ANTI ANTI-CODE MODIFICATION

Boonpoj Thongakaraniroj, *Secure D Center*

MiSSConf [SP5]
6 July 19

About Secure D

We operate across the ASEAN region with offices established in **Kuala Lumpur, Malaysia** and **Bangkok, Thailand**. We are independent and committed to your long-term goals by bringing a fresh, independent perspective, high passion to do an outstanding job and delivering cost-effective, innovation and high value services using global methodology and framework with our best cyber expertise certified by well-known cyber security certifications. We trust you will recognize our pragmatic 'hands on' style in the way we have structured our team, our approach and our deliverables. Our approach is based on the simple notion that the success of this project is measured by the results obtained and not just successful completion.

Introduction to Anti-Code Modification

SecureD

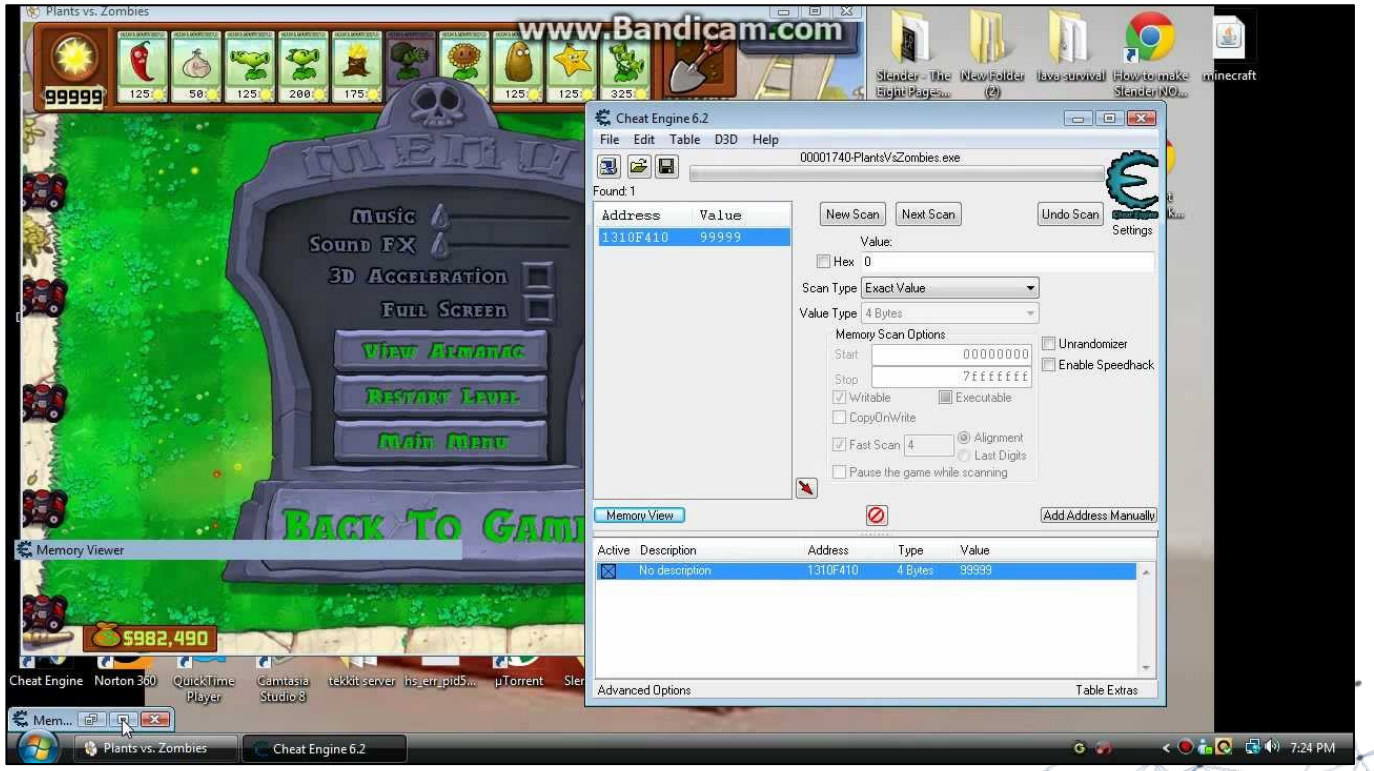
Introduction to Anti-Code Modification

Code Modification in Context of Mobile Application

What is Mobile Application ?

- A mobile app or mobile application is a **computer program** or **software application** designed to run on a mobile device such as a phone/tablet or watch.

- The PC software threats are applicable for Mobile application

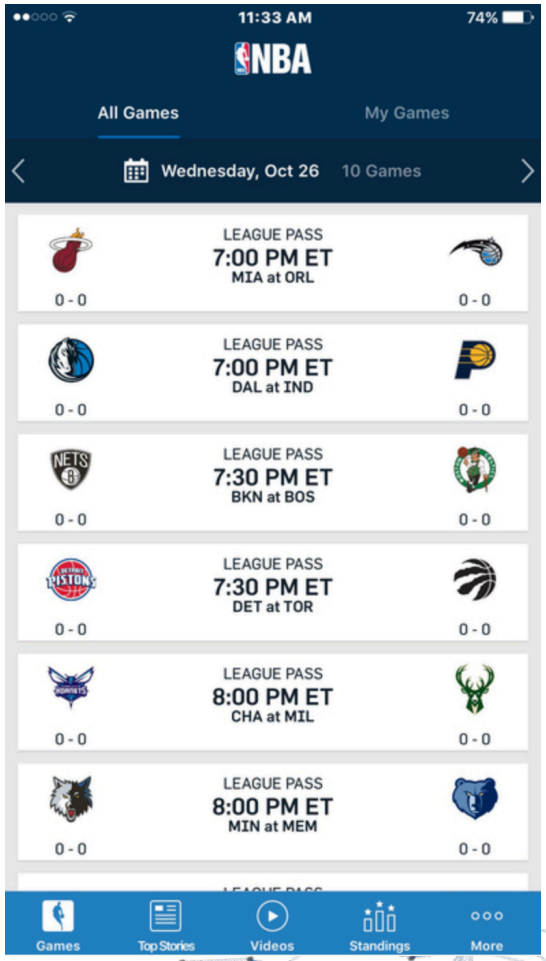


Introduction to Anti-Code Modification

Case study for Binary patching (Code Modification)

NBA Hack Online

- Watch live regular season NBA games all season long with the purchase of a NBA LEAGUE PASS subscription (Blackout restrictions apply).
- See schedules to check live scores, game times, and in-game stats.
- Customize your experience by following your favorite teams and players.
- Stay up-to-date with the latest NBA news and top stories.
- Watch top plays from around the league with video highlights and recaps.
- Follow league standings for the whole NBA.
- Listen to Live games
- CarPlay support



