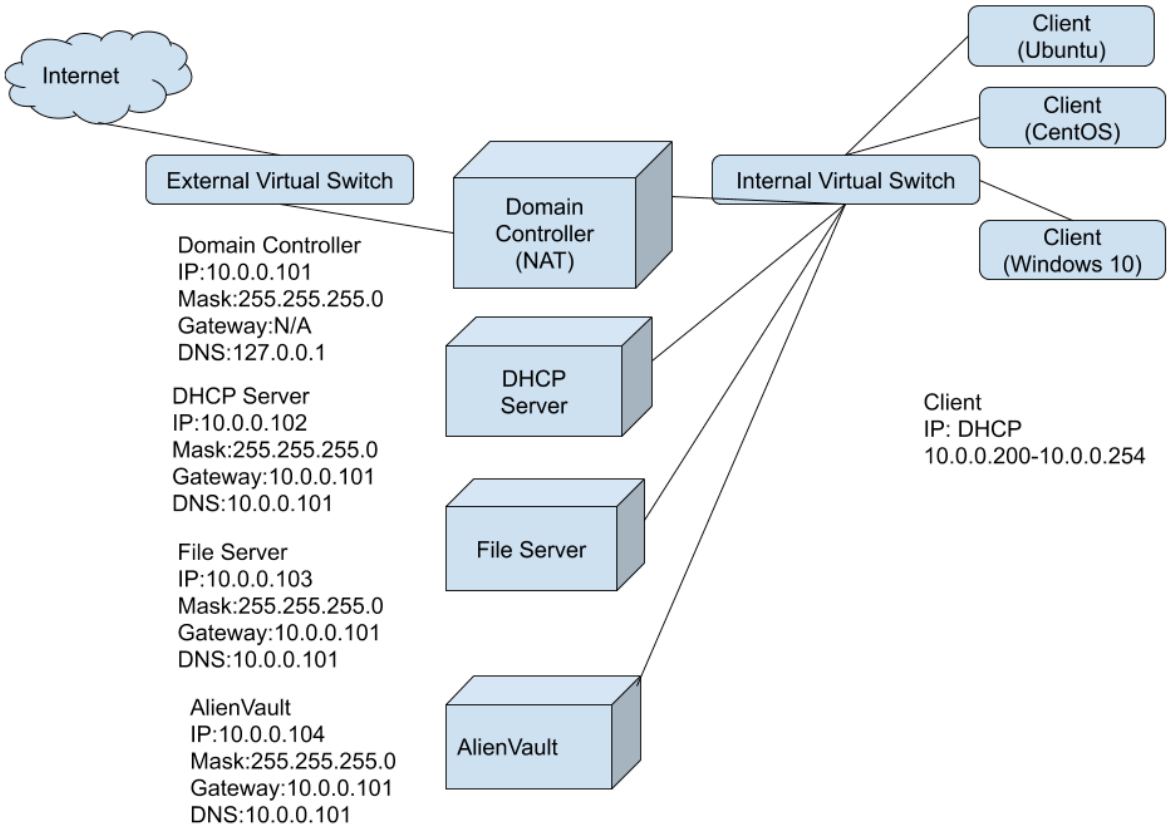
The first phase of the research would involve collecting data on the data collection practices of Indian and US mobile app developers. This data would be collected using a variety of methods, such as:

- Content analysis: The researchers would analyze the privacy policies of mobile apps to identify the types of data that are collected and the purposes for which the data is used. Decipher and Decrypt the Prompt (for Data Privacy and Permission etc.), Logs and Packets for Data Communication between the Client side of the Mobile Application and the Server.
- User surveys: The researchers would conduct surveys of mobile app users to gather their perceptions of the appropriateness of different data collection purposes. As well gather their consent and views for the next and most important Interview Portion.
- Interviews: The researchers would interview mobile app developers to gain insights into their data collection practices and their motivations for collecting data.

Phase 2: Data analysis
The second phase of the research would involve analyzing the data collected in Phase 1. The researchers would use a variety of statistical methods to identify the most common data collection purposes, compare the data collection practices of Indian and US mobile app developers, and assess the perceptions of mobile app users on their Data Privacy Concerns and Security.

Phase 3: Framework development

The third phase of the research would involve developing a framework for measuring and comparing the appropriateness of data collection purposes in mobile apps created by Indian and US developers. The framework would be developed based on the findings of the first two phases of the research, as well as existing research on data privacy and ethics.



Data Collection Methods Planned to be Used:-

Run the Applications Offline and Analyze using Android Studio

To access, decrypt, and analyze the logs/prompts of mobile applications simulated on Android Studio for data privacy and security analysis, you can follow these steps:

1. Access the logs/prompts

To access the logs/prompts of a mobile application simulated on Android Studio, you can use the following methods:

- Logcat: Logcat is a tool that allows you to view and analyze the logs generated by Android applications. To open Logcat, select View > Tool Windows > Logcat.
- Systrace: Systrace is a tool that allows you to record and analyze the system traces of Android applications. To open Systrace, select Run > Start profiling.

2. Decrypt the logs/prompts

If the logs/prompts are encrypted, you will need to decrypt them before you can analyze them. There are a number of tools that you can use to decrypt logs/prompts, such as:

- Android Debug Bridge (adb): Adb is a tool that allows you to communicate with Android devices. To decrypt logs/prompts using adb, you can use the following command:

  ```
  adb shell cat /sdcard/logs/logcat.txt | openssl enc -d -aes-128-cbc -K YOUR_KEY
  ```

3. Analyze the logs/prompts

Once you have accessed and decrypted the logs/prompts, you can start analyzing them. You can use a variety of tools to analyze logs/prompts, such as:

- Logcat: Logcat provides a number of features for analyzing logs, such as filtering, searching, and exporting.
- Systrace: Systrace provides a number of features for analyzing system traces, such as viewing the timeline of events and identifying performance bottlenecks.

