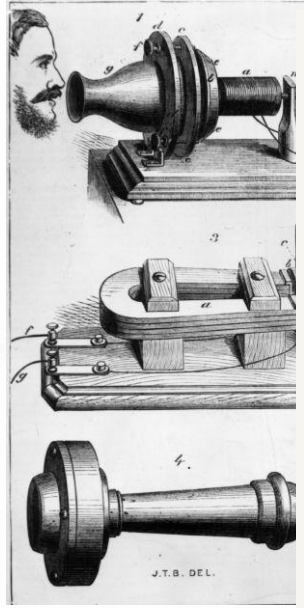
ملاحظات امنیتی برنامک ها در سامانه های هوشمند

ارایه دهنده: محمد حسام تدین
tadayon@itrc.ac.ir

معاون پژوهشکده امنیت و عضو هیات علمی
پژوهشگاه ارتباطات و فناوری اطلاعات

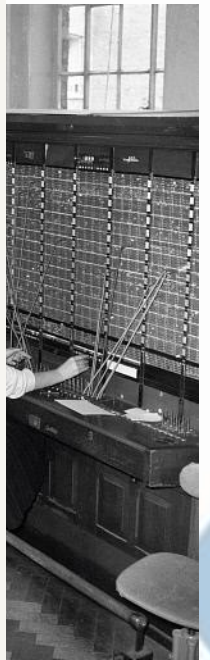
سامانه‌های هوشمند

دارای سیستم عامل
قابلیت نصب برنامه کاربردی
قابلیت پردازش و ذخیره‌سازی اطلاعات


J.T.B. DEL.

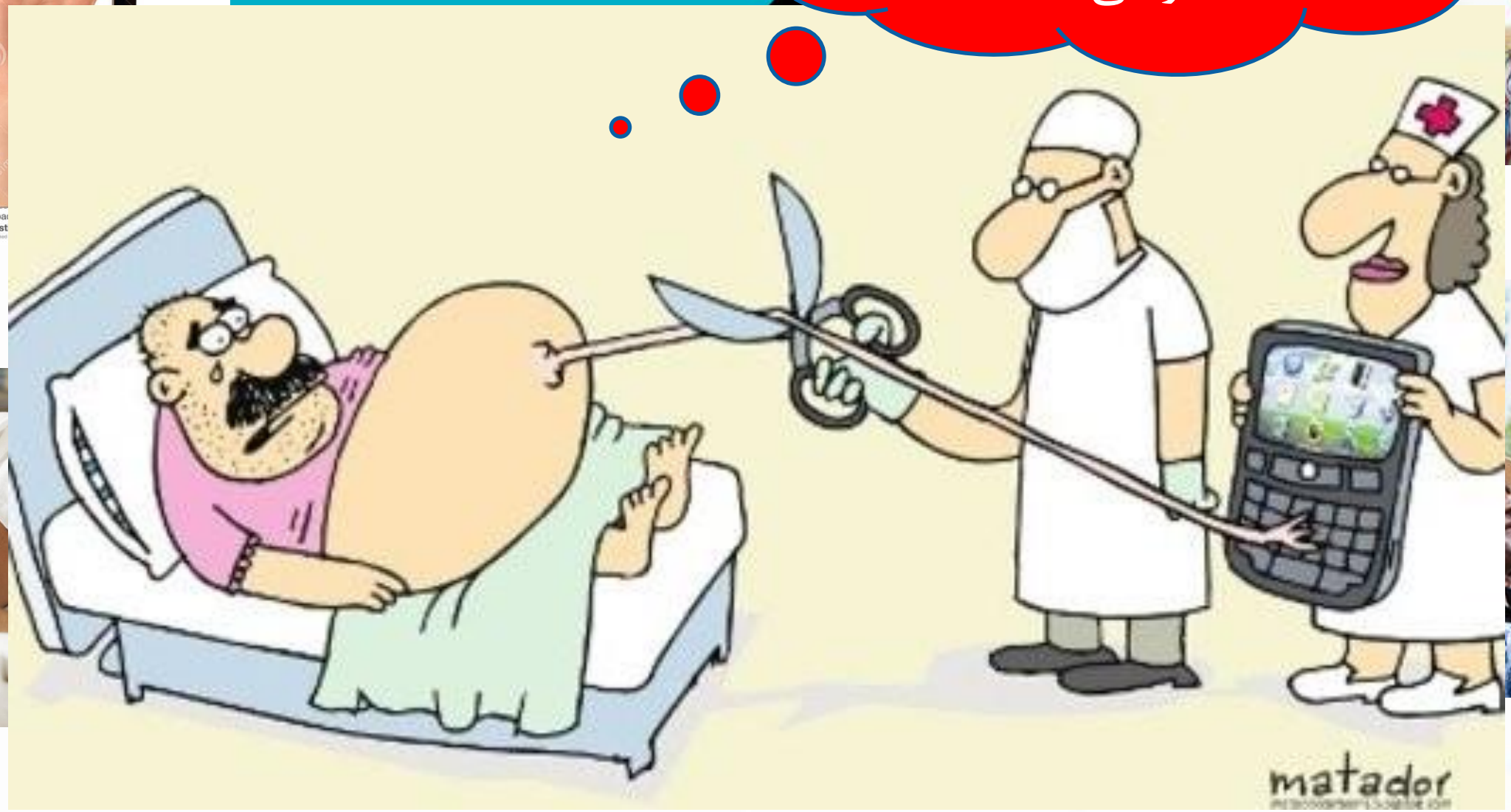

Evolution of the Mobile Phone
1982
1983
1991
1996
1999
2002
2014
WHAT'S NEXT?
IEEE

۳۰ سال
تکامل

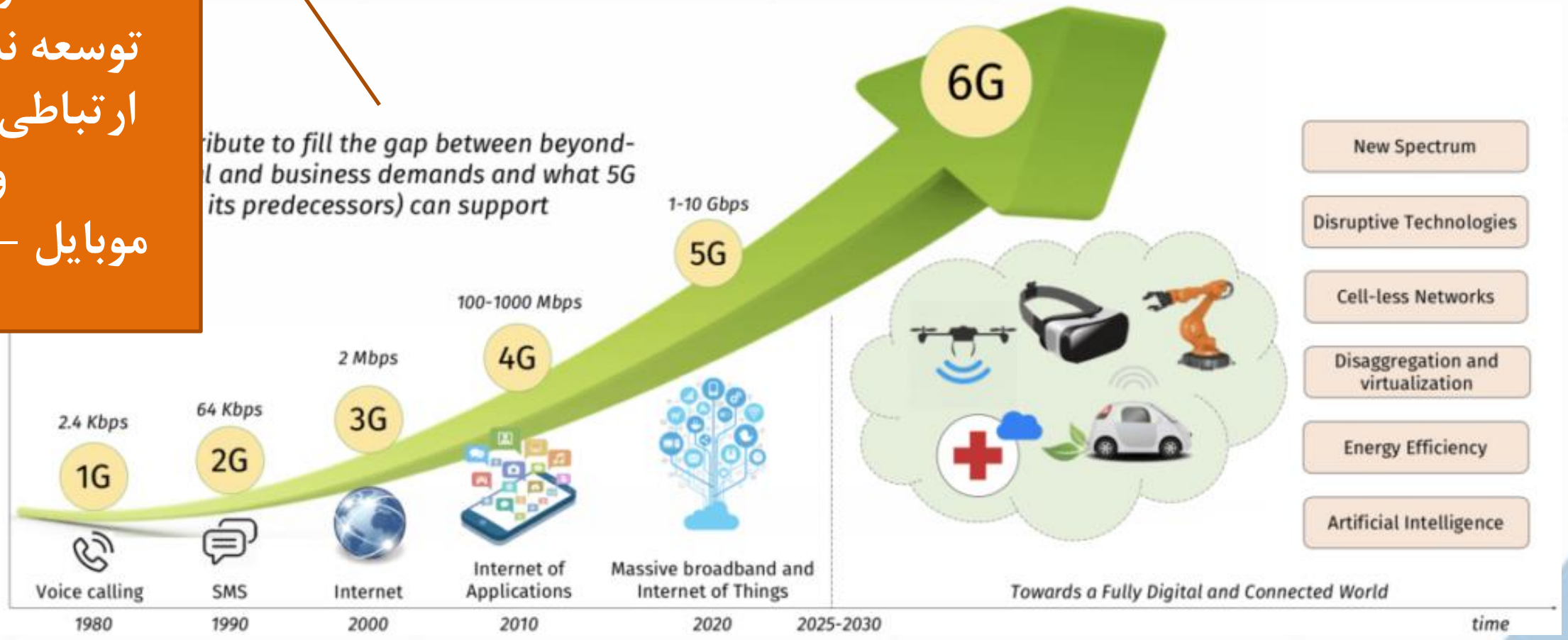


سامانه های هوشمند همراه

نوموفوبیا
(ترس از دست دادن
گوشی)



خط زمانی
توسعه نسل های
ارتباطی موبایل
و
موبایل – اینترنت



۴۰ سال تکامل 1G تا 5G

سرمایه سایبری چیست؟

- **ارزش اقتصادی بالای سرمایه** و در نتیجه **قابلیت تأثیرگذاری بر اقتصاد ملی**

- فراگیری افزون بر ۹۰ درصدی و در نتیجه
  - **قابلیت تأثیرگذاری بر اعتماد و آرامش عمومی**
  - قابلیت جریان سازی اجتماعی و سیاسی

- **حاوی اطلاعات ارزشمند**

سوال:

چرا سامانه های هوشمند (گوشی های هوشمند) واقعا نیازمند توجه هستند؟

چون:

سرمایه ملی هستند؟ واقعا؟ چطوری؟

یک برآورد حداقلی

۶۰میلیون گوشی هوشمند

×

۵ میلیون تومان متوسط قیمت هر گوشی هوشمند

+

اگر روی هر گوشی میزان اطلاعات هر شخص را به طور تقریبی **۲ میلیون تومان** برآورد کنیم (میزان پولی که یک شخص حاضر است بپردازد تا اطلاعات یا عکس های خانوادگی اش را که دزیده شده یا از بین رفته برگردد)

=۴۲۰ همت = ۹ میلیارد دلار

یک برآورد حداقلی

۶۰ میلیون گوشی هوشمند ✕ متوسط ۱۲۰ دقیقه استفاده روزانه از گوشی هوشمند × ۳۶۵ روز

=

۴۳۸۰۰ میلیون ساعت در سال

=

معادل هزینه کار ۲/۰۰۰/۰۰۰ نفر در سال

=

(با برآورد دستمزد هر ساعت ۵۰ هزار تومان)

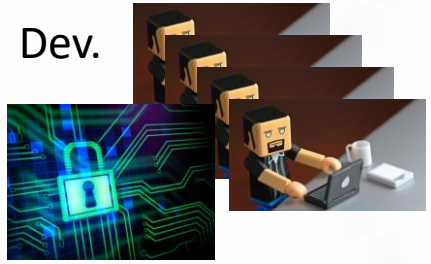
۴۸ میلیارد دلار ارزش وقت گذانده شده

یک برآورد حداقلی

- درآمد نفتی کشور ۳۸ میلیارد دلار
- هزینه استخراج ۱۸ میلیارد دلار هزینه برای برداشت ۲ میلیون بشکه
- هزینه صرف شده: ۵۷ میلیارد دلار (۱۴ درصد تولید ناخالص داخلی و ۴۰ درصد اقتصاد)

سوال:
چه میزان هزینه، تلاش و برنامه ریزی صرف پیشگیری، تشخیص، مقابله و حفاظت از سامانه های هوشمند همراه می شود؟؟

Dev.



