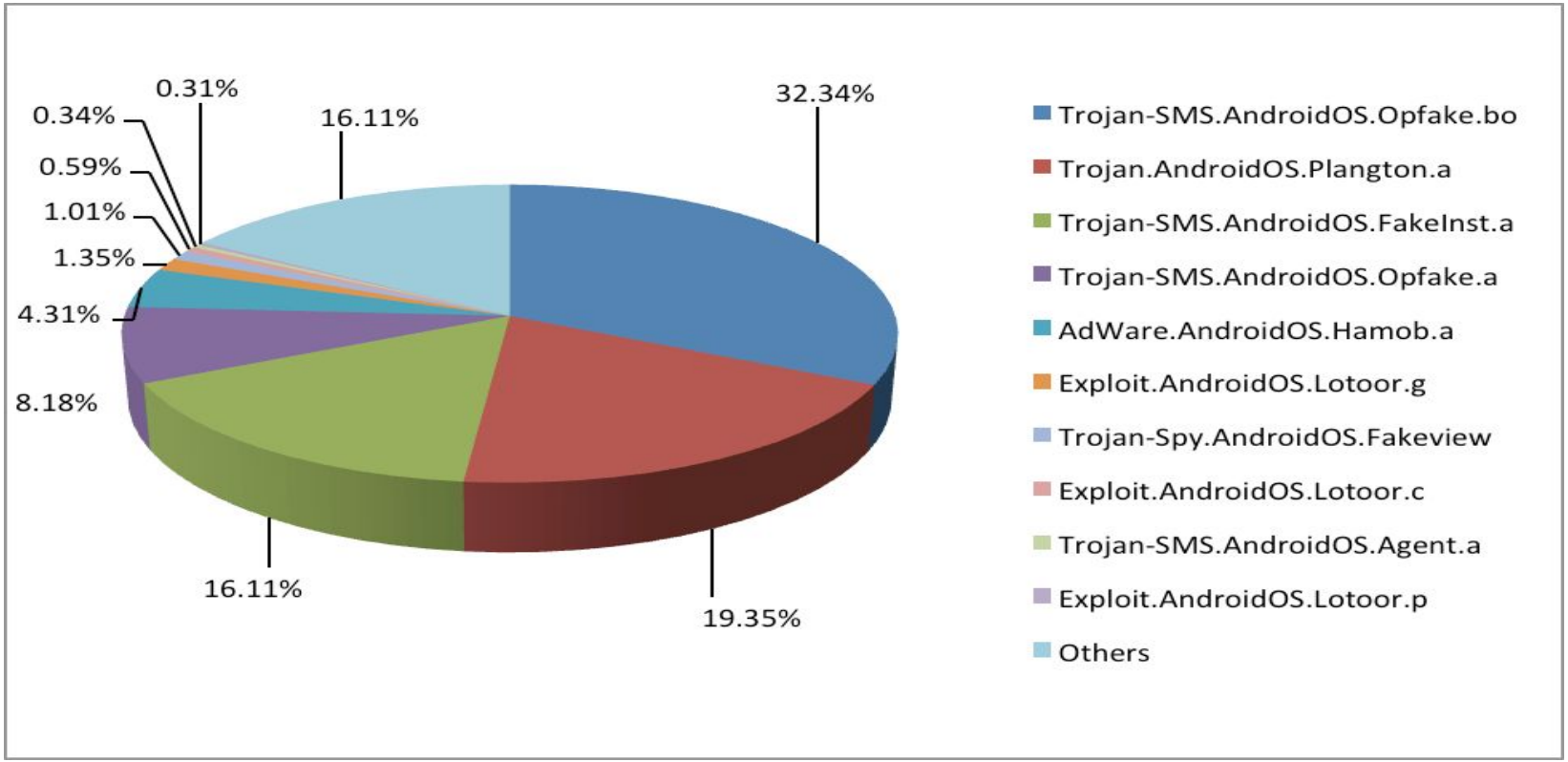
Android Security

Diffusione Android e Malware

- Android e` un sistema operativo open che ha raggiunto fin'ora una diffusione **globale maggiore del 60%**.
- E` il sistema di smartphone piu` colpito da attacchi informatici.
- Secondo gli analisti di Kaspersky lab, Il **rapporto fra malware android e altri mobile system e` giunto ad un valore di 9:1**

