KnockKnock Documentation

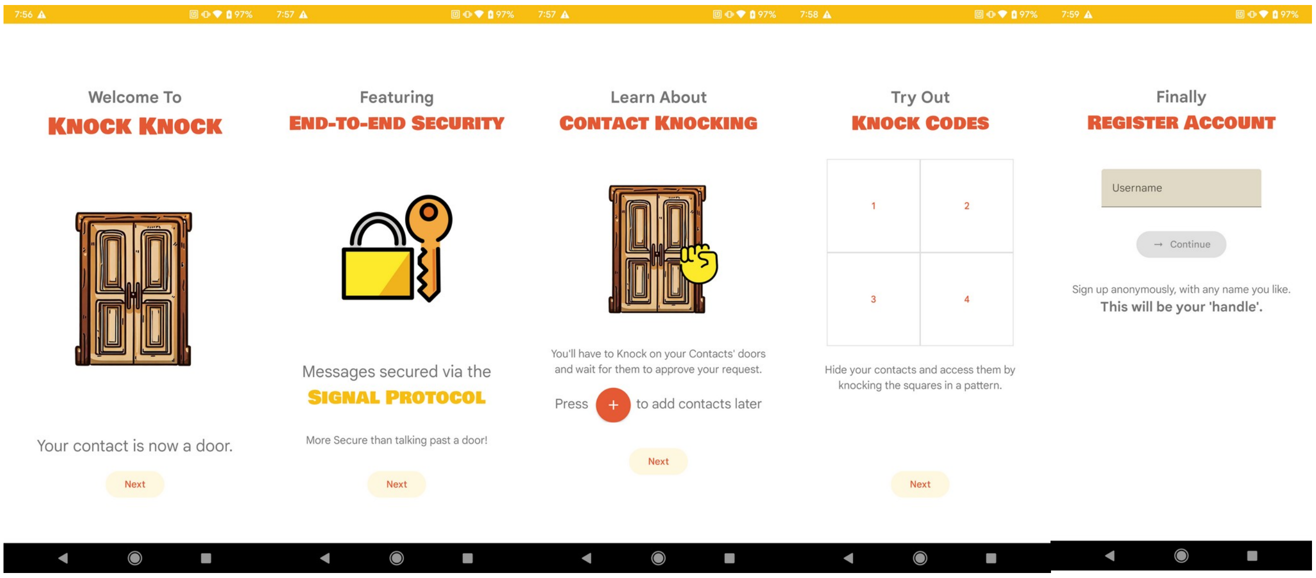
By Wong Yue Heng, M23404

Overview

KnockKnock is an Android text-messaging application, focused on privacy and security. It features the Signal Protocol, an open-source industry level end-to-end message encryption standard; message database encryption; and hiding contacts via a ‘Knock Code’. This level of security is much higher than any commericial messaging app, including popular messenger WhatsApp and privacy-focused Signal (from which the Signal Protocol was created).

As such, the target audience will also be very privacy-focused, such as celebrities, government officials, or even at-risk individuals. In addition to benifiting from end-to-end message encryption, anonymous sign-ups allow individuals who need to hide their identities (wanted individuals etc.) to contact the outside world, while the two-way messaging agreement, in which both parties that message one another have to mutually agree to chat via exchanging of contact handles, means that those who have very public identities (celebrities etc.) who do not wish to have many people contacting them will also benefit from having less clutter in the form of spam/unwanted messages or ‘message requests’ as present in other applications. The contact hiding is also useful for individuals who wish to hide the people they contact or the contents shared between them.

