RSA CONFERENCE 2014
ASIA PACIFIC & JAPAN

Share.
Learn.
Secure.
Capitalizing on
Collective Intelligence

Split and Conquer!
Don’t Put All Your Keys in One Basket

SESSION ID: MBS-T11

Kayvan Alikhani

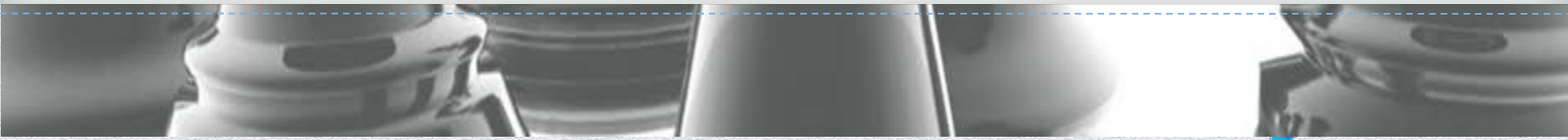
Senior Director of Technology
RSA

Salah Machani

Senior Technologist
RSA

Split and Conquer!

- Approach for building mobile security solutions, with a focus on splitting cryptographic material or user credentials on the mobile device, on wearable/carry-ables or on (cloud) servers.
- Paramount to this design is achieving a proper balance between usability and the security strength.



What is Being Stored?

Certificates & Cryptographic Keys

- Trusted certificates
- Personal certificates
- Signing keys
- Encryption/Decryption keys

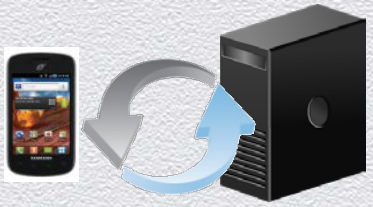
Application Data

- User credentials
- Other app data

Where Are They Being Persisted?

DEVICE — **HYBRID** — **SERVER**

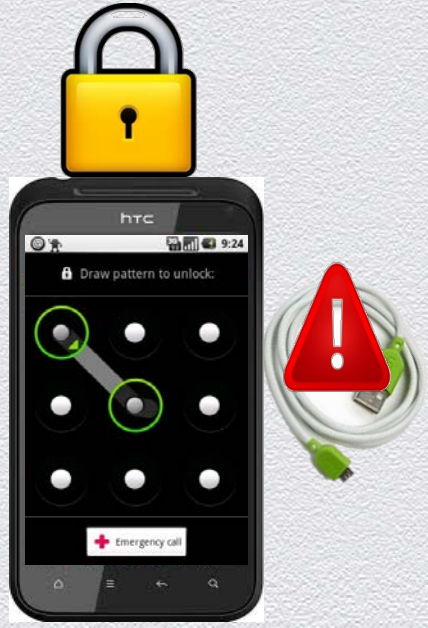
- File system
- Mobile database
- Platform key store
- Security hardware



- Stored locally, synced with server
- Backup services & systems
- **Split between user device (s) and server(s)**

- Stored on server side
- Managed on-demand from device to/from server

File System Storage



TODAY

- Android
  - Device-protected with user password
  - Files stored in internal storage are private
  - Files stored in external store e.g. SD card are public
  - All files can be accessed via USB
- iOS
  - Default encryption of application data in iOS 7 or later
  - Centrally erasable metadata
  - Cryptographic linking to specific device

RISK

- Android: No password needed to access data via USB
- Android: Files in SD card are public

Platform Key Store

TODAY

- Android
  - KeyStore for cryptographic keys
  - Keystore doesn't have an inherent protection
- iOS
  - KeyChain for cryptographic keys, passwords and login tokens
  - KeyChain items are encrypted using device UID and user passcode
  - iOS service determines which keychain items each process can access

RISK

- Given physical access, specialized tools used to read data from keychains and key stores.