Diffusione Android e Malware



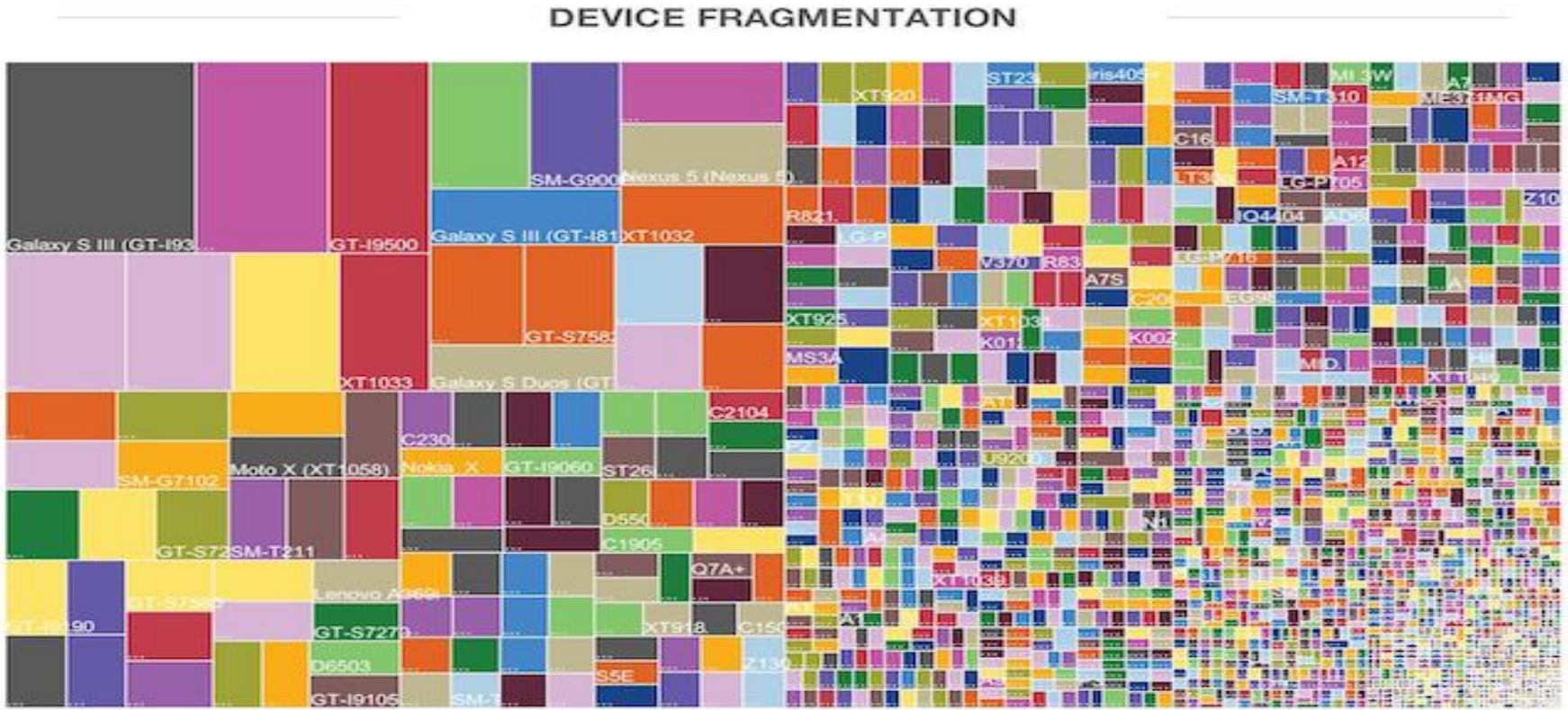
Frammentazione e Fattori di rischio

- Il principale problema della sicurezza di android e` derivato dalla **frammentazione**.
- La frammentazione e` legata alla diffusione del **sistema operativo**, spesso in **versioni non recenti**, su tipologie di HW prodotte da differenti aziende per lo piu` closed source.
- Un’analisi statistica ha rilevato su un campione di circa 700000 dispostivi ha consentito di rilevare l’esistenza di **3997 modelli distinti 599 produttori di device diversi.**

Frammentazione e Fattori di rischio

- Indagini fatte dallo stesso Google 2013 dice che c'e` la presenza di mutamenti abbastanza rilevanti:
  - Prevalenza delle versioni 4.1-4.2 Jelly Bean installate sul 40% dei device
  - La versione Icecream Sandwich sul 22,5% dei device
  - La versione Gingerbread sul 33% dei device
- Nonostante:
  - Il codice di **Android sia Open Source**
  - **Esiste la compatibility Test Suite** che automatizza il processo di testing.
  - **Le patch di sicurezza portano benenfici notevoli** contro la compromissione del device

