ETH ZURICH & SUPERCOMPUTING SYSTEMS

SEMESTER THESIS

Key Signing App for Proof of Personhood

Author:
Noa MELCHIOR

Supervisor:
Prof. Dr. Srdjan Capkun, ETH
Mridula Singh, ETH
Alain Brenzikofer, SCS

System Security Group
Department of Computer Science

June 23, 2019

Declaration of Authorship

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

Authored by (in block letters):
For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**

With my signature I confirm that
- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date	**Signature(s)**

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.

ETH ZURICH & SUPERCOMPUTING SYSTEMS

Abstract

Department of Computer Science

Key Signing App for Proof of Personhood

by Noa MELCHIOR

This semester thesis examines security aspects of the key signing ceremony of the novel Encointer [3] cryptocurrency and implements a prototype key signing app based on the findings of the security analysis. The prototype Android application uses Android Nearby [4] for peer-to-peer communication and the Encointer blockchain to determine input correctness of meetup attendees. This is the first work investigating the security of Encointer meetups and discoveries are integrated in the future development of Encointer.

Acknowledgements

I would like to thank my supervisors Professor Capkun, Mridula Singh and Alain Brenzikofer for their continuous support during my semester thesis making this endeavor possible. A special thanks to Marc Röschlin whose expertise helped form some of the key aspects in the security analysis.

Contents

List of Figures

List of Tables

List of Abbreviations

PKI	**P**ublic **L**ey **I**nfrastructure
PKL	**P**ublic **K**ey **L**ist
PoP	**P**roof-**o**f-**P**ersonhood
RTT	**R**ound-**T**rip **T**ime
RUS	**R**andomized **U**ser **S**election
TEE	**T**rusted **E**xecution **E**nvironment
UIN	**U**ser **IN**put

1 Introduction

This work extends on the idea of the Encointer white paper [3] which proposes a new blockchain based cryptocurrency with an ecological consensus mechanism and an egalitarian money supply policy. It introduces dPoET, a permissionless version of proof-of-elapsed-time (PoET, [6]) relying on trusted execution environments (TEEs). Monetary supply is not capped like in other cryptocurrencies and not directly linked with block mining as for example in Bitcoin [8]. While incentive for validators is created by a proportional transaction fee, money issuance is used as an incentive for proof-of-personhood (PoP) where users of Encointer attend monthly meetups in real life, attesting each others identity and in return receive Encointer currency and an attested digital identity.

These meetups allow users to generate a small income through the attestation of the identity of other users which helps develop communities, create a self-regulated currency and electronic identities. However, this also incentives attackers to attend more than one meetup to earn multiple rewards which would be against the idea of Encointer and eventually inflate the currency to the point where it becomes worthless. Thus, meetups need to be secured against a range of attacks where malicious users try to cheat the system, see chapter 4.

The goal of this semester thesis is the development of a prototype application to be used in money issuance meetups. The application implements all necessary steps for a user to take part in the money issuance process in a simple Android based smartphone application. The application also contains security measurements preventing attack vectors against Encointer meetups while trying to keep the impact on usability as small as possible. The security requirements for the application follow the description in the Encointer white paper and the security analysis discusses security relevant meetup scenarios which emerged during the process of this thesis.

The rest of this report is organized as follows. In chapter 2, goals and assumptions are listed, relevant aspects of the bigger Encointer ecosystem are explained and three different communication technologies are evaluated. It is explained why Android Nearby [4] builds the main communication layer for the application. Afterwards, chapter 3 introduces the system architecture where our approach and the meetup phases are described. Further, chapter 4 discusses the threat model and chapter 5 describes the main findings of the security analysis. Lastly, chapter 6 explains the prototype implementation while chapter 7 concludes this thesis.

2 Background

2.1 Application Goals

2.1.1 Security

The application should provide security mechanisms to prevent attacks emerging from the findings of the security analysis. If not all attacks can be mitigated by the application it should be noted where the application can be attacked and further mitigation can be enforced in the next phase of money issuance, e.g. by denying currency on the Blockchain.

2.1.2 Usability

The goal of this application is to provide basic security without compromising too much usability through lengthy verification processes such as pairwise QR scanning for identity checking. Usability also includes low false positives during the meetup to keep the number of restarts as low as possible. While connection issues cannot be eliminated completely, the application uses Android Nearby [4] which proved to be most stable compared to WiFi P2P [5] and UDP broadcasting, see section 2.5. The downside of using Android Nearby is that it depends on Google Play Services. This is not so much a usability issue for standard users but more advanced users might explicitly choose not to include Google Play Services on their device.

