DEVELOPMENT OF A PRIVACY PRESERVING LIFERAY PORTAL DOCUMENT SYNCHRONIZER FOR ANDROID

BY

MAX PERRY PERINATO

Thesis Supervisor: Prof. Michele Bugliesi

DEPARTMENT OF ENVIRONMENTAL SCIENCES, INFORMATICS AND STATISTICS

MASTER OF SCIENCE IN COMPUTER SCIENCE

A.Y. 2011/2012

VENICE, 1 MARCH 2013

Motivation

- Bring-Your-Own-Device is becoming an *inevitable trend* (Juniper Research)
- Employees are bringing their own smartphones and tablets to work

  ➔ Access to documents anytime, anywhere

  – Private information concerning the enterprise
  – Personal information about employees and clients

  ➔ Confidentiality and liability issues arise

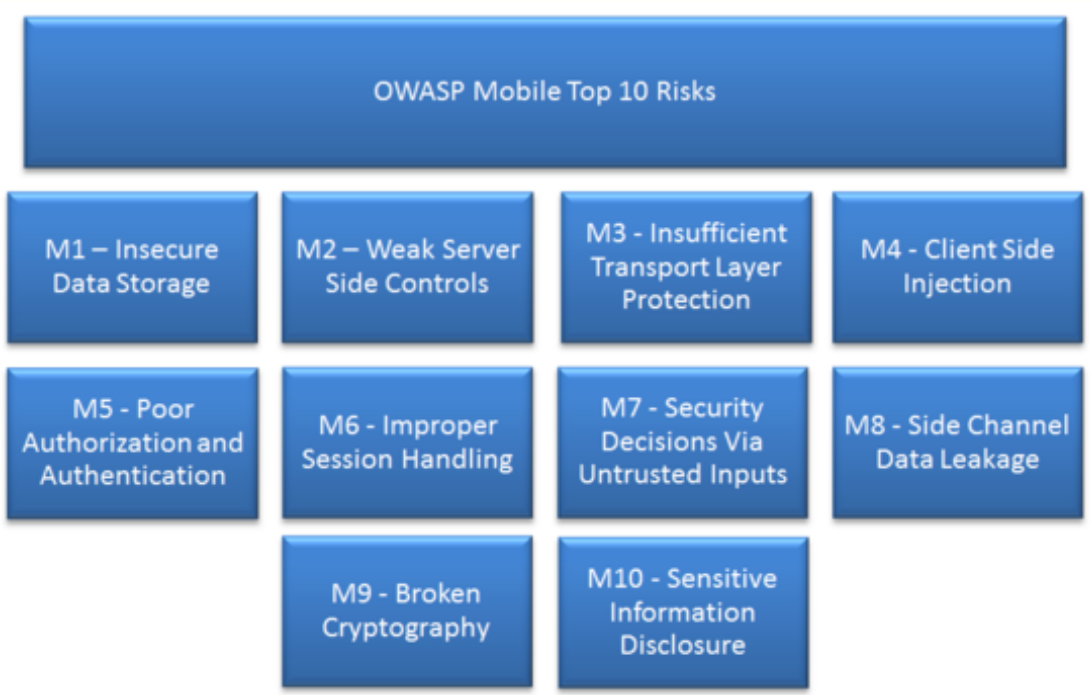
- *Security and data breach are the greatest barriers for BYOD* (Trend Micro)