Introduction to Anti-Code Modification

Case study for Binary patching (Code Modification)

BIGO LIVE Hack Features

1.) Unlimited Diamonds

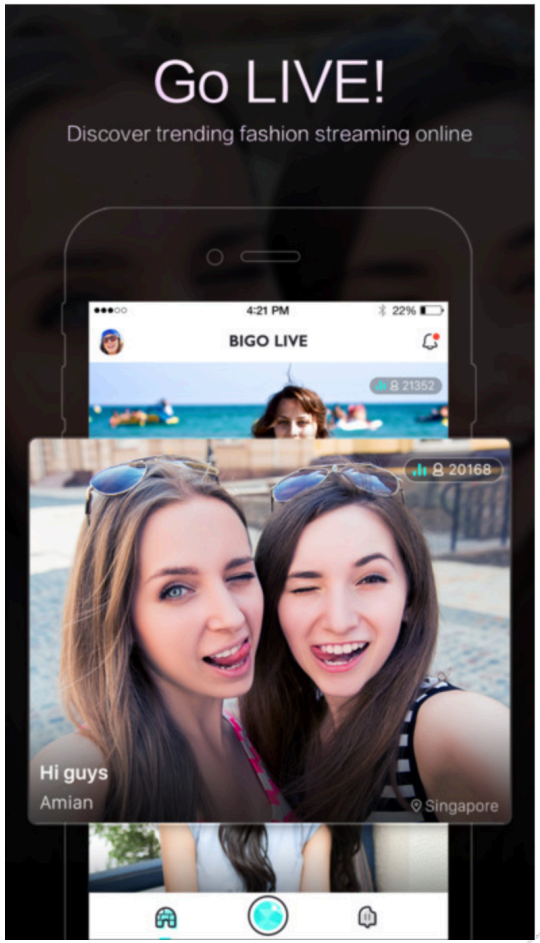
2.) 100% Safe and Working.

3.) No Jailbreak or root needed to use.

4.) Designed for iOS and Android devices.

5.) Daily Updates!

100% No-Virus Free and Fast Download Server! Guaranteed!



Introduction to Anti-Code Modification

Anti-Code Modification is the risk mitigration

Code Modification

❑ Static

- Understand app logic flow
- Modify app logic flow or malware in order to rebuild the app and contribute outside Google Play Store

❑ Dynamic

- Using Frida script to trace the application at runtime to determine loaded class/method in order to override method (Method Swizzling) for app manipulation
- Understand app at the run-time using debugging technique in order to manipulate the logic

Anti-Code Modification

❑ Static

- Code Obfuscation (Normally, Developer use ProGuard)
- Anti-Patching

❑ Dynamic

- Anti-Frida
- Anti-Debugging

Introduction to Anti-Code Modification

Dangerous Mindsets

- We don’t need to protect on server-side since the application is protected via Anti-code modification
- We have an End-to-end encryption on application, Nobody can manipulate the data over HTTP(s)
- We use ProGuard for binary obfuscation, Nobody can understand our application logic
- We have root detection, Rooted device cannot run our application
- We also have Anti-patching and Anti-Debugging solution, Nobody can modify or manipulate the data our application at the runtime
- We deploy Google SafetyNet then we are SAFE !!

Bypass Anti-Static Analysis Protection

SecureD

Bypass Anti-Static Analysis Protection

Code Obfuscation

Original Source Code

```java
package test;

import java.util.ArrayList;
import java.util.List;

public class Something {
    public static final String SOME_USEFUL_BIG_STRING_CONSTANT =
            "a45nv7s9j0s9a1b9v78xy";
    private List<Person> persons;

    public Something() {
        persons = new ArrayList<>();
    }

    public void superUsefulMethod(Person person) {
        persons.add(person);
    }

    public List<String> doSomethingUseful() {
        List<String> list = new ArrayList<>();
        for(Person p : persons) {
            list.add(p.doSomething());
        }

        return list;
    }
}
```

Minified using ProGuard

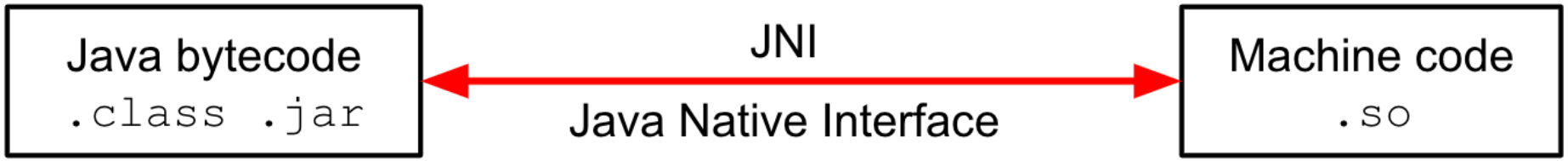
```java
package a;

import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;

public class b
{
  public static final String a = "a45nv7s9j0s9a1b9v78xy";
  private List<a> b = new ArrayList();

  public List<String> a()
  {
    ArrayList localArrayList = new ArrayList();
    Iterator localIterator = this.b.iterator();
    while (localIterator.hasNext()) {
      localArrayList.add(((a)localIterator.next()).e());
    }
    return localArrayList;
  }

  public void a(a parama)
  {
    this.b.add(parama);
  }
}
```

Bypass Anti-Static Analysis Protection

Native Code in Action

What is Native Code?

- ❑ C/C++ code is compiled to machine code ".so"
- ❑ Much faster than Java/Dalvik components
- ❑ **Hard to reverse more than on Java or Kotlin code**