2.1.3 Compatibility

Encointer could especially help developing countries where low-end and second-hand smartphone markets are rapidly growing. Hence, the app uses Android Nearby which is available on all Android phones with API level 16, i.e. Android version 4.1 *Jelly Bean*, and working Google Play Services. Android Nearby uses Bluetooth and Wi-Fi which are available in the majority of all modern smartphones and does not rely on more cutting edge technologies such as near-field communication or Wi-Fi RTT (section 2.5).

2.2 Encointer

The Encointer white paper proposes many new aspects in its novel cryptocurrency. The main contributions are directly cited from the conclusion of the white paper:

"i) A novel global egalitarian approach to monetary policy allowing for a universal basic income (UBI). ii) A new definition of trustless pseudonym key signing parties for proof-of-personhood (PoP), incentiviced by *encointer* tokens. iii) A novel unpermissioned consensus algorithm dPoET, combining PoET with PoP to achieve decentralication of power by ecological means. iv) Private token transfers with microtransaction-friendly fees and low storage footprint. v) Scalable trustless private off-chain smart contracts." ([3])

This report will mainly focus on ii) which describes key signing ceremonies in more detail. The Encointer blockchain will be referenced in this report without technical details but rather for the implications to the ceremony process discussed in the security analysis of this thesis.

2.3 Proof-of-Personhood

The Encointer white paper defines proof-of-personhood (PoP) as follows:

"These Encointer meetups are at the same time the basis of a self-sovereign identity claim called proof-of-personhood (PoP) [2], proving a one-to-one relationship between a person and her digital identity. One person can only maintain one individuality claim because ceremonies are designed to make it impossible to attend two meetups physically as they happen in different places concurrently." ([3])

The definition considers PoPs as bijective by requiring that it is impossible to attend multiple meetups at the same time. Analyzing our approach against attacks breaking this requirement will be the focus of the security analysis in chapter 5.

2.4 Assumptions

While the security analysis forms a substantial part of the semester thesis, the scope is limited by time and resources. The analysis and the threat model try to capture all aspects of a meetup and the overlapping surrounding Blockchain ecosystem.

The Encointer white paper already defines many aspects of the ceremony. The most important ones for this thesis are:

- Ceremony participants need to pre-register with a one-time public key and are randomly assigned to one of many meetups in groups of 3-12 people
- Participants know the public key of all co-participants at the same meetup
- All meetups happen at precisely the same time within a certain geographical range

The security analysis assumes that strictly more than half of all attendees are honest users with no malicious intent. This assumption is heavily used later for mitigation of attacks and justified by the random selection process of users for meetups.

The random selection protects meetups from an attacker that tries to register for a meetup with multiple identities to reach the threshold of 50% which would mean the attacker controls the outcome of the meetup, see section 5.4.

The application is going to be open-sourced. Thus, everyone has access to the application code excluding underlying proprietary closed-source software, i.e. Android Nearby. It is important to note that while the source-code of our implementation, highlighted in chapter 6, is open-sourced, a user who runs a modified version of the code is immediately considered an attacker and so are curious users.

The threat model assumes an attacker who owns a rooted smartphone. Hence, an attacker is assumed to be able to virtually represent multiple identities in a meetup and create any sort of messages during the meetup.

2.5 Wireless Communication Layer

Before the system architecture is discussed in detail in chapter 3, this chapter evaluates different wireless communication technologies and whether they are suited for the application. The first approach used distance measurements as the main security feature. This had to be discarded as neither Wi-Fi Round-Trip Time (RTT) [9] nor UDP broadcast fulfilled all necessary criteria to be used in the application. Finally, Android Nearby [4] was chosen which is described in chapter 3. There are three criteria which were evaluated:

- **Compatibility:** It is important that the application runs on as many smartphones as possible. Hence, it should also run on older Android versions and not use cutting edge technology which exclude large parts of the community.
- **Necessary Infrastructure:** Meetups should required as little preparation as possible and additional infrastructure such as routers, internet connection or similar should be avoided by all means. Peer-to-peer communication is preferred to master-slave setups, e.g. hotspot.
- **Security:** Security features provided by the communication layer are not required but a bonus since the Encointer blockchain acts as a public key infrastructure (PKI).

2.5.1 Android Nearby

Android Nearby [4] combines Bluetooth and Wi-Fi to establish a communication channel between nearby Android devices. The prototype is implemented with Android Nearby as it is available from Android version 4.1 *Jelly Bean* and supports peer-to-peer communication by default. Little is known about the provided security guarantees as the implementation of the API is proprietary, closed-source and obfuscated. Recent work discovered several attacks based on reverse-engineering of the API [1]. The attacks concern connection manipulation and range extension attacks. However, the attacks are later discussed and pose no direct threat to the system architecture, see section 5.6.

