

mobile app sec. training

Ahmethan Gültekin – Mehmet Faris Acar

About Us

- > Ahmethan Gültekin
- > founder / mobile security researcher at Byteria
- > consulting and product development
- > public / private trainings



byterialab.com
ahmethan@byterialab.com

About Us

- > Mehmet Faris Acar
- > mobile security researcher at Byteria
- > mobile security researching



mehmet.faris@byterialab.com

Contents

- > android internals
- > android native internals
- > use of reverse engineering tools
- > detection mechanisms (frida,xposed e.g)
- > some hooks and analysis
- > detection bypass techniques
- > native side detections
- > crypto methods
- > ???

All presentation and lab files;

> <https://github.com/byterialab/altinbas-online-bootcamp-2025>

Android Internals

- > Android linux çekirdeğini kullanan bir işletim sistemidir
- > projeler .apk dosya uzantısında paketlenir.
- > her uygulamanın kendine ait klasörü, bu klasörlerin de kendi içinde permissionları vardır.
- > uygulama DEX (dalvik executable) bytecode formatında çalışır
- > yeni cihazlarda ART (Android Runtime), eski cihazlarda DVM (Dalvik Virtual Machine) kullanılır



> DVM (dalvik virtual machine):

- > eski versiyonlar için kullanılır.
- > uygulamalar **DEX** formatında çalıştırılır.
- > **JIT** (Just-In-Time) derleme kullanır. bytecode'u çalışma anında **makine koduna** dönüştürür.

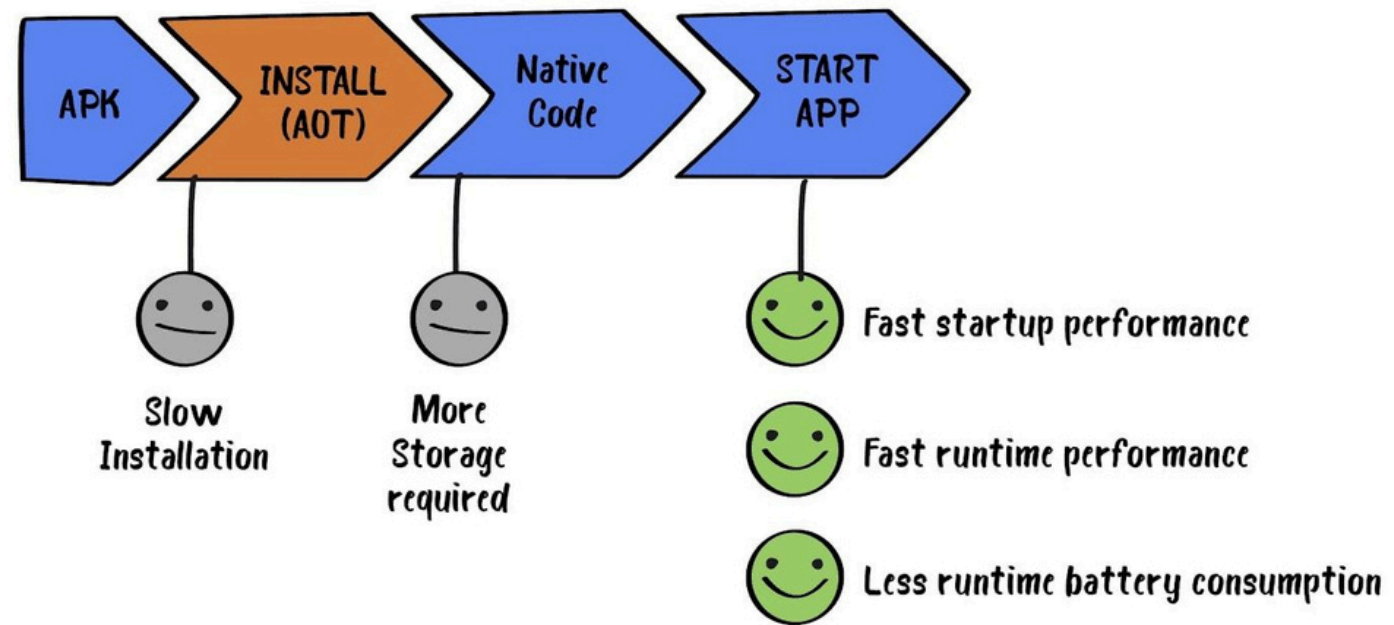
> ART (Android Runtime):

- > yeni versiyonlar için kullanılır. **Lollipop** (5.0) versiyonu ile gelmiştir.
- > uygulamalar DEX formatında çalıştırılır.
- > **AOT** (Ahead-Of-Time) derleme kullanır. DEX bytecode'u yükleme / kurulum anında **makine koduna** dönüştürür.
- > sık kullanılan kod parçaları önbelleklenir ve daha performanslı çalışır.

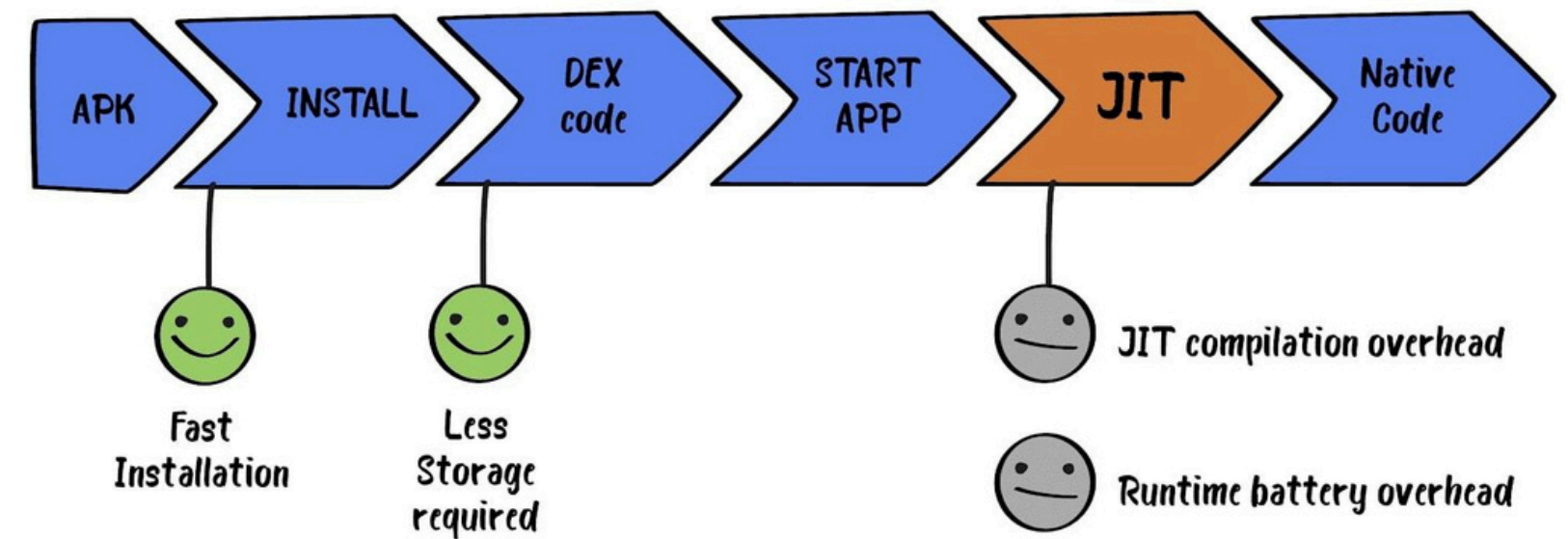


AOT vs JIT

Android Marshmallow Ahead-Of-Time (AOT) Solution



Android KitKat Just-In-Time (JIT) Solution



ART vs Dalvik

> Depolama Alanı:

> ART, **AOT** yaklaşımını kullandığı için **daha fazla** depolama alanına ihtiyaç duyar.

(önbellekleme, geçici derleme dosyaları)

> Dalvik, **JIT** yaklaşımını kullandığı için kod runtimeda derlenir. kod kalıcı olarak saklanmaz, bellekten yürütülür.

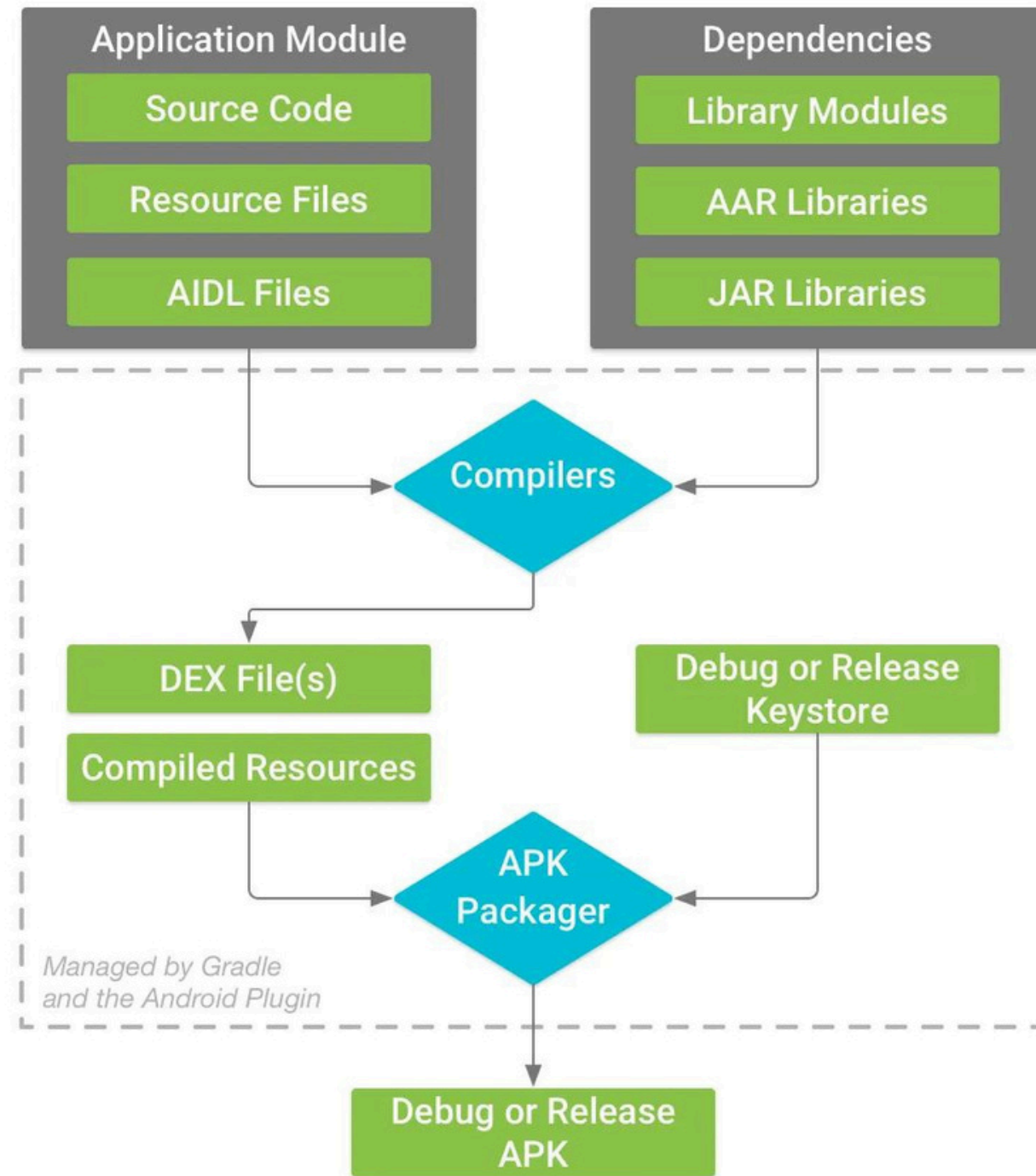
> Önyükleme Süresi (booting time):

