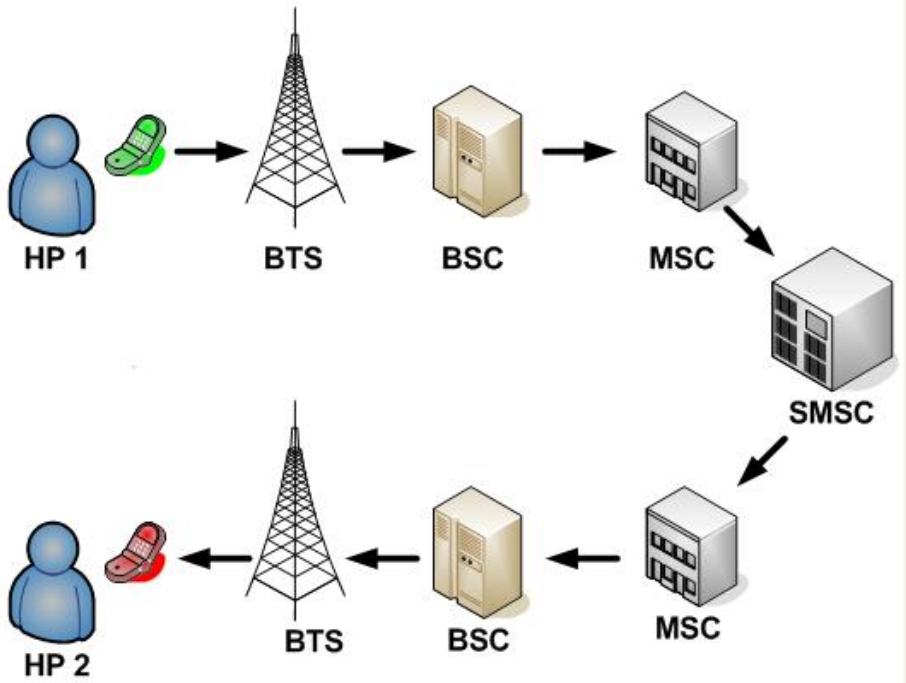
PGP Messaging

Elliptic Curve PGP Implementation for SMS

Final Presentation

Short Message Service (SMS)



HP: Mobile Devices

BTS: Base transceiver station

BSC: Base station controller

MSC: Mobile switching center

SMSC: Short message service center

Musbikhin. (2012, June). Short Message Service (SMS) [Online]. Available: http://www.musbikhin.com/short-message-service-sms

Existing SMS Encryption Applications for Android

- Symmetric
  - SMS Encryption Droid
  - TxT Encrypt

Drawbacks:

- Suffer from key distribution problem
- No non-repudiation

- Asymmetric
  - TextSecure
  - PGP SMS
  - ECC SMS

Drawbacks

- Multiple SMS messages to send single message
- No implementation of ring of trust
- No non-repudiation
- Require centralized server