2.5.2 Wi-Fi P2P

Wi-Fi P2P [5] is also know as Wi-Fi Direct and can be interpreted as the predecessor of Wi-Fi Nearby. It only uses Wi-Fi to connect and tests revealed fairly unstable and inconsistent connections. Although used for direct peer-to-peer communication, it is most often used in one-to-one scenarios making a meetup with up to 12 users slow as connections have to be established one by one.

2.5.3 Wi-Fi RTT

The publication of the task group *mc* of the IEEE 802.11 working group introduced fine timing measurements which enable distance measurements between devices and access points with a precision of 1-2 metres [7] using round-trip times of packets sent between two devices. Fine timing measurements were introduced to Android 9 *Pie* called Wi-Fi RTT. Usually, this is done between an access point and a user device but it can also be used between two devices which have Wi-Fi RTT enabled. While Wi-Fi RTT is not a communication standard in itself, distance measurements could be used as a security feature. Unfortunately, very few devices currently support Wi-Fi RTT and no security guarantees can be given since implementation depends on hardware manufacturers and is mostly closed-source.

2.5.4 UDP Broadcast

Finally, UDP broadcast was tested as an alternative method for measuring rount-trip time of packets for distance measurements. Unfortunately, this either requires additional infrastructure, i.e. a Wi-Fi access point, to broadcast the UDP packets, a master device which all other devices connect to or rooted devices which can directly access the wifi chipset and extract packets directly without being connected to a message broker. Additionally, the latency increased to around 1-10ms which already drastically decreases precision.

3 System Architecture

The application closely follows the requirements of the Encointer white paper [3] and this chapter reasons about the exact specifications of our approach where the description of the white paper allows for different solutions. The white paper defines key signing ceremonies as reoccurring events in 41-day time intervals where a user has to sign up at least 24h hours prior with a one-time public key and various parameters necessary for meetup creation. A ceremony takes exactly 24h hours as each meetup will start exactly at high sun in the defined location.

Once a user attends more than half of all ceremonies with the same identity a proof-of-personhood claim becomes valid for this identity. This requires that it is not possible for a user to attend meetups during a single ceremony with more than one identity. This requirement relies partly on the application developed in this thesis.

3.1 Our Approach

The approach presented in this thesis uses the user input and the collected signatures of the other users to detect fraudulent users when they try to claim their reward after the meetup on the blockchain. See the security analysis in chapter 5 for the discussion of which attacks can be detected and/or prevented with our approach.

3.1.1 Randomized User Selection (RUS)

The registered users are randomly selected from the pool of identities who signed up for the meetup, as shown in Figure 3.1. This lowers the possibility of an attack where a malicious user registers for a meetup with multiple identities to manipulate PIC.

3.1.2 User Input (UIN)

At the beginning of the meetup, each user enters the number of attendees. Assuming that more than 50% of the attendees are honest, the user input can be used to derive a trusted number of attendees. As before, the user input alone cannot be trusted by default as the user can provide any input from 3 to 12 without the device being able to verify correctness on its own. The input with the assumption that more than 50% of attendees give honest input can be used to find out the correct number of attendees.

3.1.3 Public Key List (PKL)

Every registree has a unique asymmetric key pair which enables cryptographic primitives at the meetup. The public key list is downloaded by the application from the blockchain. Therefore, this information can be trusted as this is computed on-chain or in a TEE. The list contains all public keys allowed to participate in the meetup enabling identified and encrypted communication in the peer-to-peer network.

3.1.4 Device Scan

While the application is running, it continuously scans for other devices advertising the same identification string. The string is not confidential hence connected devices cannot be considered honest nor true attendees. A scan can reveal less or more devices with one or more identities than there are physically attending users due to connection issues or users bringing more than one device.

3.1.5 Signature

A signature contains the number of attendees signed by the user's private key. Additionally, the signature also contains a timestamp and it can be arbitrarily extended to fit the needs of the meetup (e.g. GPS location). The signature can be assigned to one of the one-time public key in the PKL since it is signed with the corresponding private key. The application will collect as many signatures as the number of attendees that has been entered before saving them collectively such that they can be sent to the blockchain. On the other hand, the application creates a signature for each user as soon as they entered their number of attendees which can then be sent to the other attendees once they are connected via Android Nearby.

3.2 Meetup

A meetup can take place if two conditions are met: i) at least 3 registered users are within each others accepted travel radius and ii) at most 2/3 of all selected users have met at the last meetup. A meetup consists of at most 12 registered users. Otherwise, the meetup is going to be split into two meetups. The meetup is guided by an application on the users' smartphones to attest each others identity, proof physical attendance and prevent attendance of multiple meetups in the same time window. Users are only considered attendees if and only if they are physically attending the meetup as Encointer intended.