Documentation


Welcome To
KNOCK KNOCK
Your contact is now a door.
Next

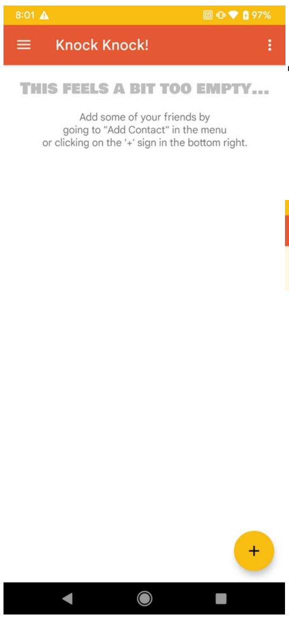
Featuring
END-TO-END SECURITY
Messages secured via the
SIGNAL PROTOCOL
More Secure than talking past a door!
Next

Learn About
CONTACT KNOCKING
You'll have to Knock on your Contacts' doors and wait for them to approve your request.
Press + to add contacts later
Next

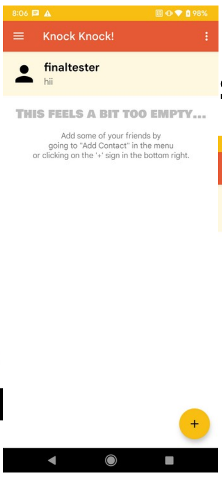
Try Out
KNOCK CODES
1 2 3 4
Hide your contacts and access them by knocking the squares in a pattern.
Next

Finally
REGISTER ACCOUNT
Username
→ Continue
Sign up anonymously, with any name you like.
This will be your 'handle'.

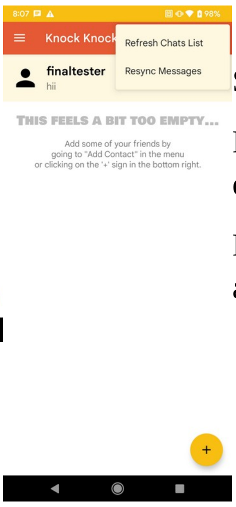
Onboarding Screens. Press ‘Next’ for all of them until you get to the last screen, where you can set your custom contact name (alphanumeric + ‘ ‘ + ‘_’, max 32 chars). Contact name must be unique.


8:01
97%
Knock Knock!
THIS FEELS A BIT TOO EMPTY...
Add some of your friends by going to "Add Contact" in the menu or clicking on the '+' sign in the bottom right.
+

The main screen, also known as the Chats List. Empty State.


8:06
98%
Knock Knock!
finaltester
hii
THIS FEELS A BIT TOO EMPTY...
Add some of your friends by going to "Add Contact" in the menu or clicking on the '+' sign in the bottom right.
+

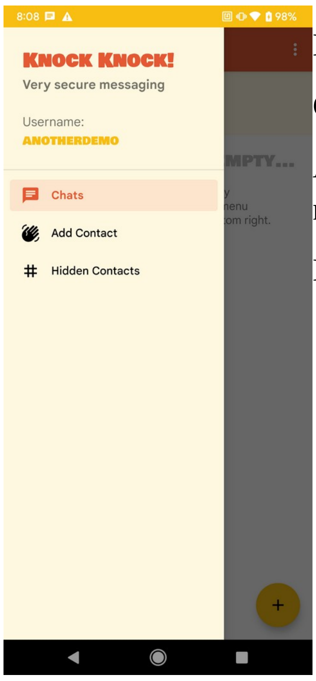
Screen with contact


8:07
98%
Knock Knock
Refresh Chats List
Resync Messages
finaltester
hii
THIS FEELS A BIT TOO EMPTY...
Add some of your friends by going to "Add Contact" in the menu or clicking on the '+' sign in the bottom right.
+

Screen with Overflow Menu

Refresh Chats List will refresh Chats List, e.g. if someone just added your contact

