How to Intercept Encrypted Messages On Android

About Me

- Natalie Silvanovich aka natashenka
- Google Project Zero Member
- Previously on Android Security Team

My (Mostly Failed) Goal

- Find remote vulnerabilities in Android messaging clients
  - Facebook Messenger
  - Whatsapp
  - WeChat
  - Signal
  - Telegram

Why E2E Encrypted Messaging?

- Some messengers do not encrypt, or encrypt from device to server and then server to device
  - Server can sanitize messages
  - Exploiting a remote server blind is *hard*
- End-to-end encrypted messages cannot be altered by server
  - Must be processed on device

Problem

- How to alter messages inside the encryption wrapper?
  - Bugs that occur pre-encryption are rare
  - Ideally want to alter a message and have decryption and signature verification succeed

Possible Strategies

- Implement the protocol
- Find existing tools
- Stubbing

Implement the protocol?

- Most messengers publish their encryption protocols
  - Since we know our own key, we should be able to replicate it
- But, but …
  - Documents are long and possibly inaccurate
  - A lot of work and very error prone

Use Existing Tools

- There’s a lot of authorized and unauthorized apps that bring mobile messengers to the desktop
- Looked at many of them, and they often use different protocols (external APIs)
- E2E encryption often not implemented

STUBBING

Basic idea:

- Find where message is encrypted
- Insert code after the message has been serialized, but before it has been signed or encrypted
- Code sends message to remote server, where it can be changed
- Altered message gets sent to test device

Finding the Encryption Point

- Start by decompiling the application APK using apktool
- Get smali files out
- Typically obfuscated
- Android applications contain a lot of unused and rarely used code

```
.method public constructor
<init>(LX/8A2;LX/0Gl;LX/0Gl;LX/89x;LX/1q1;LX/1Xs;LX/0wj;LX/0Gl;LX/1pr;LX/0wQ;LX
/0oS;LX/0dK;LX/0wO;LX/0Gl;LX/1q5;LX/0wm;)V
    .locals 10
    invoke-direct {p0}, Ljava/lang/Object;-><init>()V
    iput-object v9, p0, LX/89y;->c:LX/8A2;
    iput-object v7, p0, LX/89y;->d:LX/0Gl;
    iput-object v6, p0, LX/89y;->e:LX/0Gl;
    iput-object v5, p0, LX/89y;->f:LX/89x;
    iput-object v4, p0, LX/89y;->g:LX/1q1;
    iput-object p4, p0, LX/89y;->h:LX/1Xs;
    iput-object v1, p0, LX/89y;->i:LX/0wj;
    iput-object v0, p0, LX/89y;->j:LX/0Gl;
```

Strategies

- Look for known libraries
  - libsignal
  - Java crypto
- Focus on natives
- Log entries

Known Libraries

- Most E2E encrypted messengers include libsignal
  - Unfortunately, full feature set is not used
  - Putting in a stub where libsignal encrypts messages (based on Signal source) did not work on most messengers

Java Crypto Libs

Cheap trick:

- Make a build of Android that has a stub in javax.crypto.Mac
- Make the stub send the digest only when it can access a file in the sandbox of the app you’re testing
- Will get a lot of stuff that isn’t messages, plus sometimes messages
- Works on about half of messengers

Java Crypto Libs

- Also possible to put log entry that outputs Java stack in Java crypto libs
- Can help you find where the app is encrypting the message
- Relies on the app actually using Java crypto
- Apps often implement their own encryption (wrap a native library), but usually use Java for signing
- Once output stacks in System.arraycopy when I was desperate
- Can also search smali, but no guarantee stuff gets called

Natives (JNI)

- Java Native Interface calls cannot be obfuscated (easily)
- Calls with ‘encrypt’ in the name are good candidates for stub locations
  - Messaging encryption is usually native
  - Be careful to separate file encryption from network encryption
- Made a script that outputs log entries for every native call

JNI Question

In a Java application, can native code be run without a JNI call?

No.