3.2.1 Before a Meetup

A user may register for a meetup with a newly generated one-time public key for privacy reasons. The registration also includes an approximate location and a maximum travel radius to calculate a place reachable by all randomly selected users through triangulation. The group for the meetup is randomly selected from the pool of registered users 24h before the meetup. This list contains the public key of each

selected user, is publicly available on the blockchain and used during meetup for validation purposes. This stage of the meetup is illustrated in Figure 3.1.

3.2.2 During a Meetup

The process is initiated by running the smartphone application which starts scanning for other devices. The user provides input for the application by specifying the number of attendees. The devices establish a peer-to-peer network with Android Nearby [4] which does not rely on additional infrastructure. This scenario offers three sources of information to the application: A list of devices from scans, input from the user and the list of public keys from the registration. Together, this information is used to provide security against malevolent users described later in the threat model. After validating the number of attendees, the meetup can be finalized.

3.2.3 After a Meetup

Once the user connects to the blockchain, the collected signatures can be written into a transaction and pushed to the distributed ledger.

3.2.4 Validation on the Blockchain

Once the window for submitting the collected signatures closed, the verification process starts. The verification algorithm is not part of this thesis but briefly summarized, there are two key aspects for successful validation: i) Calculate what number each user voted for and ii) count the majority vote. After the validation process, currency issuance can commence. If need be, actions against absent or fraudulent users are taken.

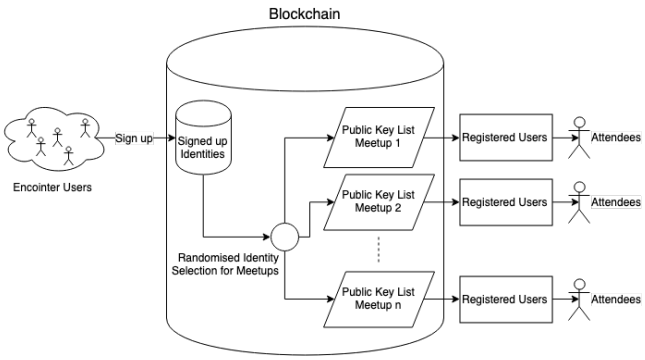

FIGURE 3.1: Users of Encointer who want to attend a meetup have to sign up and go through a random selection process beforehand.

4 Threat Model

4.1 Multiple Identities

As chapter 1 already mentioned, an attacker will need multiple identities to claim more rewards than it is intended from the Encointer white paper [3]. Maintaining multiple valid identities, i.e. attending more than 50% of all meetups with more than one identity, is going to break the bijective nature of PoPs. This can be used by an attacker to earn multiple rewards at meetups, as seen in Figure 4.1, or to gain any other sort of unfair profit which relies on identity attestation provided by Encointer.

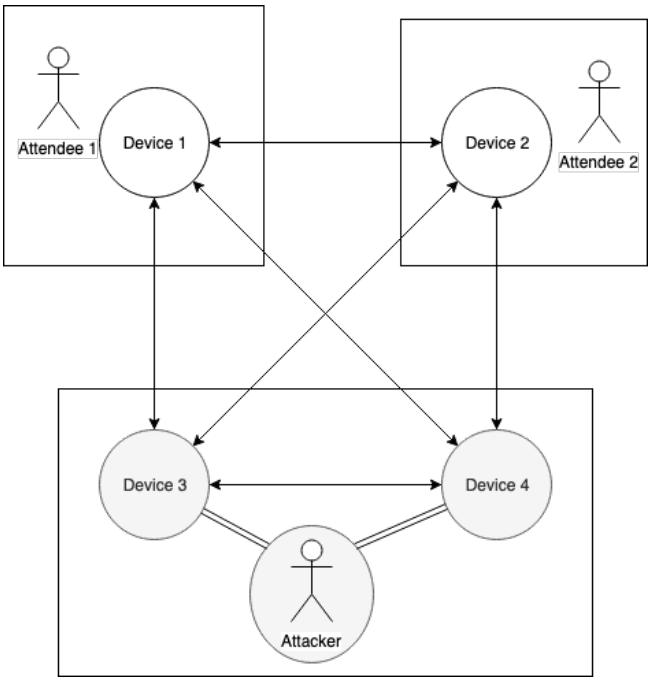

FIGURE 4.1: An attacker with multiple identities at a meetup.

4.2 Financial Gain

Collecting mutliple rewards from a single meetup or from multiple meetups during the same time window. While this does not necessarily include breaking proof-of-personhood, it is considered the main incentive for an attack.

4.3 Remote Attendance

Describes attending a meetup without being physically present, as pictured in Figure 4.2. Remote attendees weaken trust in Encointer as their identity cannot be verified by other participants. Finally, remote attendance might lead to the attacker attending multiple meetups in the same time window, therefore, receiving multiple rewards.