Mobile Security Risks

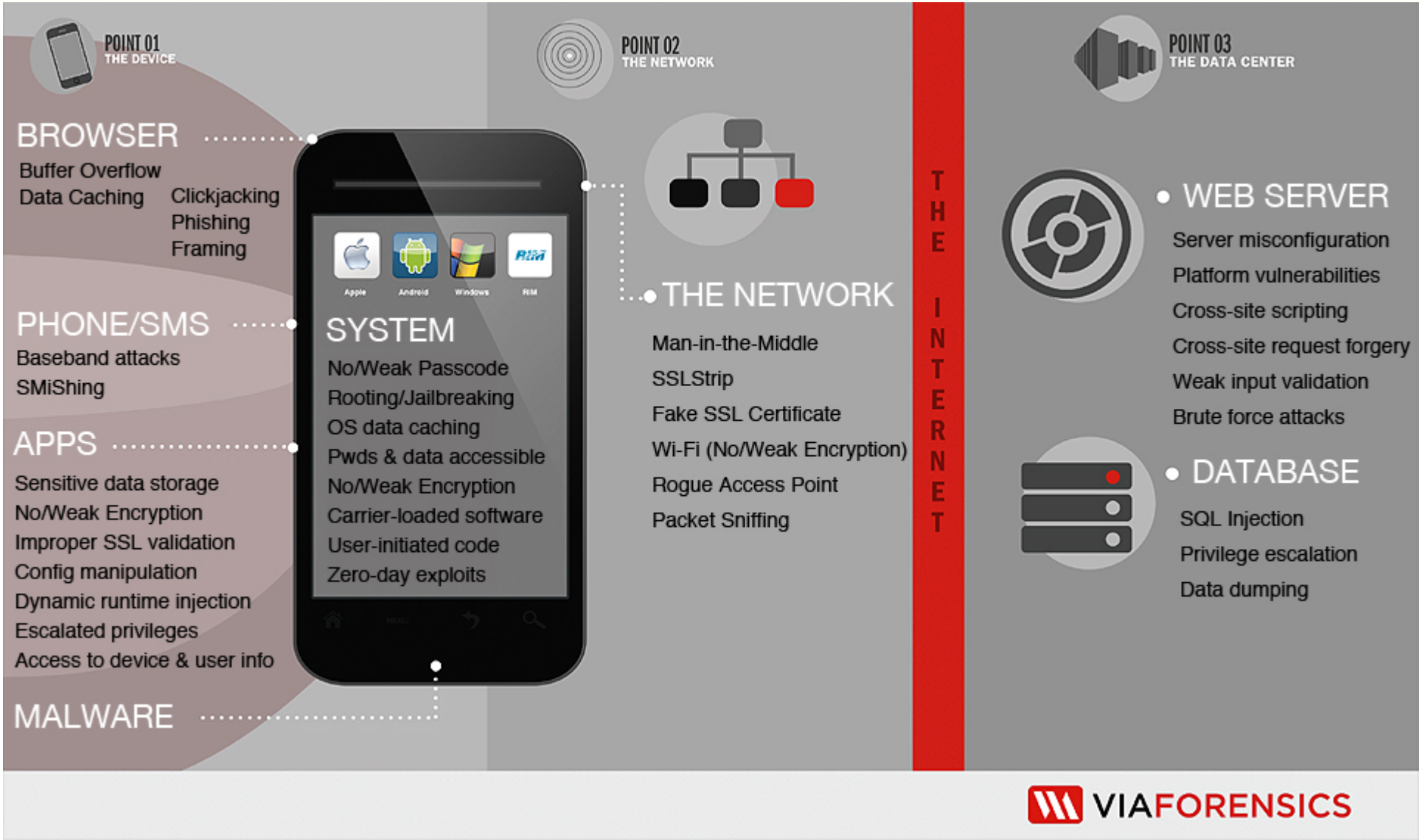
- Mobile device security model erroneously based on security model of predecessor: laptop computer
- Mobile devices are always turned on and almost always connected

  ➔ new set of security risks and attack vectors

- Information discloure via flash memory or RAM
- Privilege escalation bugs
- Bad design and insecure coding practices