Resync Messages will sync your messages and restart the Message Sync Worker

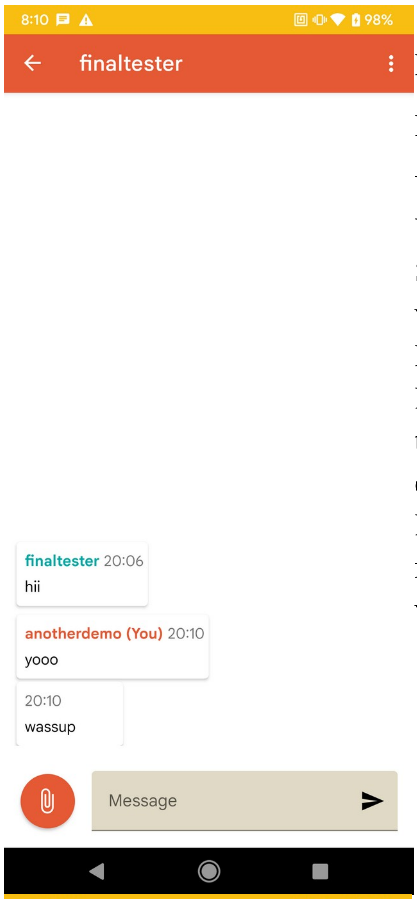


Drawer Layout, allows you to transit between the different screens:

Chats – The Chats List (shown above), access what contacts you ahve

Add Contact – Same as '+' FAB in Chats List, allows you to add/request new contacts

Hidden Contacts – Access Hidden Contacts, if any


8:10 98%

finaltester

finaltester 20:06
hii

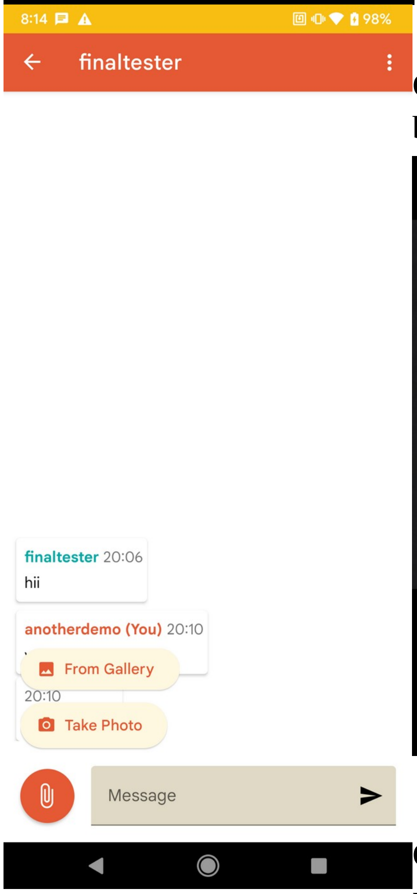
anotherdemo (You) 20:10
yooo

20:10
wassup

Messaging Screen. You can message your contacts here.

Explaination of code:

Upon pressing send, the contents of the EditText are serialized into UTF-8 Bytes. This is then encrypted into a SignalMessage/PreKeySignalMessage by the Signal libraries, which are encrypted. This encrypted message is then serialized via Base64 and is posted to the server via the Retrofit library, in a POST request. On the other side, the Message Sync Worker gets the new message, deserializes the message (how it does this is very deep as it goes in to the initial PreKey exchange and Key Ratcheting (deriving keys from previous keys) and I would much rather not explain it), and feeds it into the RecyclerView which will show the message in a CardView.