> **ART**, kurulum esnasında derleme yapar. bundan dolayı boot süresi daha **uzundur**.

> Dalvik, **JIT** yaklaşımını kullandığı için runtimeda derleme yapar. boot süresi daha **kısadır**.



App compilation process



Smali

- > android'e özel geliştirilmiş java'ya benzer bir dil
- > java kodunun dex bytecode'a dönüşmeden önceki hali

```
public class Calculation {  
    public int add(int a, int b) {  
        return a + b;  
    }  
}
```

java code

```
.class public LCalculation;  
.super java/lang/Object  
  
.method public constructor <init>()V  
    .registers 1  
    invoke-direct {p0}, Ljava/lang/Object;-><init>()V  
    return  
.end method  
  
.method public add(II)I  
    .registers 3  
    .parameter "a"  
    .parameter "b"  
  
    .prologue  
    .line 4  
    add-int/2addr v0, v1  
    return v0  
.end method
```

smali code

APK File Structure

lib: projede kullanılan C/C++ librarylerinin olduğu klasör

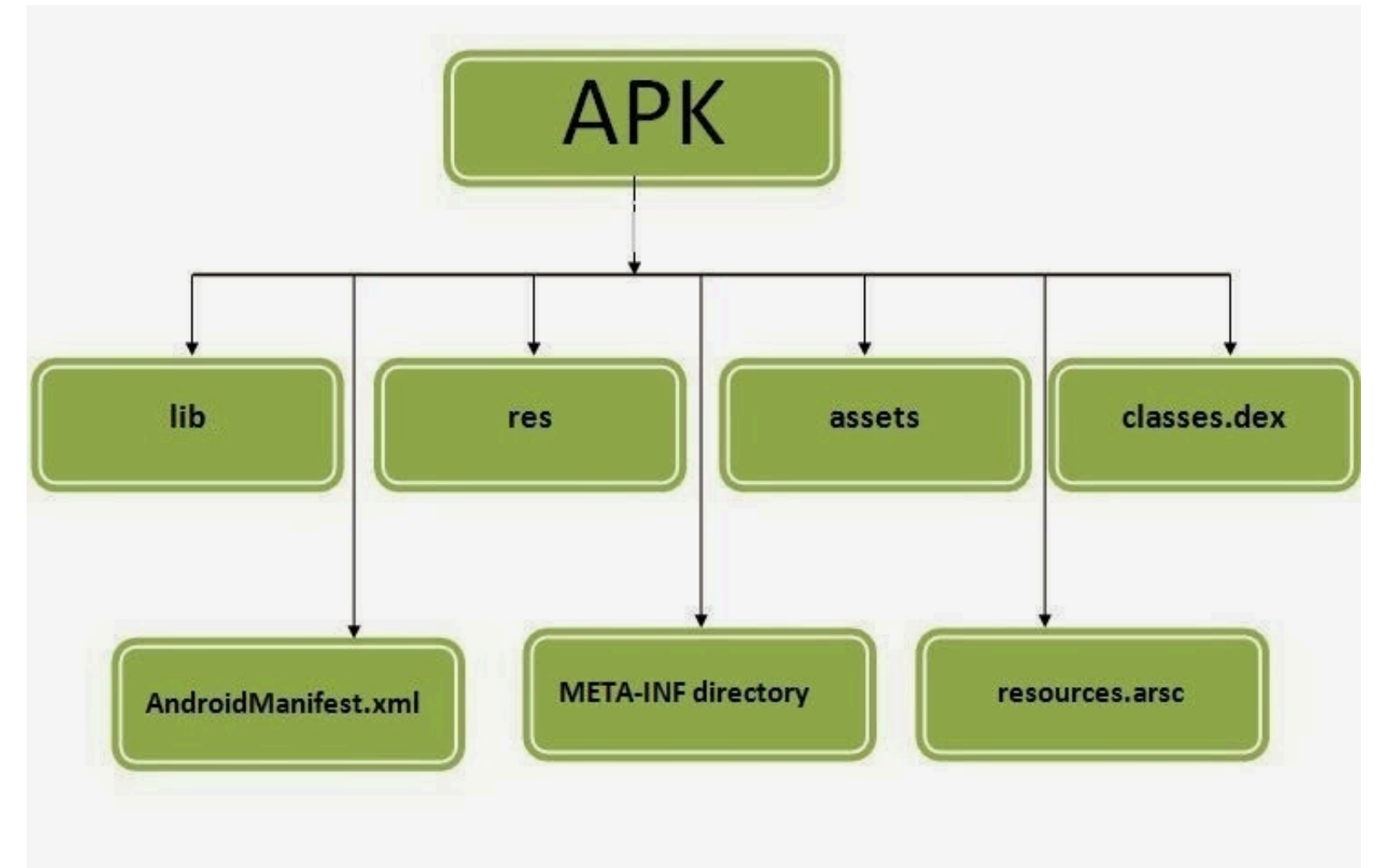
res: projedeki ui,strings, renk tanımlarını içeren klasör
.xml formatında veri tutar.

assets: uygulamanın sahip olduğu tüm assetleri içeren klasör

classes.dex: java kodunun derlenip dex bytecode formatına dönüştürülmüş hali

META-INF: apk'nın imza ve sertifika bilgilerini içeren klasör

AndroidManifest.xml: uygulamanın tüm permission ve activity - receiver bilgilerinin bulunduğu dosya



- > android cihaz ile ana makine arasında köprü sağlayan tool
- > cihaz ile ilgili bir çok işlemi yapabilir (install, shell, file push/pull vs.)
- > Android Studio ile birlikte gelir (platform-tools)



Android Security Model

> permissions:

- > kullanıcılar, uygulamaların erişebileceği verilere ve özelliklere izin verir
- > izinler, uygulamaların sadece gerekli ve istenilen kaynaklara erişimini sağlar.
- > hassas verilere erişim kullanıcı kontrolündedir

> sandboxing:

- > her uygulama, kendi izole edilmiş kullanıcı kimliğinde (UID) çalışır ve kendine özel dizinleri vardır
- > uygulamalar birbirlerinin verilerine ve sistem kaynaklarına erişemez.
- > izolasyon güvenliğini sağlar ve SELinux ile entegreli çalışır.



Android Security Model

> SELinux (Security-Enhanced Linux) :

- > zorunlu erişim kontrolü sağlar, sistem seviyesinde güvenlik politikalarını uygular
- > uygulamaların ve sistem süreçlerinin kaynaklara erişimini denetler.
- > güvenliği çekirdek seviyesinde artırır.

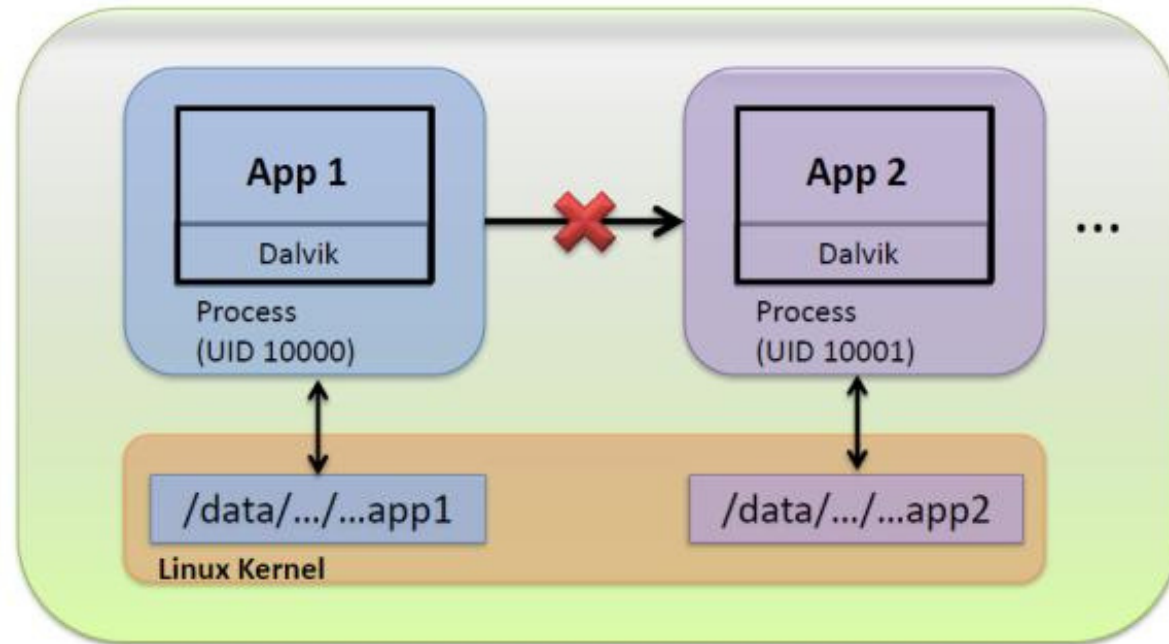
> Enforcing vs Permissive Mode

- > Enforcing, SELinux politikalarını katı bir şekilde uygular ve izin verilmeyen işlemleri engeller.
- > Permissive: politikaları loglar ancak engellemez, test ve hata ayıklama için kullanılır