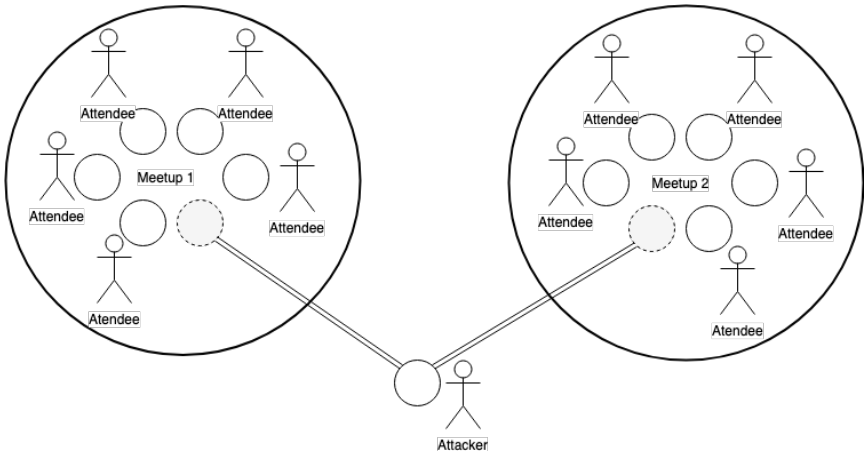

FIGURE 4.2: Attacker attends multiple meetups remotely.

5 Security Analysis

5.1 Proof of Input Correctness (PIC)

Assuming more than 50% honest physically present attendees and distinguishable signed messages, devices are able to share and collect the signed input of other devices. In fact, $\lceil \#attendees/2 \rceil$ unique messages with an equal input are enough to reach consensus about the number of persons attending the meeting if every attendee registered one identity. The public key list is used to filter unsigned or falsely signed messages while the user input provides the threshold that needs to be reached to complete the meetup. Device scanning is not used as a security feature but rather to establish connections. When the application has collected at least $\lceil \#attendees/2 \rceil - 1$ signed inputs from other devices that match its own input, it has proven the correctness of its input as the majority of attendees agree. PKL, UIN and RUS are necessary to perform a valid proof of input correctness, as shown in Figure 5.1. While it is technically possible to verify the proof of every user by their peers during the meetup, this is not done in our implementation. Rather, the proof is pushed to the blockchain where validation is easier. However, this leads to special cases where an attack can be detected but not pinpointed to the attacker's device. Thus, making the meetup invalid for all users, see the attack described in section 5.3.

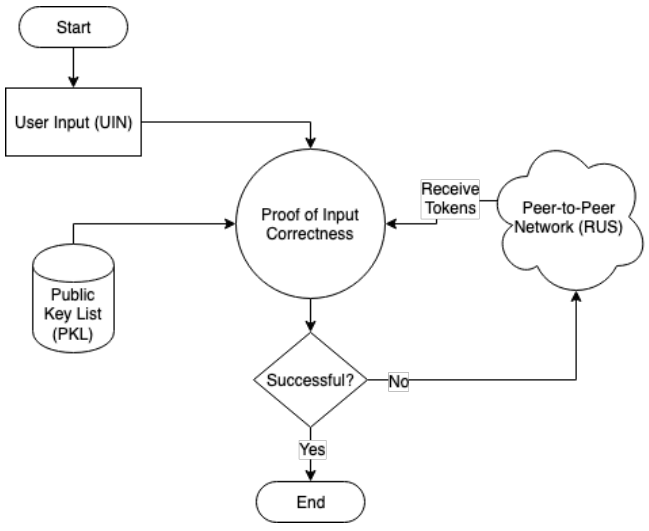

FIGURE 5.1: Calculating a proof of input correctness is based on RUS, UIN and PKL.

5.2 Scenarios

One has no control over how many identities a user registers or how many registrees are going to a meetup in the end. Therefore, there are nine scenarios in Table 5.1 that ought to be covered: i) If all registrees attend the meetup, no additional user input is needed. Each user can collect more than half the number of attendees' token to proof that the meeting was attended and earn the reward. ii) Not all registrees show up to the meetup, then there is additional input needed to verify how many registrees are eligible for the reward. iii) More attendees than registered users is not further discussed as there is no incentive for a user with no registration to attend a meetup. The public key list will prevent money issuance even if a not registered user is accepted by the rest of the meetup. The same argument can also be made for both other cases to exclude scenarios where real attendees are replaced by not registered users.