8:14 98%

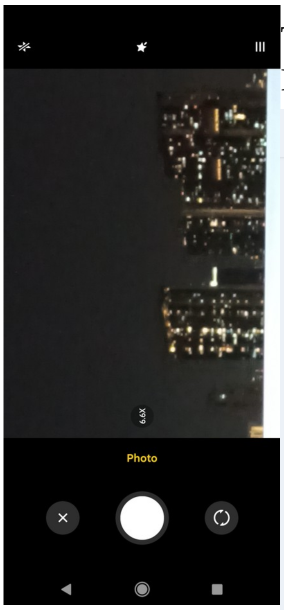
finaltester

finaltester 20:06
hii

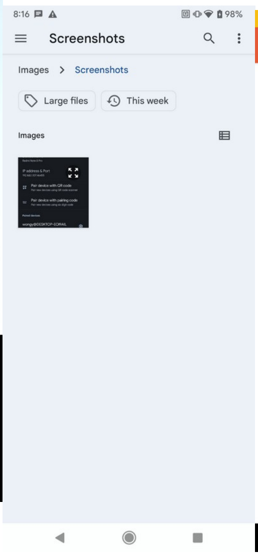
anotherdemo (You) 20:10

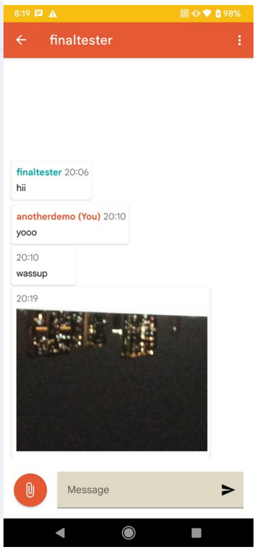
20:10

Observe the bottom left hand corner. Upon tapping the attach button, two new options appear.



Take Photo option leads to this screen, while From Gallery leads to the one below:




8:19 98%

finaltester

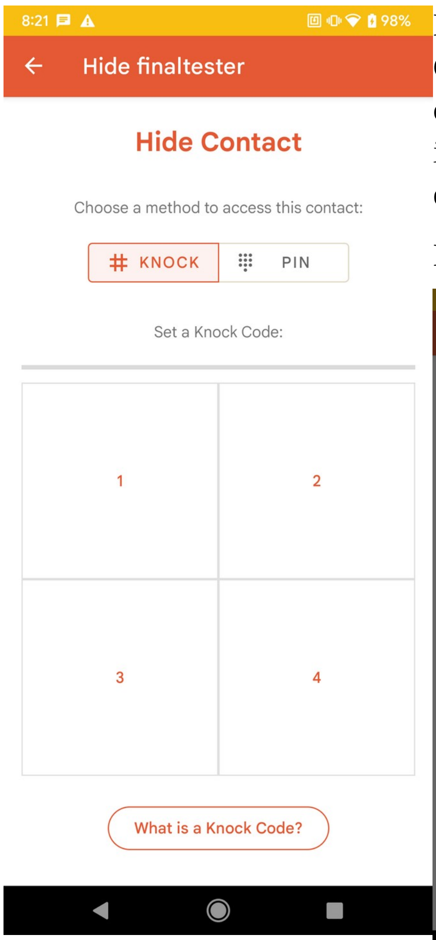
finaltester 20:06
hii

anotherdemo (You) 20:10
yooo

20:10
wassup

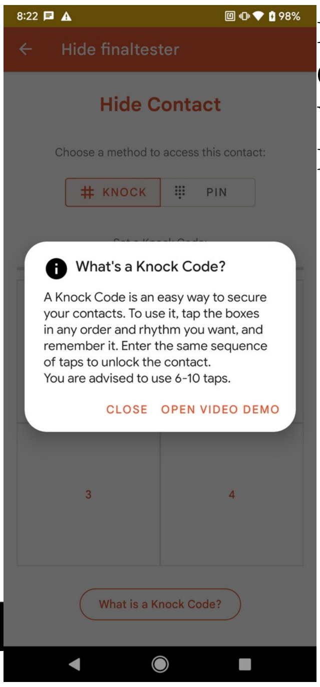
20:19

Once image is selected, it is serialized and sent just like a text message. It is shown as in the right-most example above.



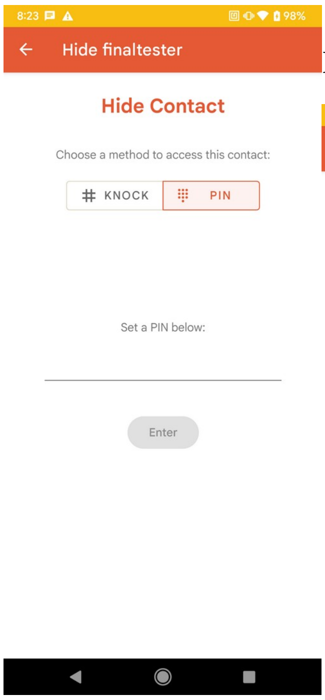
Hide contact screen. You can hide your contact by using a Knock Code, or a PIN. The Knock Code works by storing the magnitude of delay between the inputs on each of the four quadrants of the input area. This allows the code to be rhythm based, e.g. if you enter the same code slowly or quickly it will still work.