PGP Messaging

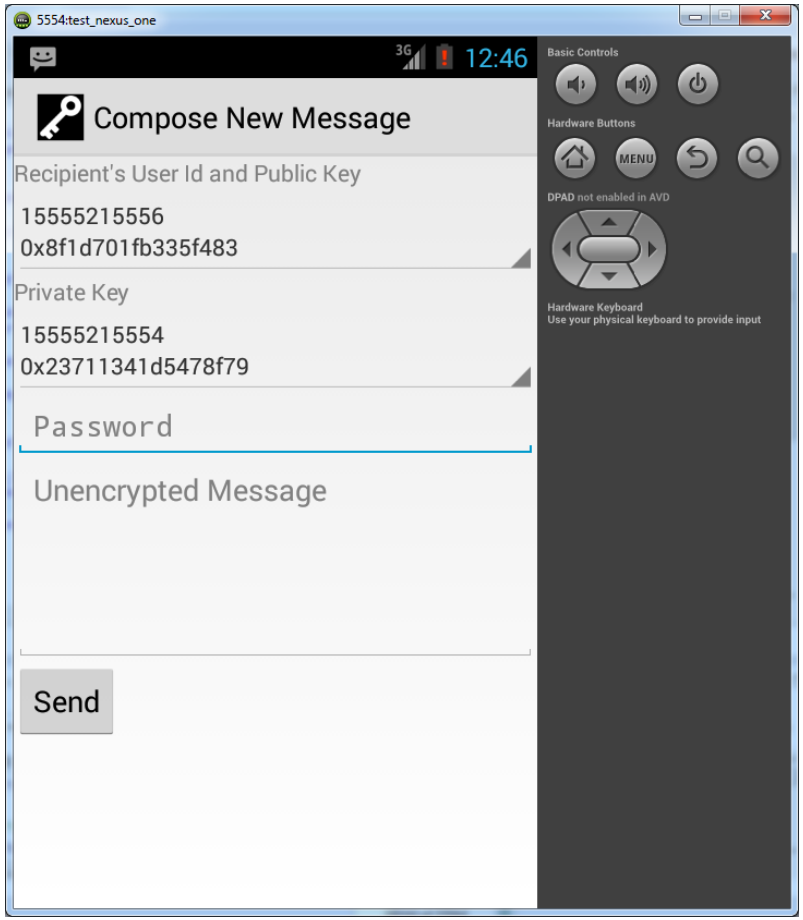
- Public key cryptosystem based on Elliptic Curve Cryptography
- Message
  - Key generated using private key and recipient public key in an elliptic curve key derivation function (KDF)
  - Resulting key used for AES-256 encryption
- Signature
  - Messages and Public Keys signed using SHA-256 and ECDSA
- Private keys - encrypted using AES-256
- Pretty Good Privacy Key Rings