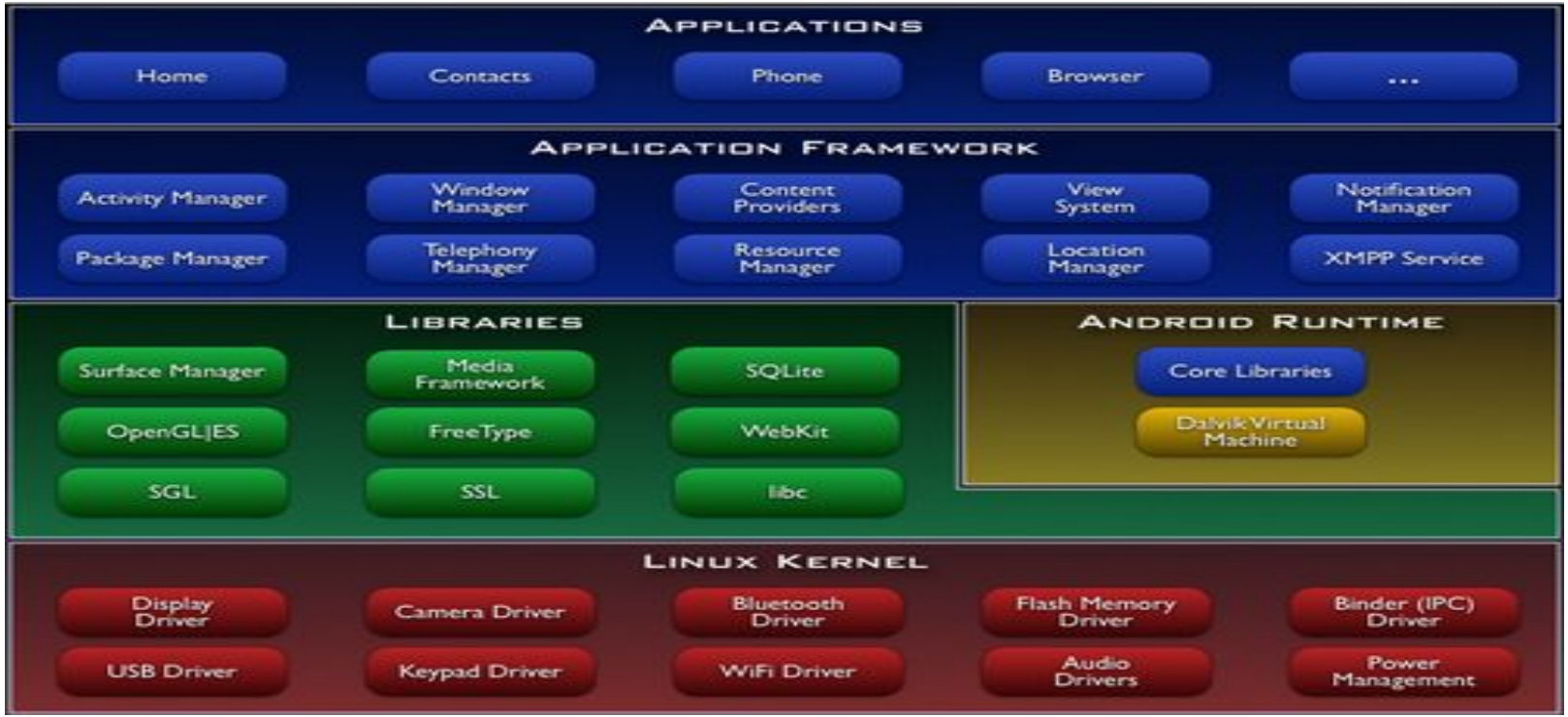
Frammentazione e Fattori di rischio



Rooting del device

- **Rooting** acquisizione dei permessi di amministratore sul device:
  - **Vantaggi**:
    - Sostituzione della ROM originale di bloatware (e.g., software promozionale)
    - Regolazione del comportamento hardware (frequenza CPU, voltaggi)
    - Installazione della ROM custom per personalizzare il telefono
  - **Svantaggi:**
    - L'ottenimento dei privilegi di amministratore implica che tutte le applicazioni installate nello smartphone abbiano questo privilegio.
    - Implicazioni serie in caso di attacco.

Android Architecture



Android Applications

- Possiamo avere 4 tipologie di applicazioni:
  - **Activities**: Sono le classiche applicazioni utente definite attraverso un interfaccia grafica.
  - **Broadcast receivers:** Sono applicazioni che richiedono l'intento di ricevere particolari messaggi per esempio ricevere un SMS da internet (e.g, android.provider.Telephony.SMS_RECEIVED)
  - **Content Providers**: Sono applicazioni che condivido certi dati tra piu` applicazioni, per esempio dati relativi ai propri contatti della rubrica telefonica oppure del calendario.
  - **Services**: Sono applicazioni che senza UI che vengono eseguite in background. Per esempio SmsReceiverService and BluetoothOppService, vengono utilizzate come servizi esterni alle app, ma possono comunicare con le app attraverso IPC.