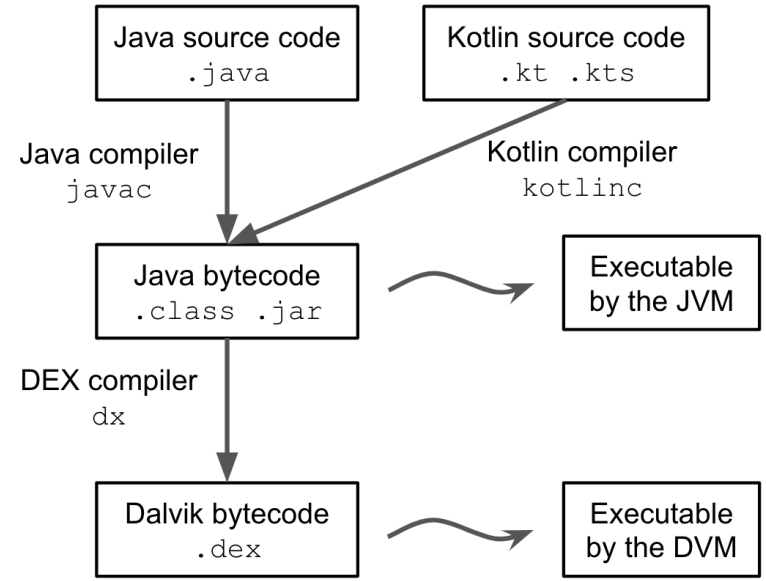


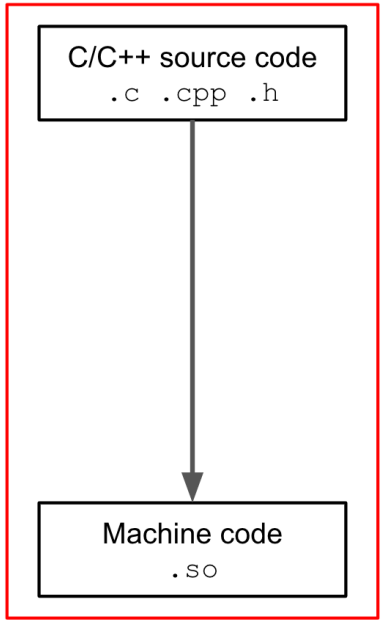
Source: https://developer.android.com/ndk/guides/concepts
https://docs.google.com/presentation/d/1r_DVzjCXw4gUum_WkRTzeivAnIArf8bw_g-rI0ywOSs/edit#slide=id.g47dd89f423_0_7

Bypass Anti-Static Analysis Protection

Native Code in Action

Native Code





- At compilation time
  - APK content
    - classes.dex
    - ...
    - lib/armeabi-v7a/native-lib.so
    - lib/x86/native-lib.so

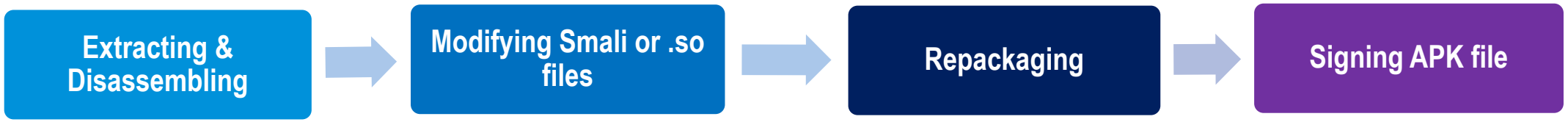
Source: https://developer.android.com/ndk/guides/concepts
https://docs.google.com/presentation/d/1r_DVzjCXw4gUum_WkRTzeivAnIArf8bw_q-rI0ywOSs/edit#slide=id.g47dd89f423_0_7

Bypass Anti-Static Analysis Protection

Anti-Patching

What is Patching ?

- ❑ Binary patching is one of the techniques used to alter the application functionalities.
- ❑ Binary patching on Android can be achieved by modification of disassembled code in .smali or .so file(s).



Bypass Anti-Static Analysis Protection

Anti-Patching

Instruction of Intel x86

Instruction	Description	signed-ness	Flags	short jump opcodes	near jump opcodes
JO	Jump if overflow		OF = 1	70	0F 80
JNO	Jump if not overflow		OF = 0	71	0F 81
JS	Jump if sign		SF = 1	78	0F 88
JNS	Jump if not sign		SF = 0	79	0F 89
JE **JZ**	Jump if equal Jump if zero		ZF = 1	74	0F 84
JNE **JNZ**	Jump if not equal Jump if not zero		ZF = 0	75	0F 85
JB **JNAE** **JC**	Jump if below Jump if not above or equal Jump if carry	unsigned	CF = 1	72	0F 82
JNB **JAE** **JNC**	Jump if not below Jump if above or equal Jump if not carry	unsigned	CF = 0	73	0F 83
JBE **JNA**	Jump if below or equal Jump if not above	unsigned	CF = 1 or ZF = 1	76	0F 86
JA **JNBE**	Jump if above Jump if not below or equal	unsigned	CF = 0 and ZF = 0	77	0F 87
JL **JNGE**	Jump if less Jump if not greater or equal	signed	SF <> OF	7C	0F 8C
JGE **JNL**	Jump if greater or equal Jump if not less	signed	SF = OF	7D	0F 8D
JLE **JNG**	Jump if less or equal Jump if not greater	signed	ZF = 1 or SF <> OF	7E	0F 8E
JG **JNLE**	Jump if greater Jump if not less or equal	signed	ZF = 0 and SF = OF	7F	0F 8F
JP **JPE**	Jump if parity Jump if parity even		PF = 1	7A	0F 8A
JNP **JPO**	Jump if not parity Jump if parity odd		PF = 0	7B	0F 8B
JCXZ **JECXZ**	Jump if %CX register is 0 Jump if %ECX register is 0		%CX = 0 %ECX = 0	E3	

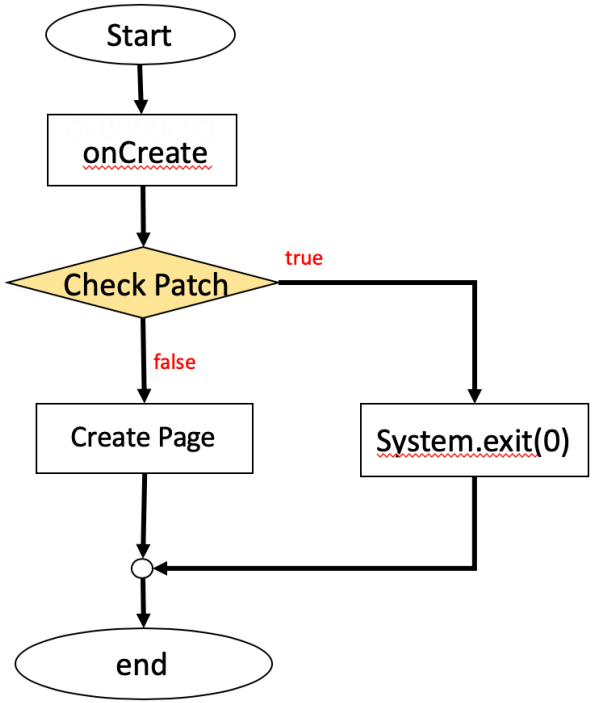
Bypass Anti-Static Analysis Protection

Anti-Patching

Example Code

```
1  boolean modified = false;
2  // required dex crc value stored as a text string.
3  // it could be any invisible layout element
4  long dexCrc = Long.parseLong(Main.MyContext.getString(R.string.dex_crc));
5  ZipFile zf = new ZipFile(Main.MyContext.getPackageCodePath());
6  ZipEntry ze = zf.getEntry("classes.dex");
7  if ( ze.getCrc() != dexCrc ) {
8    // dex has been modified
9    modified = true;
10 }else {
11   // dex not tampered with
12   modified = false;
13 }
```