Cryptographic Libraries

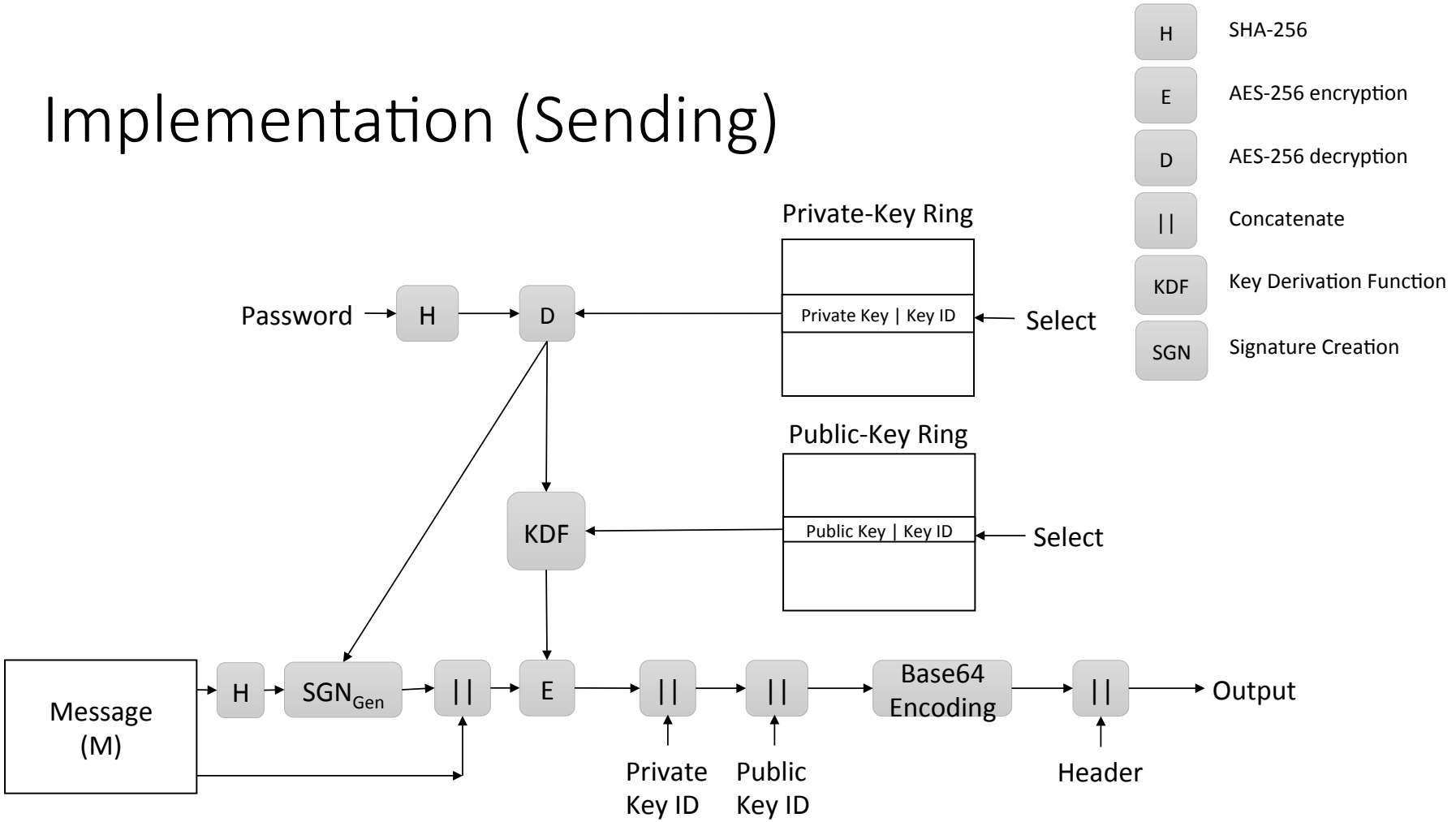
- Spongy Castle
  - Open source lightweight implementation of Bouncy Castle specifically for Android
  - Downloadable at http://rtyley.github.io/spongycastle/
  - Provides
    - AES Encryption/Decryption
    - Elliptic Curve Key Generation (prime192v1)
    - Key Derivation Function (ECDH)
    - Elliptic Curve Digital Signature Algorithm (SHA256withECDSA)
- Android SDK
  - SHA-256