Each delay must be less than 1 second long.

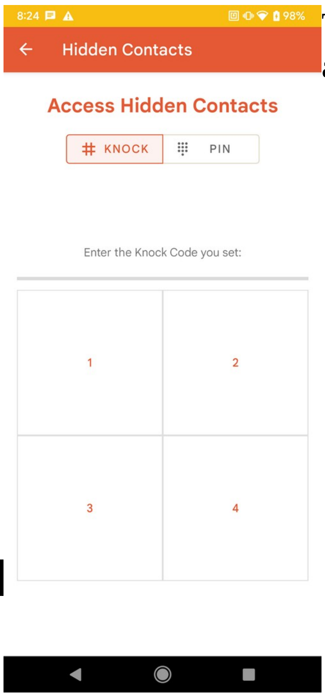


A Knock Code is an easy way to secure your contacts. To use it, tap the boxes in any order and rhythm you want, and remember it. Enter the same sequence of taps to unlock the contact.
You are advised to use 6-10 taps.

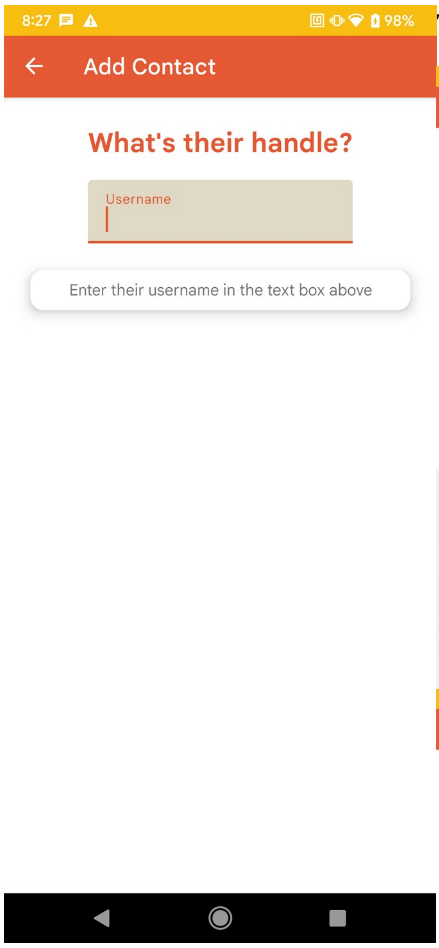
Dialog pop-up, detailing what a Knock Code is. A Video Demo is available on YouTube upon pressing the ‘Open Video Demo’ button.



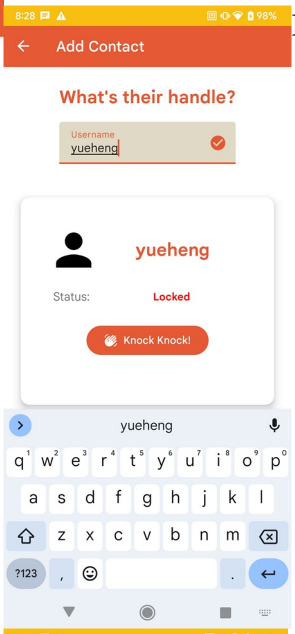
Hide using PIN, in case one is uncomfortable with Knock Codes



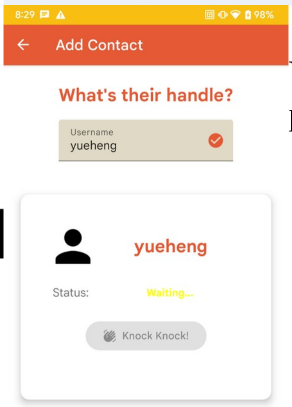
The screen to unlock your hidden contacts, accessible via the app drawer.