	A < R	A = R	A > R
Identities < Attendees	PIC	PKL & RUS	not discussed (PKL)
Identities = Attendees	PIC	PKL & RUS	not discussed (PKL)
Identities > Attendees	PIC	PKL & RUS	not discussed (PKL)

TABLE 5.1: A proof of input correctness is only required if there are less attendees (A) than registrees (R).

5.3 Remote Attacker

A remote attacker signs up and is selected for a meetup but does not physically attend the meetup. The prototype implementation currently has visual aids (identicon, identification token) to manually check which device belongs to whom. The worst case scenario would be if all attendees also send their signature to the remote attacker and collect the signature of the remote attacker. Figure 5.2 shows as an attacker in a meetup of six registrees where honest users also sent their signature to the attacker due to a lack of manual validation. Hence, all six registrees will publish their collected transactions on the blockchain. It is easy to verify that there has been an attack to the meetup because no matter what the input of the attacker was, the honest users will report maximum five attendees in their signature. However, the attacker can only be identified if the signature of the attacker reports another number of attendees than the majority of the attendees. If the input of the attacker is the same as the attendees' it is not possible to determine who the attacker was. Therefore, the whole meetup has to be invalided, i.e. no rewards are issued at all. While this is a hard punishment for honest users, it should prevent this attack because there is no incentive for an attacker other than damaging others (DoS attack) since the attack can be detected just not backtracked to the attacker.

5.4 Flood Attack

The sign up process for the next ceremony uses one-time asymmetric key pairs. It is therefore very simple to generate new accounts to sign up for the next ceremony. In section 2.4, it is assumed that RUS will prevent an attacker to be able to register for

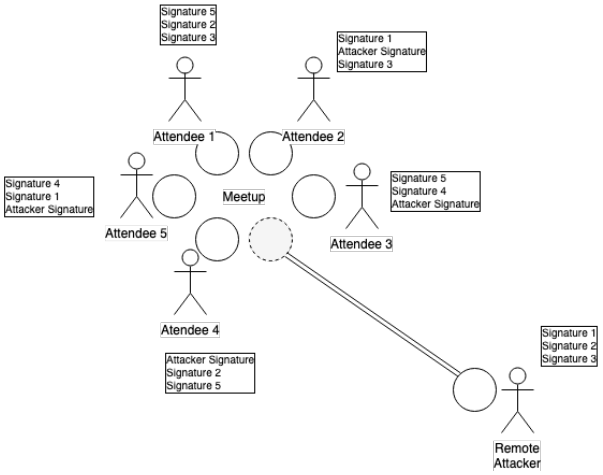

FIGURE 5.2: In the worst case, a remote attacker can invalidate a meetup.

the same meetup with multiple identities. This assumption does not hold for large areas with very few residents and even in densely populated areas an attacker could simply spawn hundreds of accounts raising the probability of being registered for the same meetup. Registering with two identities to one meetup will immediately be detected by the verification before money issuance. The attack becomes more sophisticated if the attacker manages to register enough identities to solely claim the majority vote of a meetup. In this scenario, the attacker is not dependent on other users but can even exclude them if they do not cooperate and still convince the verification algorithm.

An additional security mechanism which can be used in meetups that are endangered by this attack is the use of identity trust. A valid PoP claim can be made if a user attends at least half of all ceremonies with one identity. The more ceremonies are attended by a user with the identity the higher the trust. An attacker which tries to flood a meetup will create a lot of new identities with zero trust, i.e. their identity claim is not valid at that point. One could limit the number of identities with zero trust in a meetup to strictly less than half. Unfortunately, this would also slow down growth of Encointer in new areas where no meetups take place as new users would need already trusted users in their meetups.

5.5 3-for-2 Attack

While it is not possible for a single attacker to claim more than one valid PoP, it is possible for two attackers to claim three valid PoPs. As the threshold of a valid PoP claim is 50%, two attackers can rotate their identity in a round-robin fashion and attend 66% of all ceremonies with three identities. One can simply increase the threshold of claiming a valid PoP but this will just increase the number of attackers needed to cooperate until an additional identity can be claimed. It can be argued that this will not be profitable when too many attackers are involved but there is further research needed to quantify an exact threshold.

5.6 Android Nearby

Recent work by Daniele Antonioli et al. [1] has revealed that Android's Nearby Connections API enables two groups of attacks: i) Range extension attacks and ii) connection manipulation attacks. Although the communication layer is not the focus of this thesis, it should be noted that threats exist. In the future, one should consider other communication channels which are open-source without the need of additional proprietary closed-source software.

5.6.1 Connection Manipulation Attacks

In [1], connection manipulation attacks include impersonation, man-in-the-middle attacks on Bluetooth and on Wi-Fi, attacker-induced physical layer switch, injection of default route via attacker access-point, DoS on all victim traffic and radio state manipulation.