Message Sender

- Choose from available public keys
  - Displays phone number and key ID
- Choose from available private keys
  - Displays phone number and key ID
- Password field for decrypting private key
- Entered plaintext message to be transmitted



Implementation (Sending)

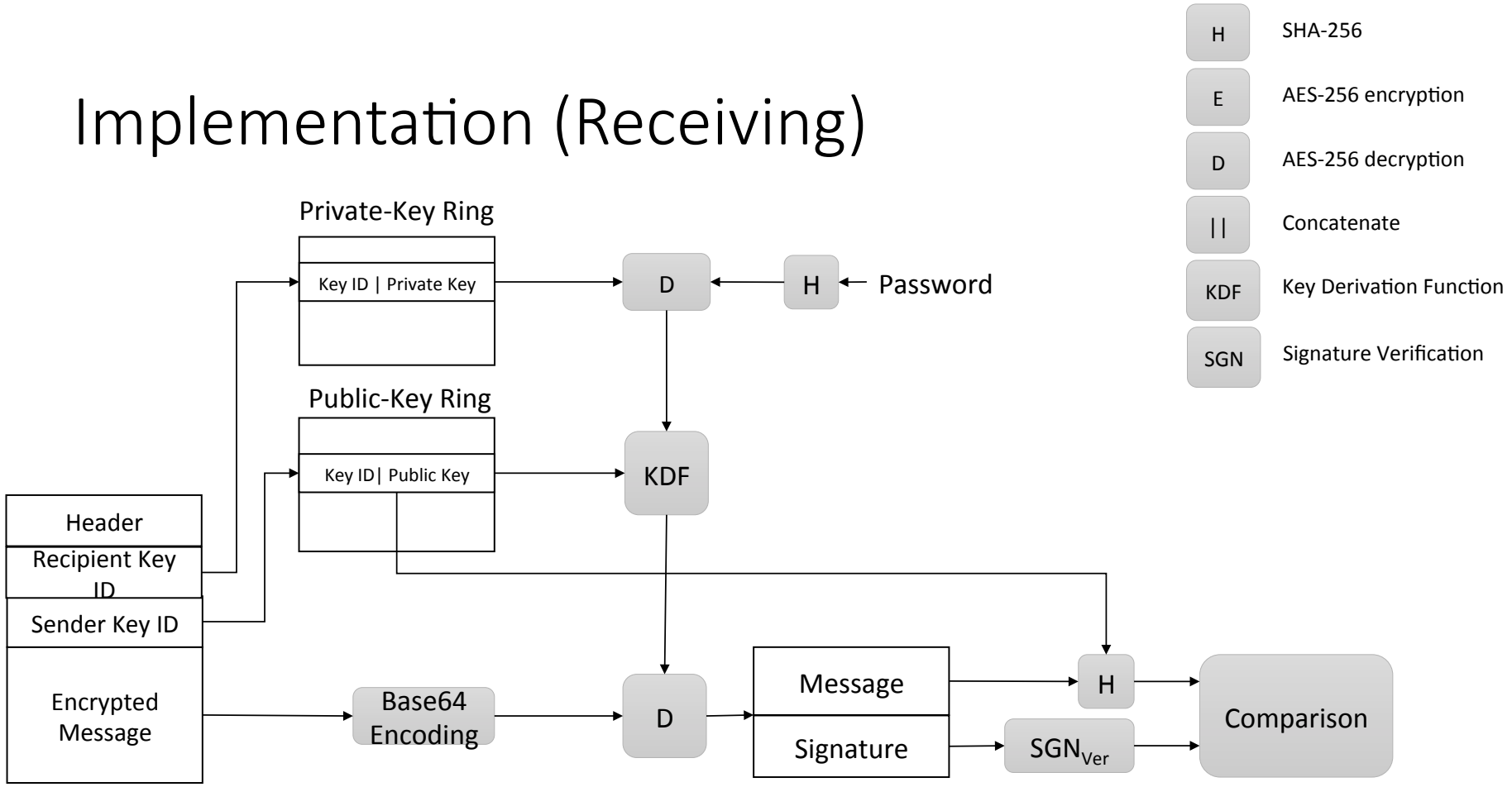


Message Format

- 160 total bytes available (SMS Standard)
- 5 characters for header
- 155 remaining becomes 113 characters (because of Base64 encoding)
- Header alerts application that following data is encrypted and contains signature
- Header and key IDs are not encrypted

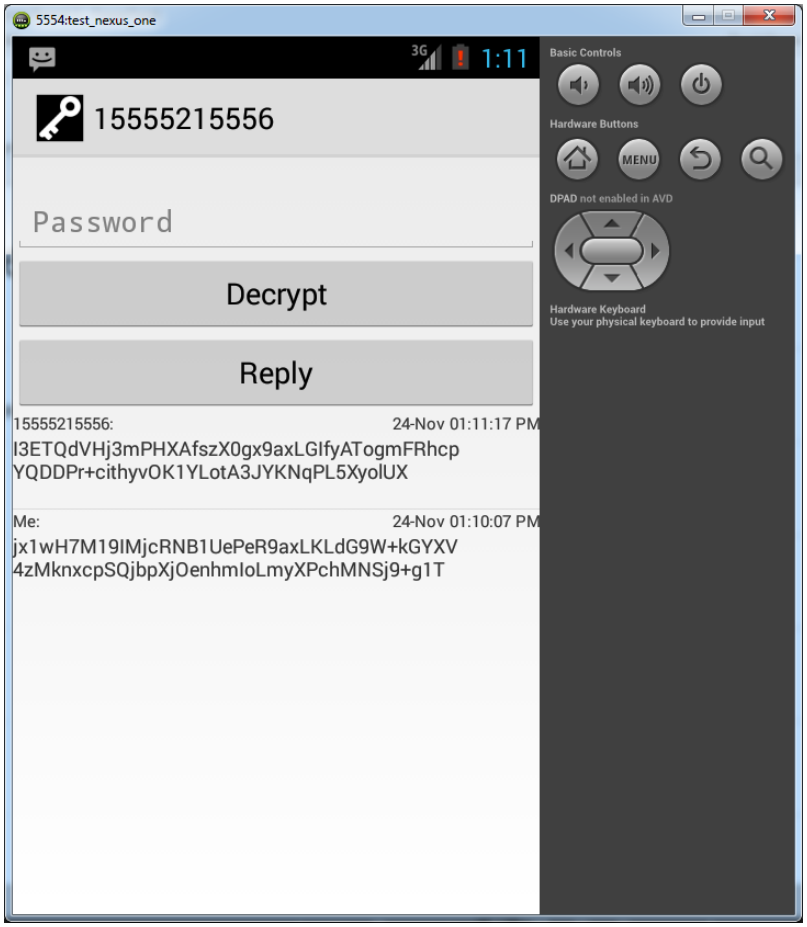
Header	PGPM:	Unencrypted
Recipient Key ID	8 bytes	Unencrypted
Sender Key ID	8 bytes	Unencrypted
Signature Length	1 byte	Encrypted with AES-256
Signature	56 bytes	
Message	39 bytes	

