Storage Side Channel Attacks in Modern OS and Networking Stacks --- How to break isolation in OS?

Zhiyun Qian
University of California, Riverside

Outline

- Background and methodology
- Android UI state inference
- Off-path TCP sequence number inference
  - Firewall-middlebox-enabled attacks
  - Host-based attacks
- Summary

Outline

- Background and methodology
- Android UI state inference
- Off-path TCP sequence number inference
  - Firewall-middlebox-enabled attacks
  - Host-based attacks
- Summary

Side channels - Real world example Mafia game

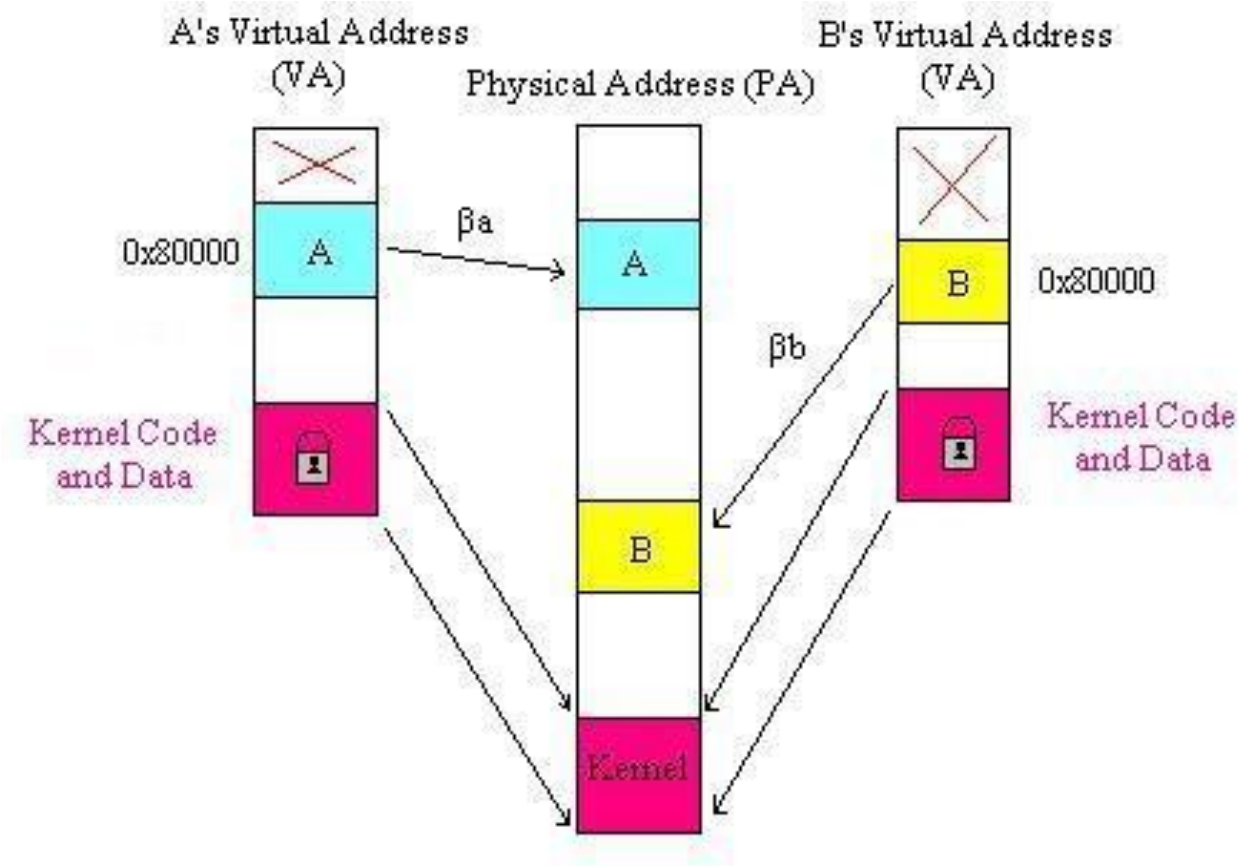
Another example

Anyone at home?

???