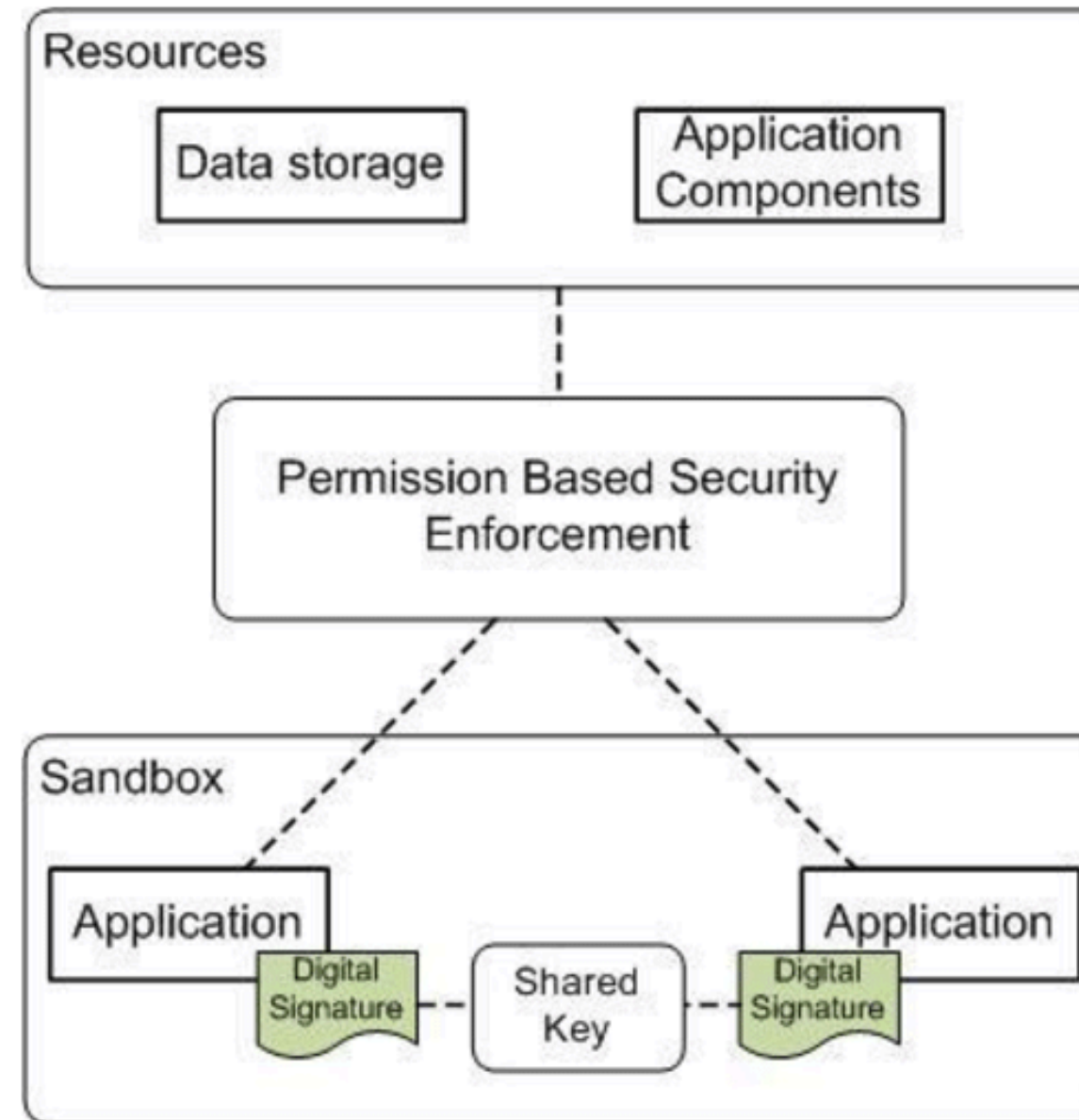


Android Security Model

> application UID model

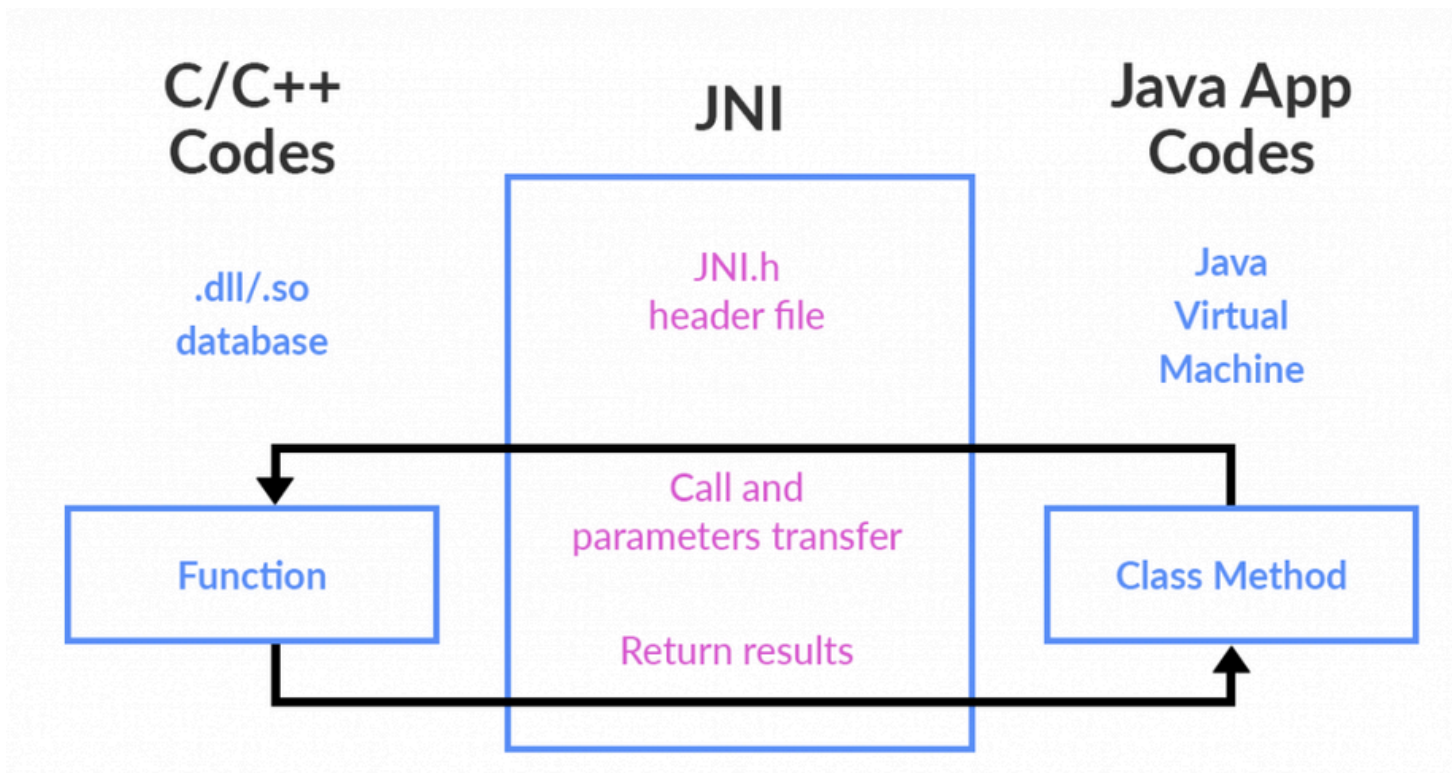


> general security model

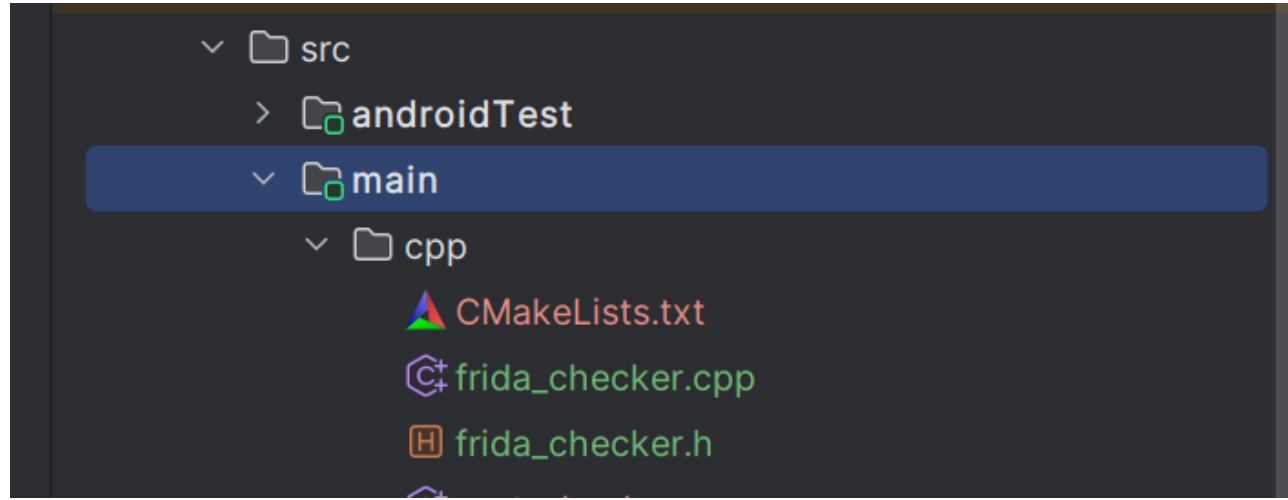


Android NDK

- > java / kotlin uygulamalara C/C++ modüllerini entegre etme
- > java ile JNI (Java Native Interface) aracılığıyla iletişim kurar
- > c syntax dillerdeki gibi .h başlık dosyası ve c/cpp dosyası kullanılır
- > performans gerektiren ses - görüntü gibi işlemlerde ve obfuscate edilmek istenen işlemlerde tercih edilir



Android NDK



- > NDK dosyaları default olarak src/main/cpp dizininde bulunur.
- > **CMakeLists.txt**: NDK projenizi derlemek için kullanacağınız cmake config dosyası
- > **.h dosyaları**: C/C++ dillerinde kullanılan başlık dosyaları
- > **.cpp dosyaları**: Modüllerin bulunduğu cpp dosyası



Android NDK- JNI

(Java Native Interface)

- > Java tarafından NDK fonksiyonlarına erişebilmek için kullanılır.
- > NDK ile Java arasında köprü görevi görür.
- > Java ile C/C++ kodları arasında dönüşümleri yapar.