Implementation (Receiving)



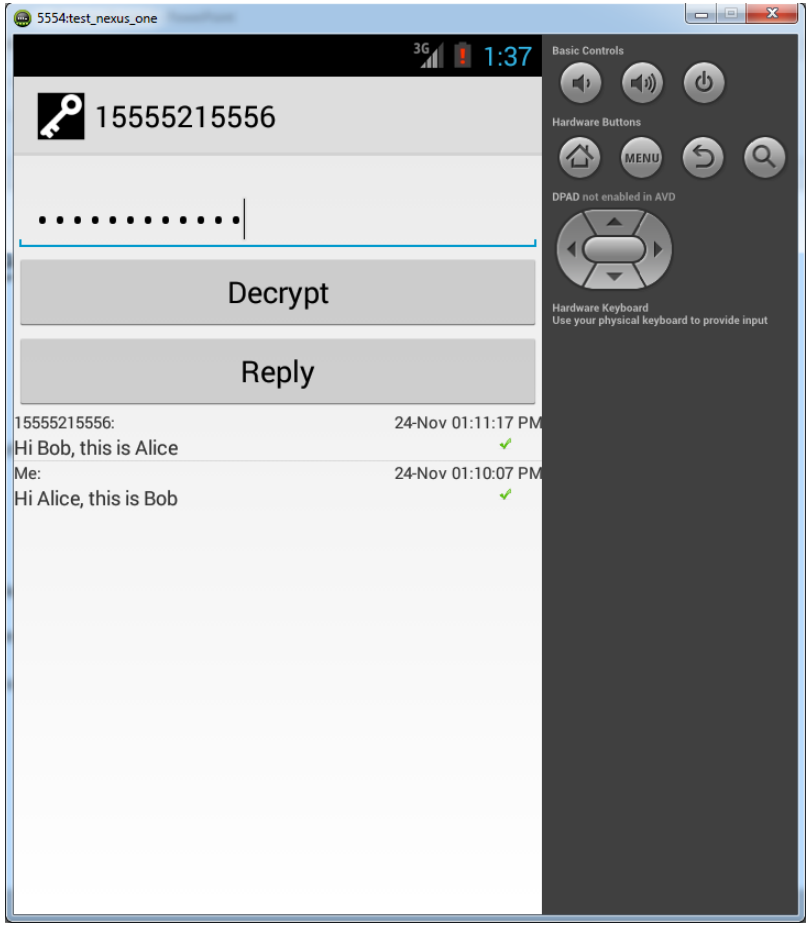
Message Reader Encrypted

- All messages in SMS database displayed
- Messages remain encrypted
- Full encrypted SMS displayed
- Header has been removed



Message Reader Decrypted

- Password entered
- Pressing Decrypt button performs:
  - Base64 Decoding
  - Decryption
  - Signature Verification
- Public Key Trust Indicator
  - Complete trust in public key used: 
  - Marginal trust in public key used: 
  - No trust (or undefined trust) in public key used: 