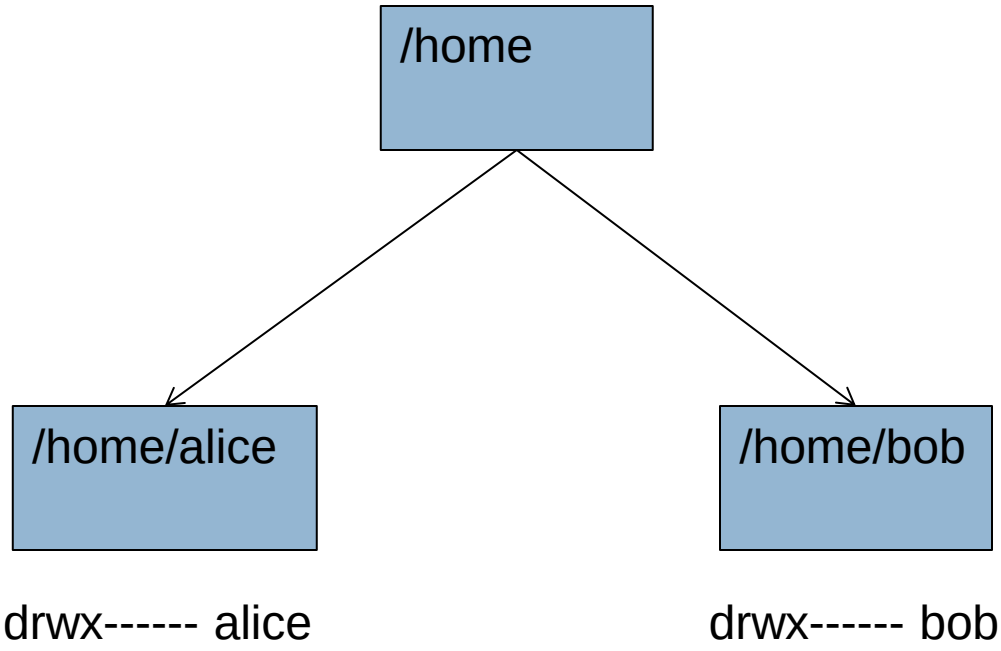
OS Security Mechanism -- Isolation

- Memory isolation



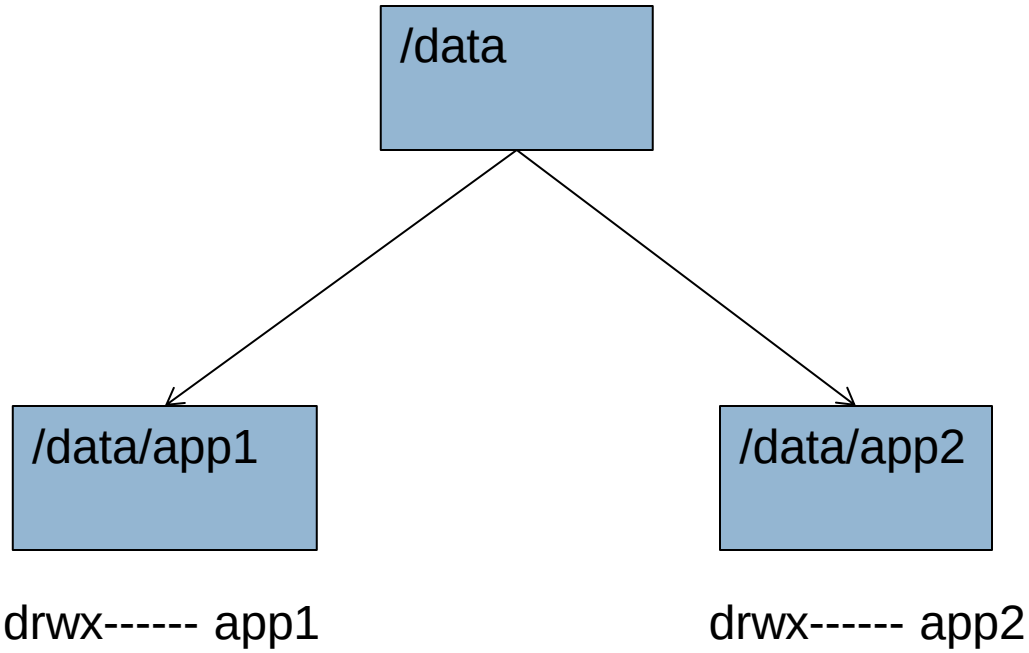
OS Security Mechanism -- Isolation

- File system isolation



OS Security Mechanism -- Isolation

- Android File system isolation



OS Security Mechanism -- Isolation

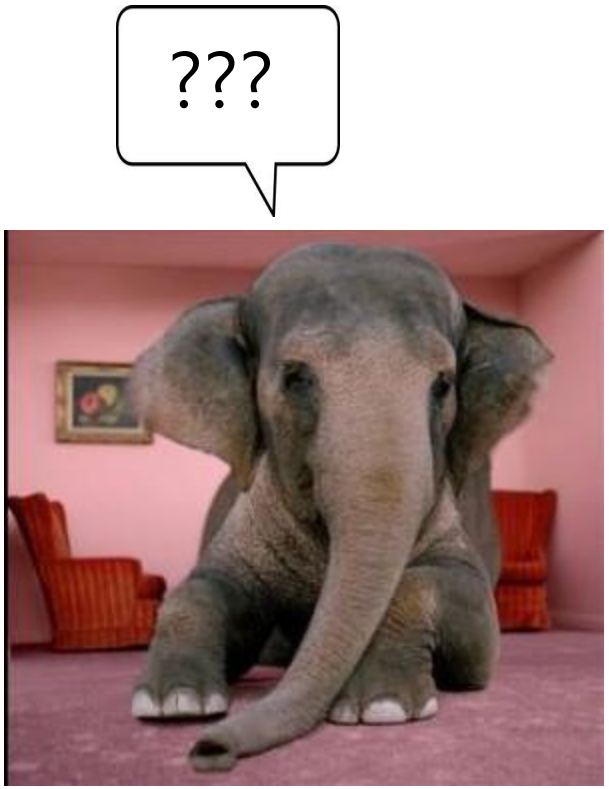
- Exceptions

  /proc/[pid]/statm

  /proc/net/netstat

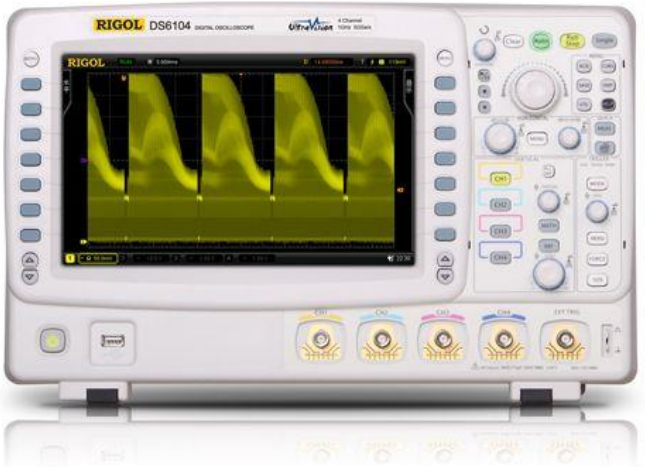
  Etc.

Breaking Isolation through Side Channel Attacks



What is a side channel attack?

- Information gained from the physical implementation of a cryptosystem, rather than brute force or theoretical weaknesses [1]
  - Timing, Power monitoring, Acoustic, Electromagnetic, etc.
  - Used as early as World War II.



[1] TEMPEST: A Signal Problem. Journal of Cryptologic Spectrum 1972

Modern side channel attacks

- Information gained from the ~~physical~~ design and implementation of a ~~crypto~~system, rather than brute force or theoretical weaknesses
  - Keystrokes (e.g., password) inference [Song01,Zhang09,Vuagnoux09,Chen10]
    - Timing, IPID, Power, Electromagnetic waves
  - Crypto key extraction through VM co-residency [Zhang12]
    - CPU cache

Timing vs. Storage side channels

- **Password authentication**

```
for(i = 0; i < len; i++)  {
    if(input[i] != password[i]) {
        failed = true;
        break;
    }
}
```

Timing

Timing vs. Storage side channels

- **Memory allocation**

```
secret_func() {
    malloc(1000KB);
    // … computation
    malloc(1000KB);
    // … computation
    malloc(1000KB);
    // … computation
}
```

```
zhiyunq@ubuntu:~$ ps ef -o pid,command,vsize,rss,size
  PID COMMAND                        VSZ   RSS  SIZE
 5302 bash XDG_SEAT_PATH=/org/fre   8248  4636  3064
 4708 bash XDG_SEAT_PATH=/org/fre   8292  4728  3108
19901  \_ ps ef -o pid,command,vs   4672   704   636
 4474 bash XDG_SEAT_PATH=/org/fre   8540  5016  3356
 2493 bash XDG_SEAT_PATH=/org/fre   8476  4916  3292
17255  \_ evince main.pdf SSH_AGE 207504 69888 91628
 2319 bash XDG_SEAT_PATH=/org/fre   8276  4708  3092
```

Storage

Research contributions

- Google
- Linux kernel
- FreeBSD kernel

Check Point
SOFTWARE TECHNOLOGIES LTD.