Java primitive type	Native primitive type
void	void
byte	jbyte
int	jint
float	jfloat
double	jdouble
char	jchar
long	jlong
short	jshort
boolean	jboolean



Android NDK- JNI

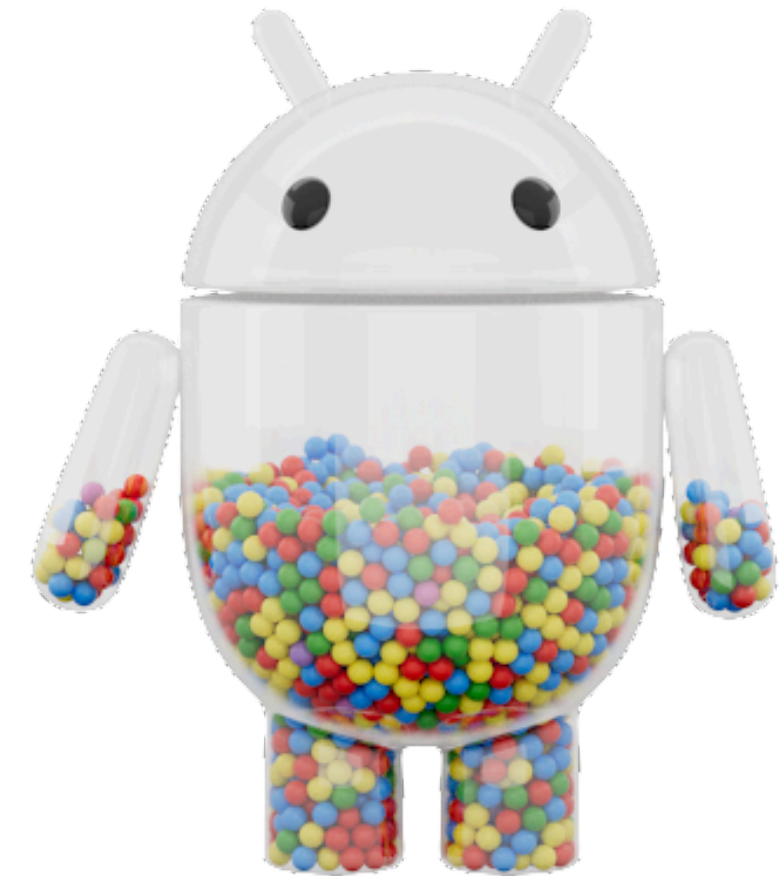
```
JNIEXPORT jboolean JNICALL Java_com_tsgk_lab_MainActivity_checkPort(JNIEnv *env, jobject, jint port) {
```

JNIEXPORT: native taraftaki C/C++ fonksiyonunu JNI ile dışarı aktarır

jboolean: fonksiyonun dönüş tipi

JNICALL: native fonksiyonun JNI ile Java tarafından çağrılabilmesi için native tarafta nasıl tanımlanacağını belirtir

Java_*: Java tarafında fonksiyonun hangi paket, hangi sınıf ve hangi fonksiyon ismiyle tanımlanacağını belirtir



mobile reversing tools

frida, frida-trace, jnitrace, objection, xposed vs.

Apktool

- > en popüler apk analiz tooludur
- > decompile edilmiş kaynak kod, uygulamanın tüm assetlerine erişilebilir
- > apk decompile edebilir - build alabilir
- > tek bir jar dosyası ile tüm sistemlerde çalışabilir

- > apktool d base.apk
- > apktool b base.apk



- > **vala** dili ile geliştirilmiş bir instrumentation tooludur
- > cihazla etkileşime girebilmesi için cihazda frida-server çalışması gerekir
- > TCP/IP üzerinden iletişim sağlar
- > hedef processe **libfrida-gadget.so** veya **frida-agent.so** isminde library enjekte eder ve injection bu şekilde gerçekleşir
- > android için ptrace(), LD_PRELOAD yöntemleri kullanılarak injection işlemi gerçekleşir
- > javascript dili ile custom hook scriptleri yazmaya olanak tanır

The logo for Frida, featuring the word "FRIDA" in white, bold, uppercase letters centered on a solid red rectangular background.

FRIDA

> ptrace() ile injection

- > ptrace() methodu bir processin başka bir processi kontrol etmesi ve izlemesine olanak sağlar
- > methodun birden fazla flagi vardır ve her flagin işlevi farklıdır
- > frida, ptrace() kullanarak süreci duraklatır ve dlopen() ile açtığı libfrida-gadget.so kütüphanesini processe inject eder. ardından processi devam ettirir.
- > android için ptrace(), LD_PRELOAD yöntemleri kullanılarak injection işlemi gerçekleşir

```
frida -> ptrace(PTRACE_ATTACH)
-> hedef process -> ptrace(PTRACE_POKETEXT) -> hedef process
-> ptrace(dlopen("libfrida-gadget.so")) -> hedef process
-> libfrida-gadget.so -> ptrace(PTRACE_DETACH)
```

The logo for Frida, featuring the word "FRIDA" in a bold, white, sans-serif font centered on a solid red rectangular background.

> ptrace()

> frida, ptrace() ile injection yaparken PTRACE_ATTACH ve PTRACE_CONT flaglerini kullanır.

```
pid_t target_pid = /* hedef işlemin PID'si */;
ptrace(PTRACE_ATTACH, target_pid, NULL, NULL);

// Hedef işlem durdurulduktan sonra gerekli işlemler yapılır.

// İşlemi devam ettir
ptrace(PTRACE_CONT, target_pid, NULL, NULL);

// İzlemeyi bırakmak için
ptrace(PTRACE_DETACH, target_pid, NULL, NULL);
```

The logo for Frida, featuring the word "FRIDA" in a bold, white, sans-serif font centered on a solid red rectangular background.

FRIDA

> LD_PRELOAD ile injection

- > LD_PRELOAD ortam deęiřkeni, bir processin alıřtırılmadan nce belirli bir kitaplıęı yklemesini saęlar
- > hedef uygulama bařlamadan nce ortam deęiřkeni olarak LD_PRELOAD deęerini set ederek processten nce verilen librarynin yklenmesini saęlar

> rnek:

```
export LD_PRELOAD=/lib/frida/libfrida-gadget.so
```

```
frida -U -f com.example.app --no-pause -l script.js
```

- > burada com.example.app yklenmeden libfrida-gadget.so ktphanesi load edilir ve frida bu sayede processe inject olur

The logo for Frida, featuring the word "FRIDA" in a bold, white, sans-serif font centered on a solid red rectangular background.

> frida script modules

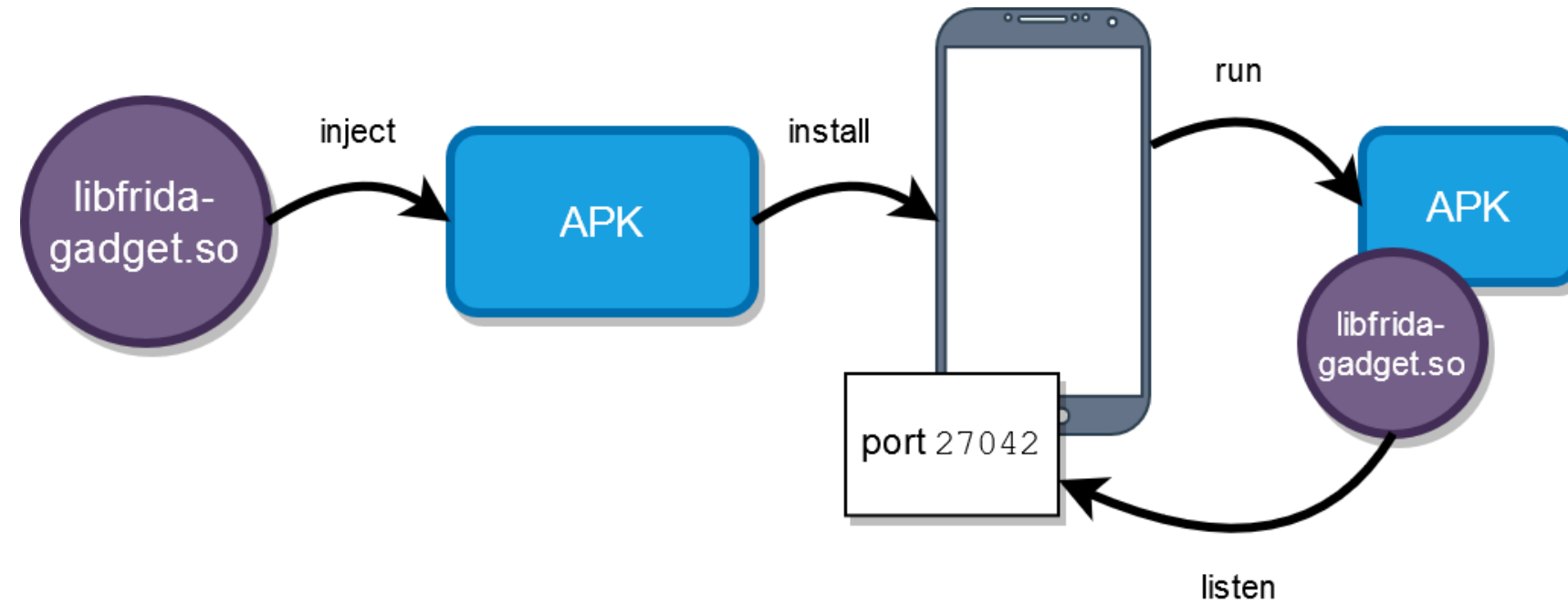
- **Interceptor.attach():**
 - Bir işlevin çalışmasını yakalar ve işlevin başında **onEnter()** ve sonunda **onLeave()** özel kod çalıştırır.
- **.overload():**
 - Aynı isimli fakat farklı imzalara sahip işlevleri ayırt etmek için kullanılır.
- **.implementation():**
 - Bir işlevin davranışını yeniden tanımlar ve yeni bir implementasyon sağlar.