- Log analysis tools: There are a number of third-party log analysis tools available, such as Splunk and Elasticsearch. These tools can provide more advanced features for analyzing logs, such as machine learning and data visualization.

Example

The following example shows how to use Logcat to access and analyze the logs of a mobile application simulated on Android Studio:

1. Open Android Studio and simulate a mobile application.
2. Select View > Tool Windows > Logcat.
3. In the Logcat window, select the All filter.
4. To view the logs of a specific process, select the process from the Processes drop-down menu.
5. To search for a specific message in the logs, enter the message in the Search text box.
6. To export the logs, click the Export all button.

The following example shows how to use Systrace to record and analyze the system traces of a mobile application simulated on Android Studio:

1. Open Android Studio and simulate a mobile application.
2. Select Run > Start profiling.
3. The Systrace window will open.
4. Click the Record button to start recording the system traces.
5. Reproduce the issue that you are trying to analyze.
6. Click the Stop button to stop recording the system traces.
7. To analyze the system traces, click the Analyze button.
8. The Systrace window will display a timeline of events.
9. To view more details about an event, click the event.
10. To identify performance bottlenecks, click the Performance tab.

By following these steps, you can access, decrypt, and analyze the logs/prompts of mobile applications simulated on Android Studio for data privacy and security analysis. This

information can be used to measure and compare the appropriateness of data collection purposes in mobile apps created by Indian and US developers.

Wireshark

Wireshark is a network traffic analyzer that can be used to capture and analyze the logs and prompts of mobile apps simulated on a computer. This can be useful for data privacy and security analysis, as it can help to identify the types of data that are being collected by mobile apps and the purposes for which the data is being used. To access, decrypt, and analyze the logs / prompts of mobile applications simulated on Wireshark for data privacy and security analysis:

1. Capture the traffic between the mobile device and the internet using Wireshark. This can be done by connecting the mobile device to the computer using a USB cable and putting the computer's network interface card (NIC) in promiscuous mode.
2. Save the captured traffic to a file. This will allow you to analyze the traffic later.
3. Install a TLS decryption certificate on the computer. This is necessary to decrypt the HTTPS traffic that is captured by Wireshark.
4. Open the captured traffic file in Wireshark.
5. Decrypt the HTTPS traffic. To do this, go to Edit > Preferences > Protocols > TLS and select the TLS decryption certificate that you installed in step 3.
6. Analyze the traffic. Wireshark provides a variety of tools for analyzing traffic, such as filters, packet viewers, and statistics.

Example:

To analyze the data collection practices of a mobile app, you would first need to capture the traffic between the mobile device and the internet using Wireshark. Once you have captured the traffic, you would need to save it to a file and decrypt it if necessary. Once the traffic is decrypted, you can open it in Wireshark and use the various tools provided by Wireshark to analyze the traffic.

For example, you could use the filters to filter the traffic to specific IP addresses or ports. You could also use the packet viewers to view the individual packets in the traffic.

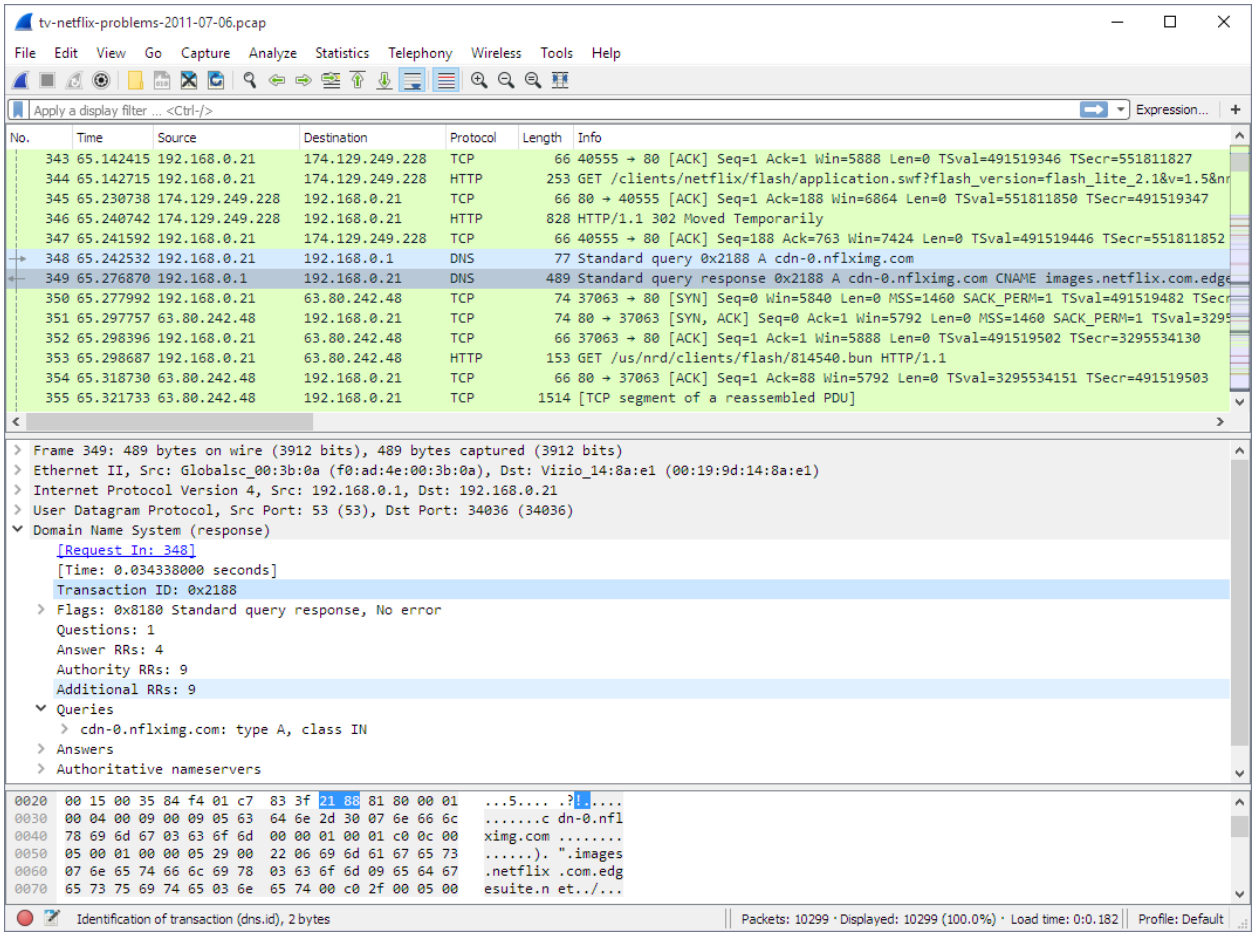
Additionally, you could use the statistics to see how much data is being transferred and which protocols are being used.