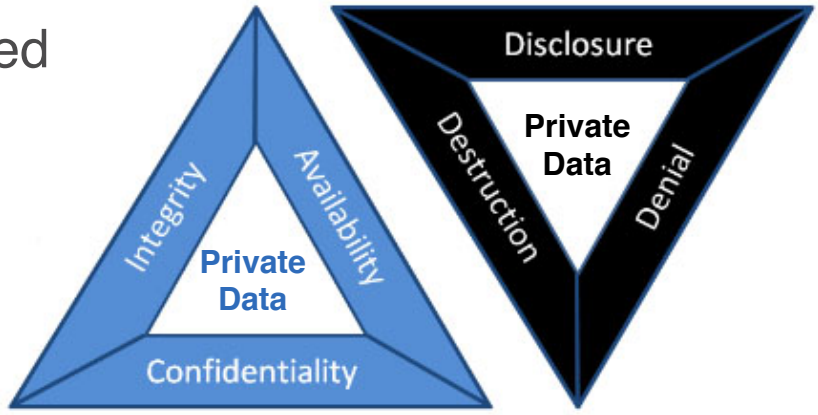


Mobile Attacks



Preserving Privacy of Enterprise Data

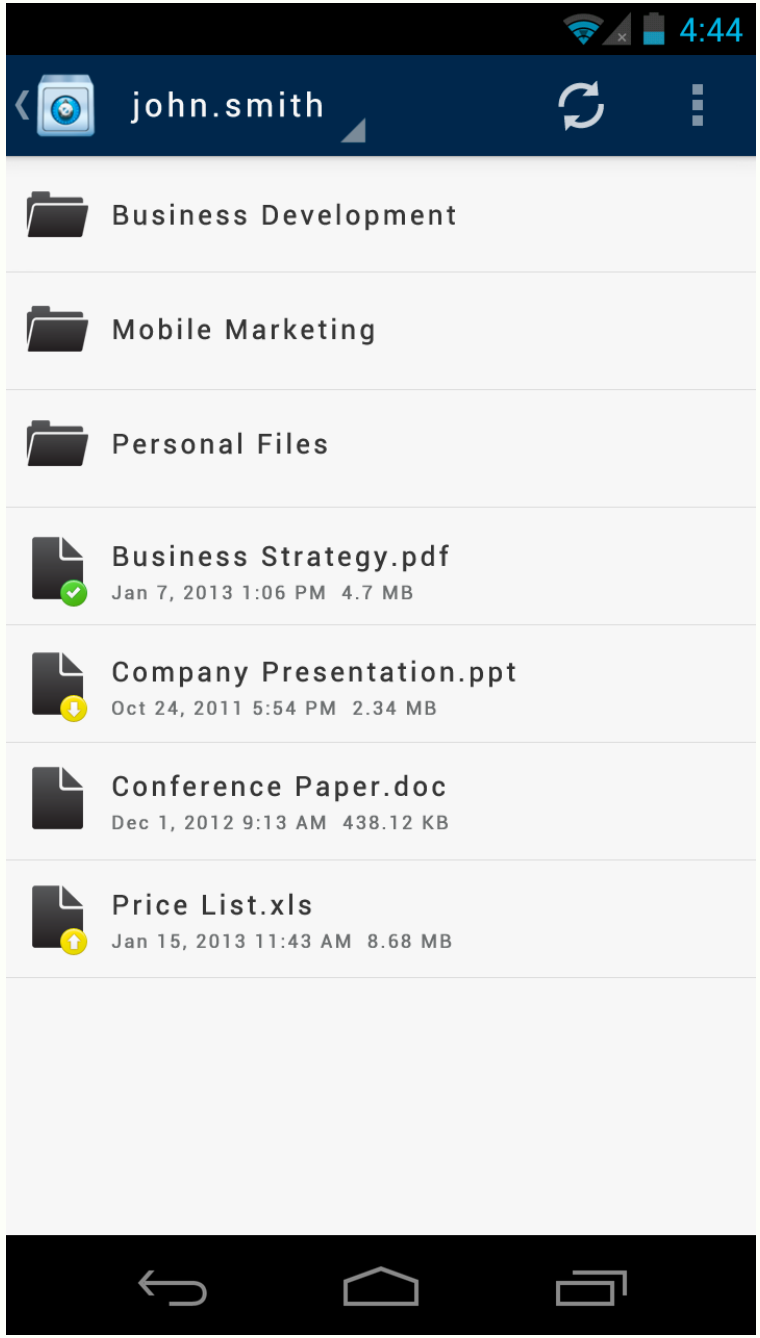
- BYOD poses one major challenge to be addressed:
  - *Protecting and securing the privacy of sensitive data at all times while allowing unrestricted access to public data*
- Information security becomes highly dependent on situational information:
  - Security of the device, its location, the user, the network and the apps being used
- Access to sensitive data can be allowed with "Security Containers"
  - Can mitigate risks surrounding CIA of resources
  - Can be trusted by enterprises