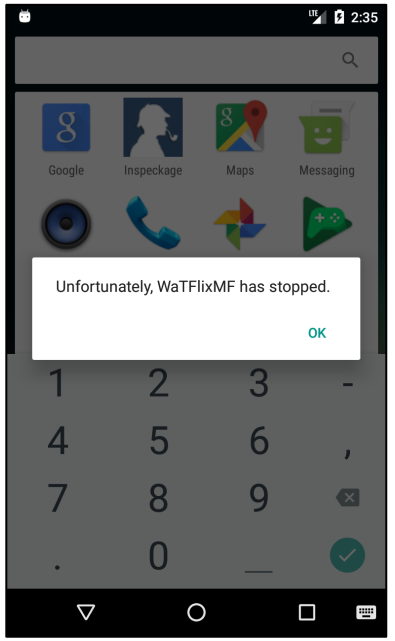
Flow of Application



Bypass Anti-Static Analysis Protection

DEMO

- ❑ Bypassing Root detection (RootBeer Library)
- ❑ Bypassing Code Patching Check



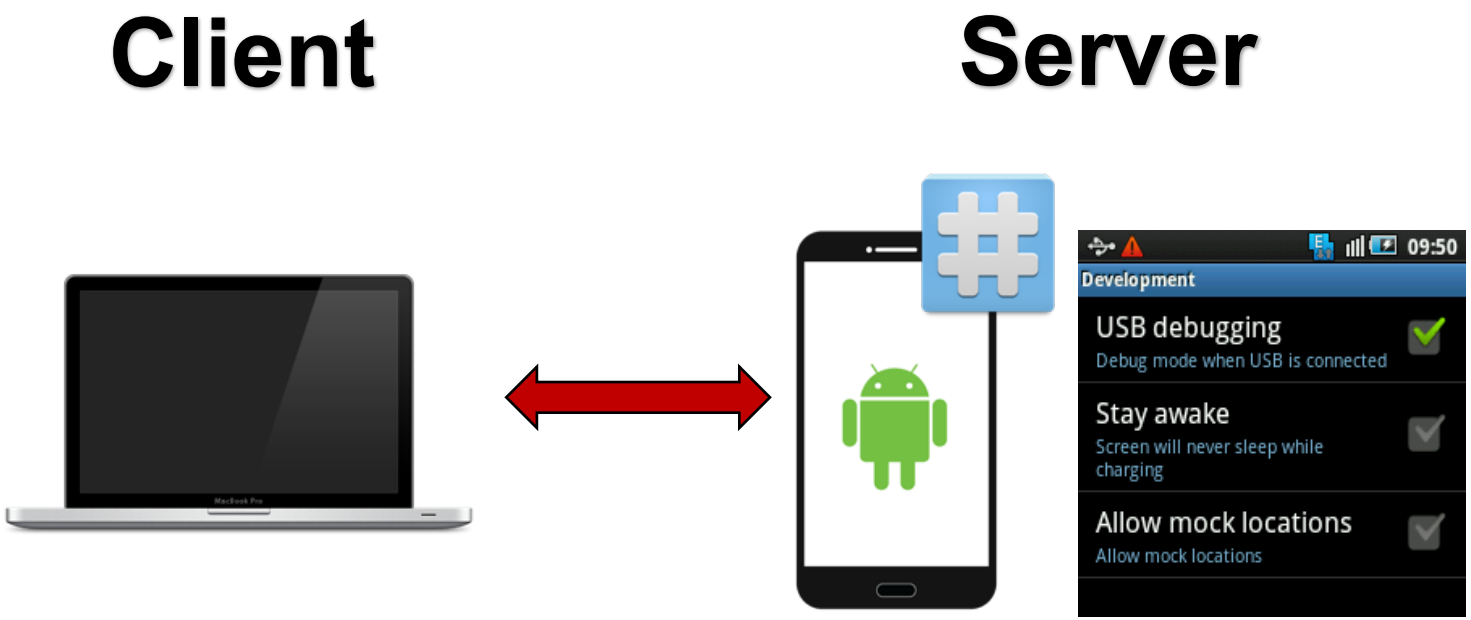
Bypass Anti-Dynamic Analysis Protection

Bypass Anti-Dynamic Analysis Protection

Anti-Frida

What is Frida ?

- Frida is hugely popular with Android reverse engineers, and for good reason: It offers runtime access to pretty much everything one could dream of, from raw memory and native functions to Java object instances.



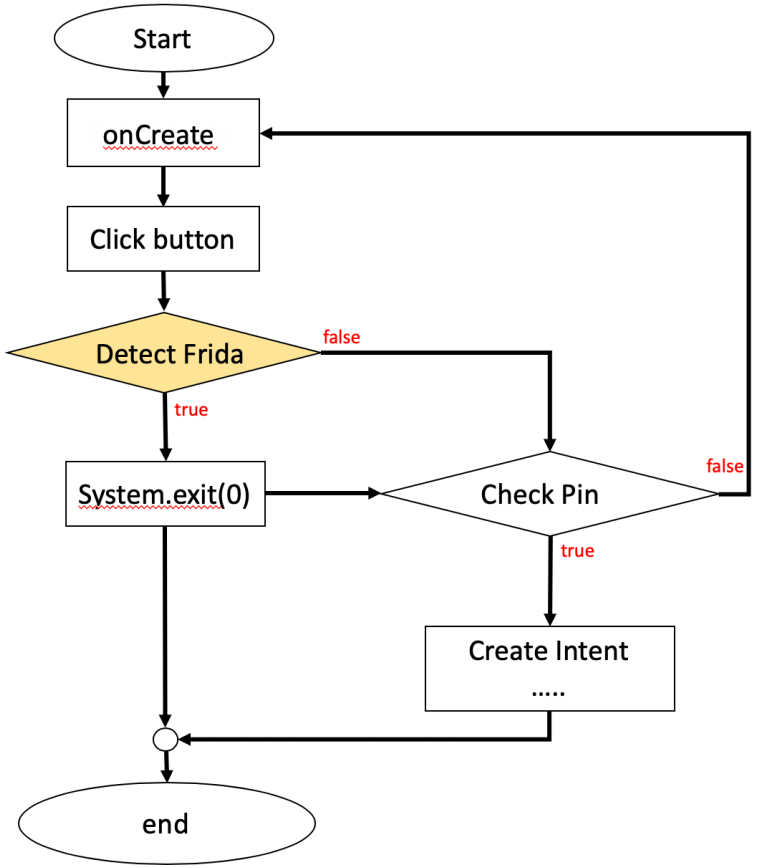
Bypass Anti-Dynamic Analysis Protection

Anti-Frida

Example Code

```
1  char line[512];
2  FILE* fp;
3  fp = fopen("/proc/self/maps", "r");
4  if (fp) {
5      while (fgets(line, 512, fp)) {
6          if (strstr(line, "frida")) {
7              /* Evil library is loaded. Do something… */
8          }
9      }
10     fclose(fp);
11 } else {
12     /* Error opening /proc/self/maps. If this happens, something is off. */
13 }
```