Global Sites | My Account

Home | Products & Services | Buy | Support | About Us

Support Center > Search Results > SecureKnowledge Details

Support Center

Search

Print Email

Check Point response to "Off-Path TCP Sequence Number Inference Attack"

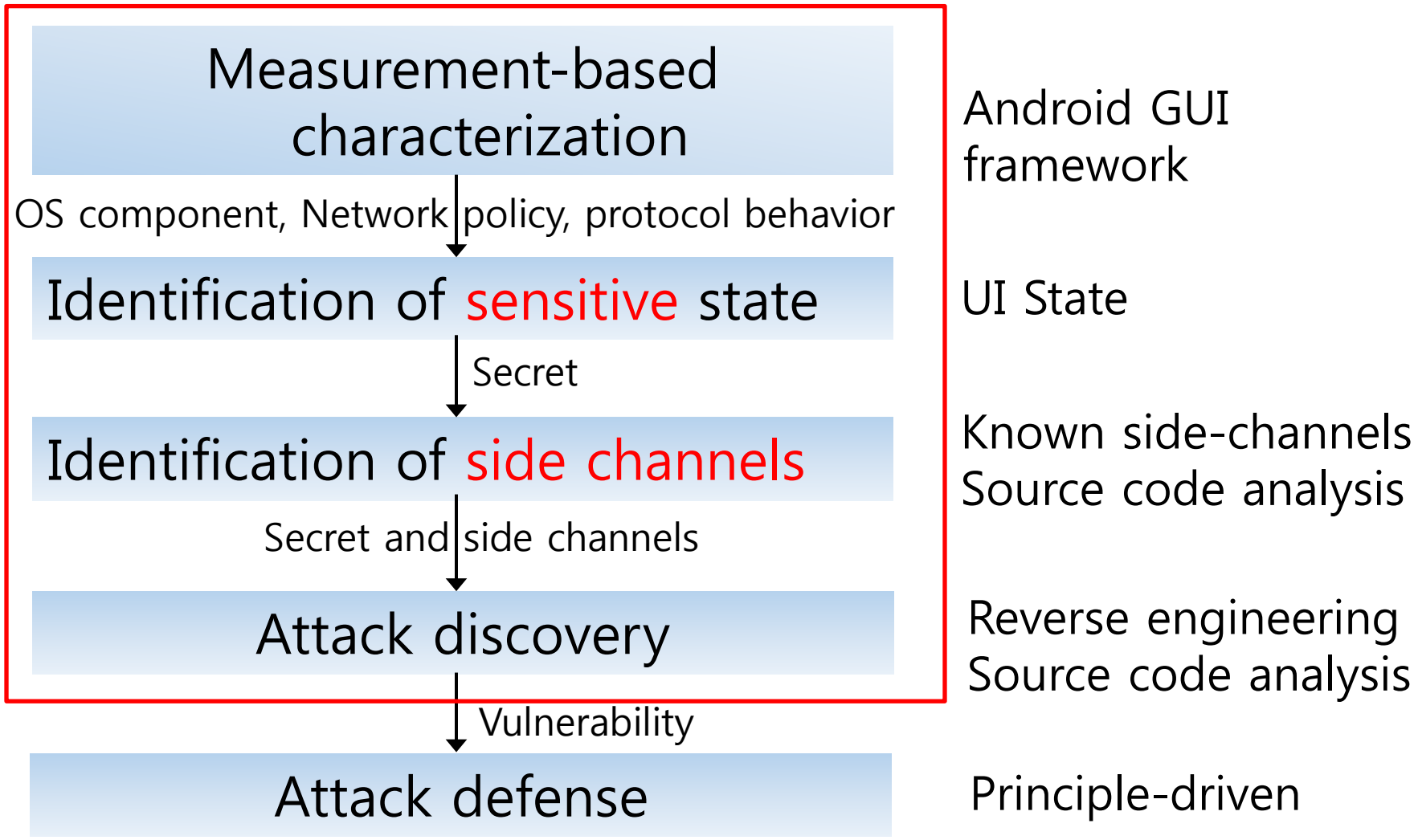
Solution ID: sk74640
Severity: Low
Product: Security Gateway
Version: All
Date Created: 24-May-2012
Last Modified: 14-Mar-2013

Rate this document
☆ ☆ ☆ ☆ ☆ [1=Worst,5=Best]

SYMPTOMS

- Researchers at the University of Michigan have published a paper "Off-Path TCP Sequence Number Inference Attack How Firewall Middleboxes Reduce Security".
- This attack identifies the current sequence range of a TCP connection, by exploiting the fact that firewalls drop out-of-window TCP packets. After the sequence range is identified, an off-path attacker may inject data or hijack the TCP connection.
- Client applications that use cleartext connections (e.g., HTTP and not HTTPS) are potential targets for these attacks.

Research methodology



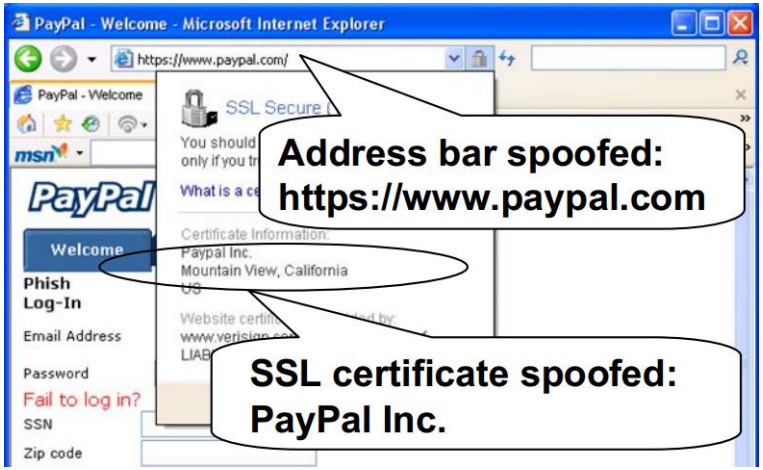
Outline

- Background and methodology
- Android UI state inference

  [USENIX SECURITY 14]
- Off-path TCP sequence number inference
  - Firewall-middlebox-enabled attacks
  - Host-based attacks
- Summary

Importance of GUI Security

- GUI content confidentiality and integrity are critical for end-to-end security
  - UI Spoofing in desktop/browsers[1]
  - Screenshot capture on Android without privilege[2]