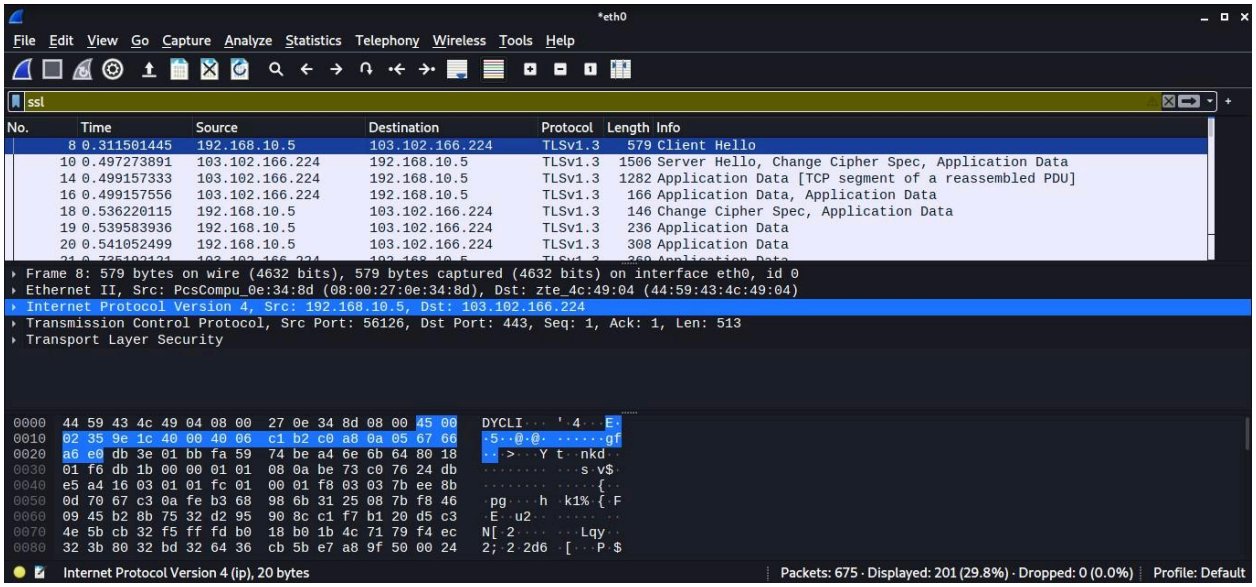
By analyzing the traffic, you can identify the types of data that are being collected by the mobile app, the purposes for which the data is being collected, and the third parties that the data is being shared with. This information can be used to assess the privacy and security risks of the mobile app.

Here are some specific examples of how Wireshark can be used to analyze the data collection practices of mobile apps:

- Identify the types of data that are being collected: You can use the filters in Wireshark to filter the traffic to specific IP addresses or ports that are known to be used by the mobile app to collect data. For example, you could filter the traffic to the IP addresses of Google Analytics or Facebook Pixel servers.
- Identify the purposes for which the data is being collected: You can use the packet viewers in Wireshark to view the individual packets in the traffic to see what data is being collected and how it is being used. For example, you could view the HTTP requests that are being sent to the mobile app's servers to see what data is being submitted.
- Identify the third parties that the data is being shared with: You can use the packet viewers in Wireshark to view the IP addresses of the servers that the mobile app is communicating with. This information can be used to identify the third parties that the mobile app is sharing data with.

By analyzing the traffic in Wireshark, one can gain a deep understanding of the data collection practices of mobile apps. This information can be used to assess the privacy and security risks of the mobile app and to make informed decisions about which apps to install and use.