The logo for Frida, featuring the word "FRIDA" in white, uppercase, sans-serif font centered on a solid red rectangular background.

FRIDA

> Gadget

- > fridanın processe inject olmasını sağlayan kütüphane
- > android / ios / windows / linux gibi tüm OS'larda çalışır
- > root & jailbreak gerektirmez



> frida-trace

- > uygulamanın fonksiyon çağrılarını izler
- > hem java hem native taraftaki fonksiyonları trace edebilir
- > regex desteği vardır. esnek trace inputu alabilir

> example:

```
frida-trace -U -j "com.tsgk.lab.MainActivity.traceTest" com.tsgk.lab
```

(java function trace)

```
frida-trace -U -i "nativeFunction" com.tsgk.lab
```

(ndk function trace)

The logo for Frida, featuring the word "FRIDA" in a bold, white, sans-serif font centered on a solid red rectangular background.

> jnitrace

- > frida-trace'nin alternatifidir, daha detaylı versiyonudur
- > frida-trace'den daha detaylı çıktı verir. (parametreler, backtrace gibi)
- > sadece JNI fonksiyonlarını trace eder.

```
/* TID 4932 */
2976 ms [+] JNIEnv->NewByteArray
2976 ms |- JNIEnv*      : 0xf4099ae0
2976 ms |- jsize        : 25
2976 ms |= jbyteArray    : 0x100025

-----Backtrace-----
2976 ms |-> 0xf308fbc9: Java_com_nativetest_MainActivity_stringFromJNI+0x489 (libnative-lib.so:0xf308f000)

/* TID 4932 */
2980 ms [+] JNIEnv->SetByteArrayRegion
2980 ms |- JNIEnv*      : 0xf4099ae0
2980 ms |- jbyteArray    : 0x100025
2980 ms |- jsize        : 0
2980 ms |- jsize        : 25
2980 ms |- jbyte*       : 0xff8579d4
2980 ms |:      0000000: 01 02 03 04 05 06 07 08 09 0A 0C 0D 0E 01 02 03 .....
2980 ms |:      0000010: 04 05 06 07 08 09 0A 0C 0D .....

-----Backtrace-----
2980 ms |-> 0xf308fc4f: Java_com_nativetest_MainActivity_stringFromJNI+0x50f (libnative-lib.so:0xf308f000)
```

```
4932 */
JNIEnv->NewByteArray
Env*      : 0xf4099ae0
ze        : 25
teArray   : 0x100025

-----Backtrace-----
F308fbc9: Java_com_nativetest_MainActivity_stringFromJNI+0x489 (libnative-lib.

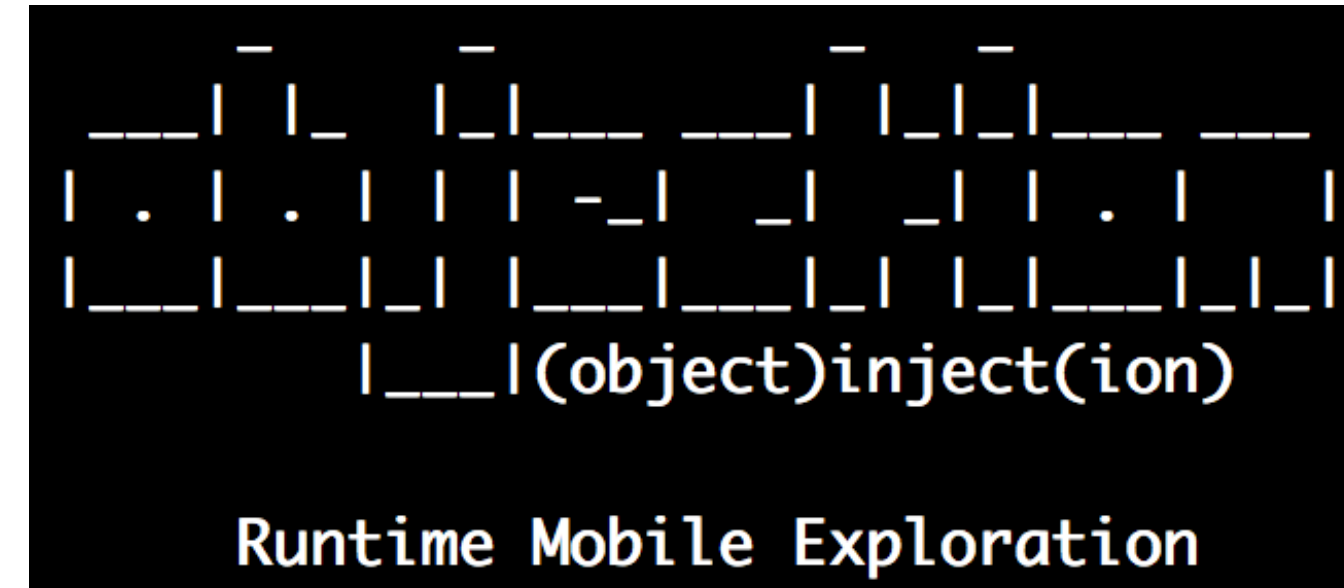
4932 */
JNIEnv->SetByteArrayRegion
Env*      : 0xf4099ae0
teArray   : 0x100025
ze        : 0
ze        : 25
te*       : 0xff8579d4
0000000: 01 02 03 04 05 06 07 08 09 0A 0C 0D 0E 01 02 03 .....
0000010: 04 05 06 07 08 09 0A 0C 0D .....

-----Backtrace-----
F308fc4f: Java_com_nativetest_MainActivity_stringFromJNI+0x50f (libnative-lib.
```


Objection

> objection

- > frida tabanlı çalışır
- > runtime'da bir çok işlemi pratikçe yapmayı sağlar
- > kendi içinde hazır modülleri vardır.
ek olarak 3. parti modül de eklenebilir



some cryptography

AES, RSA, SHA256 ...

XOR (Exclusive OR)

> iki bitin karşılaştırılmasıyla yapılan bir mantıksal işlemdir

> İki bit farklıysa sonuç 1, aynıysa 0 olur

- $0 \text{ XOR } 0 = 0$
- $0 \text{ XOR } 1 = 1$
- $1 \text{ XOR } 0 = 1$
- $1 \text{ XOR } 1 = 0$

> veriyi sabit uzunluklara böler ve blok şeklinde şifreleme yapar

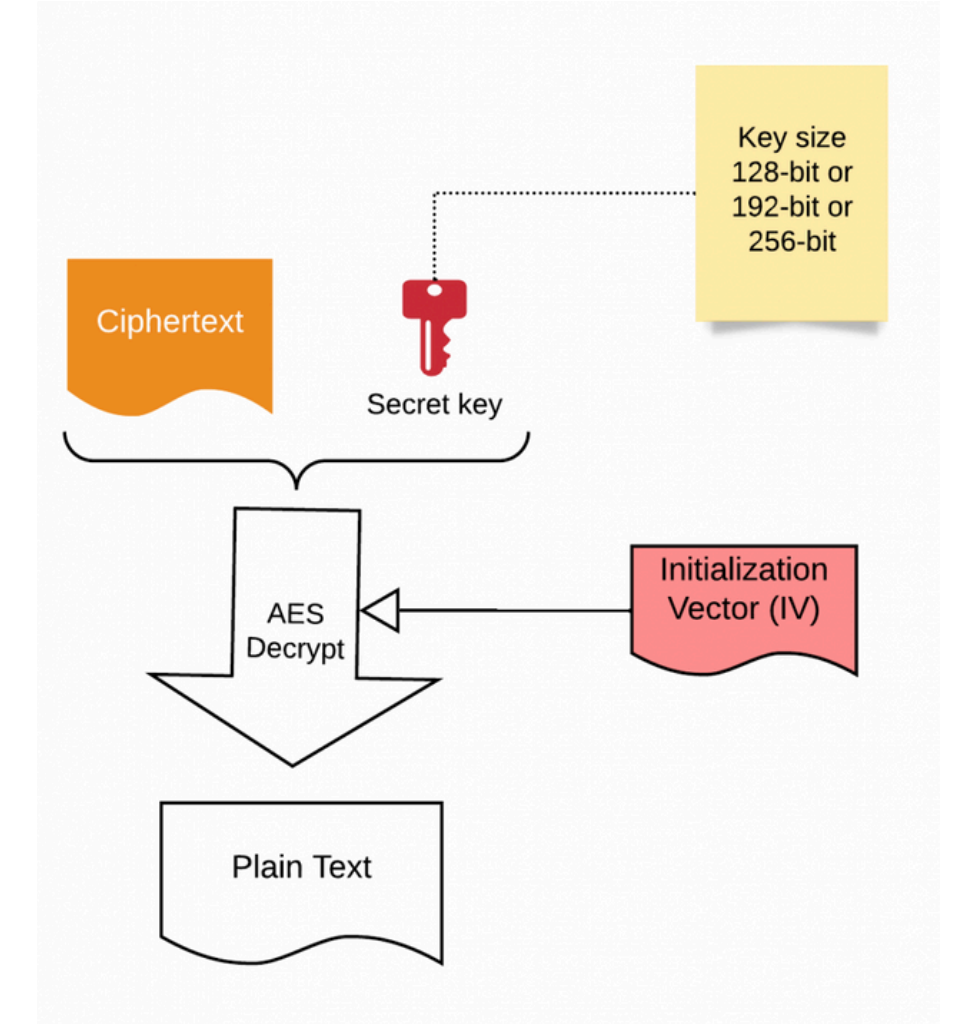
- Bit Dizisi A: 1101
- Bit Dizisi B: 1010
- XOR Sonucu: 0111