[1]Chen, Oakland'07

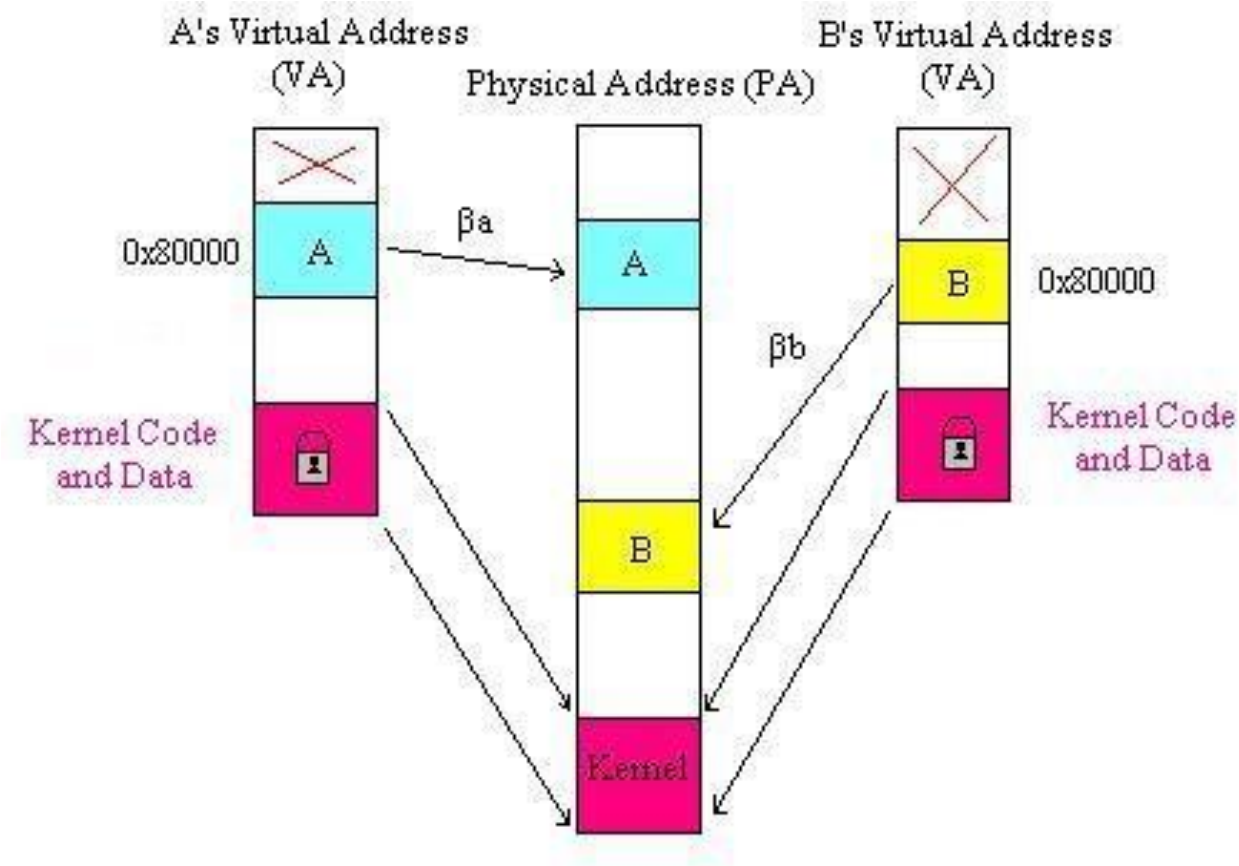
[2]ScreenMilker, NDSS'14

Android OS

- App no root privilege
- App can request limited permissions (users have to agree)
- Apps isolated from each other

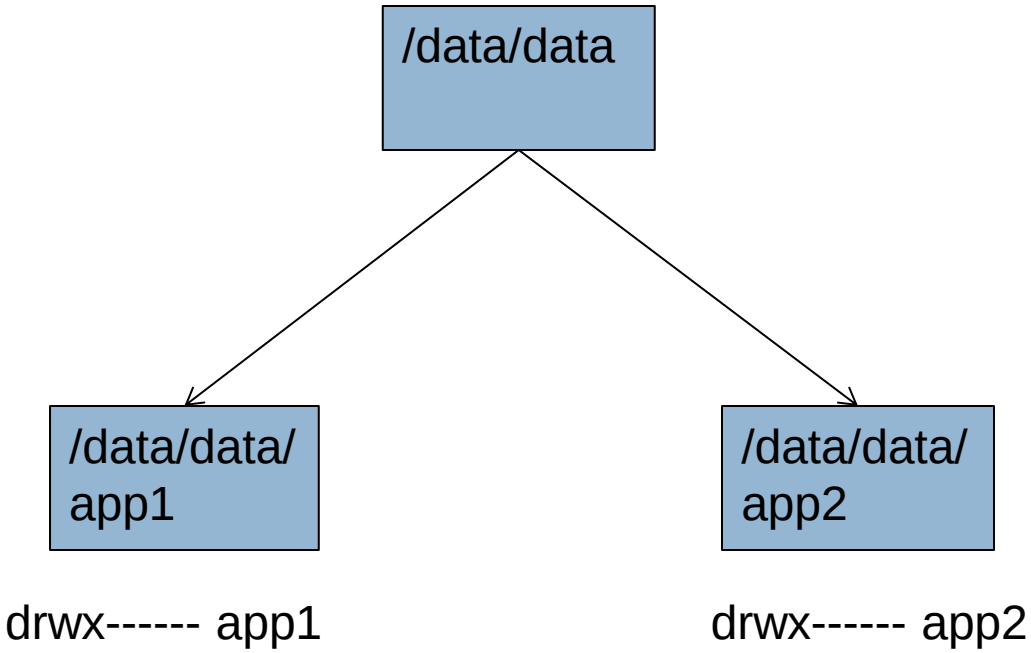
Android Security Mechanism -- Isolation

- Memory isolation



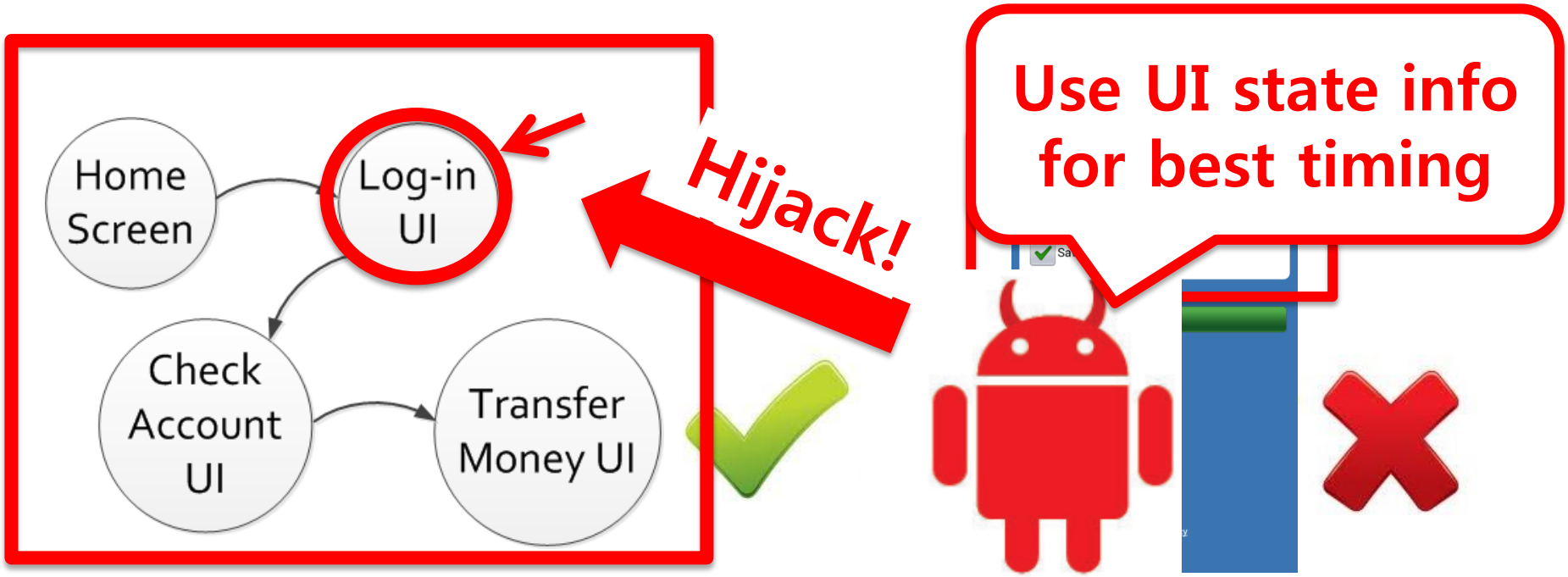
Android Security Mechanism -- Isolation

- File system isolation



Another Form of GUI Confidentiality Breach

- A weaker form
  - UI state an app is in (e.g., login state) **without knowing the exact pixels of the screen**