Android Sandbox

- Il modello adottato da Android per la gestione dei permessi sui file e` quello simile a UNIX, dove **ogni applicazione ha i propri permessi** (e.g. UID e GID)
- La gestione della Sandbox viene assicurata dai classici meccanismi implementati nei sistemi operativi moderni, **Linux implementa anche permessi piu` fine-grained come le capabilities**.

Permessi App Android

- Ogni componente dispone di una **access permission label** e ogni applicazione all'atto dell'installazione richiede un **elenco di permission label necessarie**.
- Una volta accettati i permessi da parte di un utente su un' applicazione (e.g. definiti nel file Androidmanifest.xml) , **il sistema controlla a runtime attraverso il sistema di esecuzione (e.g, DalvikM) e attraverso i permessi dell'applicazione definite dai permessi UNIX l'accesso alle risorse.**

Permessi Android

Enforcement of Android Permission

I permessi Android sono controllati da 3 principali oggetti:

- **API Permission**: Sono quelle che vengono controllate per l'accesso alle risorse tramite Android Framework. (e.g., READ_PHONE_STATE, INTERNET) questo permesso viene controllato a runtime.
- **File System Permission**: Questi permessi vengono utilizzati per l'implementazione della sandbox (DAC permission) e servono per definire l'accesso alle risorse sul filesystem.
- **IPC Permission:** Sono quelle relative alla comunicazione tra le differenti app.

IPC Permessi Android

- IPC permission: **ogni componente** (e.g. activity, service, content provider etc.) definisce un **access permission label.**

- Quando un processo vuole comunicare con un altro processo deve utilizzare il **protocollo Binder.**

- A questo punto Il chiamante puo` controllare i permessi del chiamato attraverso due principali funzioni:

  - **CheckCallingPermission(String permission)**
  - **CheckCallingPermissionOrSelf(String permission)**

Caso Kakao Talk

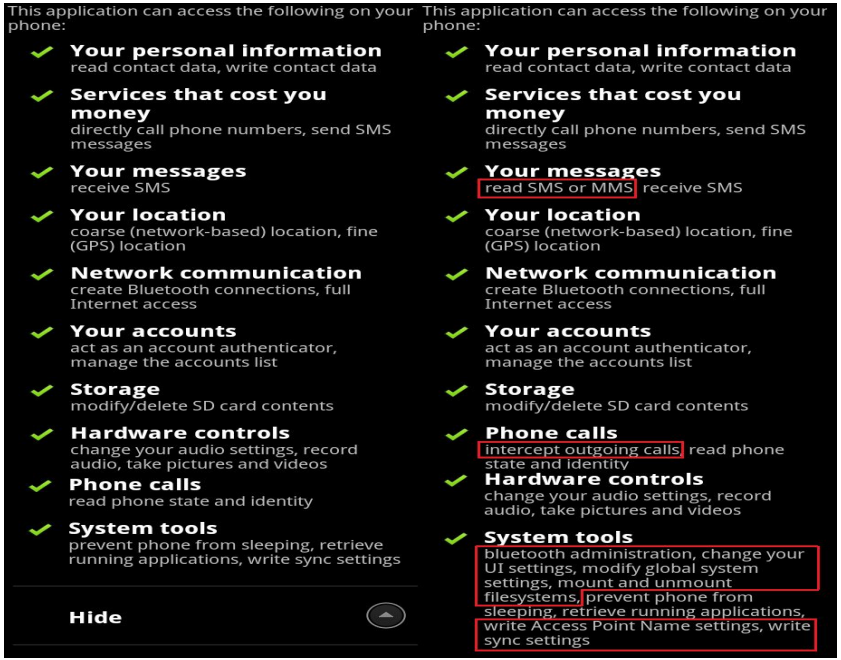
- Effettuare il **repack di un'applicazione famosa** per poi offrirla gratis e` il metodo piu` veloce per rubare dati personali.

- **Kakao Talk e` un client di messaggistica istantanea** in voga nella comunita` tibetana ed e` stato oggetto di un attacco volto a rubare i dati personali

- **Il vettore di infezione e` stato un messaggio di posta elettronica** contenente il pacchetto infetto.