tv-netflix-problems-2011-07-06.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...
No. Time Source Destination Protocol Length Info
343 65.142415 192.168.0.21 174.129.249.228 TCP 66 40555 → 80 [ACK] Seq=1 Ack=1 Win=5888 Len=0 TSval=491519346 TSecr=551811827
344 65.142715 192.168.0.21 174.129.249.228 HTTP 253 GET /clients/netflix/flash/application.swf?flash_version=flash_lite_2.1&v=1.5&nr
345 65.230738 174.129.249.228 192.168.0.21 TCP 66 80 → 40555 [ACK] Seq=1 Ack=188 Win=6864 Len=0 TSval=551811850 TSecr=491519347
346 65.240742 174.129.249.228 192.168.0.21 HTTP 828 HTTP/1.1 302 Moved Temporarily
347 65.241592 192.168.0.21 174.129.249.228 TCP 66 40555 → 80 [ACK] Seq=188 Ack=763 Win=7424 Len=0 TSval=491519446 TSecr=551811852
348 65.242532 192.168.0.21 192.168.0.1 DNS 77 Standard query 0x2188 A cdn-0.nflximg.com
349 65.276870 192.168.0.1 192.168.0.21 DNS 489 Standard query response 0x2188 A cdn-0.nflximg.com CNAME images.netflix.com.edge
350 65.277992 192.168.0.21 63.80.242.48 TCP 74 37063 → 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=491519482 TSecr
351 65.297757 63.80.242.48 192.168.0.21 TCP 74 80 → 37063 [SYN, ACK] Seq=0 Ack=1 Win=5792 Len=0 MSS=1460 SACK_PERM=1 TSval=3295
352 65.298396 192.168.0.21 63.80.242.48 TCP 66 37063 → 80 [ACK] Seq=1 Ack=1 Win=5888 Len=0 TSval=491519502 TSecr=3295534130
353 65.298687 192.168.0.21 63.80.242.48 HTTP 153 GET /us/nrd/clients/flash/814540.bun HTTP/1.1
354 65.318730 63.80.242.48 192.168.0.21 TCP 66 80 → 37063 [ACK] Seq=1 Ack=88 Win=5792 Len=0 TSval=3295534151 TSecr=491519503
355 65.321733 63.80.242.48 192.168.0.21 TCP 1514 [TCP segment of a reassembled PDU]
Frame 349: 489 bytes on wire (3912 bits), 489 bytes captured (3912 bits)
Ethernet II, Src: Globalsc_00:3b:0a (f0:ad:4e:00:3b:0a), Dst: Vizio_14:8a:e1 (00:19:9d:14:8a:e1)
Internet Protocol Version 4, Src: 192.168.0.1, Dst: 192.168.0.21
User Datagram Protocol, Src Port: 53 (53), Dst Port: 34036 (34036)
Domain Name System (response)
[Request In: 348]
[Time: 0.034338000 seconds]
Transaction ID: 0x2188
Flags: 0x8180 Standard query response, No error
Questions: 1
Answer RRs: 4
Authority RRs: 9
Additional RRs: 9
Queries
cdn-0.nflximg.com: type A, class IN
Answers
Authoritative nameservers
0020 00 15 00 35 84 f4 01 c7 83 3f 21 88 81 80 00 01 ...5.... .?!.....
0030 00 04 00 09 00 09 05 63 64 6e 2d 30 07 6e 66 6c ....c dn-0.nfl
0040 78 69 6d 67 03 63 6f 6d 00 00 01 00 01 c0 0c 00 ximg.com ....
0050 05 00 01 00 00 05 29 00 22 06 69 6d 61 67 65 73 ......). ".images
0060 07 6e 65 74 66 6c 69 78 03 63 6f 6d 09 65 64 67 .netflix .com.edg
0070 65 73 75 69 74 65 03 6e 65 74 00 c0 2f 00 05 00 esuite.n et../...
Identification of transaction (dns.id), 2 bytes
Packets: 10299 · Displayed: 10299 (100.0%) · Load time: 0:0.182
Profile: Default


*eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
ssl
No. Time Source Destination Protocol Length Info
8 0.311501445 192.168.10.5 103.102.166.224 TLSv1.3 579 Client Hello
10 0.497273891 103.102.166.224 192.168.10.5 TLSv1.3 1506 Server Hello, Change Cipher Spec, Application Data
14 0.499157333 103.102.166.224 192.168.10.5 TLSv1.3 1282 Application Data [TCP segment of a reassembled PDU]
16 0.499157556 103.102.166.224 192.168.10.5 TLSv1.3 166 Application Data, Application Data
18 0.536220115 192.168.10.5 103.102.166.224 TLSv1.3 146 Change Cipher Spec, Application Data
19 0.539583936 192.168.10.5 103.102.166.224 TLSv1.3 236 Application Data
20 0.541052499 192.168.10.5 103.102.166.224 TLSv1.3 308 Application Data
Frame 8: 579 bytes on wire (4632 bits), 579 bytes captured (4632 bits) on interface eth0, id 0
Ethernet II, Src: PcsCompu_0e:34:8d (08:00:27:0e:34:8d), Dst: zte_4c:49:04 (44:59:43:4c:49:04)
Internet Protocol Version 4, Src: 192.168.10.5, Dst: 103.102.166.224
Transmission Control Protocol, Src Port: 56126, Dst Port: 443, Seq: 1, Ack: 1, Len: 513
Transport Layer Security
0000 44 59 43 4c 49 04 08 00 27 0e 34 8d 08 00 45 00 DYCLI··· '·4···E·
0010 02 35 9e 1c 40 00 40 06 c1 b2 c0 a8 0a 05 67 66 ·5··@·@· ·····gf
0020 a6 e0 db 3e 01 bb fa 59 74 be a4 6e 6b 64 80 18 ···>···Y t··nkd··
0030 01 f6 db 1b 00 00 01 01 08 0a be 73 c0 76 24 db ·····s·v$·
0040 e5 a4 16 03 01 01 fc 01 00 01 f8 03 03 7b ee 8b ·····{··
0050 0d 70 67 c3 0a fe b3 68 98 6b 31 25 08 7b f8 46 ·pg····h ·k1%·{·F
0060 09 45 b2 8b 75 32 d2 95 90 8c c1 f7 b1 20 d5 c3 ·E··u2·· ····· ··
0070 4e 5b cb 32 f5 ff fd b0 18 b0 1b 4c 71 79 f4 ec N[·2···· ···Lqy··
0080 32 3b 80 32 bd 32 64 36 cb 5b e7 a8 9f 50 00 24 2;·2·2d6 ·[···P·$
Internet Protocol Version 4 (ip), 20 bytes
Packets: 675 · Displayed: 201 (29.8%) · Dropped: 0 (0.0%)
Profile: Default

TCP-DUMP

Accessing, Decrypting, and Analyzing Mobile App Logs/Prompts in Wireshark (via tcpdump) for Data Privacy and Security Analysis. To access, decrypt, and analyze mobile app logs/prompts in Wireshark (via tcpdump), you will need the following:

- A computer with Wireshark installed.
- A mobile app simulator.
- A mobile app that you want to analyze.
- A root certificate installed on the computer.