Flow of Application



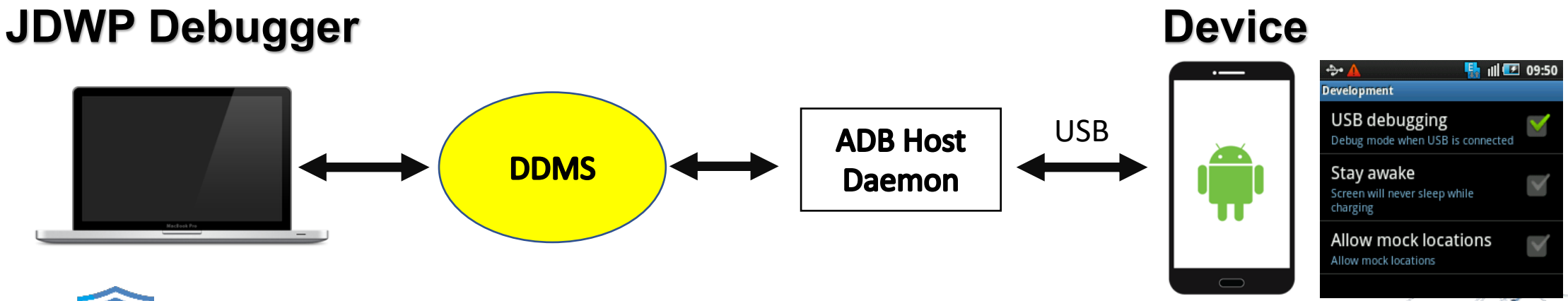
Bypass Anti-Dynamic Analysis Protection

Anti-Debugging

What is JDWP ?

- The Java Debug Wire Protocol (JDWP) is the protocol used for communication between a debugger and the Java virtual machine (VM)
- A JDWP debugger allows you to:
  - step through Java code
  - set breakpoints on Java methods
  - inspect and modify local and instance variables.

Source: https://docs.oracle.com/javase/8/docs/technotes/guides/jpda/jdwp-spec.html
https://github.com/OWASP/owasp-mstg/blob/master/Document/0x05c-Reverse-Engineering-and-Tampering.md



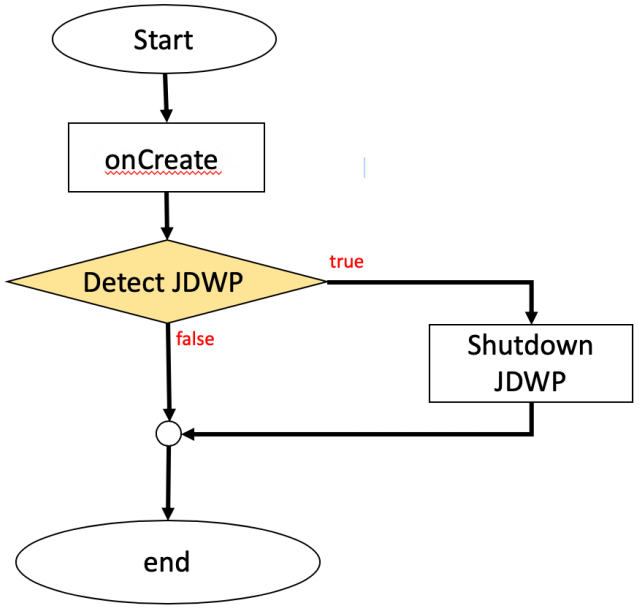
Bypass Anti-Dynamic Analysis Protection

Anti-Debugging

Example Code

```
void* lib = dlopen("libart.so", RTLD_NOW);
if (lib == NULL) {
    log("Error loading libart.so");
    dlerror();
}else{
    struct VT_JdwpAdbState *vtable = ( struct VT_JdwpAdbState *)dlsym(lib, "_ZTVN3art4JDWP12JdwpAdbStateE");
    if (vtable == 0) {
        log("Couldn't resolve symbol '_ZTVN3art4JDWP12JdwpAdbStateE'.\n");
    }else {
        log("Vtable for JdwpAdbState at: %08x\n", vtable);
        // Let the fun begin!
        unsigned long pagesize = sysconf(_SC_PAGE_SIZE);
        unsigned long page = (unsigned long)vtable & ~(pagesize-1);
        mprotect((void *)page, pagesize, PROT_READ | PROT_WRITE);
        vtable->ProcessIncoming = vtable->ShutDown;
        // Reset permissions & flush cache
        mprotect((void *)page, pagesize, PROT_READ);
    }
}
```

Flow of Application



Bypass Anti-Dynamic Analysis Protection

DEMO

- ❑ Bypassing Frida detection
- ❑ Bypassing Anti-Debugging detection

```
mobsec@mobsec-virtual-machine:~$ frida -U com.watf.watflixmf
     ____
    / _  |   Frida 12.4.0 - A world-class dynamic instrumentation toolkit
   | (_| |
    > _  |   Commands:
   /_/ |_|       help      -> Displays the help system
   . . . .       object?   -> Display information about 'object'
   . . . .       exit/quit -> Exit
   . . . .
   . . . .   More info at http://www.frida.re/docs/home/

[Android Emulator 5554::com.watf.watflixmf]-> Process terminated
```

```
mobsec@mobsec-virtual-machine:~$ jdb -attach localhost:12345
java.net.SocketException: Connection reset
        at java.net.SocketInputStream.read(SocketInputStream.java:210)
        at java.net.SocketInputStream.read(SocketInputStream.java:141)
        at com.sun.tools.jdi.SocketTransportService.handshake(SocketTransportService.java:130)
        at com.sun.tools.jdi.SocketTransportService.attach(SocketTransportService.java:232)
        at com.sun.tools.jdi.GenericAttachingConnector.attach(GenericAttachingConnector.java:116)
        at com.sun.tools.jdi.SocketAttachingConnector.attach(SocketAttachingConnector.java:90)
        at com.sun.tools.example.debug.tty.VMConnection.attachTarget(VMConnection.java:519)
        at com.sun.tools.example.debug.tty.VMConnection.open(VMConnection.java:328)
        at com.sun.tools.example.debug.tty.Env.init(Env.java:63)
        at com.sun.tools.example.debug.tty.TTY.main(TTY.java:1082)

Fatal error:
Unable to attach to target VM.
```