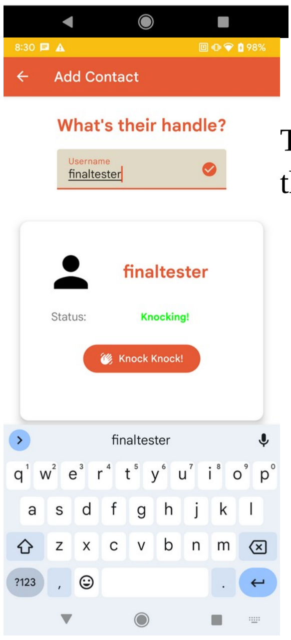
The add contact screen. Input another person's handle to add them.



Neither person has requested each other



You have requested the person, but the other person has not requested you



The other person has requested you, clicking the Knock Knock button starts a new chat.

Usability Study/User Feedback

A usability study was conducted on people of varying backgrounds to assess the usability of the application. For those I met face-to-face, I provided them with an Android phone with a fresh install of the application, while for those I asked online, I sent them an APK of the app to install and run.

Testers were tasked with:

1) Creating an account

2) Adding me as a contact

3) Sending a selfie of themselves

4) Hiding the contact

These four tasks were chosen as they represent the majority of the functionality in the app, encompassing message encryption, multimedia file transfer and contact hiding.

Feedback was recorded via phsyical/verbal cues in real life (e.g. general confusion, pauses) while online the only source on feedback is their questions to me when they are setting up the app. An additional set of questions were asked at the end of every testing session, which included questions like what they liked/disliked, and what they features they wished were in the app.

This study was conducted on 5 individuals.

1) Hiuk Ming Hong, a CS Student with deep and insightful knowledge on cryptography,

2) Tran Duc Nam, a CS Student with not much insight into cryptography and encryption,

3) Kristy Wong, a teenager with no knowledge of CS,

4) Will Wong, a 50 year old man with slight knowledge of technology, and

5) Jeanie So, my mom who has no idea or clue on anything relating to technology

Generally, all users found the set-up of the application easy as all they needed to do was enter their username to create an account. Some were also entertained by the animations in the onboarding screens. However, all of them were initially confused on what to do at the 'Add Contact' screen and did not understand that I had to approve their request before they could send messages to me. However, after they got past that, they knew how to message and send pictures with ease, owing to familiar iconography from other text messengers and clear text labels. However, problems arise once again in the 'Hide Contact' screen, as despite them managing to find it, many were still confused on Knock Codes, and some even mistakenly used them, leading to them being locked out of the contact forever.

Overall, feedback was that the animations and the user interface of the application was well designed. However, the performance of the app left much to be desired, with long loading times especially in places which require heavy networking such as sending images or adding new contacts. Users also commented on sent messages not immediately appearing, however, I justify this as the messages only show after being successfully sent to the other device, hence ensuring both sides are synced. Users also requested for the addition of group chats, and increased customizability including adding of profile pictures.

Reflections

I learned a great deal from this project. Firstly, I learned the basics of cryptography and encryption, including private/public keys, curves, signing, as well as the essential skills to go with it such as serialization and charsets. I also learned a lot about networking, such as GET/POST requests, packets and ports, HTTP vs HTTPS, as well as the existence of ngrok which would have saved me hours of trouble setting up port forwarding. I also learned how to make a server in Java and Spark, instead of my usual Python and Flask.

The major difficulties I had were with the Signal libraries, which are poorly documented, with many sources online being outdated due to it being used very rarely. I struggled a lot until I learned about Android Studio/IntelliJ's Jump To Source button which allowed me to look into how the functions were implemented and in turn create the application around that. Another one was networking, especially Android's numerous security protocols and obscure exceptions that make me tempted to surround all my code with a 'catch (e: Exception)' block as there were many different exceptions that could pop up at different times.

If I had more time, I would definitely focus a lot more on user experience, such as sending more types of media like audio/video, making voice calls, as well as creating group chats.