Steps:

1. Connect the mobile app simulator to the computer using a USB cable.
2. Start the mobile app simulator and launch the mobile app that you want to analyze.
3. Open Wireshark and start a new capture session.
4. In Wireshark, go to Capture > Interfaces and select the network interface that the mobile app simulator is using.
5. Click Start to start the capture session.
6. On the mobile app simulator, perform actions that will generate logs and prompts.
7. After you have finished generating logs and prompts, stop the capture session in Wireshark.

To decrypt the captured traffic, you will need to import the root certificate into Wireshark. To do this:

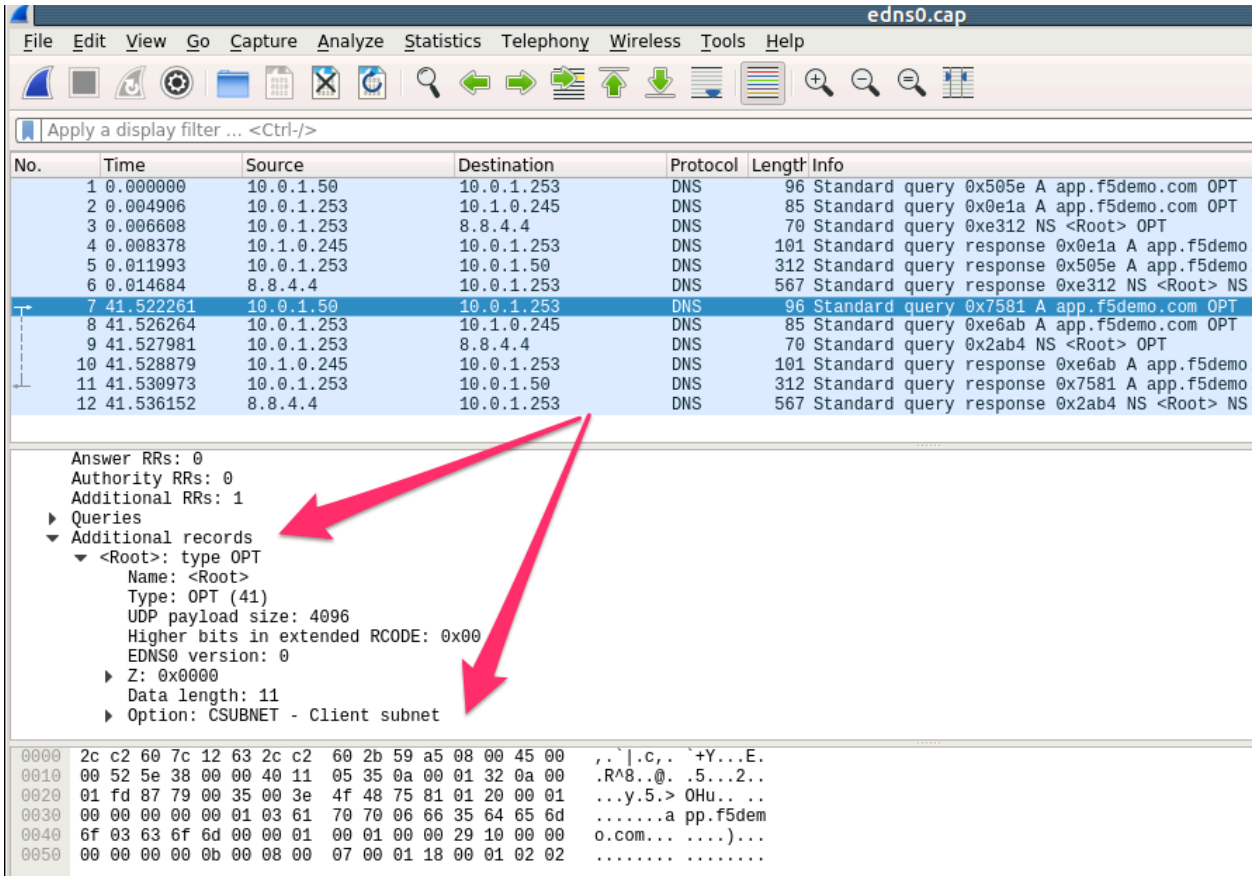
1. In Wireshark, go to Edit > Preferences.
2. Click the Protocols tab.
3. Expand the TLS tree and select the Pre-Master Secret Decryption section.
4. Click the Edit button.
5. Click the Add button and select the root certificate file.
6. Click OK to save your changes.

Once you have imported the root certificate, you can decrypt the captured traffic by:

1. Right-clicking on a captured packet and selecting Decrypt.

2. In the Decrypt Packet window, select the TLS protocol and click Decrypt.

Once you have decrypted the captured traffic, you can analyze the logs and prompts to identify the types of data that are being collected by the mobile app and the purposes for which the data is being used.



Example

The following example shows how to access, decrypt, and analyze the logs/prompts of a mobile app simulated on an Android emulator using Wireshark (via tcpdump):

1. Connect the Android emulator to the computer using a USB cable.
2. Start the Android emulator and launch the mobile app that you want to analyze.
3. Open Wireshark and start a new capture session.
4. In Wireshark, go to Capture > Interfaces and select the network interface that the Android emulator is using.