- JNI can start threads, etc, but native code **always** starts with a JNI call in and Android Java application

Log Entries

- Some apps have a lot of helpful log entries (and some don't)

```
const/4 v10, 0x0
monitor-enter v4
:try_start_0
iget-object v0, v4, LX/8B3;->d:Ljavax/crypto/Mac;
if-nez v0, :cond_10
sget-object v1, LX/8B3;->a:Ljava/lang/Class;
const-string v0, "Could not verify Salamander signature - no SHA256HMAC"
invoke-static {v1, v0}, LX/00T;->b(Ljava/lang/Class;Ljava/lang/String;)V
:try_end_0
.catchall {:try_start_0 .. :try_end_0} :catchall_0
```

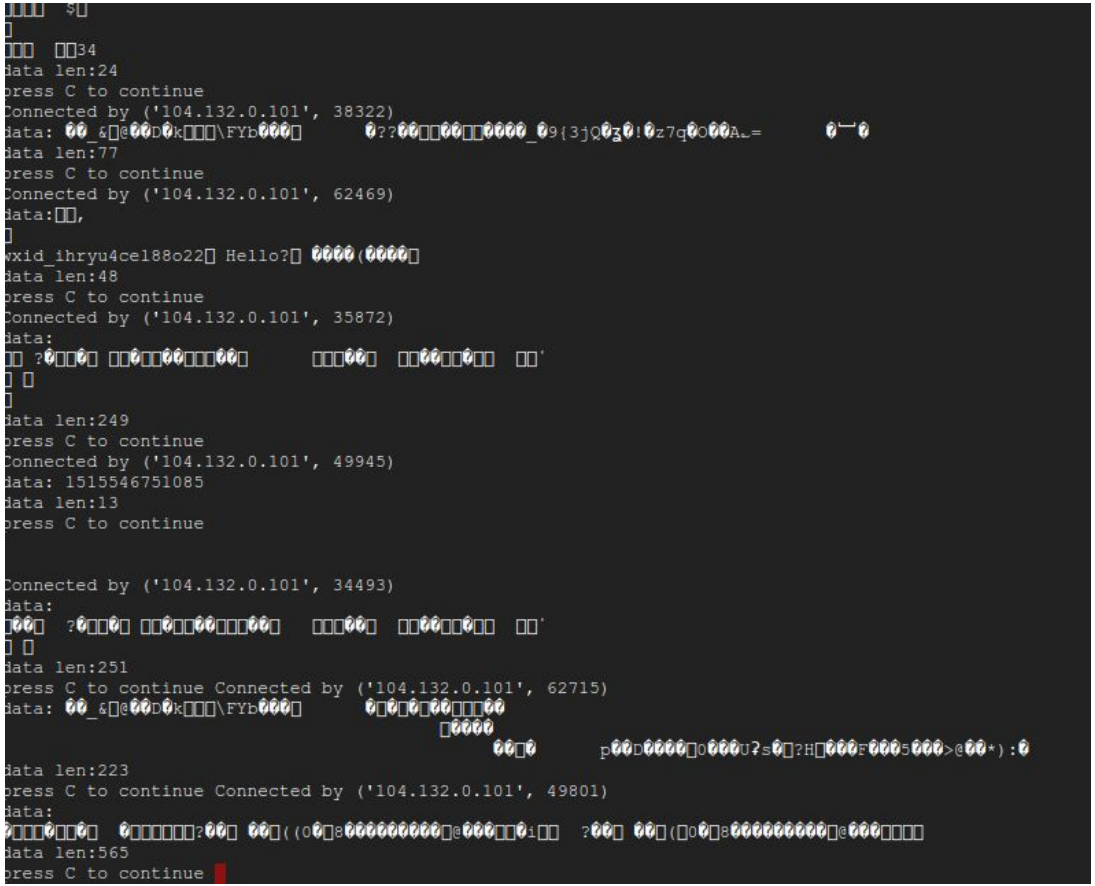
Log Entries

- Signature verification failure is a good log entry to look for
- Remember, you can add your own log entries

More About Message Encryption

- Apps usually have more than one location where they encrypt messages
  - Messages
  - Attachments
  - Typing/presence indicator
  - Notification content
- Usually need to add multiple stubs

Messages!


data len:24
press C to continue
Connected by ('104.132.0.101', 38322)
data len:77
press C to continue
Connected by ('104.132.0.101', 62469)
data:
wxid_ihryu4ce188o22 Hello?
data len:48
press C to continue
Connected by ('104.132.0.101', 35872)
data:
data len:249
press C to continue
Connected by ('104.132.0.101', 49945)
data: 1515546751085
data len:13
press C to continue
Connected by ('104.132.0.101', 34493)
data:
data len:251
press C to continue Connected by ('104.132.0.101', 62715)
data:
data len:223
press C to continue Connected by ('104.132.0.101', 49801)
data:
data len:565
press C to continue

Bug

- One remote code execution vulnerability in Telegram

Questions

natalie@natashenka.ca

@natashenka