Mobile apps&
App Stores
Web sites

IOT ،ابر

Security Gateway

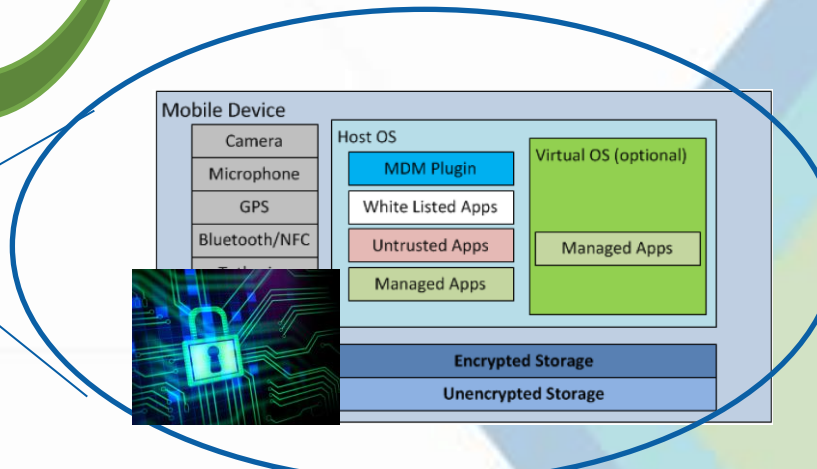
اینترانت

Internet

SMS

صوت

امنیت زیرساخت بوها سامانه‌های هوشمند همراه



تهدیدات برنامک های سامانه های هوشمند

پیشران ها در توجه به امنیت برنامک ها

There are approximately
7.21 billion smartphones
in the world

تهدیدات برنامک ها

- تقریباً از هر شش برنامه، یکی دارای آسیب‌پذیری‌های شناخته شده بوده است
- تقریباً دو سوم برنامه‌ها از رمزگذاری شکسته یا ضعیف استفاده می‌کردند

- https://www.nowsecure.com/blog/2025/04/30/525600-assessments-later-top-mobile-app-risks-since-2022/

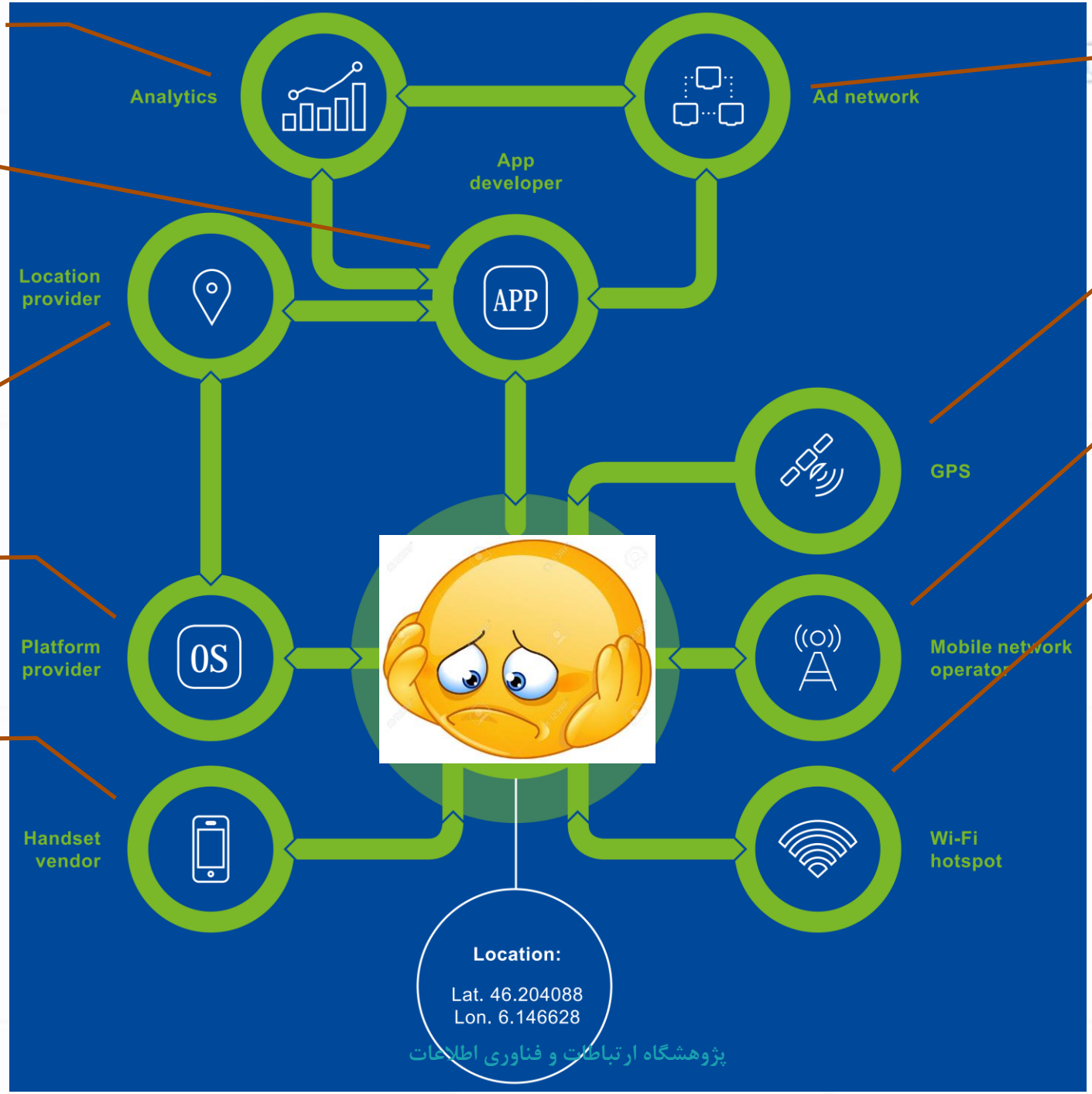
تحلیل گران

برنامه های کاربردی

فرهم کنندگان خدمات مکانی

فراهم سازنده سیستم عامل

سازنده گوشی



شبکه های تبلیغاتی

مکان یاب GPS

مکان یاب اپراتورها

مکان یاب شبکه وای فای

حریم خصوصی؟!

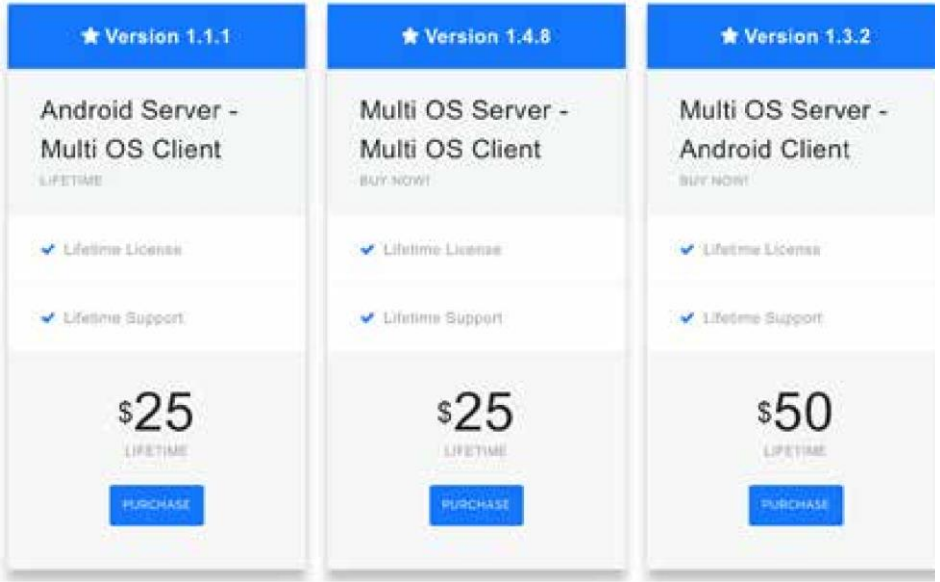
آیا گوشی شما آنچه را که شما در تلوزیون تماشا می کنید، می شنود؟!!!

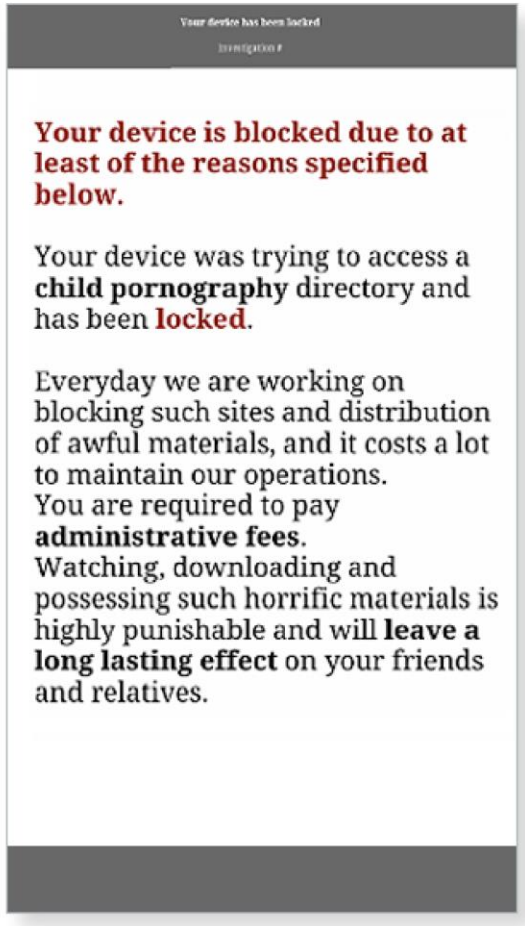
SilverPush یک برنامه تنها نیست، بلکه بد افزاری است که در سایر برنامک ها جاسازی شده و بدون اجازه کاربر اجرا می شود. تنها راه خلاصی از آن حذف برنامک مربوطه از روی گوشی است. توسط یک کمپانی هندی و برای شرکت های تلوزیونی ورادیویی برای بحث تبلیغات تولید شده است. اصوات را بعد از فعال سازی میکروفن و شناسایی آنها، برای متقاضیان مربوطه ارسال می کند.