Mobile Database Storage

TODAY

- SQLite and extensions for encryption
- Encrypted credentials stored in local device database
- Password-based encryption

RISK

- DB password is often weak, not well protected, and ‘cached’ in clear text
- Files susceptible to brute-force attacks if device is rooted

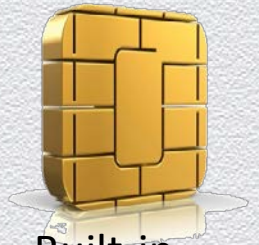
Security Hardware

Secure Elements (SE) in Various Forms

LOCAL — **IN PROXIMITY** — **REMOTE**

SIM Cards

SD Card



Built-in
Smart Cards

NFC

Bluetooth

USB Connected
Smart Card

Second Device:

- Keys paired
- Mediated by server

Server Side:

- Keys stored on server
- Compared on server side

Security Hardware

Challenges with SIM-based SE

Not all devices have SIM cards

Not all SIMs are programmable

Dependent on OS-based support

Cannot fully protect if root access is available

Requires customized “Global Platform” load

Security Hardware

SE-based Storage: Pairing Challenges

Something else user must carry

Requires user provisioning and enrollment

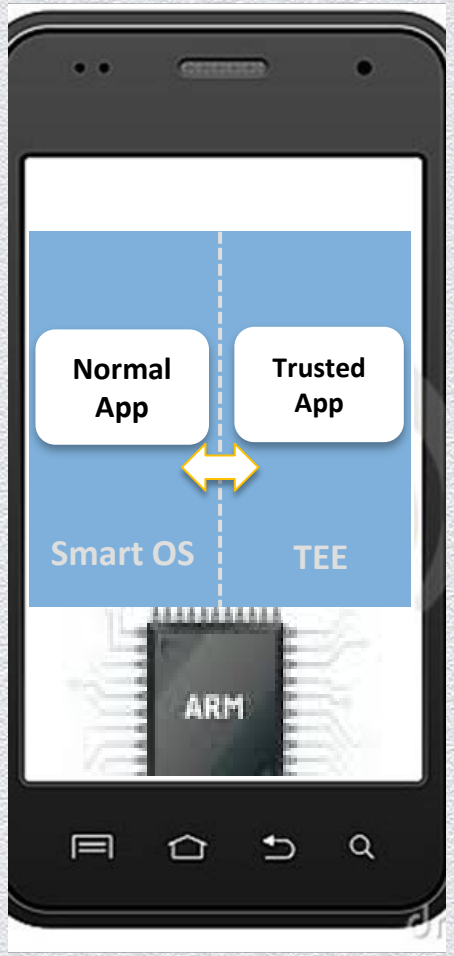
Bluetooth/NFC must be ‘ON’ → battery drain

Network approach requires network connectivity

Risk of unintentional pairing with rogue-devices

Security Hardware

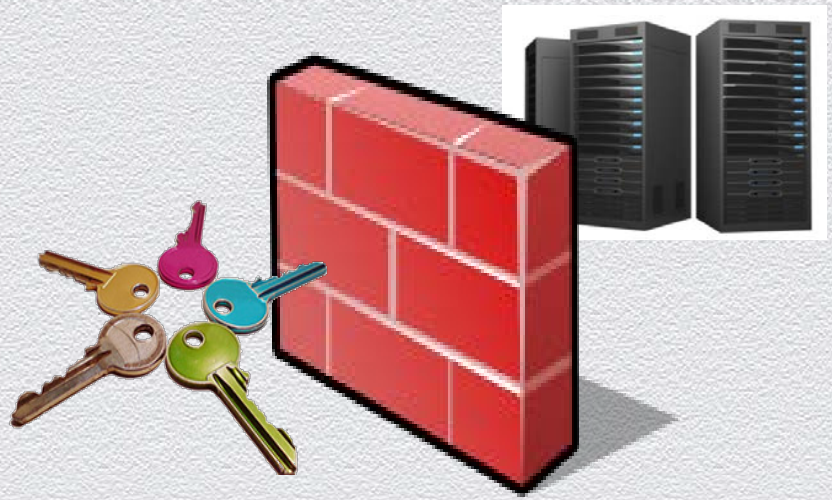
TEE in Mobile Devices



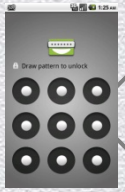
- Secure area in main processor
  - Predominantly based on ARM TrustZone
- Data stored, processed and protected in trusted environment
  - Confidentiality, integrity and data access rights
- Currently enabled in few smartphones