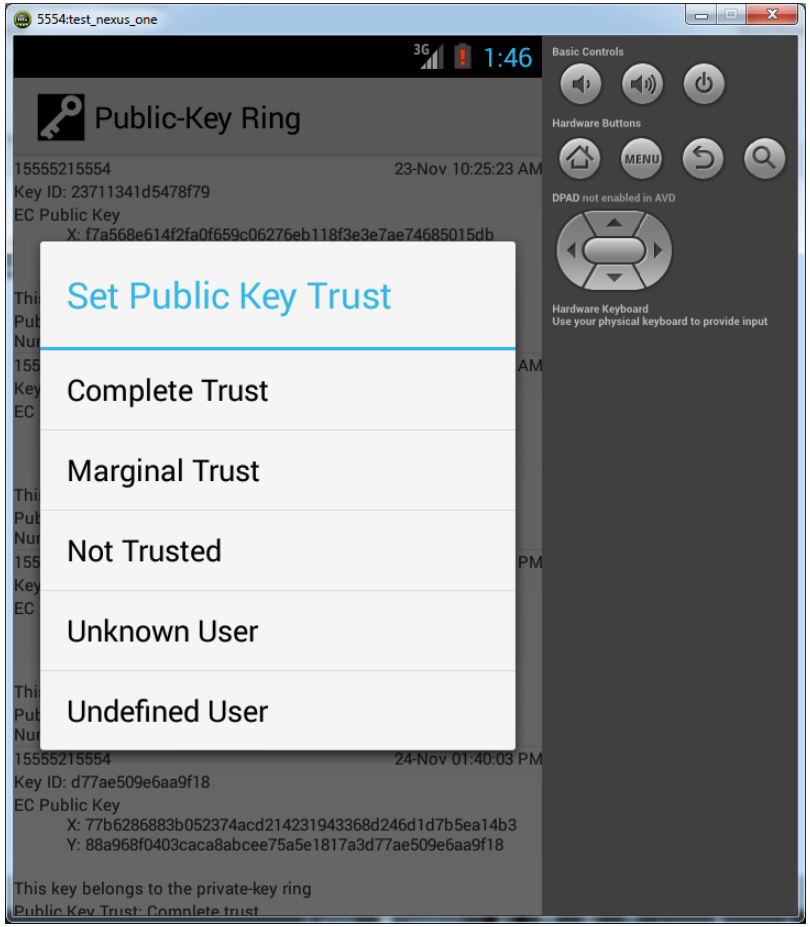


Public-Key Ring

- Displays the following information
  - User ID – 11 digit phone number
  - Key ID – last 64 bits of public key
  - Date added to public key ring
  - Public Key values
  - Owner trust field
  - Public key trust
  - Number of associated signatures