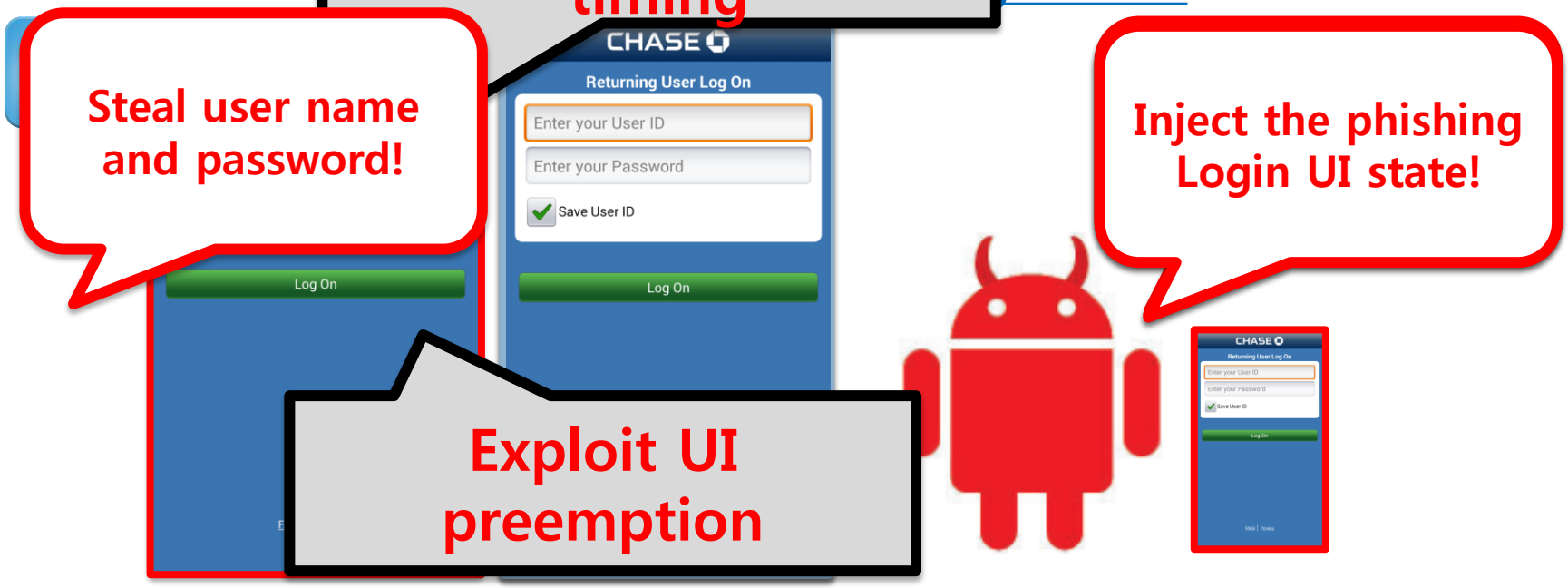
Serious security implications!