Remote Storage/Backup

- Server should have ability to back up data
- Should **NOT** have keys to decrypt data
  - Recent findings: Use Android? You're Probably Giving Google All Your Wifi Passwords



How Are They Being Protected?



Locally verified PIN/password

Encrypted local data (AES or other ciphers)

Protected with Multi-Factor Verification (MFV)

Secure data transmission (SSL, AES, etc)

Common Ways for Securing Data on Devices

- Data encrypted with strong encryption algorithm where:
  - Master encryption key is derived from user password or
  - A stronger master key (MK) is generated and protected with user password

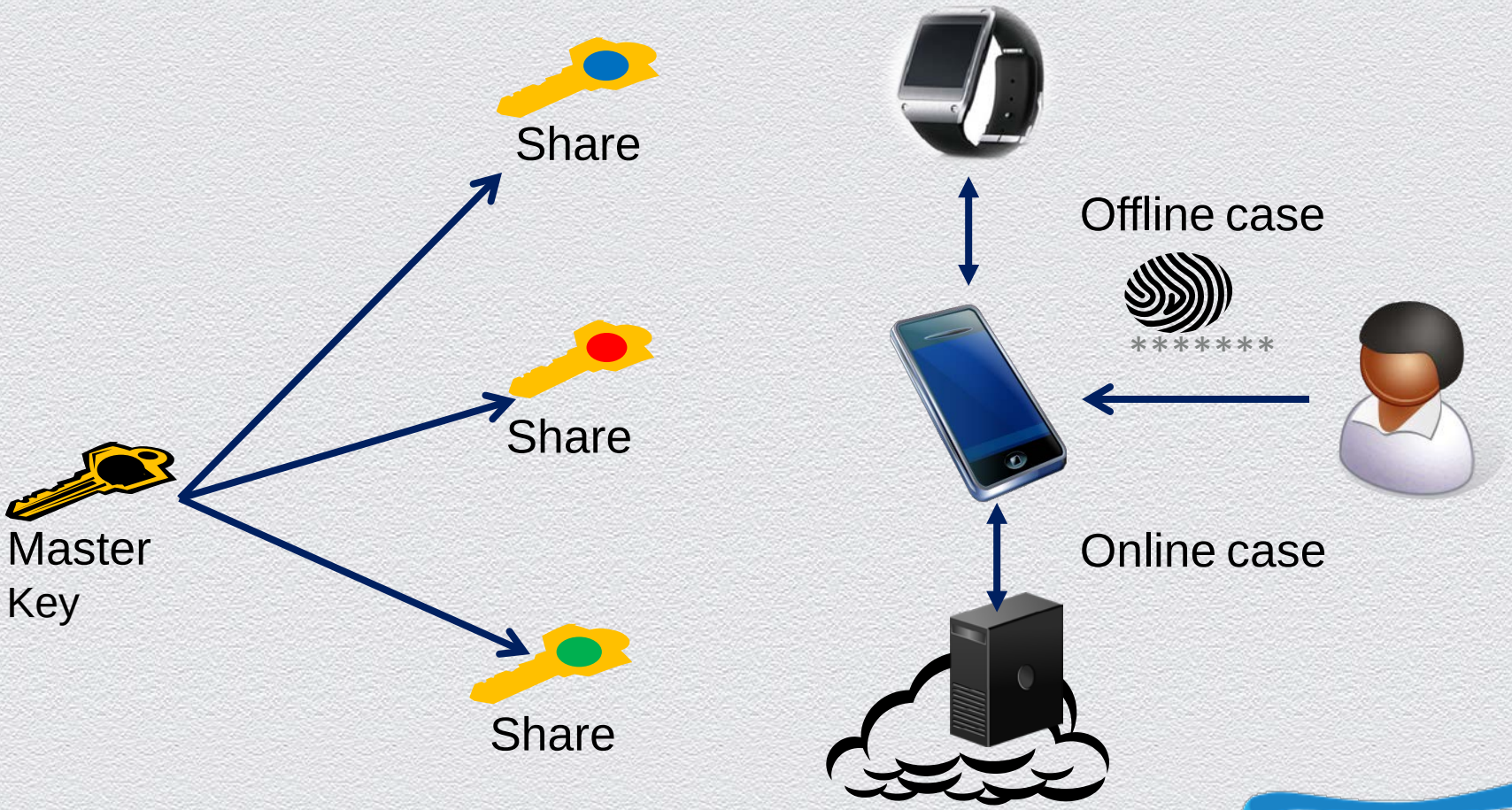
Threats

- Attacker has access to at least one of:
  - Lost/stolen device
  - Device backup/image
  - Application database
- The attacker's goal is to:
  - Extract sensitive data stored in application database

What is Secret Splitting/Sharing

- Method for distributing secret amongst group of users