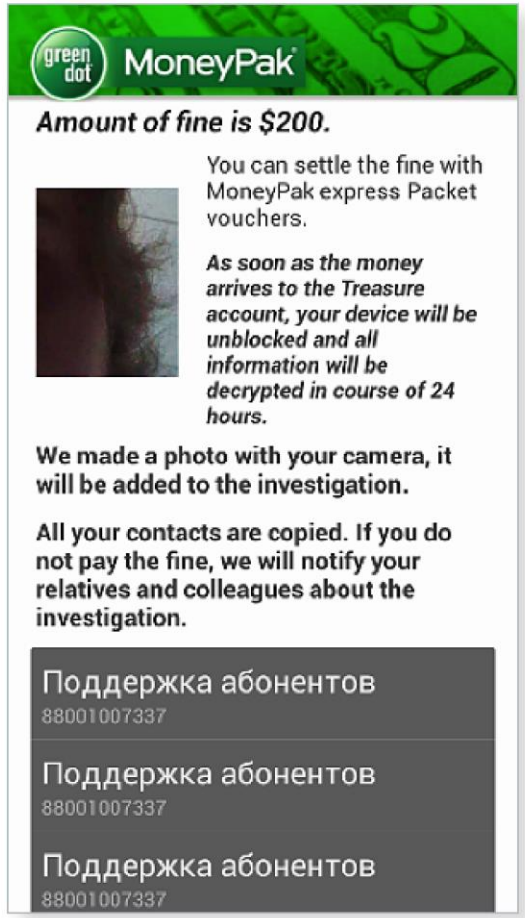
منبع: مک آفی

Remote access tools (RAT) and Ransomwares

خرید و فروش ابزار دسترسی از راه دور و رمز کننده اطلاعات





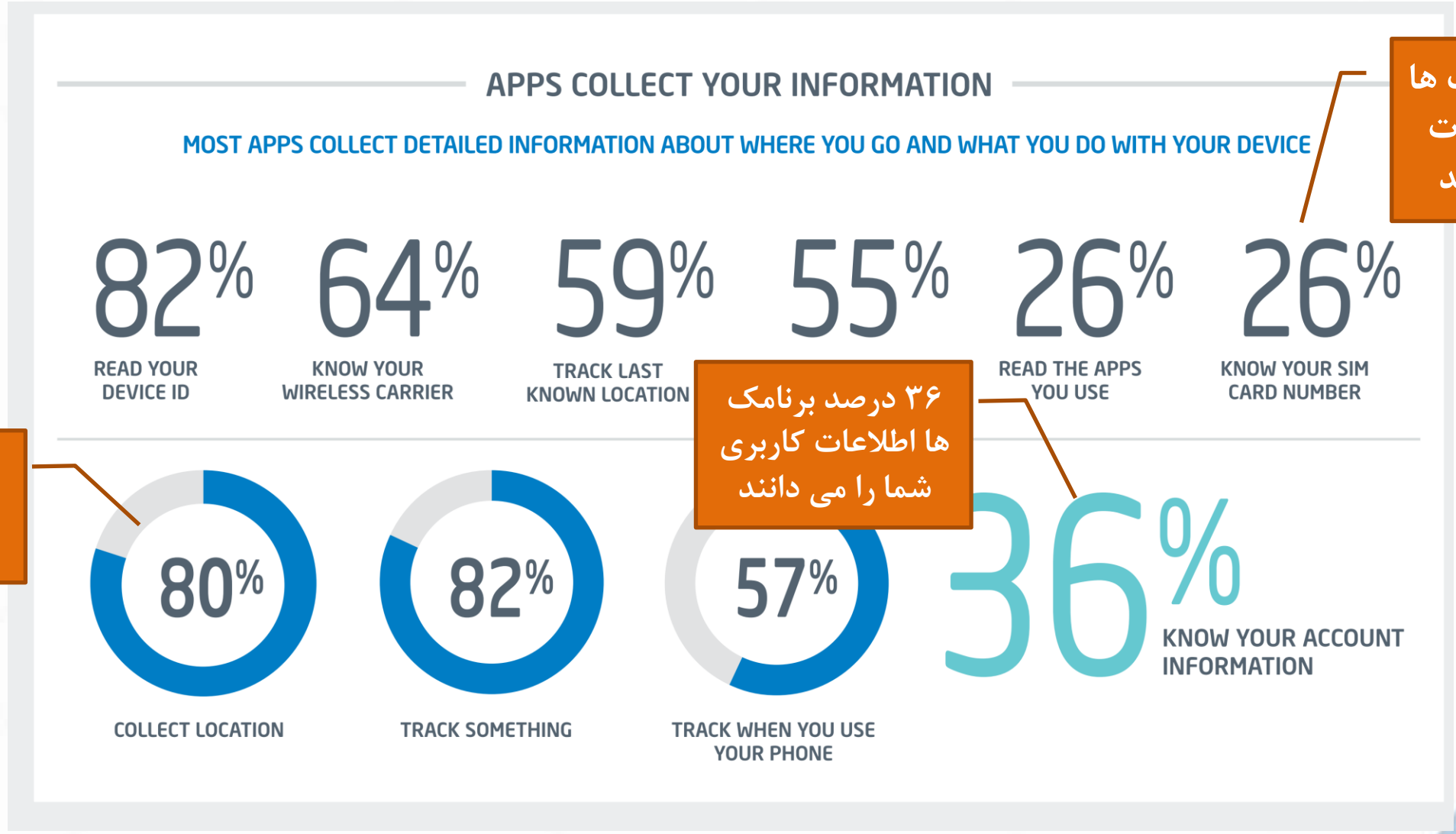


پیش بینی تعداد برنامک های موبایل دانلود شده در جهان از سال ۲۰۱۷ تا ۲۰۲۵

روی هر گوشی به طور میانگین ۳۰ برنامک نصب است. برنامک بیشتر یعنی افزایش تهدیدات

۱۵۰/۰۰۰/۰۰۰/۰۰۰ میلیارد برنامک
۵/۰۰۰/۰۰۰/۰۰۰ نفر دارای گوشی هوشمند

برنامک ها اطلاعات شما را جمع آوری می کنند
منبع: مک آفی



۲۶ درصد برنامک ها شماره سیم کارت شما را می دانند

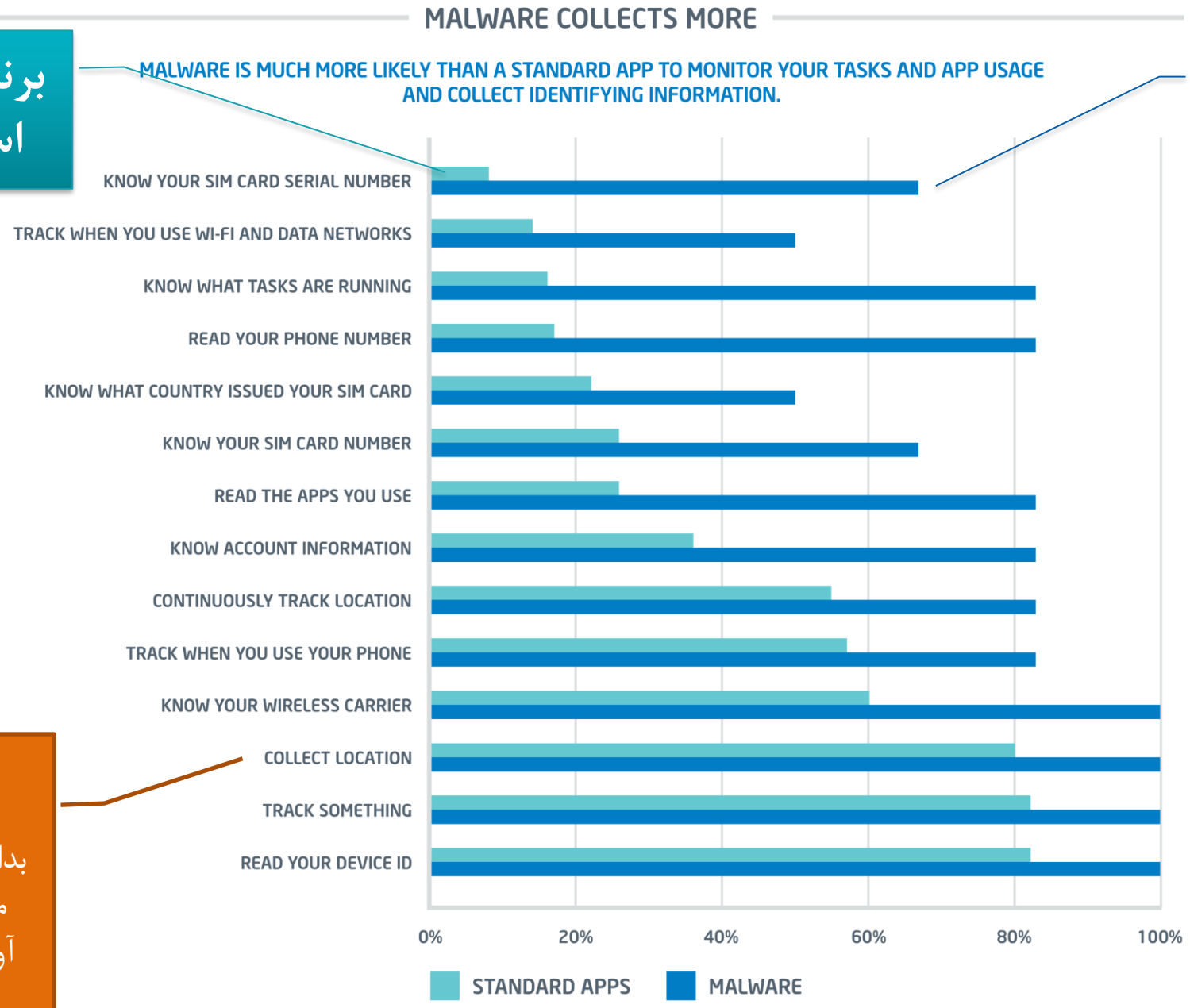
۳۶ درصد برنامک ها اطلاعات کاربری شما را می دانند

۸۰ درصد برنامک ها جمع آوری اطلاعات مکانی

برنامک های استاندارد

بد افزارها

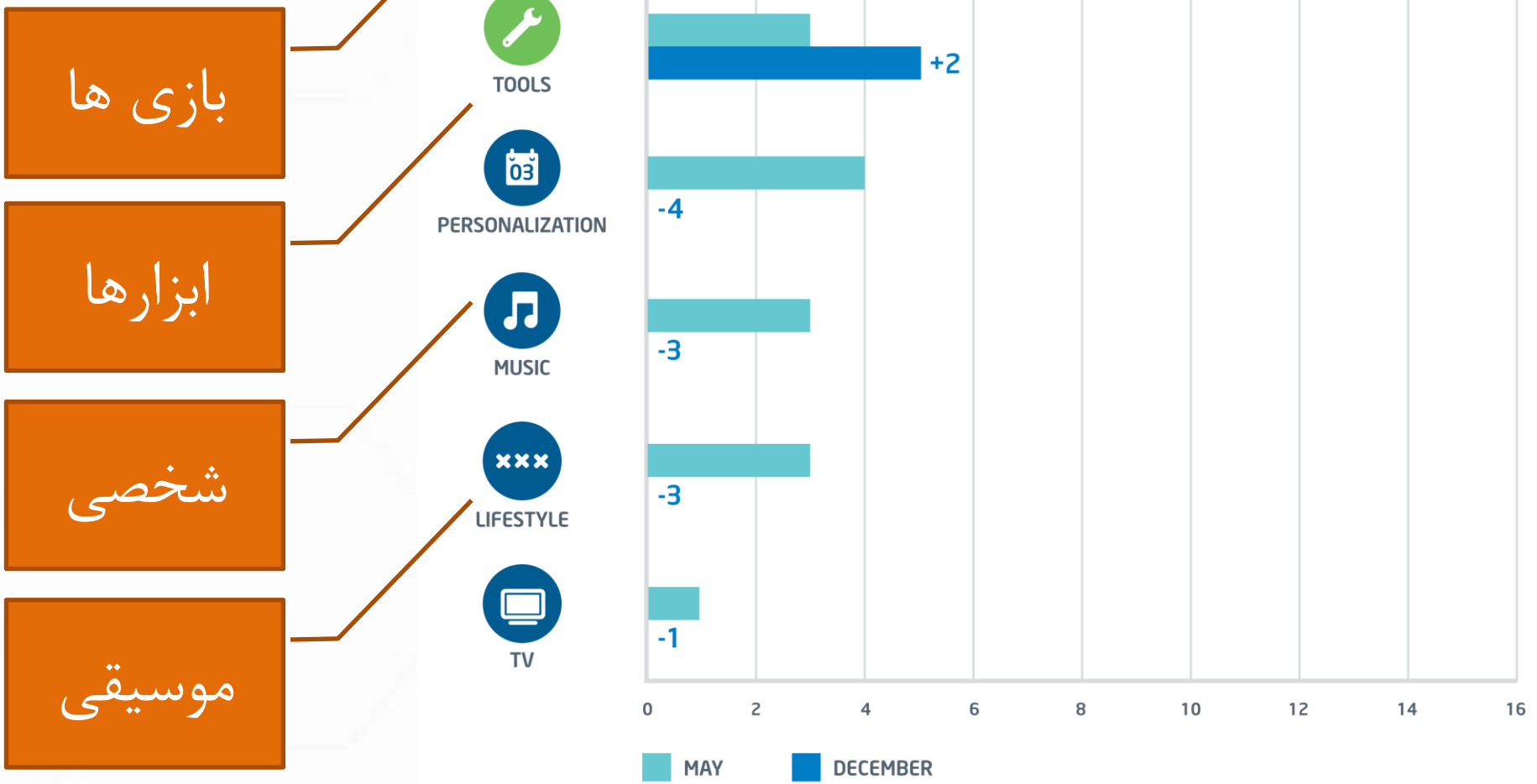
هر چه برنامک ها جمع آوری می کنند،
بد افزارها بیشتر از آنها جمع آوری می کنند!



۱۰۰ درصد بدافزارها اطلاعات مکانی را جمع آوری می کنند.