Enabled Attack: UI State Hijacking

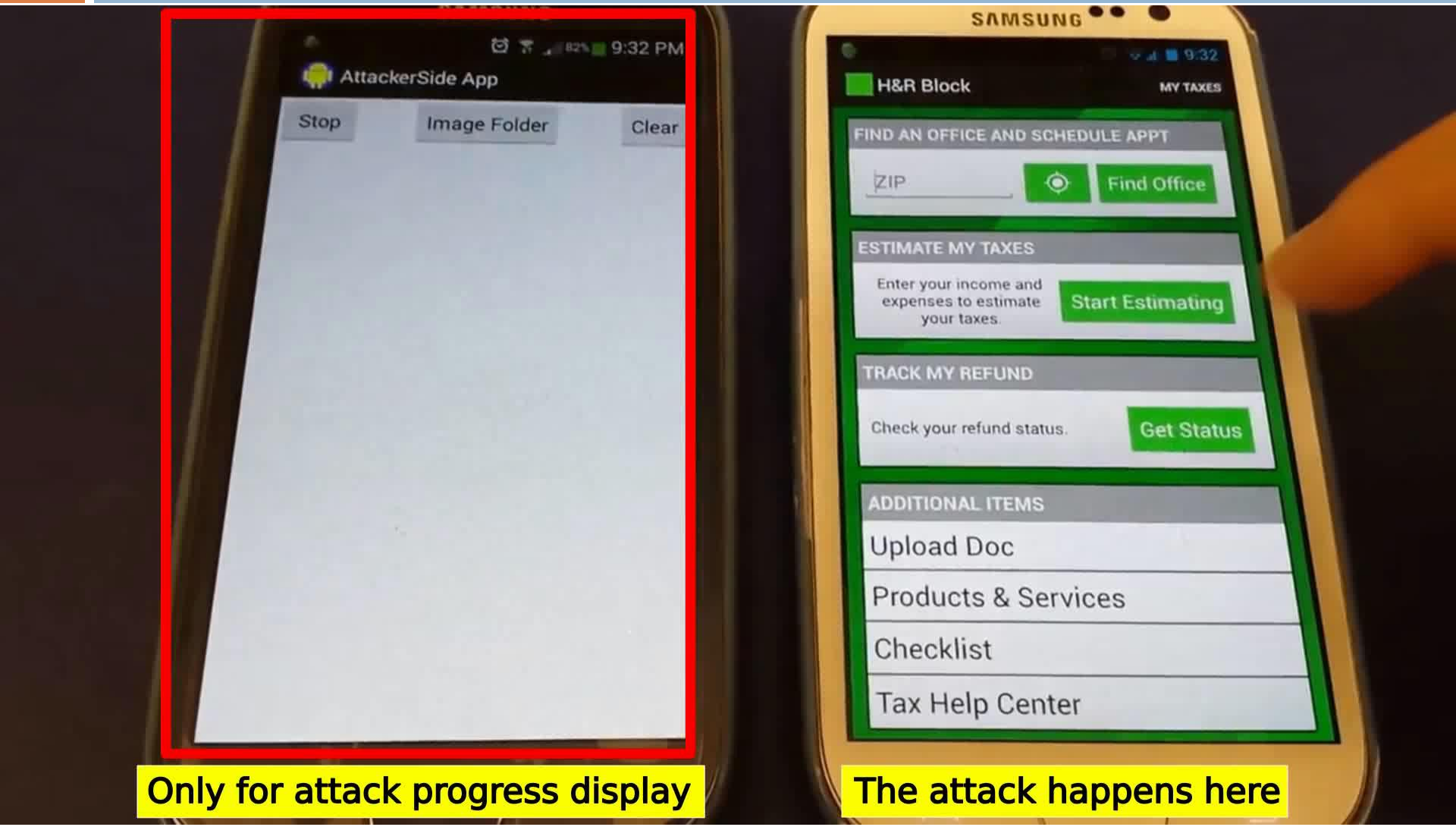
- Hijack sen… l private input

No glitches as we **disable the animation** + **precise attack timing**

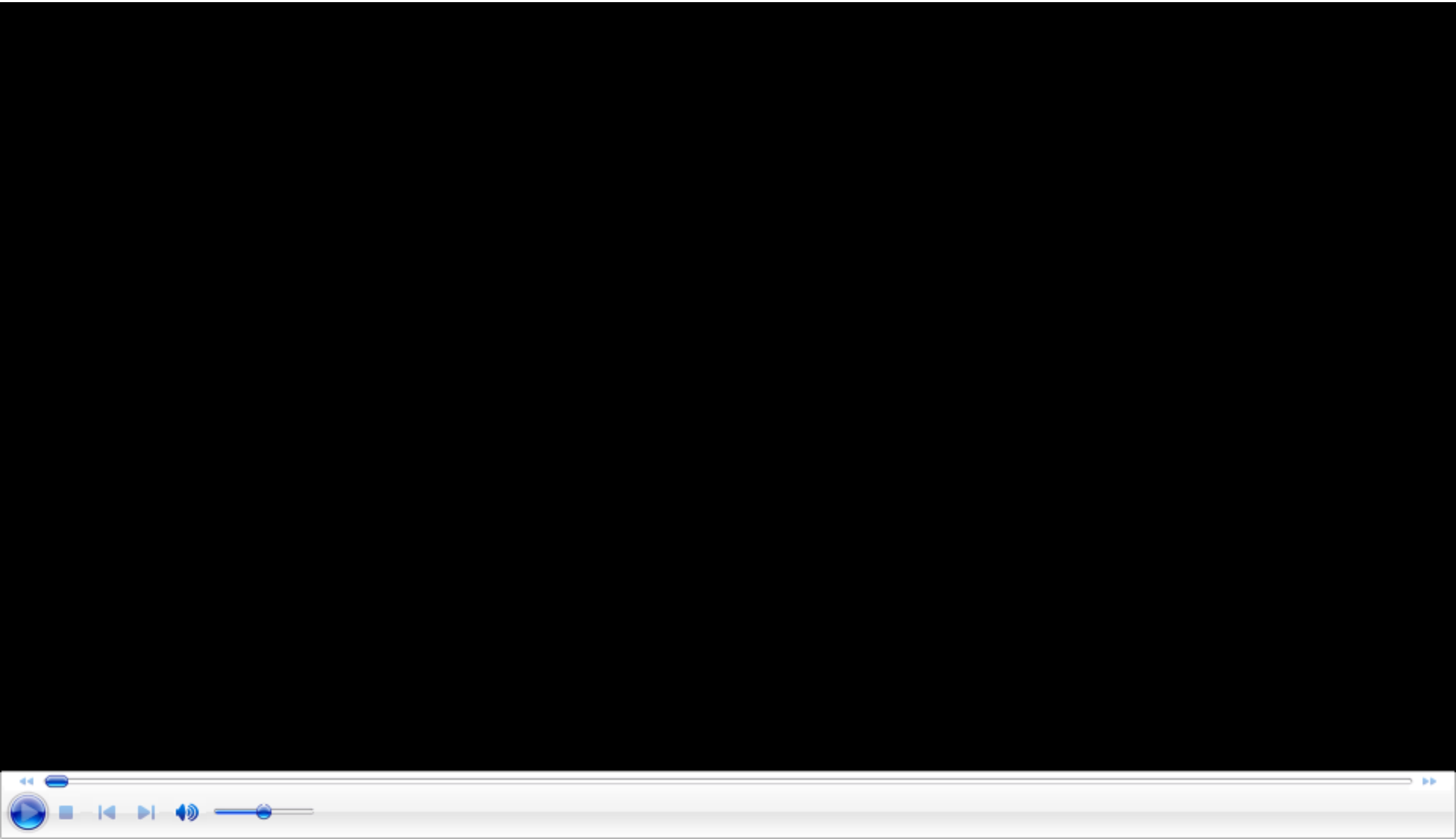
Foregr… …round:



UI State Hijacking Attack Demo



Camera Peeking Attack Demo



UI State Leakage is Dangerous

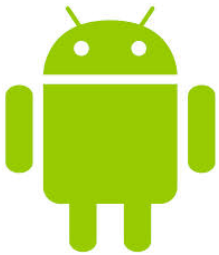
- Lead to both GUI **integrity** and **confidentiality** breaches
- UI state information **is not protected well**
  - An **unprivileged application** can track another app's UI states in real time

UI State Inference Attack

- **UI state**: a mostly consistent UI **at window level** for certain functionality (e.g., log-in)
  - On Android: **Activity (full-screen window)**
- Also called **Activity inference attack**
  - An unprivileged app can infer the foreground Activity in real time
  - Requires **no permission**