Google SafetyNet

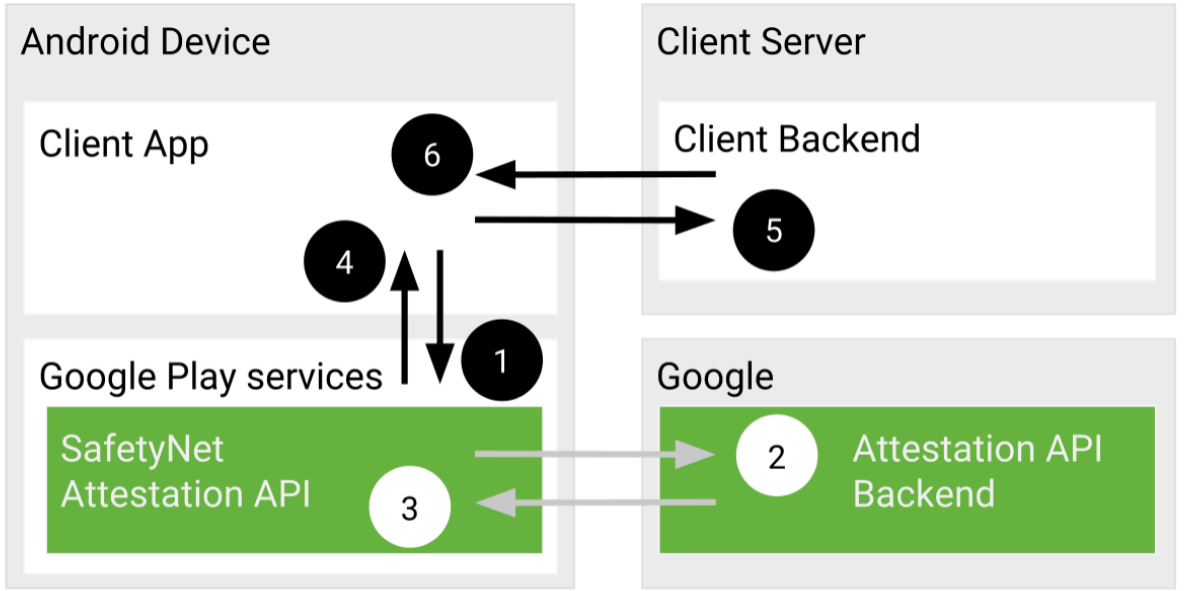
Google SafetyNet

Definition

What is Google SafetyNet ?

- SafetyNet provides a set of services and APIs that help protect your app against security threats, including **device tampering**, bad URLs, potentially harmful apps, and fake users.

- The SafetyNet Attestation API provides a cryptographically-signed attestation, **assessing the device's integrity**. In order to create the attestation, the API examines the device's software and hardware environment, looking for integrity issues, and comparing it with the reference data for approved Android devices. The generated attestation is bound to the nonce that the caller app provides. The attestation also contains a generation timestamp and metadata about the requesting app.



Source: https://developer.android.com/training/safetynet/attestation

Google SafetyNet

Anti-SafetyNet

SNetKiller *(https://github.com/iGio90/SNetKiller)*

- The app uses Frida to inject an agent into google services and prevent it to access certain files.
- com.google.android.gms

```
Interceptor.attach(Module.findExportByName(null, 'faccessat'), {
    onEnter: function(args) {
        const path = args[1].readUtf8String();
        log(path, 'faccessat');
        this.path = path;
    },
    onLeave: function(ret) {
        if (
            strstr(this.path, '/data/local') ||
            strstr(this.path, '/system')) {
            log('*filtered*', 'faccessat-ret');
            ret.replace(-1);
        }

        log(ret, 'faccessat-ret');
        return ret;
    }
});
```

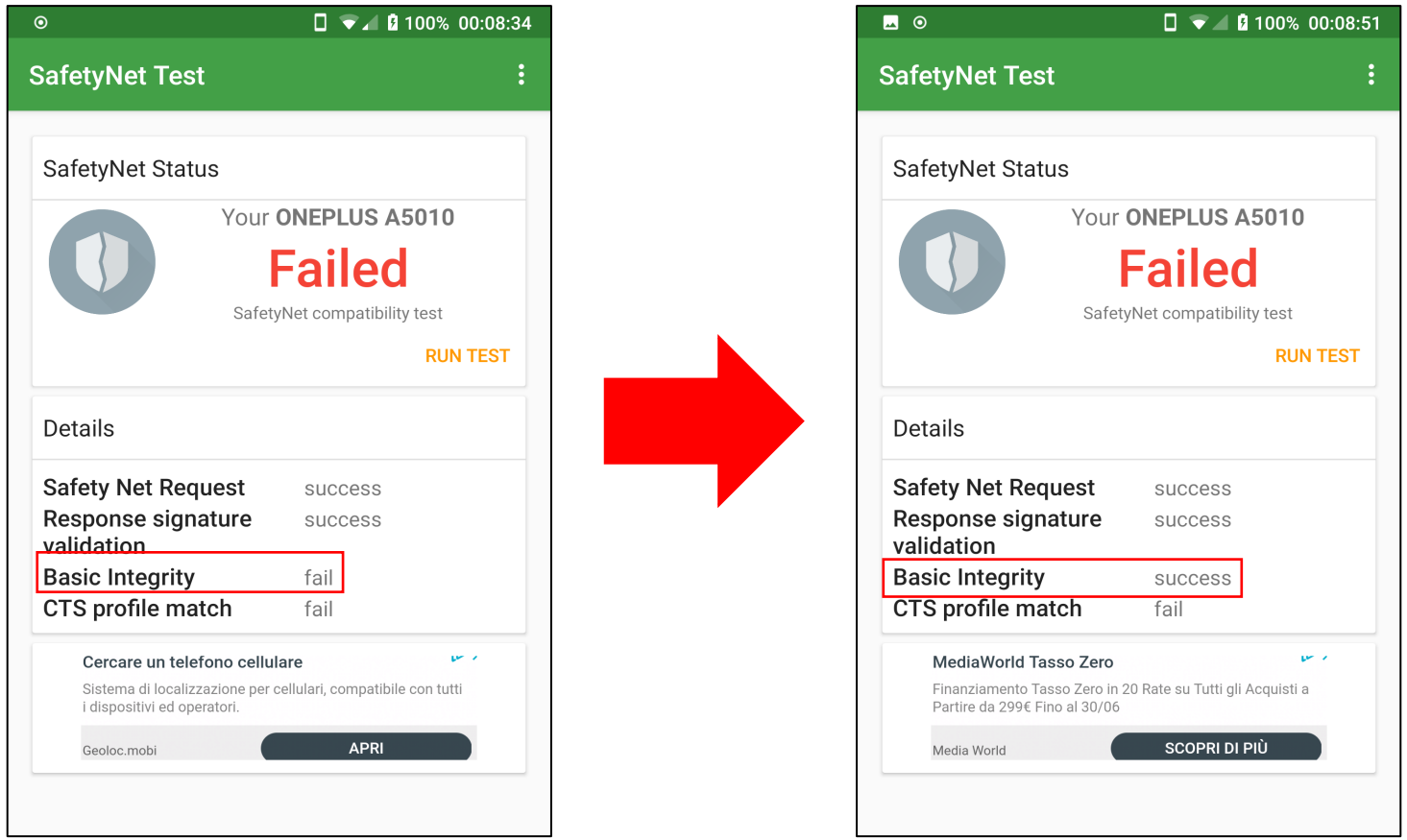
```
Interceptor.attach(Module.findExportByName(null, 'open'), {
    onEnter: function(args) {
        const path = args[0].readUtf8String();
        log(path, 'open');
        this.path = path;

        if (this.path.indexOf('.apk') >= 0) {
            this.path = this.path.replace('root', 'xxxx');
            args[0].writeUtf8String(this.path);
        }
    },
    onLeave: function(ret) {
        log(ret.toString() + ' - ' + this.path, 'open-ret');
        if (this.path === '/sys/fs/selinux/enforce') {
            selinuxFd = parseInt(ret);
        }
    }
});
```

```
Interceptor.attach(Module.findExportByName(null, 'stat64'), {
    onEnter: function(args) {
        const path = args[0].readUtf8String();
        log(path, 'stat64');
    },
    onLeave: function(ret) {
        ret.replace(-1);
    }
});
```