5.6.2 Range Extension Attacks

The paper also contains REArby, a toolkit developed while reverse engineering the Android Nearby Connection API allowing nearby connections with devices outside the intended bound.

6 Prototype Implementation

The application consists of three phases:

1. When the application is started, the user sees the first screen where a name can be picked which is later displayed for the other attendees to see.
2. The second screen asks the user how many people are attending the meeting including themselves. This is later used to calculate the number of signatures they have to collect to successfully complete the meetup.
3. The final screen shows available devices and the signature exchange can take place.

6.1 Start Screen

The start screen, Figure 6.1, is very simplistic and starts the meetup by asking the user for a name which is then displayed when advertising to other devices. The name generates an identicon which is a unique pixel image depending on the name for easier identification during the connection process, as seen in the right upper corner of each screenshot in Figure 6.3.

6.2 Enter Number of Attendees

The next screen, Figure 6.2, is crucial for the security of the meetup, as discussed in chapter 5. Assuming that strictly more than half of all attendees enter the correct number of attending people, malicious users can be detected and punished when they try to earn their reward on the Encointer blockchain.

6.3 Signature Exchange

The signature exchange screen is where users see from whom they received signatures and send their own signature. First, as soon as this screen starts the device starts advertising itself and discovering other devices. All found devices are listed as seen on the left of Figure 6.3. Once the user wants to connect to a device, tapping on it will initiate the connection and as soon as the connection is established the word *disconnected* is replaced by a cryptographic token which is shown in the middle screenshot of Figure 6.3. The token can be compared before sending the signature to prevent connecting to the wrong device. In the next phase, users can exchange signatures. This has to be done individually to each user in both directions as it is not clear before whether all users gave the right input, i.e. entered the

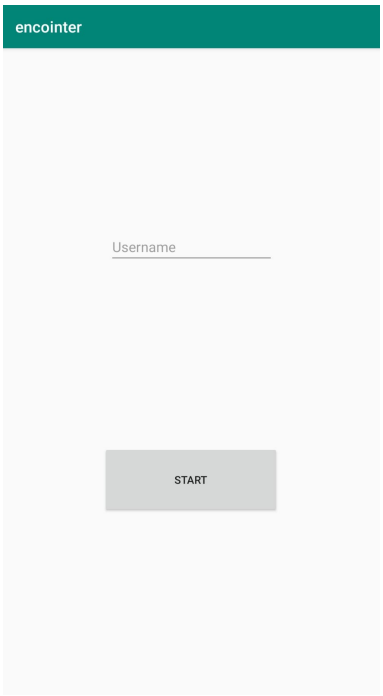

FIGURE 6.1: The application start screen where a username can be chosen.

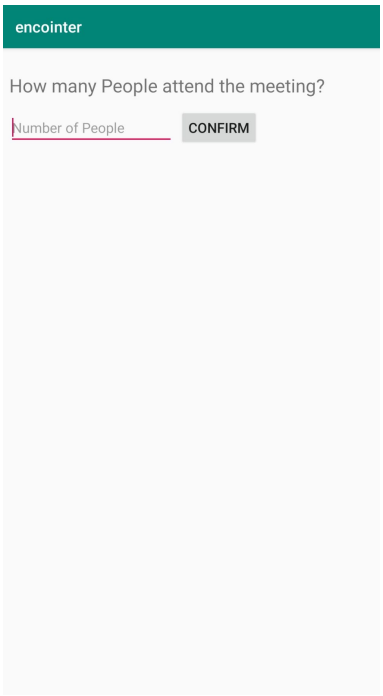

FIGURE 6.2: The second screen asks the user how many people attend the meeting.

correct number of attendees and therefore need all signatures. Finally, when the user received as many signatures as the entered number of attendees the application will save all signatures which then can be sent to the Encointer blockchain. The application asks whether the meetup should be finalized by closing the application or other participants have not yet received the users signature. A third option can be chosen if the entered number of attendees was false. Then the application will return to the second screen and repeat the process. It is in the responsibility of each user to reconnect with all other users whom they have sent a false signature and to send them the correct signature. If a user uploads the false signature of a user, the user who signed the false signature will not be able to claim a reward.