Caso Kakao Talk

- Il malware richiedeva una notevole quantita` di permessi sul telefono:



Caso Kakao Talk

- L’applicazione memorizzava ad i**ntervalli regolari e in forma criptata la rubrica dell’utente, cronologia SMS e registro delle chiamate**.
- Dopo aver contattato un server **Command & Control scaricava informazioni come URL con credenziali**.
- Inoltre intercettava SMS che contenevano determinati codici e inviava coordinate GPS per **geolocalizzare in maniera precisa il device**.

Sistemi di Difesa: Google Bouncer

- **Google Bouncer** e` un sistema di scansione automatica delle applicazioni pubblicate
- Le varie tecniche di scansione applicate sono:
  - **Analisi statica** delle applicazioni per ricercare minacce note.
  - **Esecuzione del software in un emulatore virtuale**
  - **Analisi dei dati immessi nel sistema** da parte degli sviluppatori

Sistemi di Difesa: Google Bouncer

- **Google Bouncer** e` il sistema di difesa piu` sensibile visto che e` sulla linea di difesa principale per poter fermare gli attacchi:
  - **L'utente medio non e` in grado di discriminare l'importanza dei permessi**
  - **Google puo` disinstallare le app, ma questo avviene solo ad attacco avvenuto**
  - **Auditing automatico dell'applicazione non e` un sistema scalabile**

Security Analysis Google Bouncer

- **Google Bouncer** e` stato soggetto ad un'analisi di sicurezza da parte di alcuni ricercatori che he hanno dedotto il comportamento:
  - **Bouncer analizza l'applicazione mandata allo store immediatamente**
  - **Bouncer cerca nell'app dei pattern sospetti (e.g., /system/bin, /bin etc.)**
  - **Bouncer ha un interattore automatico dell'interfaccia grafica**
  - **Bouncer controlla la provenienza delle app solo da certi indirizzi di rete**
  - **Bouncer usa blocchi di rete ben definiti.**

Security Analysis Google Bouncer

- I ricercatori hanno poi realizzato **un'applicazione che sia in grado di eludere i meccanismi di controllo di Bouncer**:
  - L'applicazione disabilita le sue funzionalita` se riconosce il **blocco di rete adibito al bouncer**.
  - L'applicazione codifica alcune funzionalita` di Javascript in modo da **caricale dinamicamente da un server su internet**.
  - L'applicazione ha poi aggiunto funzionalita` **benigne che richiedono nuovi permessi.**

Security Analysis Google Bouncer

- Problematica del Bouncer:
    - **Code Coverage** non totale
    - Facilita` di **rivelazione dell'emulatore**
    - IP dispositivo **Blocchi fissi**.
- Migliorie del Bouncer:
    - Milgiorie nella **gestione dei permessi**
    - Potenziare **l'analisi statica**
    - Potenziare **l'analisi dinamica**

Security Enhanced Android SELINUX

- E` possibile inoltre progettare un sistema di permessi piu` sofisticato per le app android chiamata anche MAC. In particolare:
  - Nell'apprendimento dinamico delle policies L'esecuzione dell'applicazione viene rappresentata come un grafo, dove i **nodi sono gli stati d'esecuzione** e gli archi, passaggi da uno **stato associabili ad un particolare permesso.**
  - Questi permessi possono essere inoltre settati staticamente, la regola di default sui nuovi dispositivi e` quella di negare ogni permesso esplicito soprattutto per le applicazioni di livello system.

Security Enhanced Android SELINUX

- Il sistema avra` poi **due principali repository**:
  - **Permission Repository**: contenente i singoli permessi richiesti nonche` le sequenze di permessi insieme agli insieme dei permessi concessi dall'utente.
  - **Policy Repository:** Contiene i pattern (sequenze) riconosciuti come dannosi:
    - **RECORD_AUDIO -> INTERNET**
    - **READ_CONTACTS -> INTERNET -> SEND_SMS**

Security Enhanced Android SELINUX

- Il sistema comunque presenta alcune debolezze:
  - **Alcune sequenze dannose sono lecite** (e.g., Falsi Positivi)
  - **Necessita` di aggiornare policy repository**
  - **Non copre tutto lo spettro di attacchi**

CopperDroid

- Il sistema presenta una piattaforma per analisi dinamica di malware Android basato su 3 principi principali:
  - **Analisi a basso livello (OS kernel) e ad alto livello** (macchine Dalvik)
  - Stimolazione delle applicazioni per raggiungere tutte le entry point del sistema e consentire **code coverage**.
  - Testing su database di malware noti e **scansioni via web di applicazioni segnalate da utenti.**

CopperDroid

- Vantaggi principali:
  - Assoluta invisibilita` alle applicazioni in esecuzione nel guest
  - La non modifica del flusso d’esecuzione.
  - Analisi approfondita delle chiamate (IPC) tra le applicazioni