5. Click Start to start the capture session.
6. On the Android emulator, open the mobile app and perform actions that will generate logs and prompts.
7. After you have finished generating logs and prompts, stop the capture session in Wireshark.

To decrypt the captured traffic, follow the steps above to import the root certificate into Wireshark.

Once you have imported the root certificate, you can decrypt the captured traffic by right-clicking on a captured packet and selecting Decrypt > TLS.

To analyze the logs and prompts, you can use the Wireshark packet inspector to view the contents of the packets. You can also use the Wireshark display filters to filter the captured traffic by protocol, IP address, or port number.

Burp Suite

Accessing, Decrypting, and Analyzing Mobile App Logs and Prompts in BurpSuite for Data Privacy and Security Analysis

BurpSuite is a powerful tool that can be used to intercept, modify, and analyze HTTP and HTTPS traffic. This makes it ideal for accessing, decrypting, and analyzing mobile app logs and prompts.

To access mobile app logs and prompts in BurpSuite:

1. Install and configure BurpSuite: Download and install BurpSuite from the BurpSuite website. Once installed, open BurpSuite and go to the Proxy tab. Select the "Intercept" checkbox and click the "Start" button.

2. Launch the mobile app: Launch the mobile app on your device and perform the actions that you want to log. BurpSuite will intercept all of the HTTP and HTTPS traffic between the mobile app and the server.
3. Save the logs: To save the logs, go to the History tab in BurpSuite and click the "Save" button. You can save the logs in a variety of formats, such as XML, JSON, and HAR.

To decrypt mobile app logs and prompts in BurpSuite:

If the mobile app is using HTTPS, the logs will be encrypted. To decrypt the logs, you will need to install the certificate authority (CA) certificate from BurpSuite on your device.

1. Install the BurpSuite CA certificate: Open BurpSuite and go to the Proxy tab. Click the "CA certificate" button and select "Install CA certificate".
2. Trust the BurpSuite CA certificate on your device: The specific steps required to trust the BurpSuite CA certificate will vary depending on your device. For example, on Android devices, you need to go to Settings > Security > Advanced > Certificate store > Trusted root certificates and add the BurpSuite CA certificate.
3. Restart the mobile app: Once the BurpSuite CA certificate is installed and trusted, restart the mobile app. BurpSuite will now be able to decrypt the HTTPS traffic between the mobile app and the server.

To analyze mobile app logs and prompts in BurpSuite:

Once the logs have been decrypted, you can use BurpSuite to analyze them for data privacy and security risks.

1. Review the logs: Open the logs in BurpSuite and review them to identify any sensitive data that is being collected or transmitted. Some examples of sensitive data include personal information, such as names, addresses, and

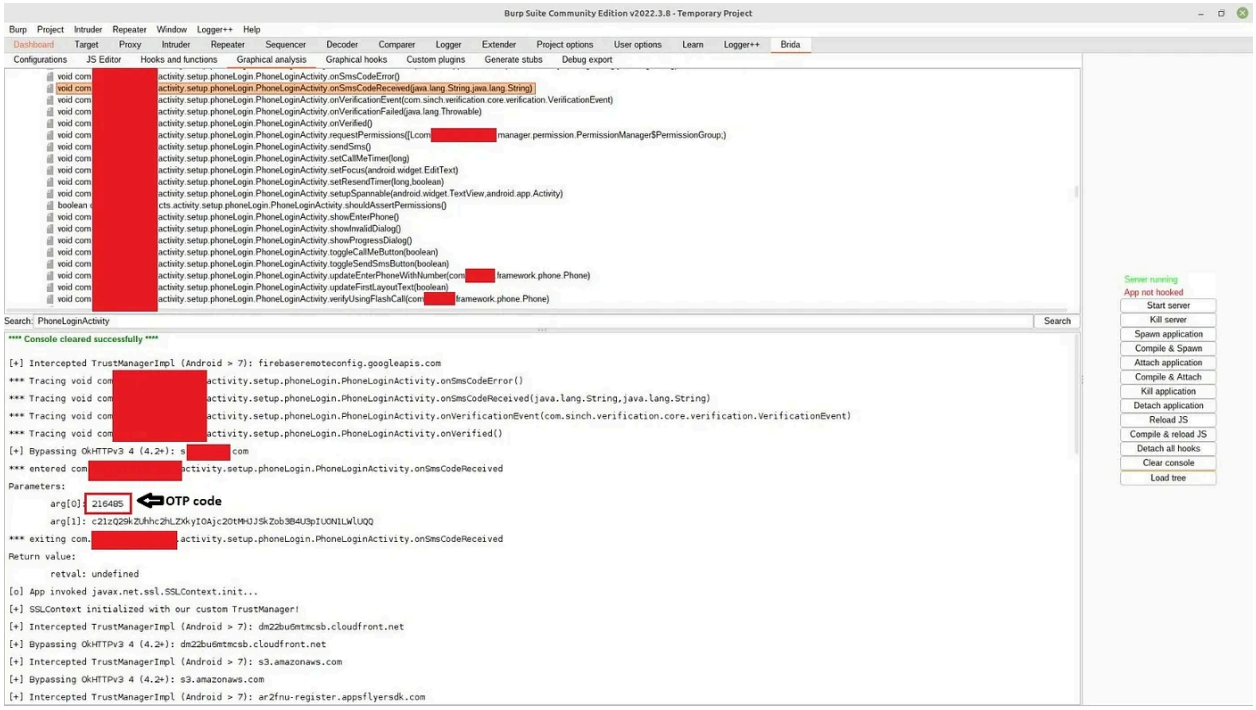
phone numbers, and financial data, such as credit card numbers and bank account numbers.