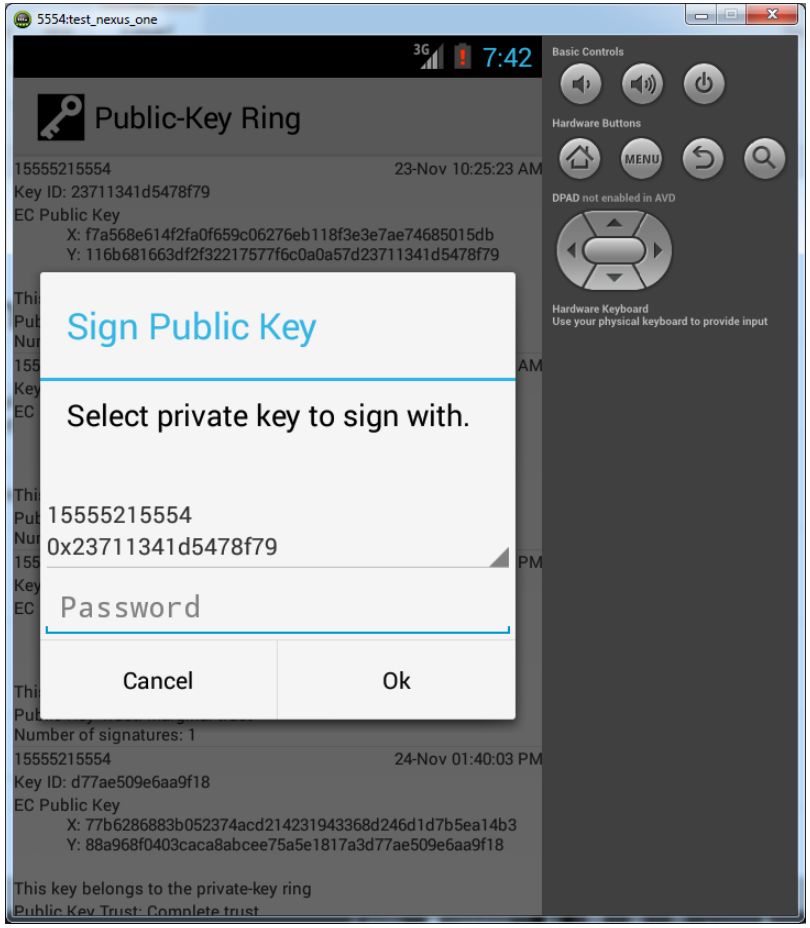
Public-Key Ring Owner Trust

- Available Fields
  - Ultimate Trust
  - Complete Trust
  - Marginal Trust
  - Not Trusted
  - Unknown User
  - Undefined User
- Assigned trust level also applies to all signatures in database



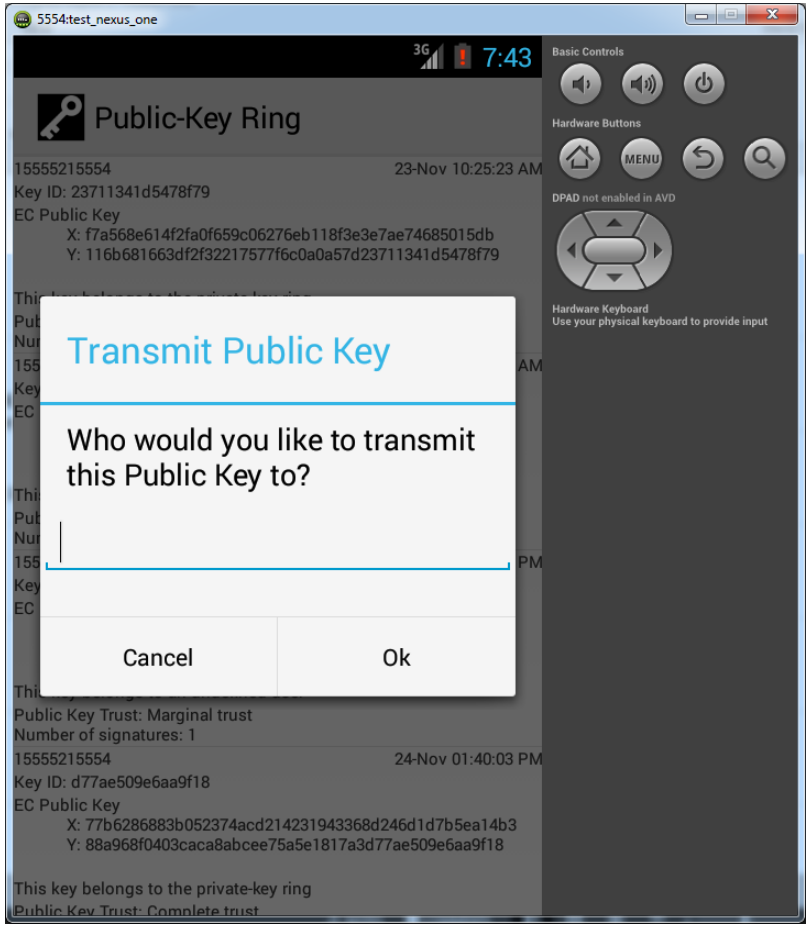
Signature Generation

- User can sign any public key in the Public-Key Ring
- Select private key to sign with
- Password needed for decrypting private key



Signature Transmission

- Any public key and accompanying signatures may be sent to another user
- Public key and all signatures are sent
  - 1 SMS per signature + 1 SMS for Public Key



Public-Key Ring
Public Key Trust

- For any given public key:
  - Has associated signatures
  - Each signature has a user configurable level of trust assigned to it (owner trust)
  - Trust in a public key is derived from the owner trust assigned to the associated signatures