CIA Triad – ISO 27001

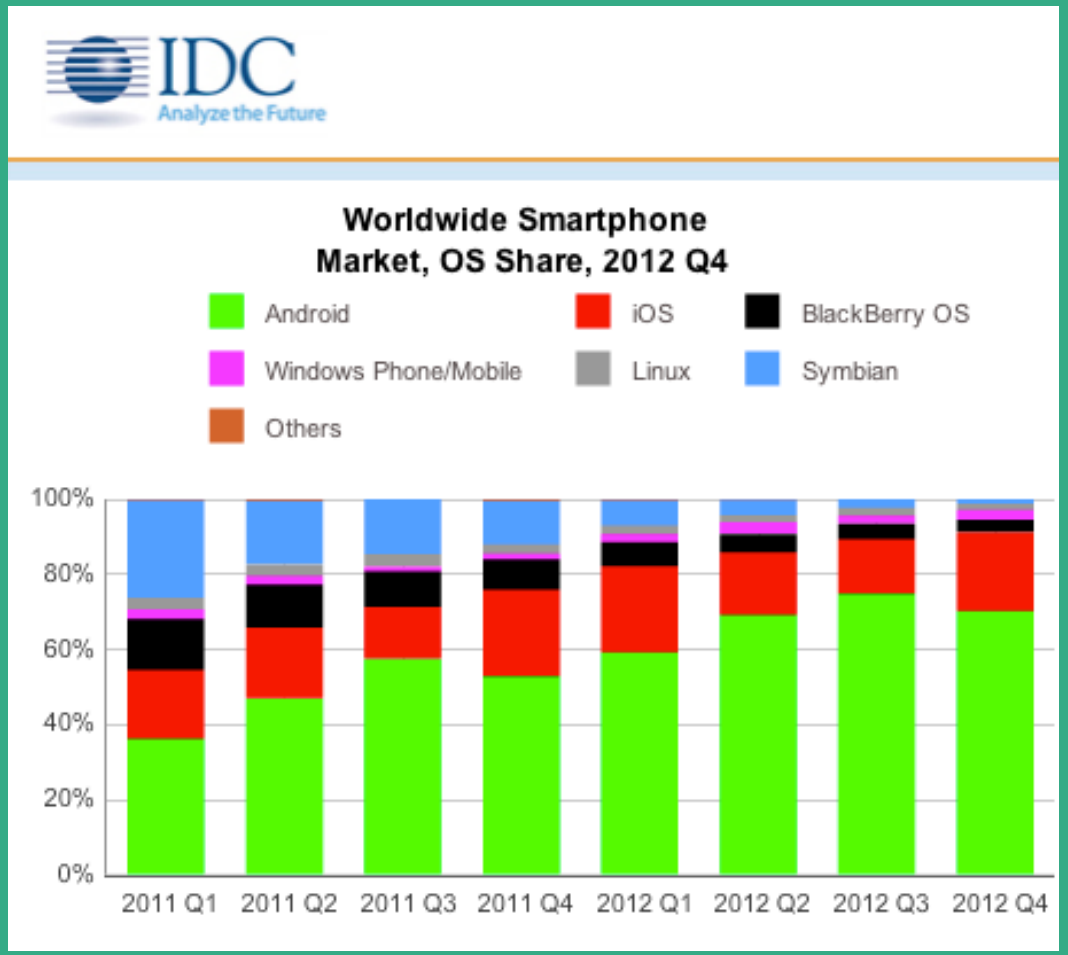
LIFERAY SAFE

- Android app for synchronization of documents with Liferay Portal
  - ➔ *"The leading open source Portal for the Enterprise"*
- Built as a Security Container
  - Data encryption
  - Data access and usage control
  - Security of data in transit
  - Security of user credentials
  - Data loss prevention: passcode enforcement, automatic/remote application lock and data wiping
  - Dynamic provisioning of user trust
- Provides security of private data and offline usage
- Protection from malicious outsiders
  - e.g., device loss or theft
- Protection from malicious insiders
  - e.g., employee leaves the company