  - Each user is given a share of the secret
- Secret can be reconstructed only when a minimum and right number of user shares are combined together
  - Individual shares are of no use

Secret Sharing Approach in Mobile Applications



Applying Secret Sharing in Mobile Platforms

- **STEP 1**: generate a strong master secret
- **STEP 2**: split secret into multiple shares
- **STEP 3**: distribute and store shares under differ

STEP 1: Generate Strong Master Secret

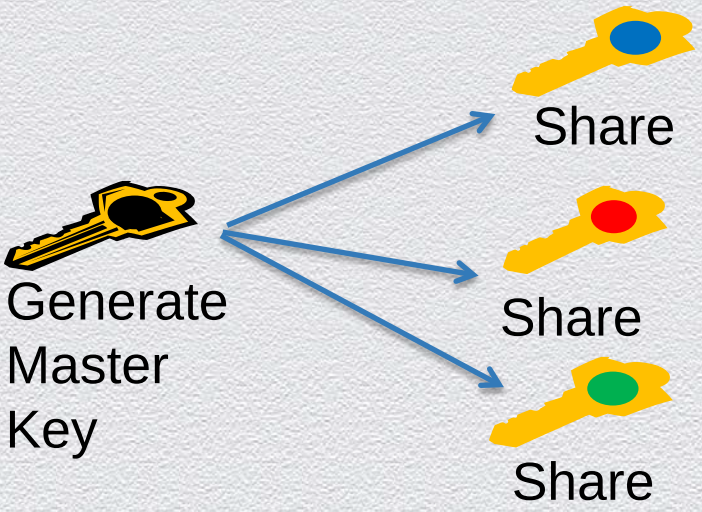
- Use Cryptographically Secure Pseudo Random Number Generator (CSPRNG)
  - Default device platform secure random generators
  - Be aware of some vulnerabilities
    - Early RNG in iOS 7 generates predictable outcomes
    - Android PRNG security vulnerabilities were exploited - patched
- Consider collecting additional entropy data from device sensors
  - Such as microphone, accelerometer, magnetometer, gyroscope, camera, GPS, etc.