> iki bit farklıysa 1, aynıysa 0 verir

x_1	x_2	$x_1 \text{ XOR } x_2$
0	0	0
0	1	1
1	0	1
1	1	0

AES (Advanced Encryption Standard)

- > simetrik şifreleme algoritmasıdır
- > 128-bit, 192-bit ve 256-bit anahtar uzunluğu
- > birden fazla şifreleme modu (her mod farklı şekilde çalışır)
- > her sütuna belirli matematiksel işlemler uygulanır
- > matematiksel işlemler, XOR ve satır kaydırma



Symmetric and Asymmetric Encryption

simetrik şifreleme

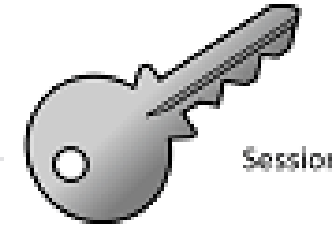
- > aynı şifre hem şifreleme, hem deşifreleme için kullanılır
- > anahtar gizlidir
- > daha hızlıdır ve yönetimi asimetriğe göre daha basittir

asimetrik şifreleme

- > iki farklı anahtar vardır (public key, private key)
- > açık anahtar (public key) şifreleme için, gizli anahtar (private key) deşifreleme için kullanılır
- > açık anahtar paylaşılabilir, gizli anahtar saklanır
- > hız olarak simetrik şifrelemeden daha yavaştır
- > anahtar güvenliği simetrik şifrelemeye göre daha güvenlidir

Symmetric Encryption

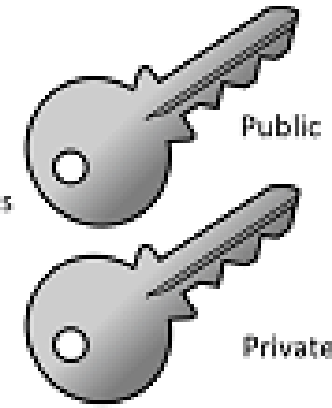
One key



Session

Asymmetric Encryption

Two keys



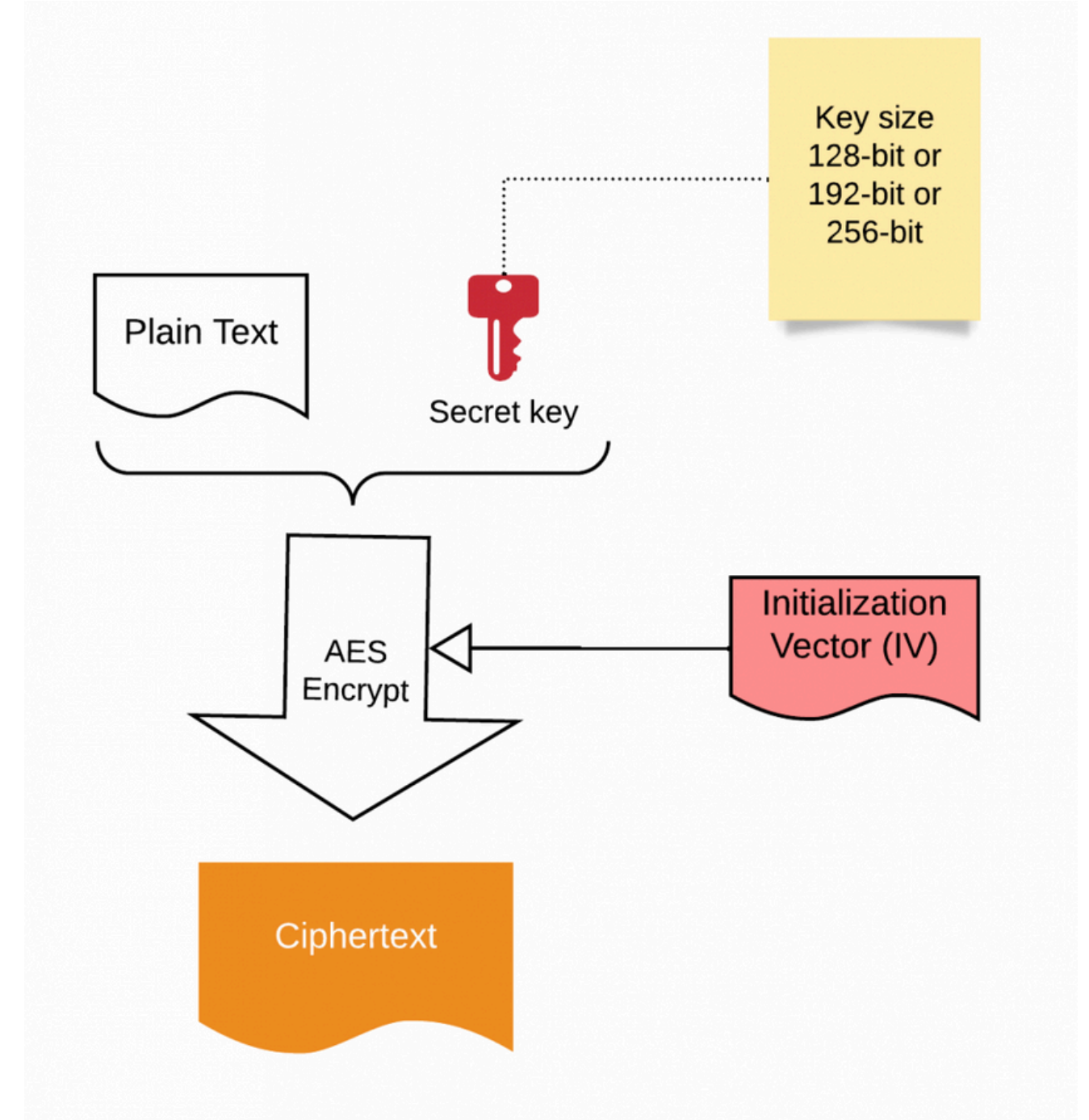
Public

Private

AES (Advanced Encryption Standard)

IV (Initialization Vector)

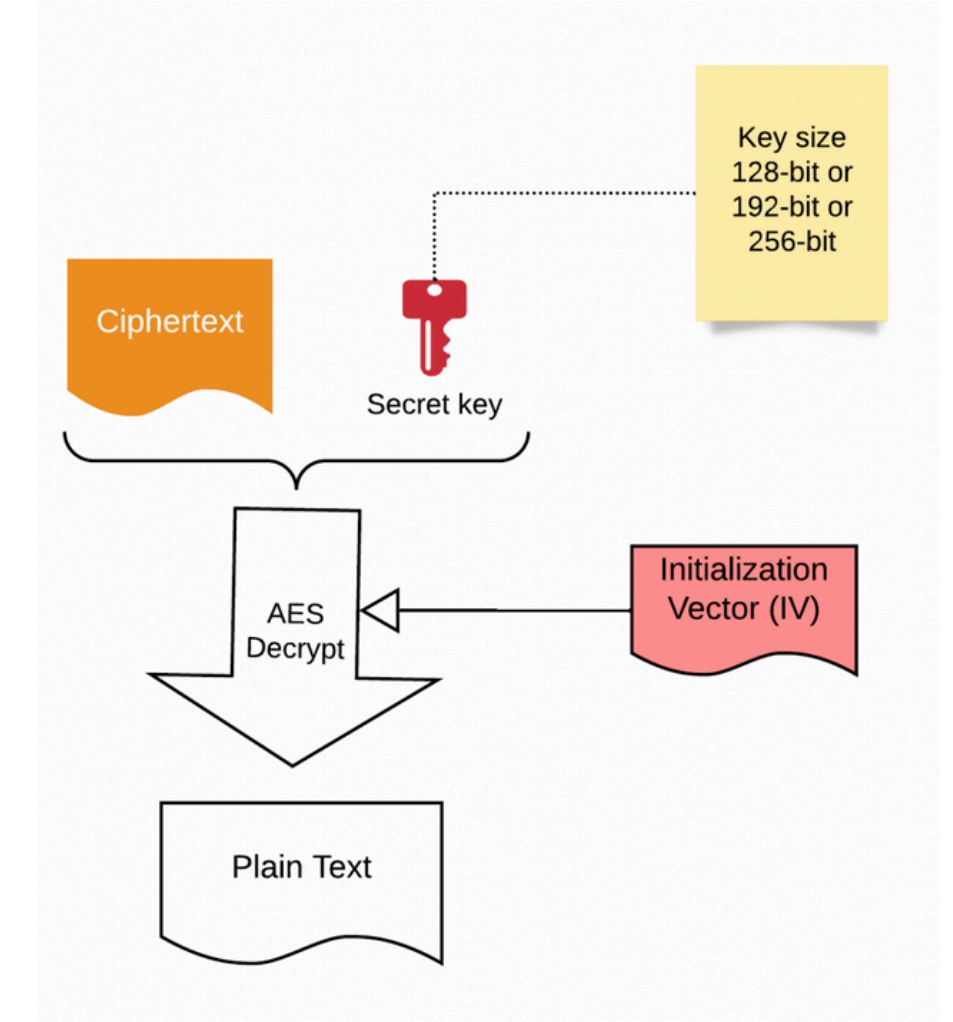
- > her işlem için rastgele ve benzersiz bir değerdir
- > rastgele olması sayesinde datalar arası benzersizliği sağlar
- > şifreleme işlemi başlamadan başlangıç noktası olarak kullanılır
- > CBC ve GCM şifreleme modları IV kullanır
- > ECB methodu IV kullanmaz, bu yüzden daha az güvenlidir