برنامک ها با بیشترین آلودگی

منبع: مک آفی



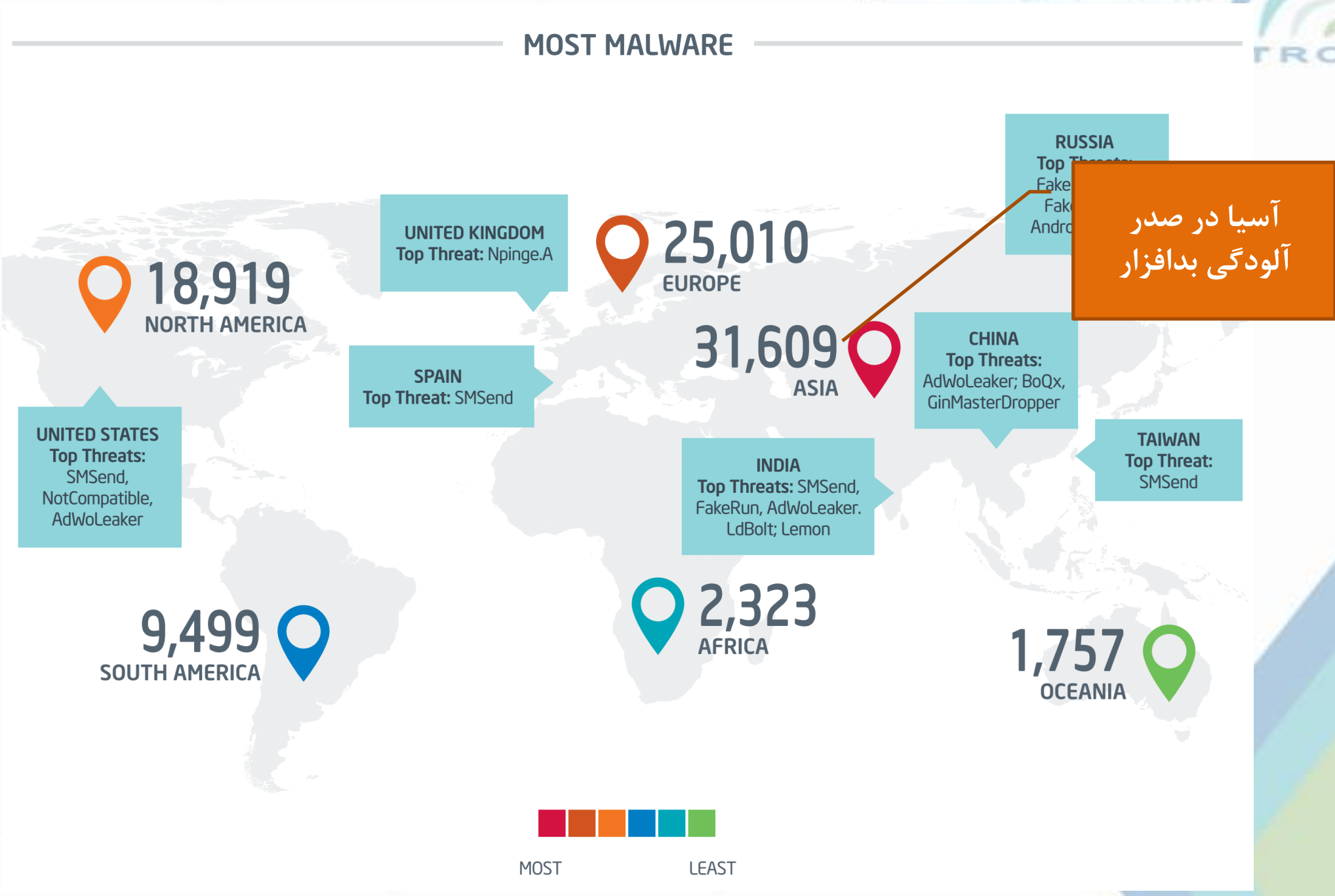
بازی ها

ابزارها

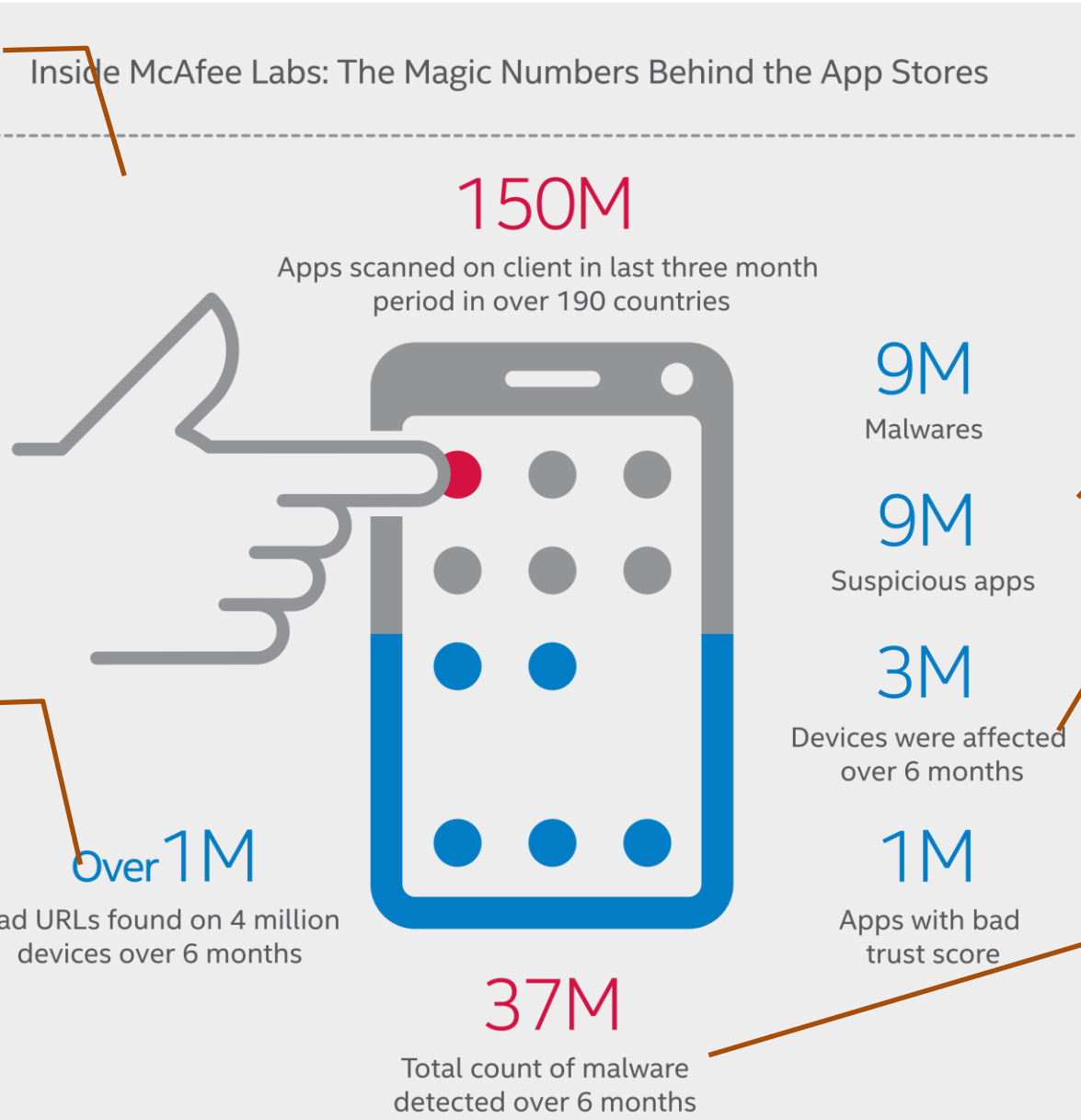
شخصی

موسیقی

پراکندگی جغرافیایی بد افزارها در دنیا منبع: مک آفی



۱۵۰ میلیون برنامک بر روی گوشی کاربران در ۱۹۰ کشور جهان توسط مک آفی پایش شده است



۹ میلیون بد افزار

۹ میلیون برنامک های مشکوک

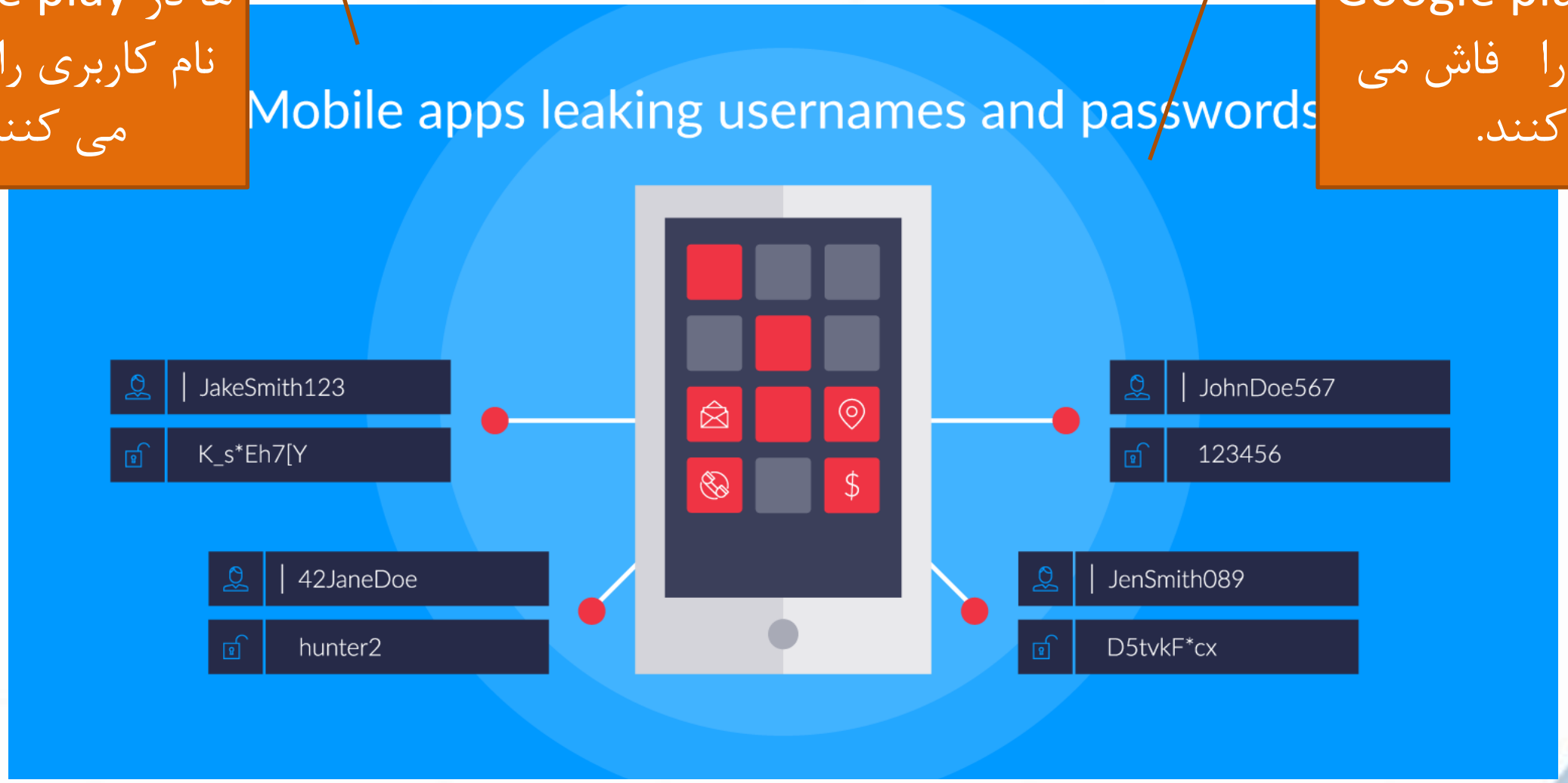
۳ میلیون گوشی در طی ۶ ماه آلوده شده اند

۳۷ میلیون بد افزار ظرف مدت ۶ ماه شناسایی شد.

طی ۶ ماه پایش، بیش از ۱ میلیون آدرس اینترنتی دارای مشکل بر روی ۴ میلیون گوشی یافت شده است.