STEP 2: Split the Secret into Multiple Shares

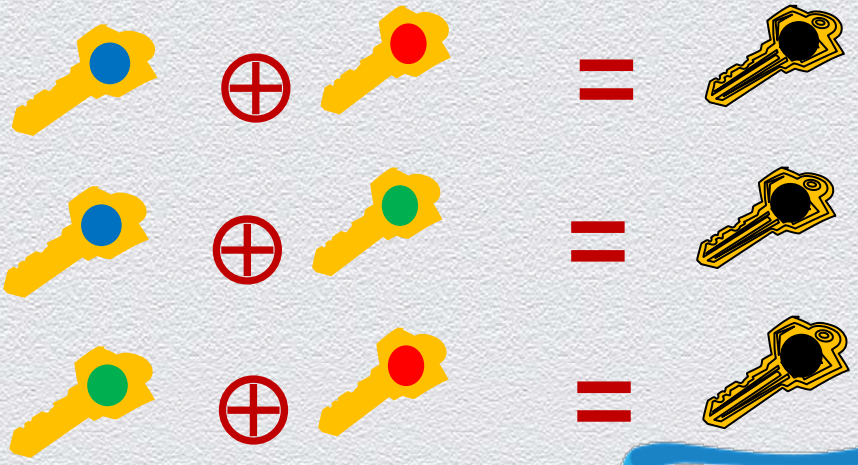
- (**t**, **n**)-threshold secret sharing scheme
  - Split secret **S** into **N** shares with threshold **T** (**1 < T <= N**) so that:
    - Any **T** of **N** shares where can be combined to reconstruct **S**
    - Less than **T** shares won't provide any info about **S**
- Arbitrary access structure
  - Split secret **S** into **N** shares so that :
    - Only desired subsets of shares can reconstruct **S**
    - Different subsets may contain different number of shares