AES (Advanced Encryption Standard)

AES ECB (Electronic Codebook)

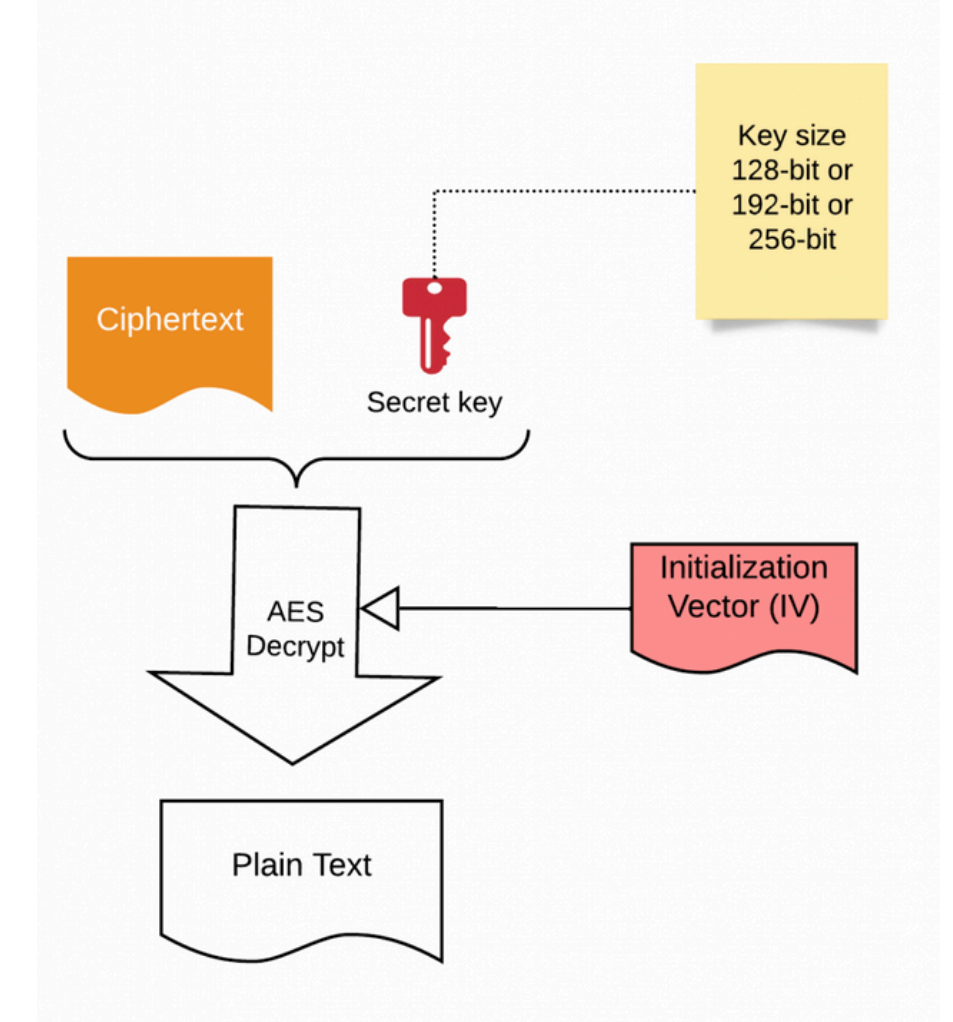
- > her veri blođunu bađımsız olarak Őifreler
- > aynı veri her zaman aynı Őifreyi üretir
- > diđer encryption modlarına göre güvenliđi daha düşük
- > IV kullanmaz



AES (Advanced Encryption Standard)

AES CBC (Cipher Block Chaining)

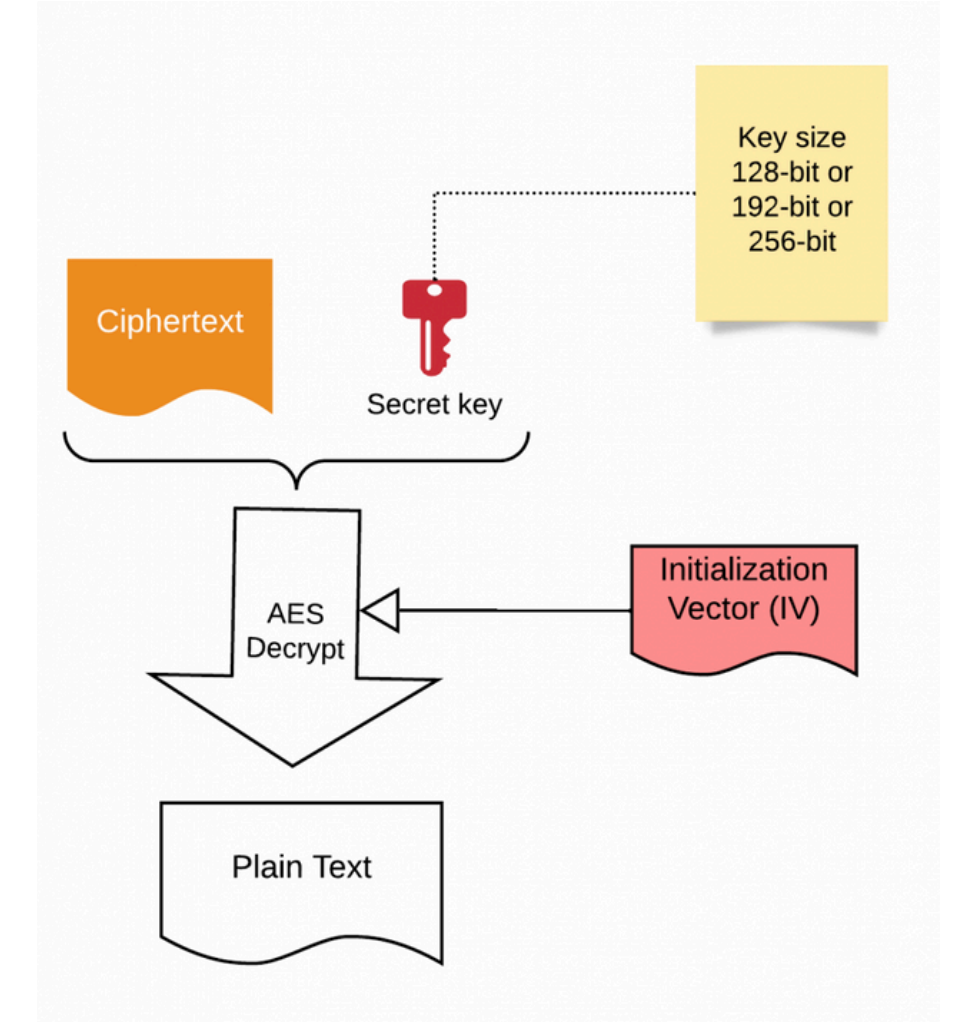
- > her veri blođunu bir önceki blođun şifreli verisi ile XOR'lar
- > her blok birbirinden bağımsız
- > diđer encryption modlarına göre güvenliđi daha düşük
- > IV (Initialization Vector) kullanılır
- > aynı veriler farklı cipherlar üretir
- > paralel işlem zordur



AES (Advanced Encryption Standard)

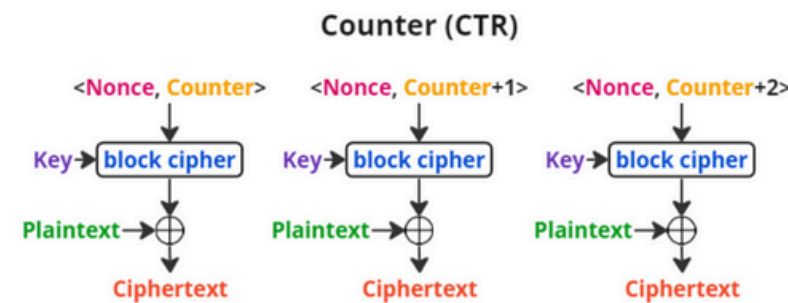
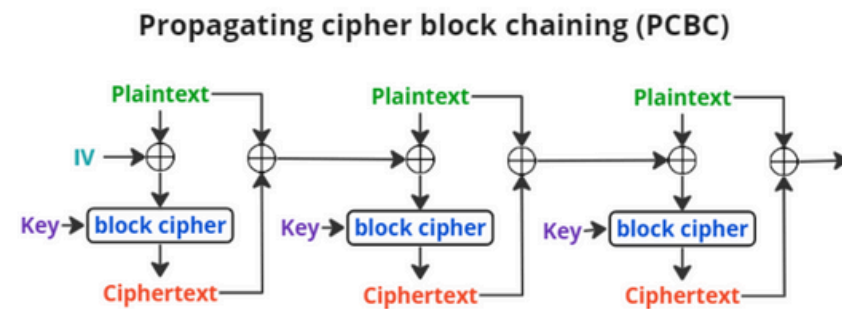
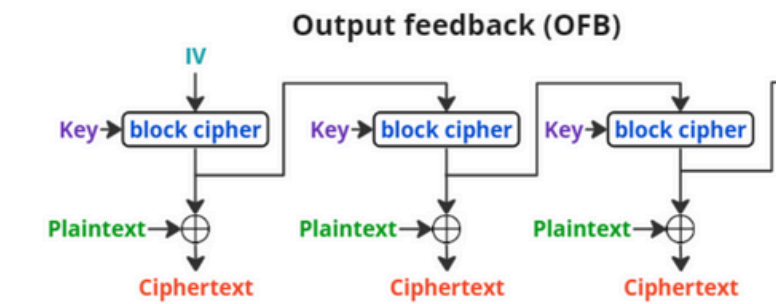
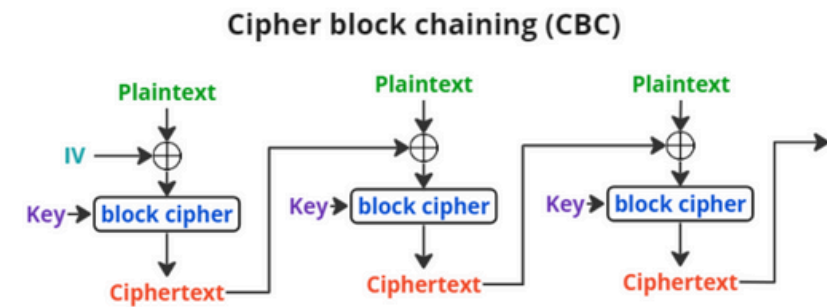
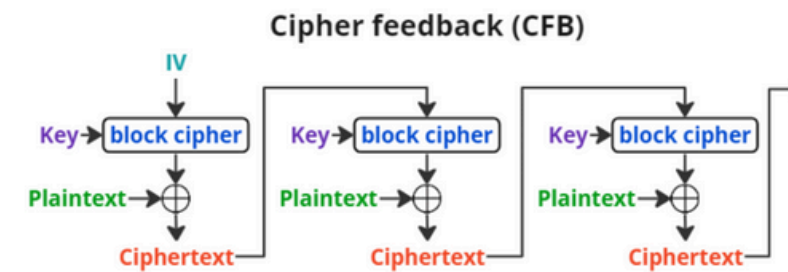
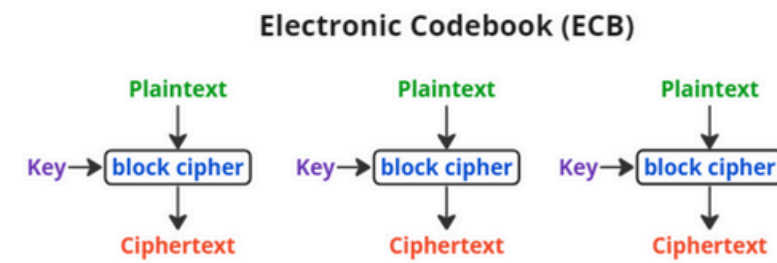
AES GCM(Galois/Counter Mode)