Google SafetyNet

Anti-SafetyNet

SNetKiller *(https://github.com/iGio90/SNetKiller)*



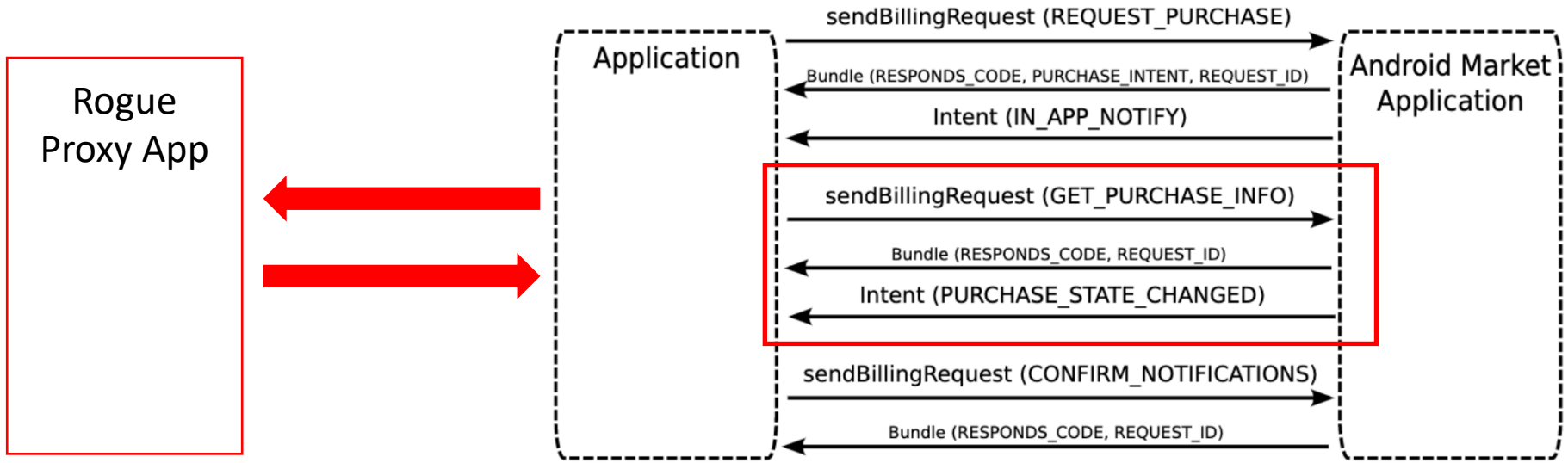
Practical Scenario
In-App Purchase

Practical Scenario In-App Purchase

Google Play Billing

Type of Google Play Billing

- ❑ One-time product
- ❑ Rewarded product
- ❑ Subscription

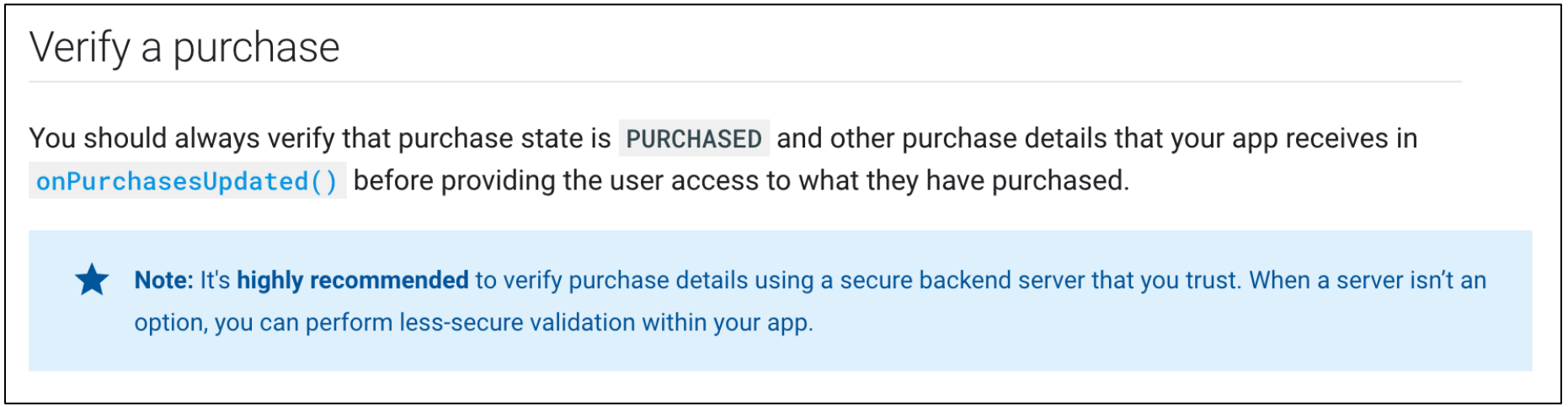


Source: https://mulliner.org/collin/publications/asia226-mulliner.pdf

Practical Scenario In-App Purchase

Google Play Billing

Google Recommendation



Verify a purchase

You should always verify that purchase state is `PURCHASED` and other purchase details that your app receives in `onPurchasesUpdated()` before providing the user access to what they have purchased.