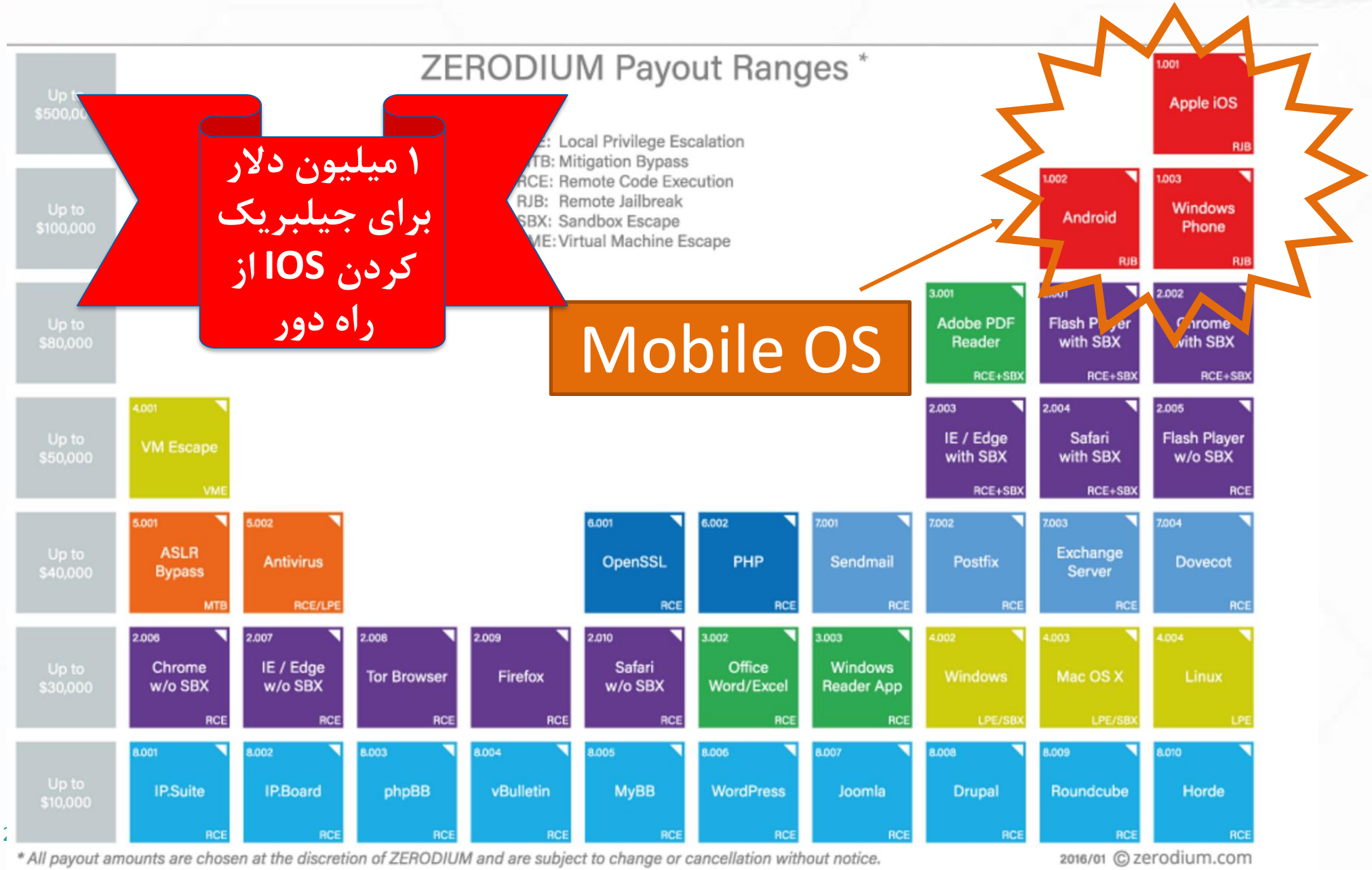
۶۰ درصد از برنامک ها در Google play نام کاربری را فاش می کنند.

۴۵ درصد از برنامک ها در Google play پسورد را فاش می کنند.



https://www.nowsecure.com/blog/2016/02/19/assessing-impact-of-leaky-apps-usernames-and-passwords/

داده ها در گوشی های هوشمند ارزشمندند



بد افزارها مساله اصلی در امنیت سامانه های هوشمند نیستند!

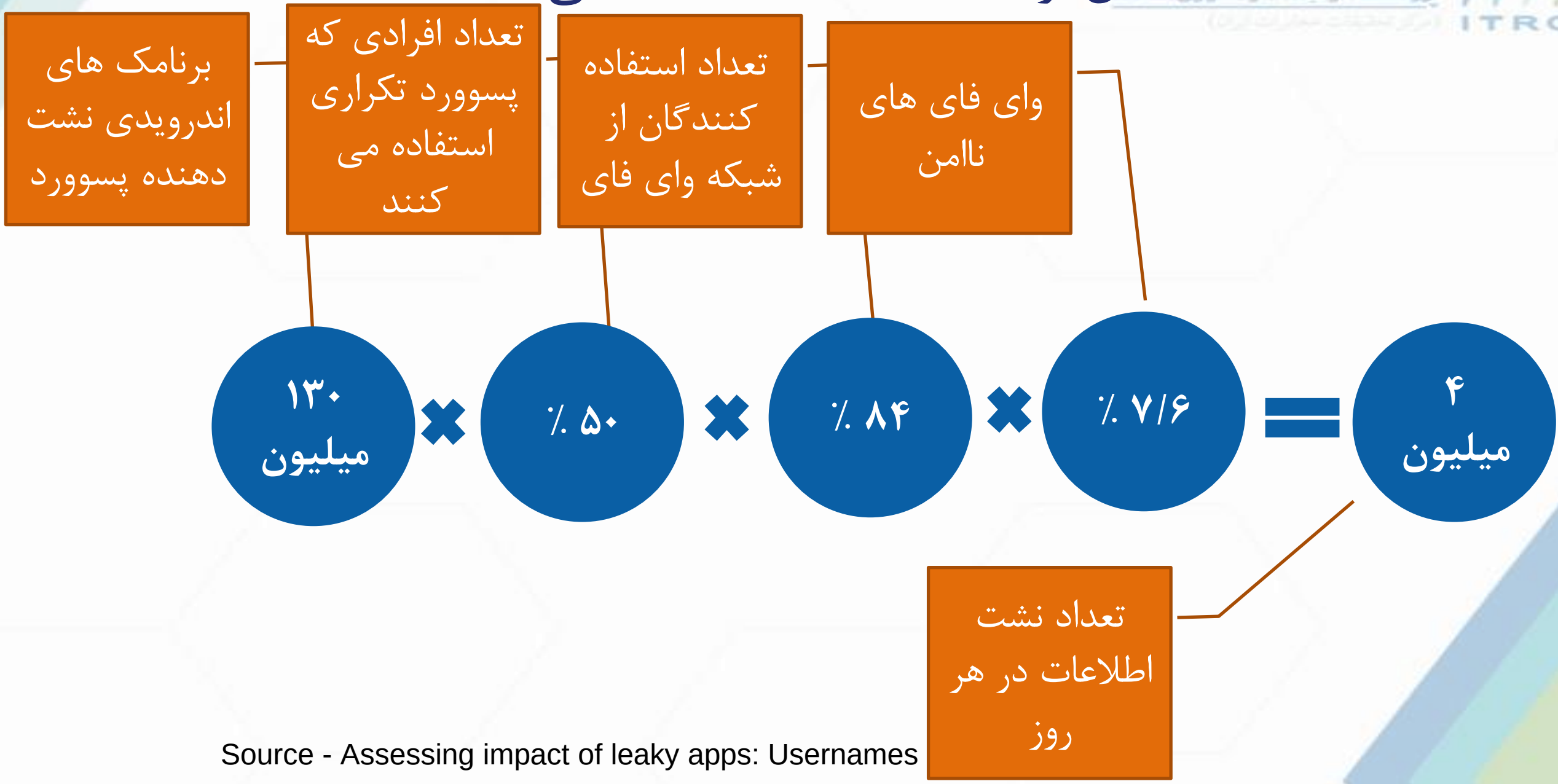
مساله فعلی امنیت در سامانه های هوشمند، خود برنامک ها هستند

برنامک های گوشی های هوشمند که داده های حساس و شخصی را افشا می کنند و نشتی اطلاعات دارند

≠

بد افزارها

مثال از نشت اطلاعات (مک آفی)



Source - Assessing impact of leaky apps: Usernames

در ماه ۵۰ درصد اتصالات در امریکا به WiFi نا امن صورت می گیرد که زمینه ساز هک شدن است.

In the United States

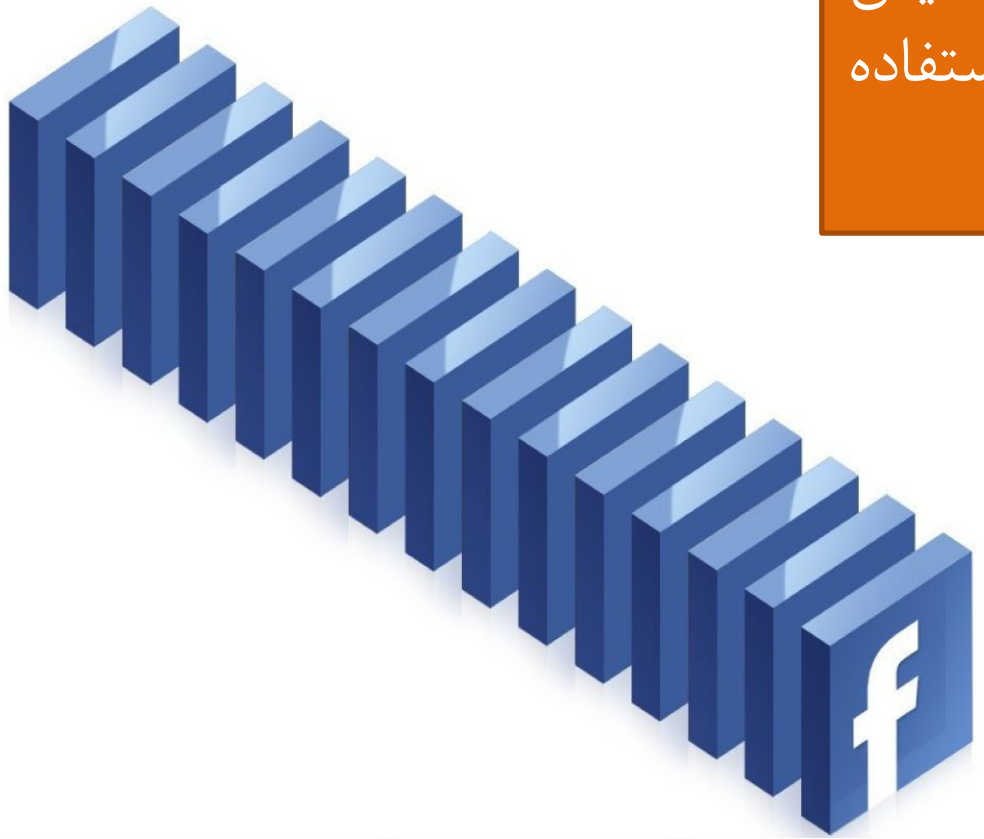
50%

of devices connect to insecure Wi-Fi at least once a month

کاربران نسخه های جدید برنامک ها را نصب و به روز رسانی نمی کنند

هم اکنون نسخه های مختلف برنامک فیس بوک در حال استفاده است!!