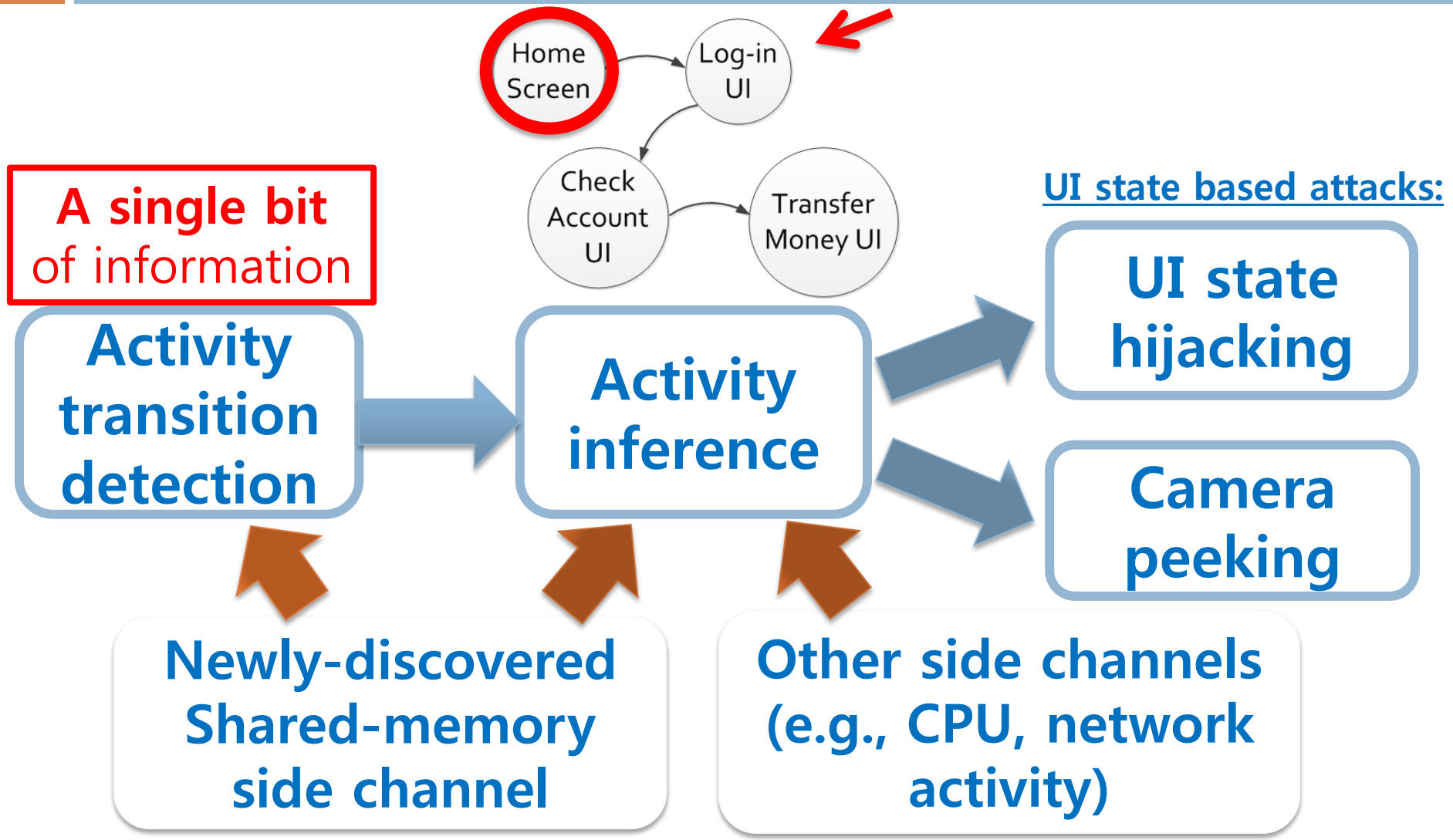
Underlying Causes

- **Android GUI framework design** leaks **UI state changes** through **a publicly-accessible side channel**
  - A newly-discovered shared-memory side channel
  - Affects nearly **all popular OSes**



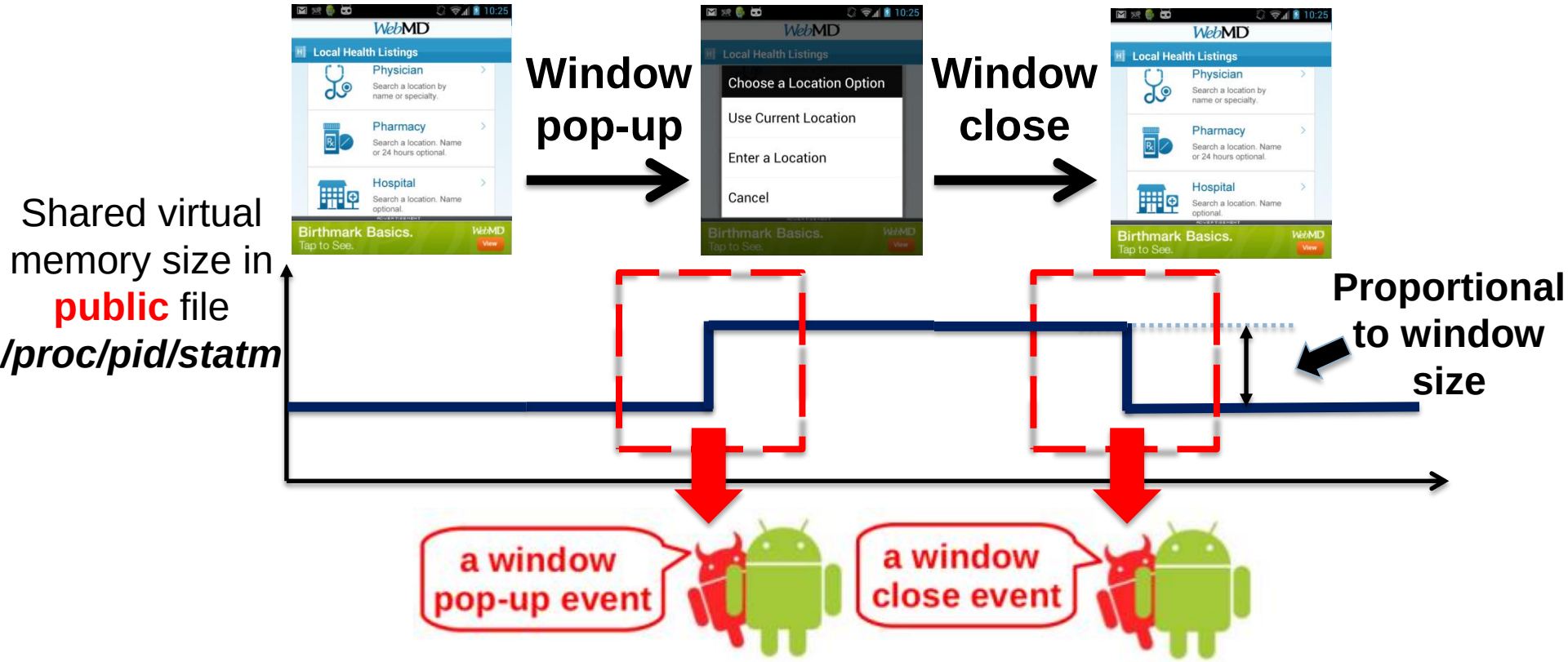


Attack General Steps



Shared-Memory Side Channel

- **Finding**: shared virtual memory size changes are correlated with Android window events