$X = User\ Defined\ Weight \geq 1$

$Y = User\ Defined\ Weight \geq 1$

$\text{Public Key Trust} = 1/X * n\downarrow trusted + 1/Y * n\downarrow \text{usually trusted}$

$\text{Public Key Trust} \geq 1$:	✔	Complete trust
$\text{Public Key Trust} > 0$:	⚠	Marginal trust
$\text{Public Key Trust} = 0$:	✖	No trust

Public-Key Ring
Public Key Trust - Example

- Public Key 0x23711341d5478f79 has three associated signatures
- Signature 1
  - Created by user with Owner Trust = Complete Trust
- Signature 2
  - Created by user with Owner Trust = Marginal Trust
- Signature 3
  - Created by user with Owner Trust = Marginal Trust

$X=2$

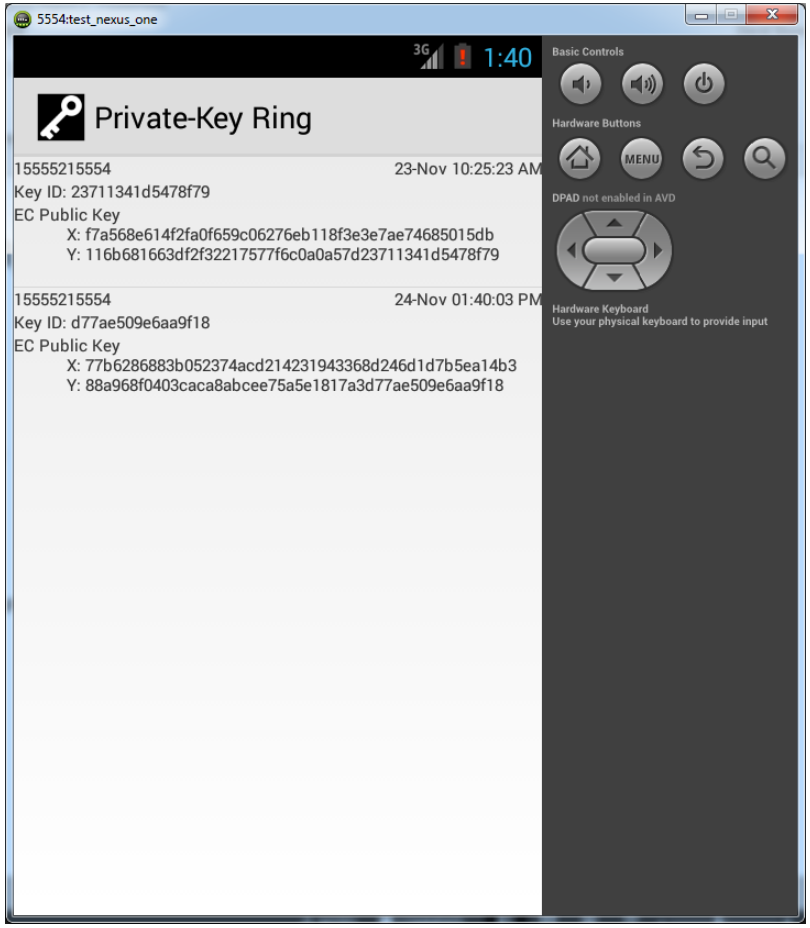
$Y=3$

$$\text{Public Key Trust} = 1/2 * 1 + 1/3 * 2 = 1.167$$

$\text{Public Key Trust} \geq 1$: ✔ Complete trust

Private Key Ring

- Displays the following information
  - User ID – 11 digit phone number
  - Key ID – last 128 bits of public key
  - Date of generation
  - Public Key values
- Does not display
  - Encrypted private key
- Different private keys may be encrypted with different passwords



Summary and Conclusions

- Android Application Developed
  - Provides confidentiality and non-repudiation
  - Implements web of trust
  - Does not require centralized servers
- Confidentiality with overhead
  - 37 effective characters per SMS
  - Security functions impose 75% overhead