Android is leading the pack…

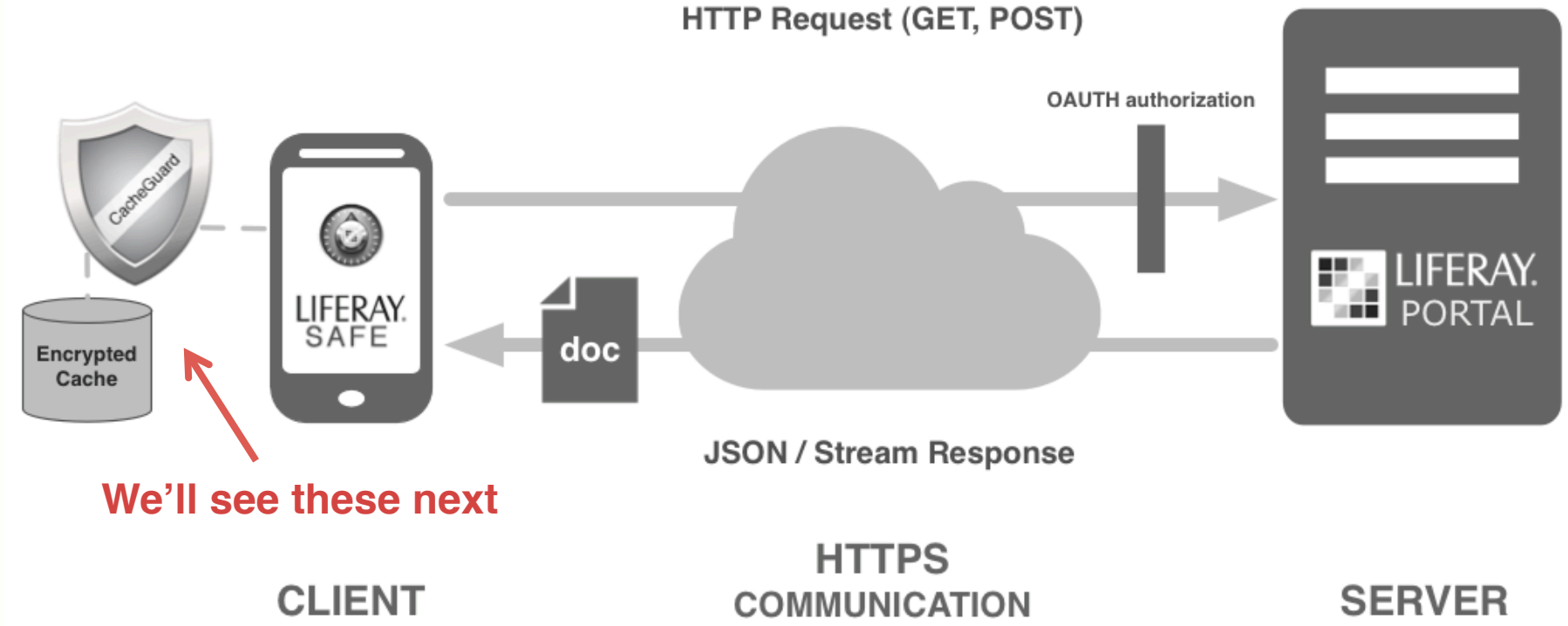
- 722.3 million smartphones shipped globally in 2012
- 68.8% (497.1 million) are Android devices



…but popularity comes at a price

- 145.000 malicious Android apps released in 3Q12 (Trend Micro)
- Lack of a control in app development and effective moderation in Google Play store
  - ➔ Can lead to exposure of private information
- Android's security model is flawed:
  - Kernel-level sandboxing
    - ➔ Allows privilege escalation attacks (Davi et al.)
  - Application-level mandatory access control
    - ➔ Allows permission misuse and insecure data flows (Fuchs et al.)
- Inter-application message passing also an attack surface.
  - ➔ Message contents sniffed, modified, stolen or replaced (Chin et al.)