- > şifreleme ve doğrulama işlemleri paralel yürütülür
- > hem şifreleme hem veri bütünlüğü
- > IV (Initialization Vector) kullanılır
- > aynı veriler farklı cipherlar üretir



AES (Advanced Encryption Standard)

AES Modes



RSA (Rivest-Shamir-Adleman)

- > asimetrik şifreleme yöntemidir
- > public key kullanılarak şifreleme, private key ile de çözme işlemi
- > güvenliği matematiksel formüllere dayanır
- > 1024, 2048 veya 4096 bit uzunluk
- > ssl/tls işlemleri, signature işlemleri

1. Anahtar Üretimi:

- $p = 61, q = 53$
- $n = 61 * 53 = 3233$
- $\varphi(n) = (61-1)(53-1) = 3120$
- $e = 17$ (genellikle küçük bir asal sayı seçilir)
- $d = 2753$ (çünkü $17 \times 2753 \equiv 1 \pmod{3120}$)

2. Şifreleme:

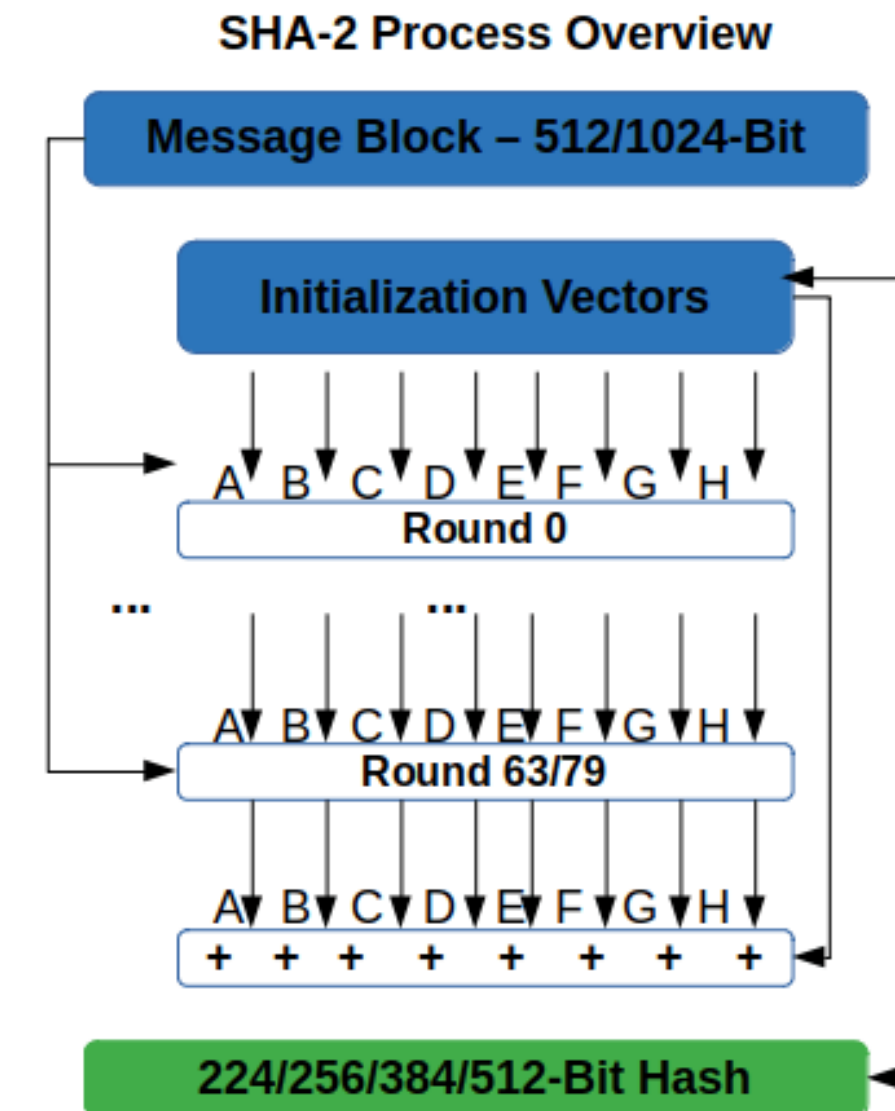
- Mesaj (m) = 123
- $c = 123^{17} \pmod{3233} = 855$

3. Deşifreleme:

- $m = 855^{2753} \pmod{3233} = 123$

SHA (Secure Hash Algorithm)

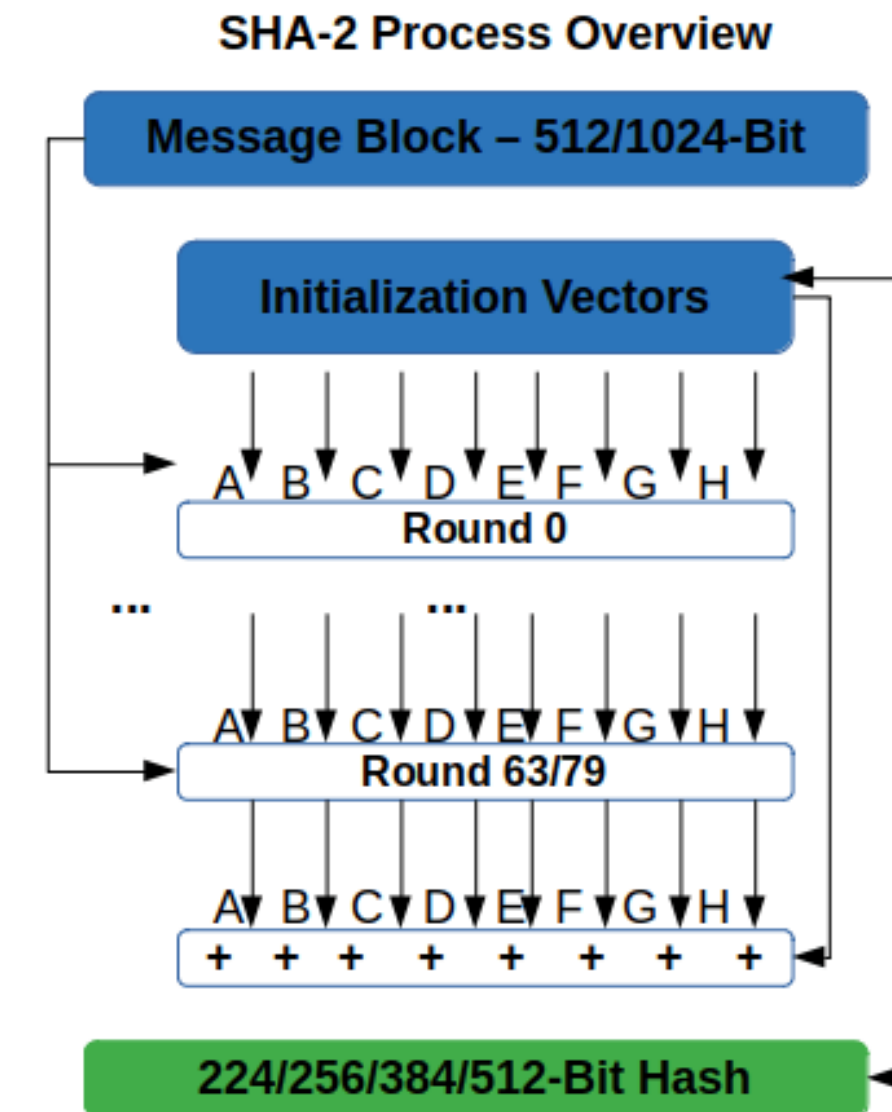
- > veri belirli bloklara bölünür
- > eğer veri blok sayısını tam karşılamıyorsa padding (doldurma) ile tamamlanır
- > her blok çeşitli matematiksel işlemler ve bit kaydırma işlemlerine tabi tutulur.



SHA (Secure Hash Algorithm)

SHA256

- > sabit uzunlukta hash üretir. (256 bitlik (32 byte))
- > 512 bitlik bloklara bölünür ve bloklar matematiksel işlemler, bit kaydırma işlemlerine tabi tutulur



practice time

let's hook some places