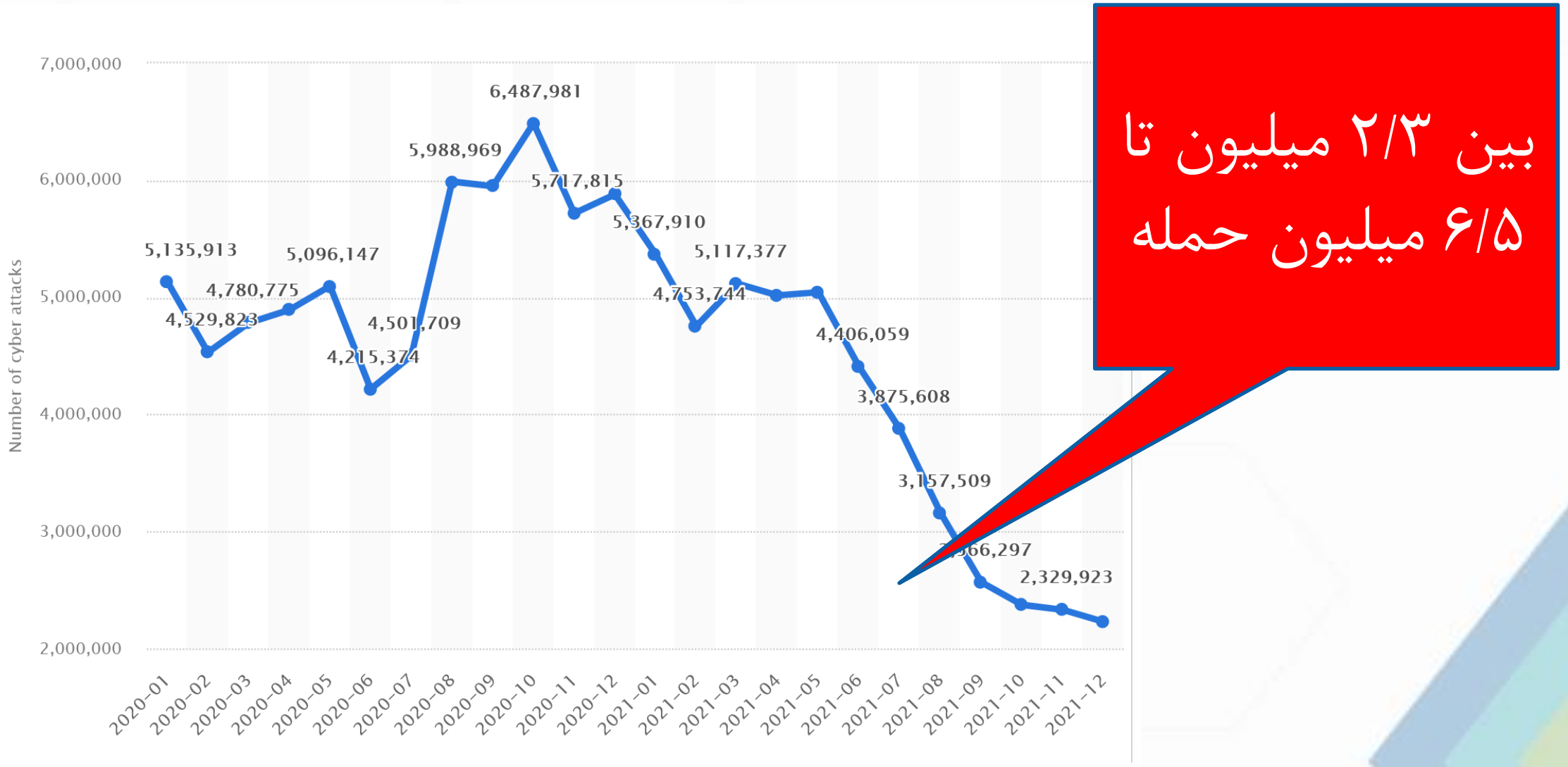
rs fail to upd ely manner



295

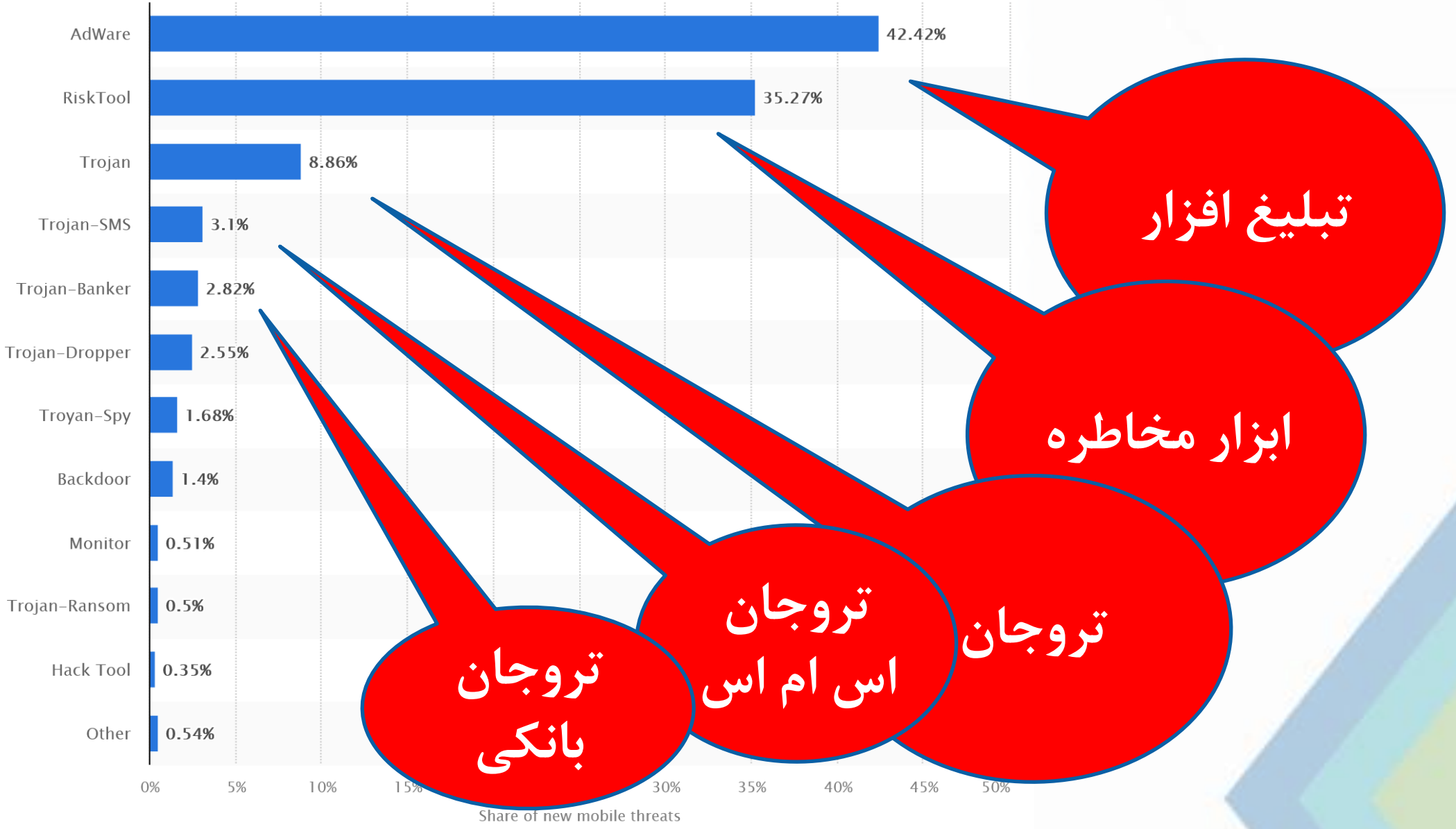
Versions of the Facebook app currently in use

تعداد حملات بر روی کاربران گوشی های هوشمند از ۲۰۲۱ تا ۲۰۲۲

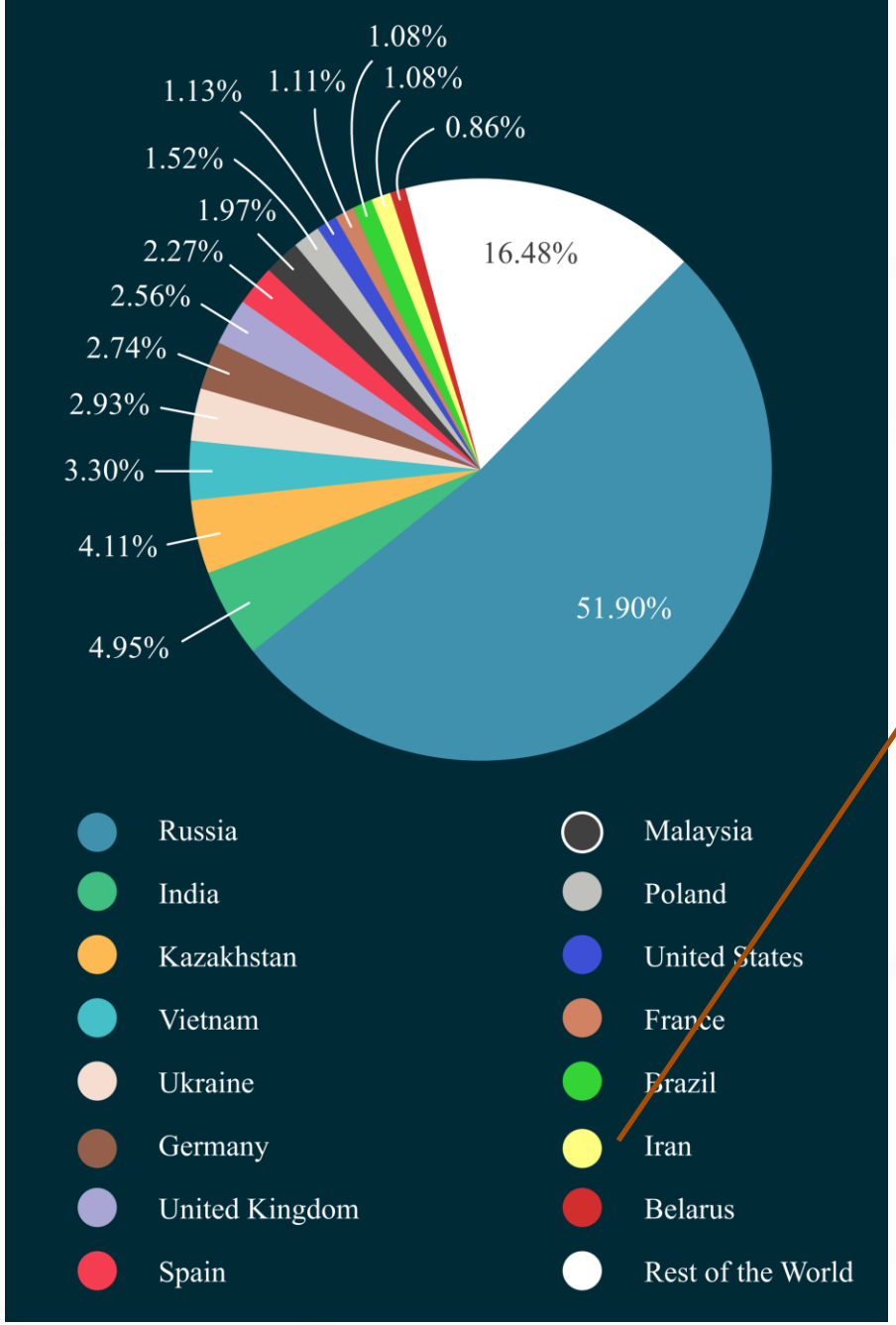


بین ۲/۳ میلیون تا ۶/۵ میلیون حمله

توزیع بدافزارها بر حسب نوع

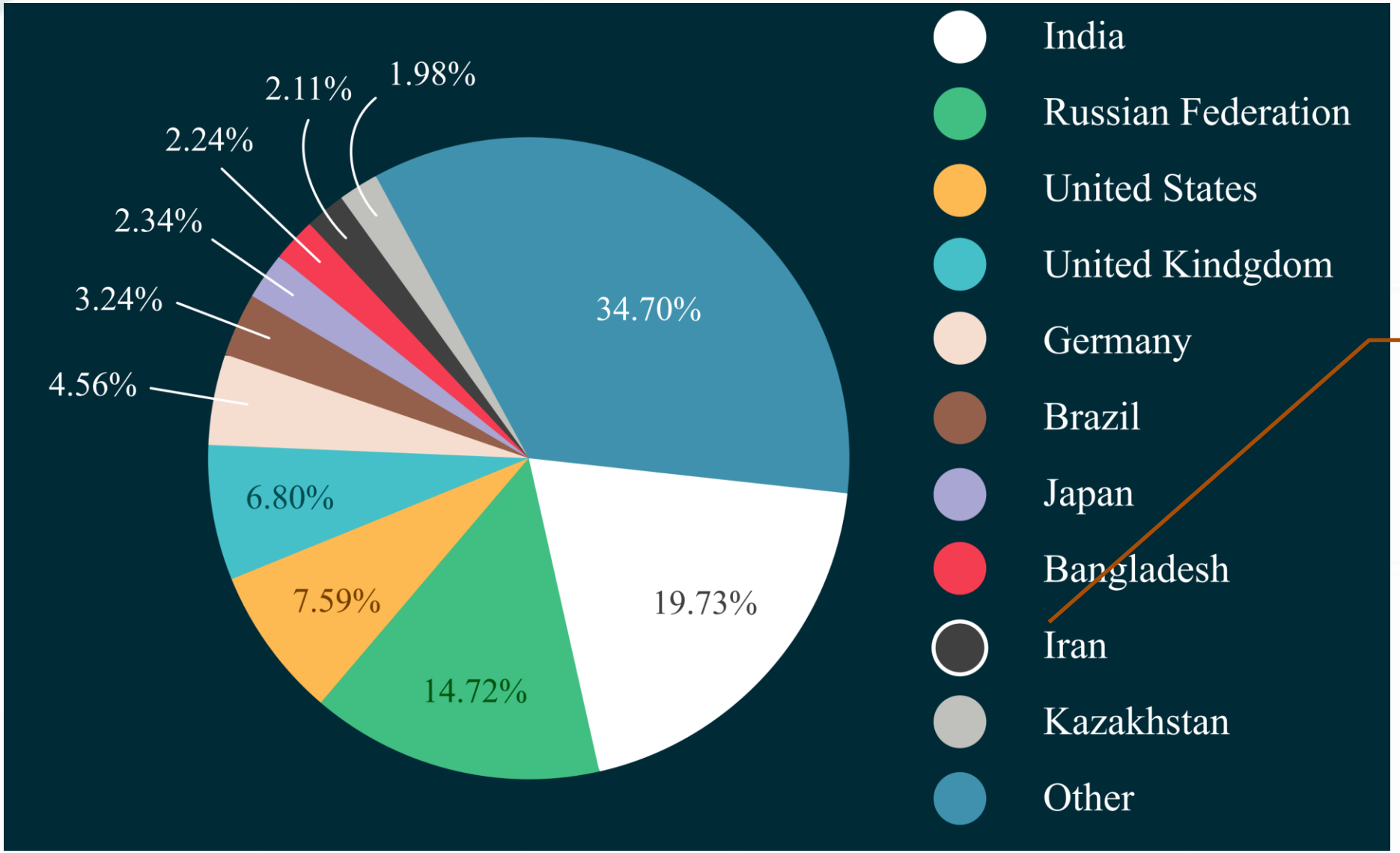


توزیع جغرافیایی حملات و کاربران برای ۱۵ کشور با بیشترین میزان شیوع



ایران

توزیع جغرافیایی جاسوس افزارهای شناسایی شده برای ۱۵ کشور



ایران

VPNs!