★ **Note:** It's **highly recommended** to verify purchase details using a secure backend server that you trust. When a server isn't an option, you can perform less-secure validation within your app.

Source: https://developer.android.com/google/play/billing/billing_library_overview

Practical Scenario In-App Purchase

Google Play Billing

Library

- Prime31

> **Purchase Validation**
>
> Google **highly recommends** always validating purchases on a secure server. The plugin will do on device validation for you but Android apps are very easily hacked so this should not be relied on.

- Unity

> **Remote validation:** For server-side content, where content is downloaded once purchased, the validation should take place on the server before the content is released. Unity does not offer support for server-side validation; however, third-party solutions are available, such as Nobuyori Takahashi's IAP project.

Practical Scenario In-App Purchase

Google Play Billing

VerifyPurchase method

```
public static boolean verifyPurchase(String str, String str2, String str3, String str4) {
    if (!TextUtils.isEmpty(str3) && !TextUtils.isEmpty(str2) && !TextUtils.isEmpty(str4)) {
        return verify(generatePublicKey(str2), str3, str4);
    }
    if (str.equals("android.test.purchased") || str.equals("android.test.canceled") || str.equals("android.test.refund
        return true;
    }
    Log.e(TAG, "Purchase verification failed: missing data.");
    return false;
}
```

Practical Scenario In-App Purchase

Google Play Billing

Verify method

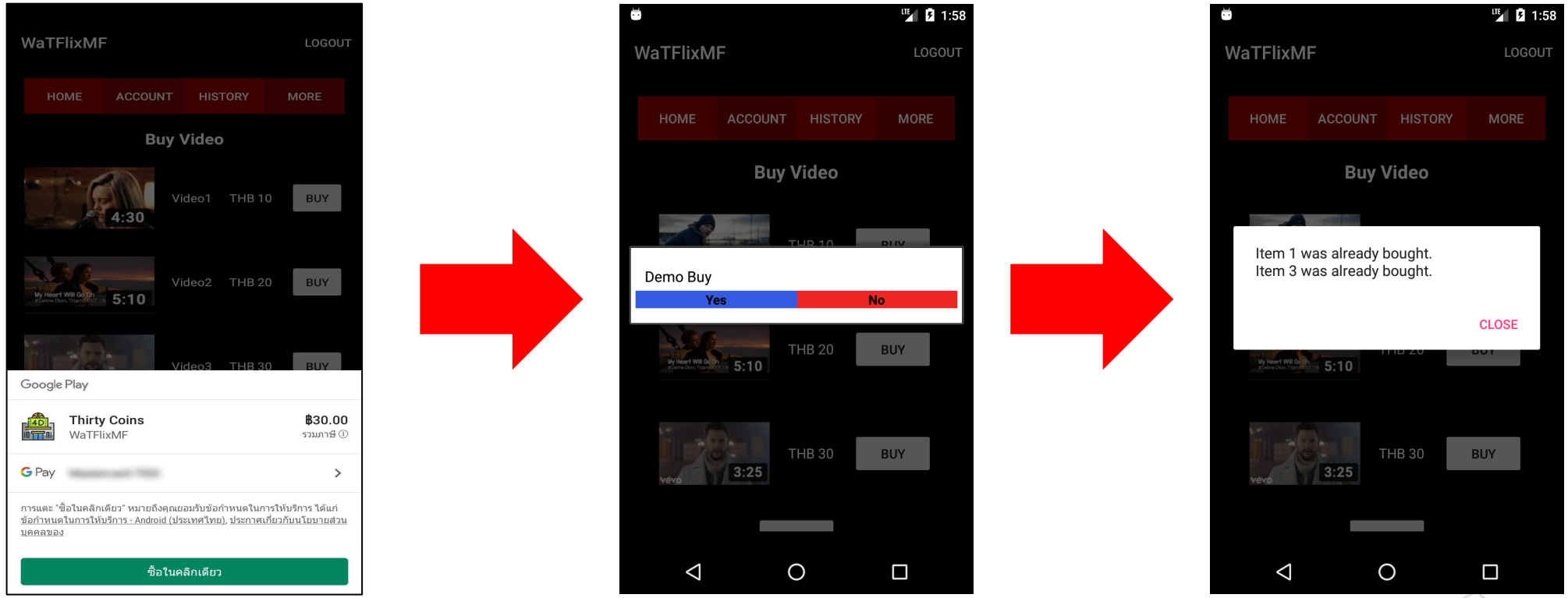
```
public static boolean verify(PublicKey publicKey, String str, String str2) {
    try {
        Signature instance = Signature.getInstance(SIGNATURE_ALGORITHM);
        instance.initVerify(publicKey);
        instance.update(str.getBytes());
        if (instance.verify(Base64.decode(str2, 0))) {
            return true;
        }
        Log.e(TAG, "Signature verification failed.");
        return false;
    } catch (NoSuchAlgorithmException unused) {
        Log.e(TAG, "NoSuchAlgorithmException.");
        return false;
    } catch (InvalidKeyException unused2) {
        Log.e(TAG, "Invalid key specification.");
        return false;
    } catch (SignatureException unused3) {
        Log.e(TAG, "Signature exception.");
        return false;
    } catch (IllegalArgumentException unused4) {
        Log.e(TAG, "Base64 decoding failed.");
        return false;
    }
}
```

```
public static boolean verify(PublicKey publicKey, String str, String str2) {
    try {
        Signature instance = Signature.getInstance(SIGNATURE_ALGORITHM);
        instance.initVerify(publicKey);
        instance.update(str.getBytes());
        instance.verify(Base64.decode(str2, 0));
        if (1 != 0) {
            return true;
        }
        Log.e(TAG, "Signature verification failed.");
        return false;
    } catch (NoSuchAlgorithmException unused) {
        Log.e(TAG, "NoSuchAlgorithmException.");
        return false;
    } catch (InvalidKeyException unused2) {
        Log.e(TAG, "Invalid key specification.");
        return false;
    } catch (SignatureException unused3) {
        Log.e(TAG, "Signature exception.");
        return false;
    } catch (IllegalArgumentException unused4) {
        Log.e(TAG, "Base64 decoding failed.");
        return false;
    }
}
```

Practical Scenario In-App Purchase

DEMO

❑ Bypassing InApp Purchase



Thank You

LET'S START

YOUR SECURITY JOURNEY WITH US.

Contact us: info@secure-d.tech