Client-Server Architecture



- Transport Layer Security (TLS) protocol for communication security
  - Prevents eavesdropping, tampering, and message forgery
- Server identity authentication
  - Full validation of CA-signed certificate
- Disabling of insecure channels and TLS validation to prevent side channel & stripping attacks

- OAuth 2.0 protocol for client authorization
  - Separates API security credentials from the User's credentials
- Access Tokens can be revoked for an individual User or the entire app
  - Unique identifier tied to the app, hard to guess, with restricted scope and limited lifetime

Challenges 1/2

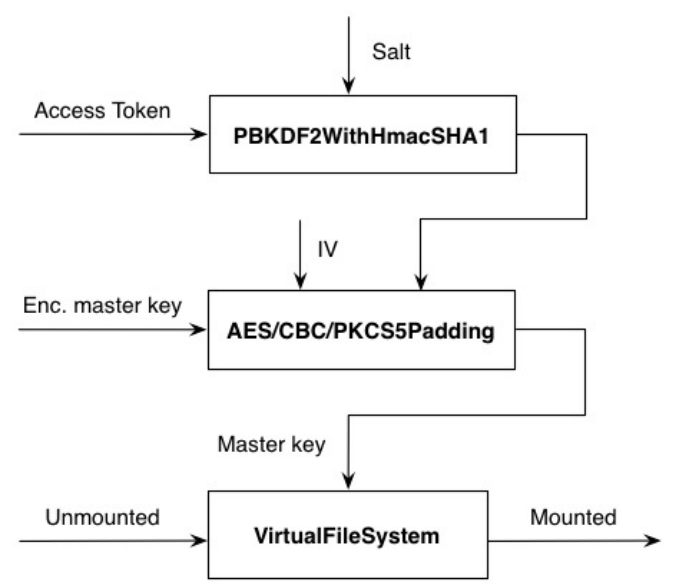
- Lack of a “root of trust”, *enterprises can trust neither its employees nor their own devices*
- Complex management and protection of encryption keys and OAuth tokens
- Offline usage hinders user revocation, and remote wiping or locking
- Little control over how devices are used and what apps are installed
- Rooting a device is easy (e.g., SuperOneClick), no 100% effective way to detect it
- On some devices fastboot allows to re-flash partitions and install a Custom Firmware (e.g., CyanogenMod)

Challenges 2/2

- Data extraction with open source forensics tools (e.g., OSAF-TK, Santoku)
- Limited internal storage. Mountable (and removable) external storage
- Impracticable data zeroization on NAND Flash memory due to wear leveling technique
- Negative impact of security provisions on user experience and battery life

Private Documents Caching and Encryption

- Encrypted caching of private data for offline usage
- App-level Virtual Encrypted Disk based on IOCipher library (by The Guardian Project)
  - Clone of the standard java.io API
  - SQLCipher (by Zetetic LLC) 256-bit AES transparent on-the-fly encryption
  - Libsqlfs (by PalmSource) POSIX style file system on top of an SQLite database
- VED initialized with random master key encrypted with a 256-bit AES key derived from the Access Token
  - Access Token has a validity of 24 hours
  - When the Token expires the master key and the VED file are erased
  - Access Token can also be revoked from the server



Access Token Management

- Access Token is secured in RAM by CacheGuard
  - In-memory obfuscation
  - Mitigates lack of a "root of trust" problem
- Exposure to memory analysis
  - ➔ *requires gaining root privileges* (Sylve et. Al)
  - Android Debug Bridge
    - ➔ *Mitigation: Enforce disabling of "USB debugging" setting*
  - Recovery Boot
    - ➔ *Assumption: Access Token is cleared after reboot*
  - Remote Exploitation
    - ➔ *Mitigation: Require minimum Android version (at least Jelly Bean)*
  - Complete access to device
    - ➔ *Mitigation: Enforce use of a password screen lock*
  - ➔ Attempt to detect if the device is rooted with a set of heuristics

Conclusions

- *Documents are safe with BYOD at a trade-off:* at the state of the art it's not possible to provide privacy preservation and offline access without posing any assumptions and constraints:
  - 24 hours limited offline access
  - Definition and enforcement of enterprise policies
  - Size limit of private documents (available RAM)
  - Minimum Android version (4.1 Jelly Bean)
  - Mandatory screen lock and disabled "USB debugging" setting
  - Reduced battery life
- *Lack of a "root of trust":* some Android devices currently embed a Trusted Platform Module (i.e., Secure Element), but it's not open to third-party apps
  - Necessary to establish a ground of truth on which to build security
  - Help increase trustworthiness of consumer devices

“Never commit to memory what can be easily looked up in books.”

- Albert Einstein