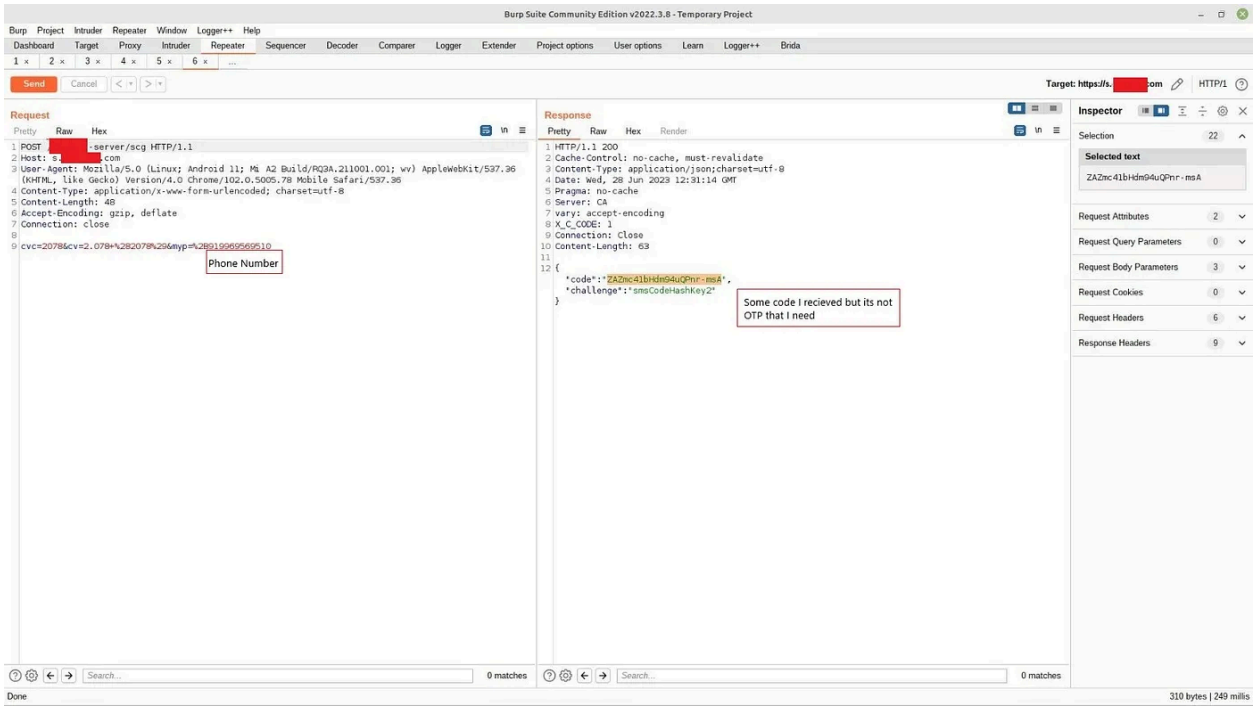
2. Identify security risks: Review the logs to identify any potential security risks. Some examples of security risks include cleartext transmission of passwords, cross-site scripting (XSS) vulnerabilities, and SQL injection vulnerabilities.

Example:

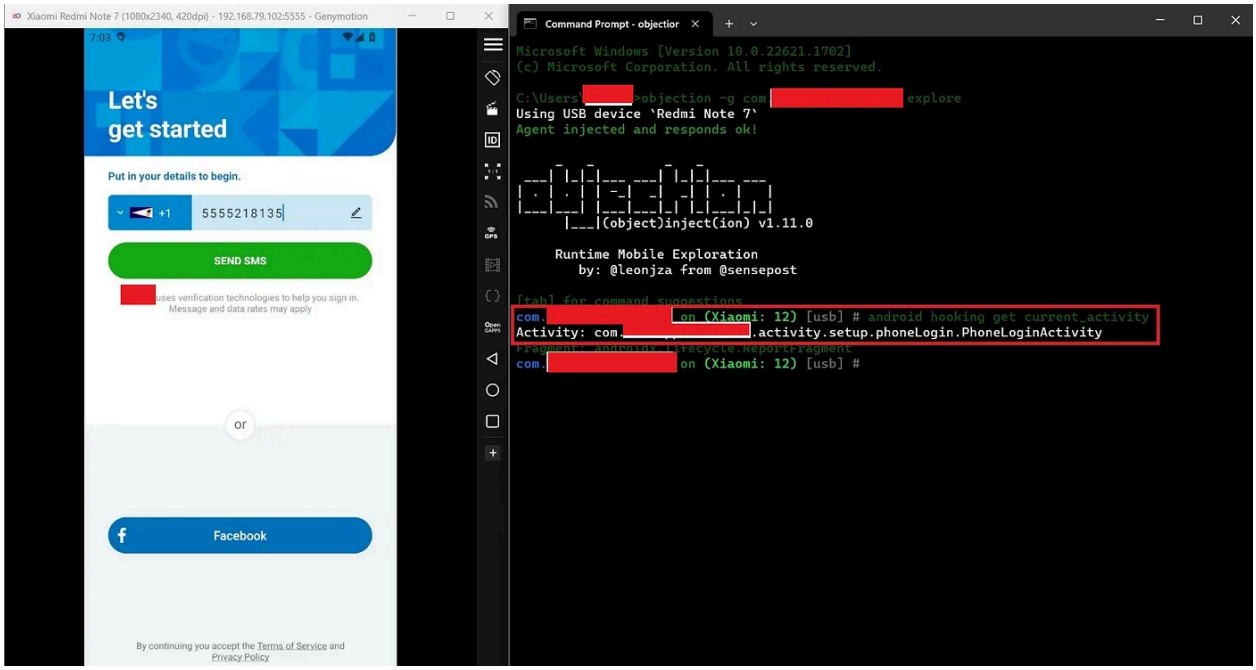
The following screenshot shows an example of a mobile app log that has been decrypted in BurpSuite:

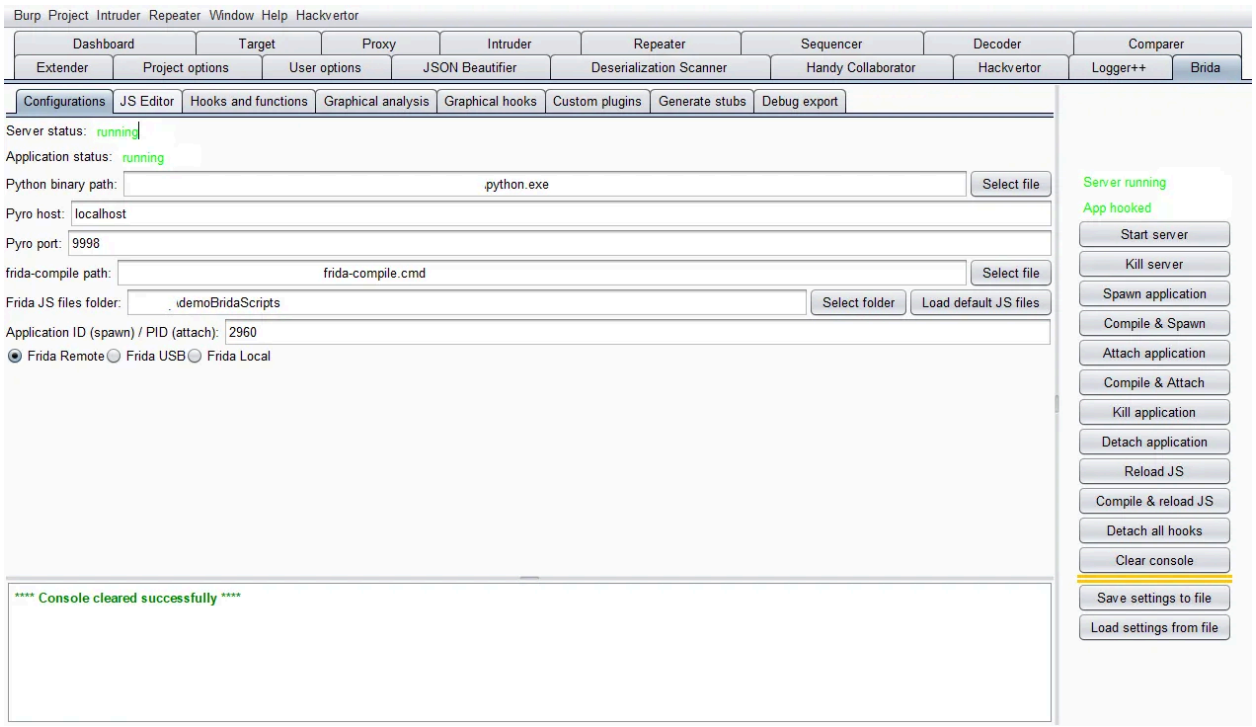
This log shows that the mobile app is collecting the user's name, email address, and phone number. This information is being transmitted to the server in cleartext, which is a security risk.






```
com                    on (Xiaomi: 12) [usb] # android hooking list class_methods com.                    activity.setup.phoneLogin.PhoneLoginActivity
private com.           .framework.phone.Phone com.                    activity.setup.phoneLogin.PhoneLoginActivity.getEnteredPhone()
private com.           .framework.phone.Phone com.                    activity.setup.phoneLogin.PhoneLoginActivity.getFixedPhoneForRegistration(com.
rk.phone.Phone)
```





Conclusion

By using BurpSuite to access, decrypt, and analyze mobile app logs and prompts, you can identify potential data privacy and security risks. This information can then

be used to improve the security of the mobile app and protect the privacy of its users.

Using BurpSuite to Measure and Compare the Appropriateness of Data Collection Purposes in Mobile Apps Created by Indian and US Developers

BurpSuite can also be used to measure and compare the appropriateness of data collection purposes in mobile apps created by Indian and US developers.

To measure the appropriateness of data collection purposes in a mobile app:

1. Identify the data that is being collected: Use BurpSuite to intercept and analyze the HTTP and HTTPS traffic between the mobile app and the server to identify the data that is being collected.
2. Compare the data that is being collected to the stated data collection purposes: Review the mobile app's privacy policy to identify the stated data collection purposes. Compare the data that is being collected to the stated data collection purposes to determine if the data collection is appropriate.

To compare the appropriateness of data collection purposes in mobile apps created by Indian and US developers:

1. Measure the appropriateness of data collection purposes in a sample of mobile apps created by Indian developers: Follow the steps above to measure the appropriateness of data collection purposes in a sample of mobile apps created by Indian developers.
2. Measure the appropriateness of data collection purposes in a sample of mobile apps created by US developers: Follow the steps above to measure the appropriateness of data collection

Conclusion

This research is important because it will help to improve the transparency and accountability of mobile app developers and protect the privacy of mobile app users. The research findings will also be valuable to policymakers who are developing privacy laws and regulations for the mobile app industry.

Few References (Final Doc Mein Tikh laga to Dal Dena)

CA Certification Creation and Linking to Computer for use in Wireshark
https://medium.com/@darryncampbell_83863/android-traffic-analysis-to-google-servers-methodology-6de1435c5157
Burp Suite
https://medium.com/@amolbhavar/how-i-get-1000-bounty-for-discovering-account-takeover-in-android-application-3c4f54fbde39
Android Studio Rooted Phone
https://medium.com/itnext/how-to-listen-apps-traffic-on-a-rooted-emulator-471a842e89bd