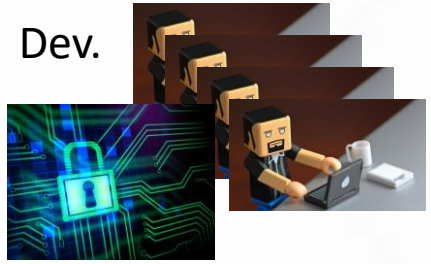
از بین ده عدد از بدترین متخلفان در بازار VPN، آمار داستان تلخی را بیان می کند:

۴۳٪ حاوی Adware بود
۱۷٪ دارای بدافزار بودند
۶٪ حاوی Riskware بود.
۵٪ حاوی نرم افزارهای جاسوسی است

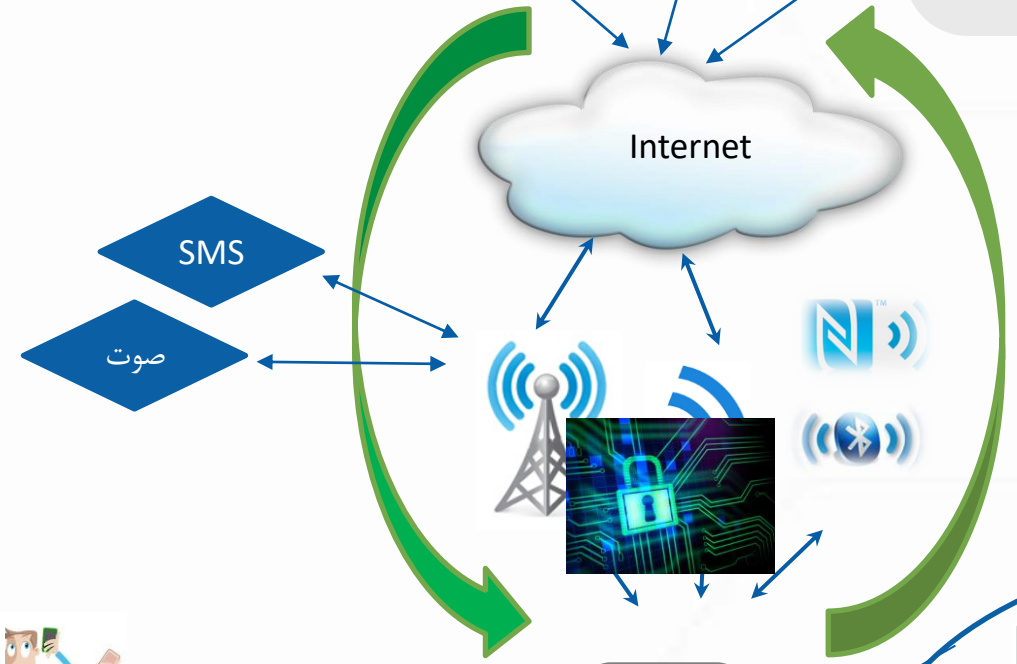
از یک مطالعه بر روی ۲۳۴ وی پی ان

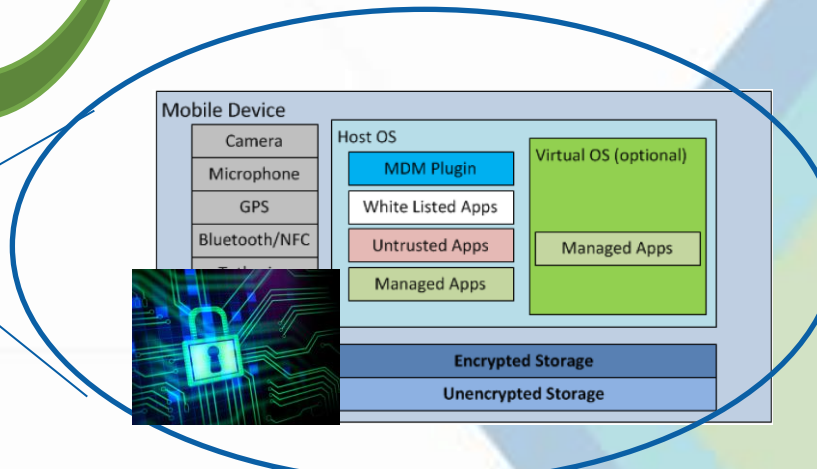
۳۸ درصد از برنامه‌های وی پی ان اندرویدی با بدافزار ساخته شده‌اند
۱۸ درصد موفق به رمزگذاری ترافیک اینترنت نشدند
۸۰٪ درخواست کردند که برنامه به اطلاعات حساس دسترسی داشته باشد



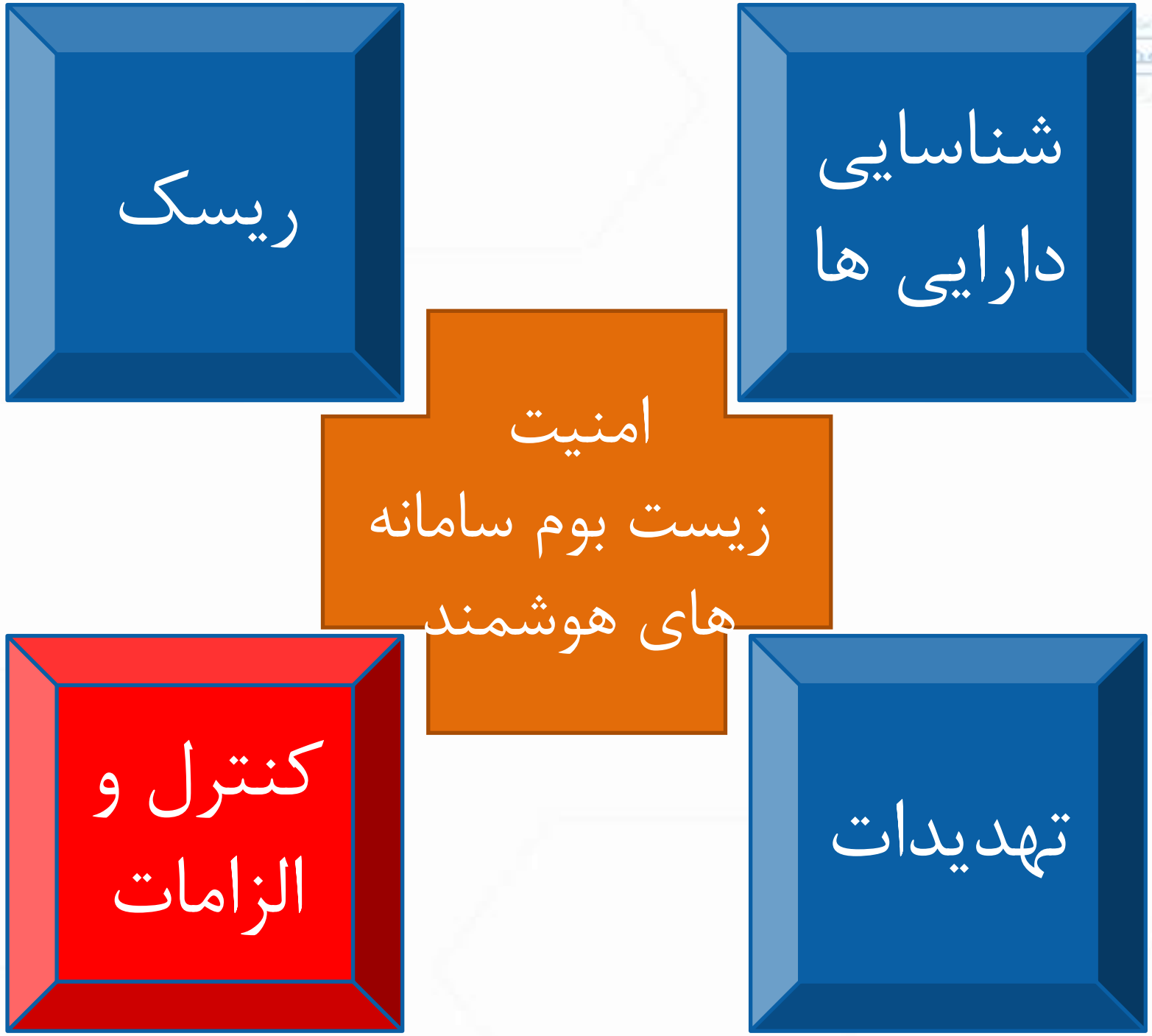


امنیت زیست بوم سامانه های هوشمند همراه





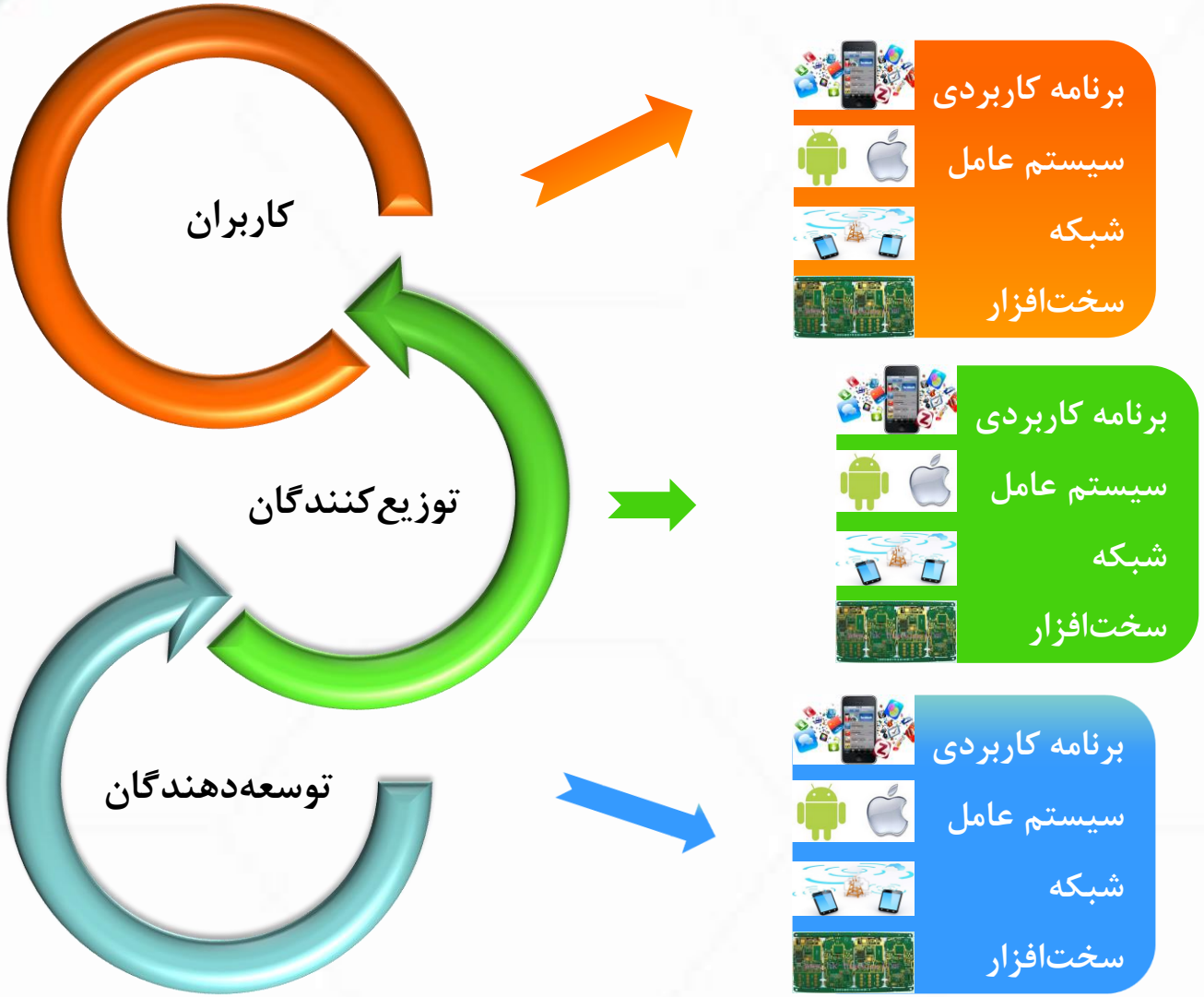
برای امنیت زیست بوم سامانه های هوشمند چه باید کرد؟