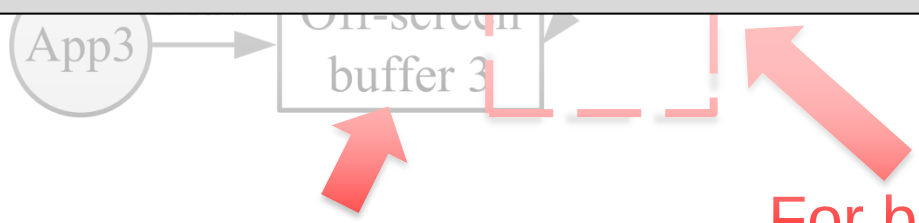


Shared-Memory Side Channel

- Root cause for this correlation

Confirmed that **shared memory is used in GUI design for many OSes,** including





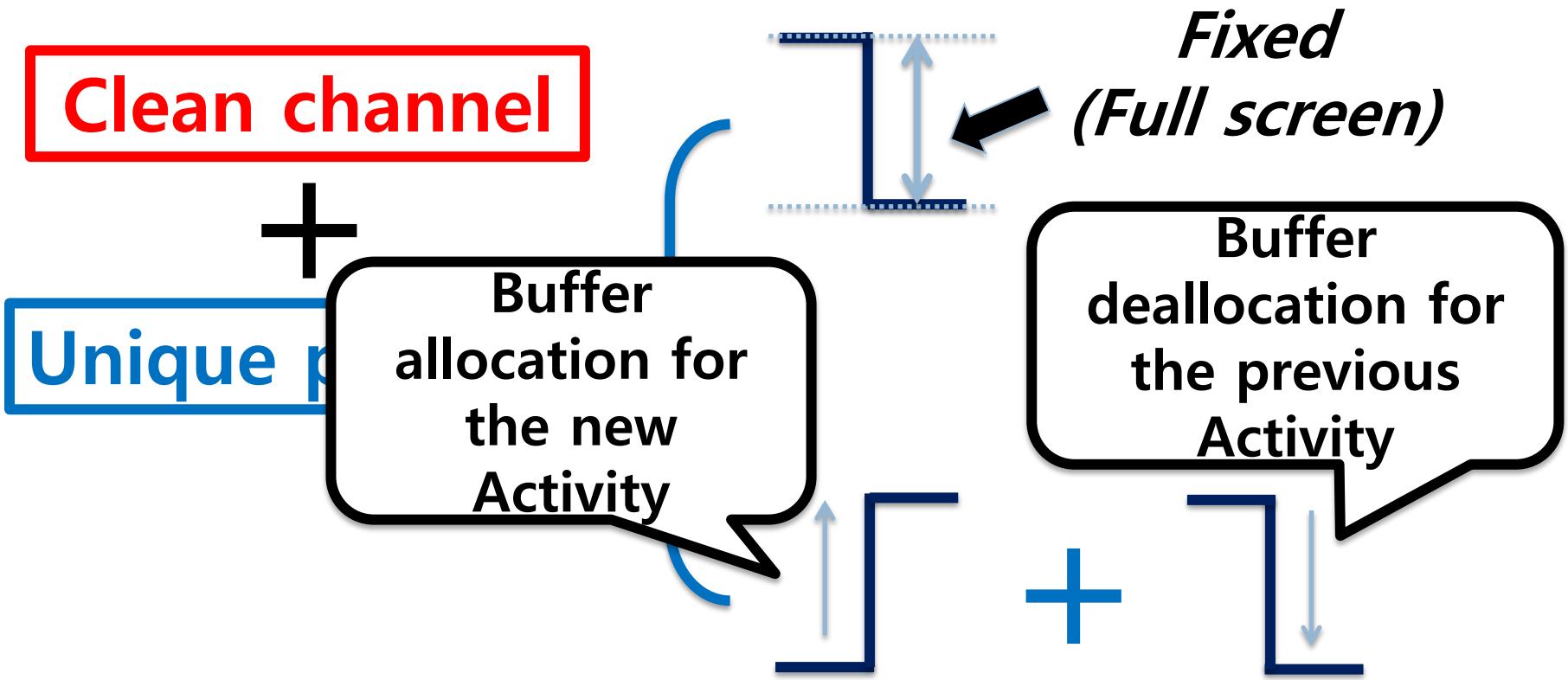
The changed size is the off-screen buffer size

For better UI drawing performance, Android uses **shared memory** as IPC

The root cause is here

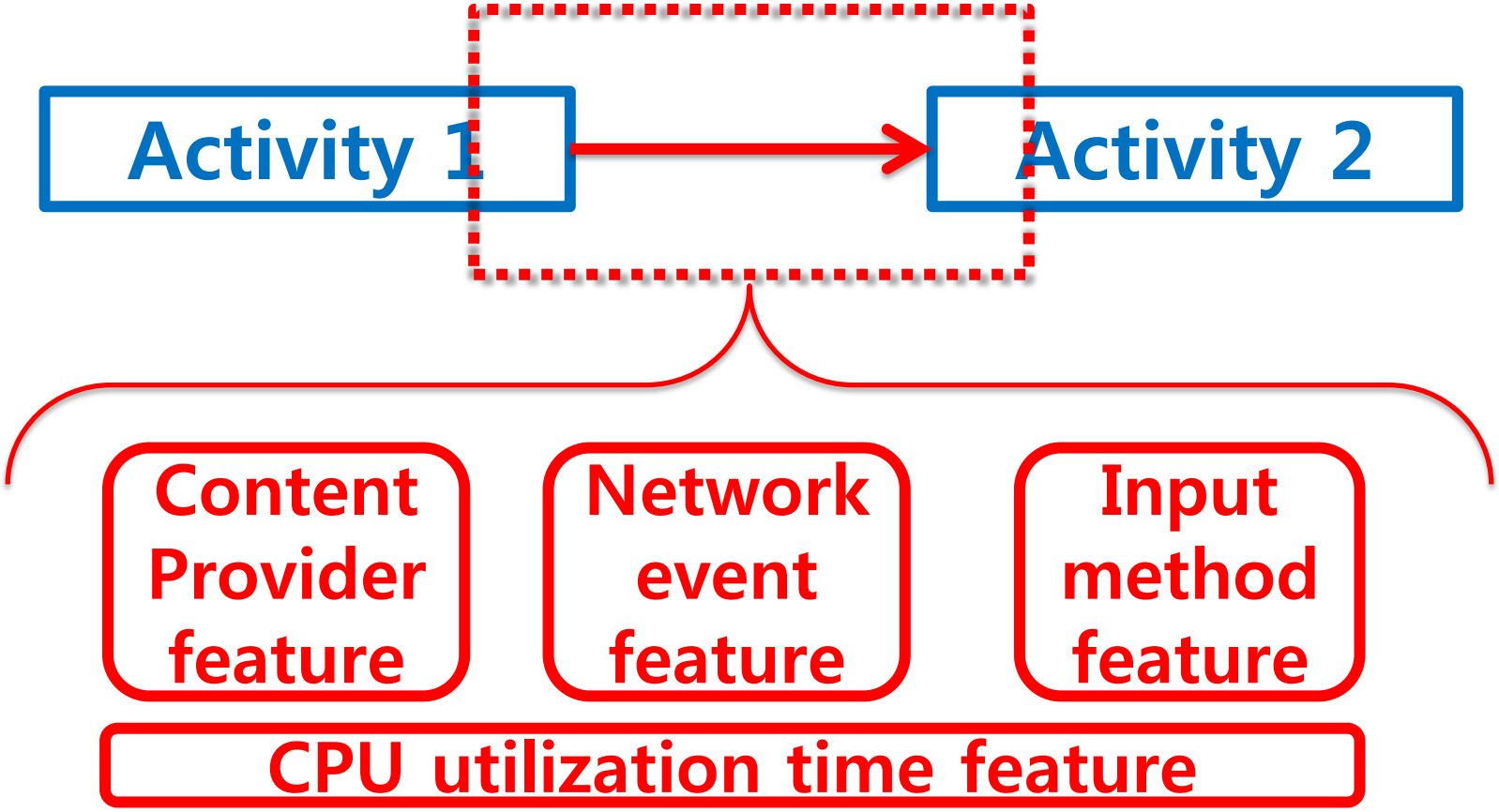
Activity Transition Detection

- Detect shared-memory size change pattern
  - **Nice properties**:



Activity Signature Design

- Consists of various features



Evaluation Methodology

- **Implementation**: ~ 2300 lines of C++ code compiled with Android NDK
- **Data collection**: using automated Activity transition tool on Samsung Galaxy S3 devices with Android 4.2
- **Experimented on 7 popular Android apps:**



Evaluation Results

- **Activity transition detection**, for all apps
  - Detection accuracy **≥ 96.5%**
  - FP and FN rates both **≤ 4%**
- **Activity inference accuracy**
  - **80–90%** for 6 out of 7 popular apps
    - **Important features**: CPU, network, transition model
- **Inference computation & delay**
  - Inference computation time: **≤ 10 ms**
  - Delay (Activity transition → inference result): **≤ 1.3 sec**
    - Improved to **≤ 500 ms** for faster and more seamless Activity hijacking
- **Power overhead**
  - **2.2–6.0%**
- **Status**
  - **Working with Google now to fix the problem**