STEP 2: Split the Secret into Multiple Shares

Example 1: (2,3)-threshold Secret Splitting Scheme

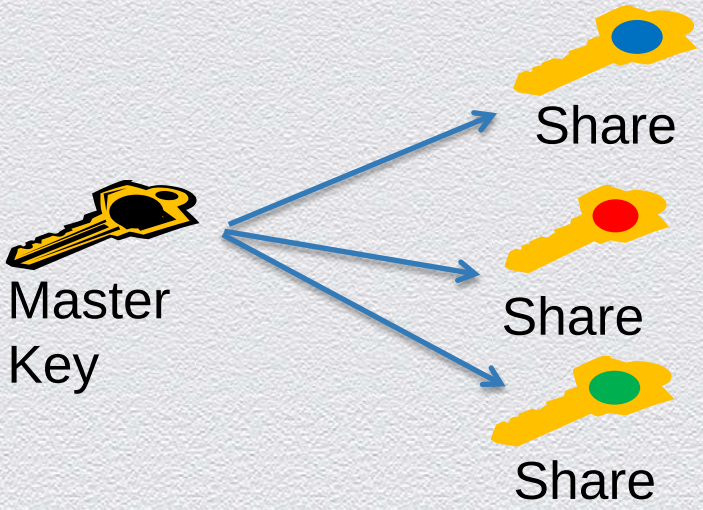


At least 2 shares must be combined to reconstruct the MK

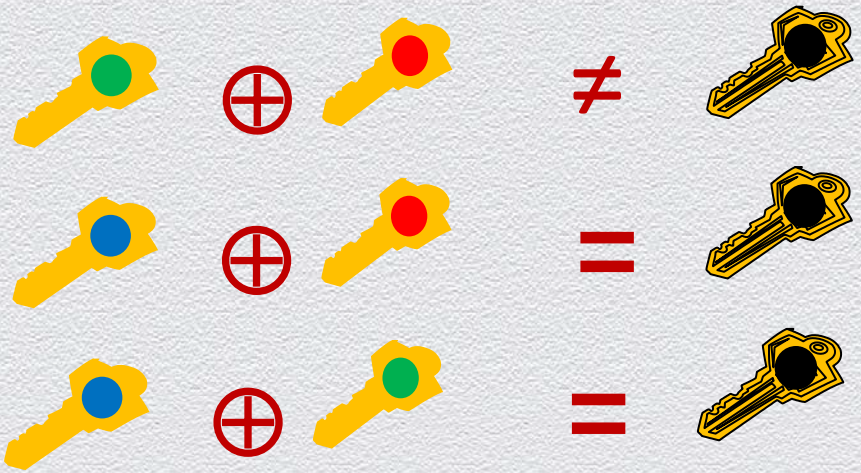


STEP 2: Split the Secret into Multiple Shares

Example #2: Arbitrary Sharing Scheme



Only a subset of combinations can reconstruct MK



STEP 3: Distribute and Store Shares

- Distribute shares across multiple secure stores in different devices
- Define access-control policies using one or multiple authentication factors
  - Secure stores make access-control decisions based on presented credentials before releasing shares
  - Master key can be reconstructed only if appropriate number of shares are released and combined

Secret Shares Management Concerns

- Secret shares replacement scenarios
  - Mobile or wearable device lost/stolen
  - Device/Application data wiped
  - Master key or shares compromised
- Consider
  - Combining shares in wearable device and on server to restore MK
  - Generate dedicated restore share and protect it in backup & restore service

Secret Splitting Approach Challenges/Weaknesses

- Crypto is well-studied and provably secure; however, it depends on correct implementations
  - Careful design for shares distribution and access-control structure
  - Can be complex to implement
- Doesn’t address memory attacks on rooted/jailbroken devices

Pros & Cons: Store It, Protect It

Storage and Protection Methods	User Convenience	Ease of Implementation	HW/OS Independent	Security Level	Withstand Rooted Access
Local File System, PW protected	Med	High	Yes	Low	Low
Mobile Database, PW protected	Med	High	Yes	Low	Low
Native Key Store, PIN/MFA protected	High	High	Yes	Med	Low
Hardware SE, PIN Protected	High	Low	No	Medium	Medium
Local/Remote Server, MFV Protected	High	Medium	Yes	Med	Med
Distributed Secret Share Stores, MFV protected	**High**	**Low**	**Yes**	**High**	**High**

Recommendations

- Avoid storing secrets on mobile devices
  - Don’t hard-code secrets or store them in clear-text files
- Use SEs when possible
  - Increases integrity of data
  - Reduces chance of data tampering
- Use secret sharing approach
  - Physical device access will be inconsequential
  - Servers and wearable devices can help with data restore

Recommendations: Continued…

- If you use secret sharing approach
  - Use native key stores and strengthen access-control with multi-factor verification
  - Use TLS or AES when transmitting shares over the network
  - Don’t back-up or cache all or subset of shares that can be combined to reconstruct master key

Security/Authentication Initiatives & Alliances

- FIDO
  Mobile OS independent, multi-factor verification
- Cloud Security Alliance
  Scalable authentication from mobile devices to multiple, heterogeneous cloud providers
- Open ID Connect
  Standards-based authentication protocol built on top of OAuth 2.0 developed by the OpenID Foundation - support for 'Android accounts'
- Global Platform

  Standards for managing applications on secure chip technology
- SIM Alliance

  Secure Element ecosystem - facilitate delivery of secure mobile apps.

References

http://en.wikipedia.org/wiki/Message_Authentication_Code

http://www.kandroid.org/online-pdk/guide/stk.html

http://code.google.com/p/seek-for-android/wiki/SecurityConcept

http://www.nearfieldcommunication.org/bluetooth.html

http://www.smartcardalliance.org/pages/publications-nfc-frequently-asked-questions

http://nelenkov.blogspot.in/2013/08/credential-storage-enhancements-android-43.html