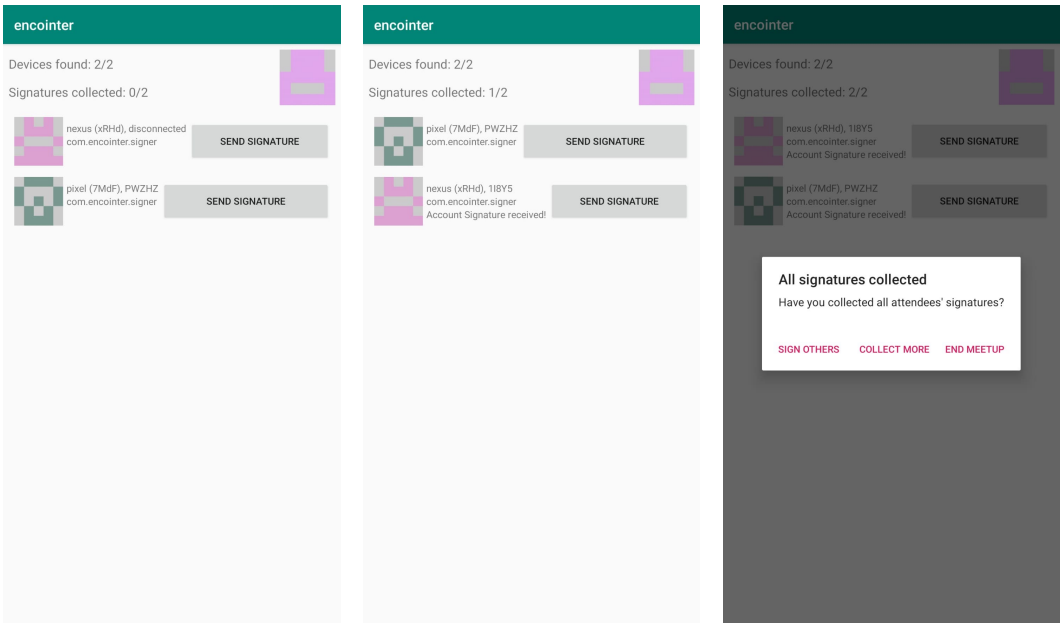

FIGURE 6.3: The screen to exchange signatures displays all discovered devices with their name, identicon, connection status and whether the signature was already received.

7 Conclusion

This thesis analyzed the security aspects of Encointer ceremonies and proposed and implemented a prototype application for Encointer meetups. The security goal was to prevent attackers from claiming multiple valid PoPs which was achieved within the assumptions and limitations described in chapter 2. Certain attacks were discovered when the assumptions are weakened and further research is needed to prevent such attacks in their entirety.

The underlying peer-to-peer communication was implemented with Android Nearby [4] which grants great overall usability in the Android ecosystem with the limitation that Google Play Services are necessary on participating devices. In future work, the communication layer could be reworked to use an open-source protocol which then could be used by users who choose not to enable Google Play Services. By leveraging the blockchain to verify the meetup, usability is not impacted by security features as the meetup only focuses on exchanging messages instead of a lengthy distributed verification process.

Finally, compatibility is guaranteed for Android devices running version 4.1 *Jelly Bean* or newer which includes over 99% of all Android devices. Thus, compatibility is also granted for developing countries where low-end and second-hand mobile devices have big market shares. As long as communication will use standard hardware such as Bluetooth and Wi-Fi chipset, the compatibility rate will remain high.

Bibliography

[1] Daniele Antonioli, Nils Ole Tippenhauer, and Kasper Bonne Rasmussen. "Nearby Threats: Reversing, Analyzing, and Attacking Google's 'Nearby Connections' on Android". In: *NDSS*. 2019.

[2] M. Borge et al. "Proof-of-Personhood: Redemocratizing Permissionless Cryptocurrencies". In: *2017 IEEE European Symposium on Security and Privacy Workshops (EuroS PW)*. 2017, pp. 23–26. DOI: 10.1109/EuroSPW.2017.46.

[3] Alain Brenzikofer. *Encointer*. https://github.com/encointer/whitepaper/raw/master/encointer_whitepaper.pdf. 2019.

[4] Google. *Android Nearby*. https://developers.google.com/android/reference/com/google/android/gms/nearby/package-summary. 2019.

[5] Google. *Wi-Fi peer-to-peer*. https://developer.android.com/guide/topics/connectivity/wifip2p. 2019.

[6] Hyperledger. *Proof-of-elapsed-Time 1.0 Specification*. https://sawtooth.hyperledger.org/docs/core/releases/1.0/architecture/poet.html. 2019.

[7] "IEEE Standard for Information technology—Telecommunications and information exchange between systems Local and metropolitan area networks—Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications". In: *IEEE Std 802.11-2016 (Revision of IEEE Std 802.11-2012)* (2016), pp. 1–3534. DOI: 10.1109/IEEESTD.2016.7786995.

[8] Satoshi Nakamoto. *Bitcoin: A peer-to-peer electronic cash system," http://bitcoin.org/bitcoin.pdf*.

[9] *Wi-Fi location: ranging with RTT*. https://developer.android.com/guide/topics/connectivity/wifi-rtt.