https://developer.android.com/training/articles/security-tips.html

https://code.google.com/p/android/issues/detail?id=57560

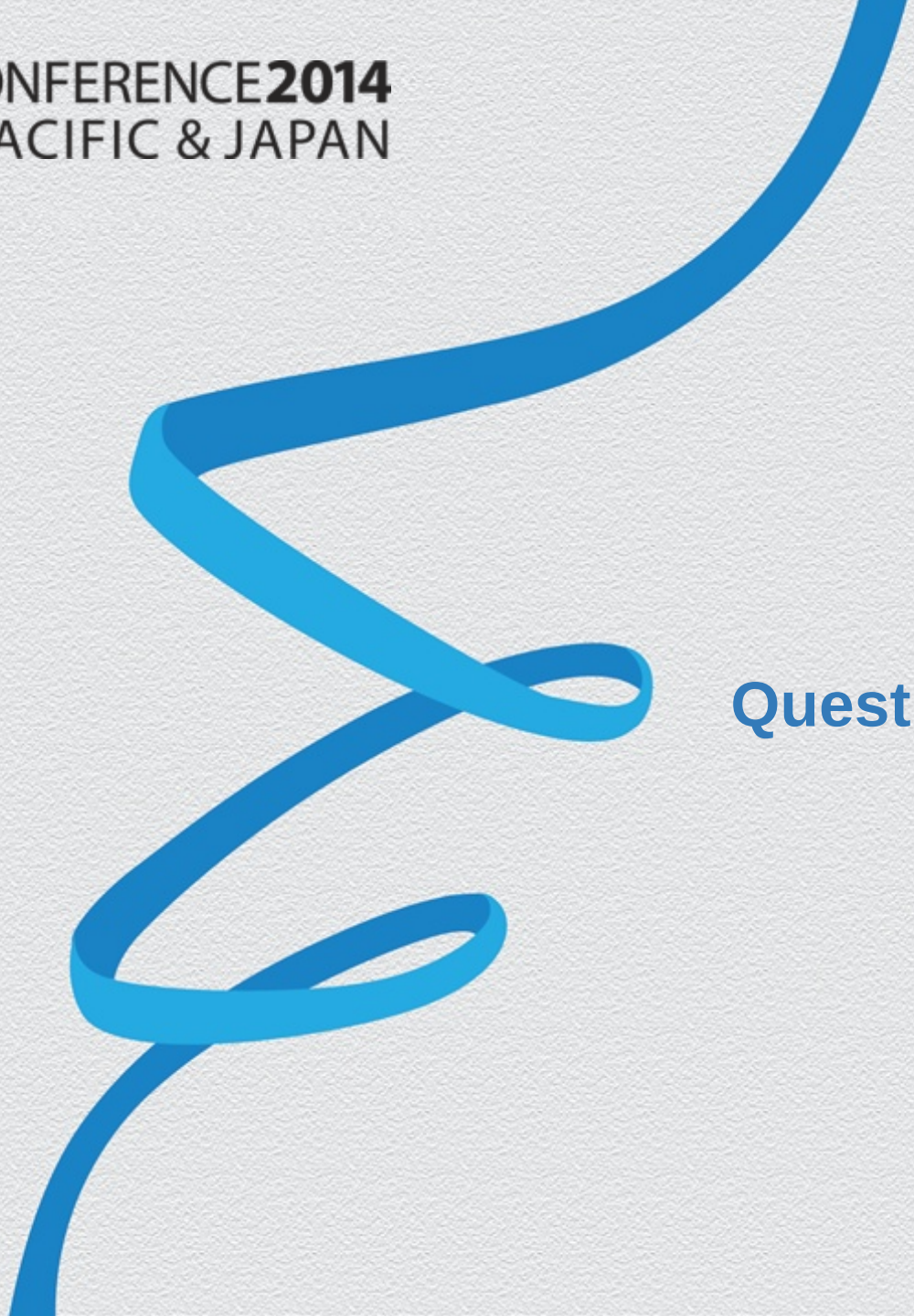
http://www.internetsociety.org/sites/default/files/02_4.pdf

http://www.elcomsoft.com/WP/BH-EU-2012-WP.pdf

http://en.wikipedia.org/wiki/Secret_sharing

http://mista.nu/research/early_random-paper.pdf

http://android-developers.blogspot.co.il/2013/08/some-securerandom-thoughts.html



Questions?