شناسایی دارایی ها
ریسک
امنیت زیست بوم سامانه های هوشمند
تهدیدات
کنترل و الزامات

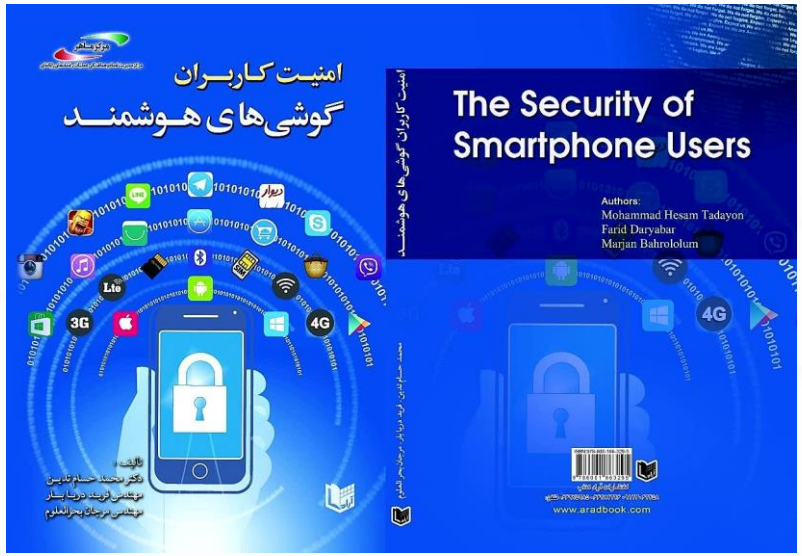
دارایی‌های یک سامانه هوشمند همراه

دارایی‌ها	توصیف	اهمیت
داده کاربر	دفترچه مخاطبان، پیشینه تماس، SMS/MMS، e-mail، تصاویر، صوت، اطلاعات بانکی، اطلاعات موقعیت مکانی، دفتر یادداشت، دستور کار و غیره	خیلی مهم
نرم‌افزار	برنامه‌های کاربردی از پیش نصب شده، برنامه‌های کاربردی نصب شده توسط کاربر، سیستم عامل و غیره	مهم
سخت‌افزار	واحد پردازش مرکزی (CPU) ، حافظه دسترسی تصادفی (RAM)، فلش، باتری و غیره	مهم

بازیگران کلیدی در بخشی از زیست بوم



لایه‌های کاربری



آخرین توصیه ها

- **امنیت Shift-Left با ممیزی هوش مصنوعی** :بررسی‌های امنیتی را در حین مرحله کدنویسی ادغام کنید، نه پس از آن.
- **Zero-Trust برایSDK ها** :هر کتابخانه شخص ثالثی را به دقت بررسی کنید.
- **تقویت وضعیت امنیت ابری** :آموزش اجباری برای توسعه‌دهندگان در مورد امنیت ابری الزامی است.
- **کمینه‌سازی داده به عنوان یک اصل طراحی** :داده‌هایی که به آن نیاز مطلق ندارید را جمع‌آوری نکنید.
- **آماده‌سازی برای تهدیدات خاص هوش مصنوعی**
- **آموزش پیشرفته کاربر** :به کاربران در مورد خطرات نصب برنامه از منابع غیرمعتبر (سایدلودینگ) هشدار دهید و به آن‌ها آموزش دهید که حملات فیشینگ پیچیده را شناسایی کنند.

با تشکر

محمد حسام تدین
tadayon@itrc.ac.ir

آسیب‌پذیری، تهدید، دفاع در برنامک‌های موبایلی

ارایه دهنده: محمد علی زمانی
m.zamani@itrc.ac.ir

اداره آزمایشگاه‌های امنیت، پژوهشکده امنیت اطلاعات و ارتباطات
پژوهشگاه ارتباطات و فناوری اطلاعات

خلاصه ارائه

- آسیب‌پذیری، تهدید، دفاع
- آسیب‌پذیری‌های متداول برنامک‌های موبایلی
- آسیب‌پذیری‌های نوین برنامک‌های موبایلی
- متدهای دفاعی
- راهبردهای OWASP

آسیب‌پذیری، تهدید، دفاع

Vulnerability	Threat (Risk)	Exploit
		
Weakness or flaw in technology, processes, or behavior.	Potential damage if a vulnerability is exploited.	Method used to exploit a vulnerability.

• آسیب‌پذیری

ضعف یا نقص در نرم‌افزار/سیستم

باگی در کد

به‌خودی‌خود خطر آنی ایجاد نمی‌کند اما زمینه را برای حمله مهیا می‌سازد

- تهدید

بهره‌برداری
آسیب‌پذیری

بالقوه

تهدید بدون
آسیب‌پذیری کاری از
پیش نمی‌برد؛ اما در
حضور ضعف، می‌تواند
منجر به حمله شود

• اکسپلویت

بهره‌برداری یا
به‌کارگیری
آسیب‌پذیری

پل ارتباطی بین
تهدید و حمله
واقعی

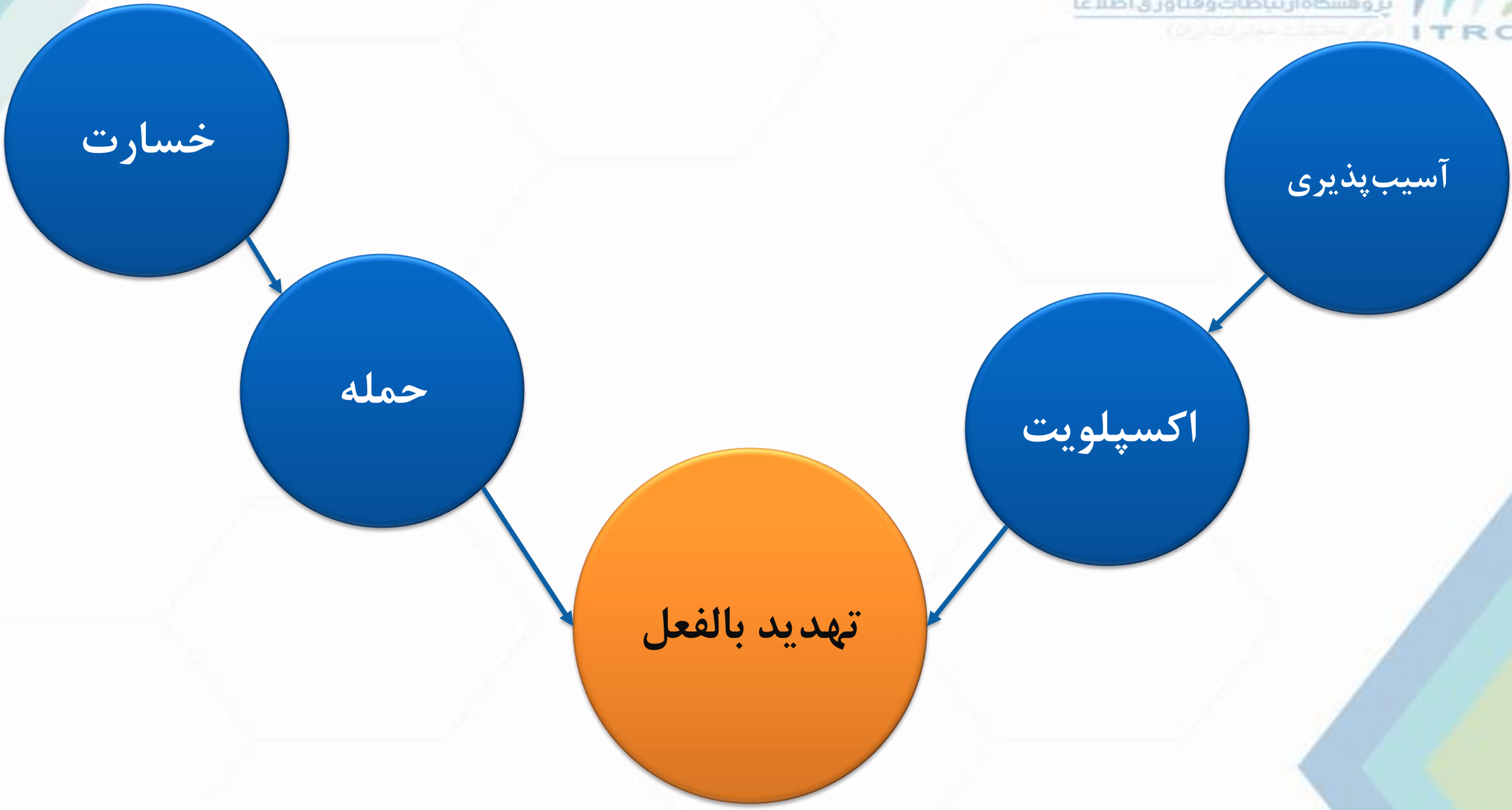
اسکریپت یا بدافزاری
که از باگ نرم‌افزار
برای نفوذ استفاده
می‌کند.

- دفاع

روش‌ها و ابزارهایی برای محافظت و کاهش ریسک

کنترل‌های پیشگیرانه و واکنشی

جلوگیری از تبدیل آسیب‌پذیری به حمله موفق و کاهش اثر تهدید


خسارت
حمله
آسیب‌پذیری
اکسپلویت
تهدید بالفعل

حملات متداول

ایده‌های جدید

OWASP™

پروژه

Open Web Application Security Project

- تاریخچه مختصر
- ساختار بنیاد
- اهداف اصلی
- جامعه جهانی
- پروژه‌های مشهور
- تأثیر در صنعت و استانداردسازی
- نقش در آموزش و همکاری

- **حملات متداول**

برنامک‌های موبایلی

- OWASP Mobile Top10
  - لیست ۱۰ مورد برتر
  - انتشار سالیانه

M1: Improper Credential Usage	استفاده نادرست یا مدیریت ضعیف اعتبارنامه‌ها
	ذخیره یا انتقال نامطمئن توکن‌ها / کلیدها
	استفاده از مکانیزم‌های احراز هویت ضعیف یا غیر استاندارد
	نشتی توکن‌ها در لاگ‌ها یا حافظه موقت
	عدم rotation امن یا پایان عمر مناسب برای کلیدها

M2: Inadequate Supply Chain Security	وابستگی به کتابخانه‌ها یا SDKهای شخص ثالث ناامن
	دستکاری در زنجیره تأمین نرم‌افزاری (مثال: آلوده شدن پکیج‌های مورد استفاده)
	عدم بررسی و تأیید امضای کتابخانه‌ها
	استفاده از نسخه‌های قدیمی یا آسیب‌پذیر وابستگی‌ها
	اتصال به منابع خارجی غیر قابل اعتماد

M3: Insecure Auth	تأیید هویت ضعیف یا مکانیزم‌های احراز هویت ناکامل
	کنترل دسترسی نامناسب به منابع حساس
	استفاده از توکن بسیار ساده یا قابل حدس
	نپذیرفتن اصول کنترل سمت سرور
	امکان تجاوز به دسترسی‌های سایر کاربران

M4: Insufficient Input / Output Validation	عدم اعتبارسنجی مناسب ورودی‌ها
	پذیرش داده‌های خطرناک یا نامطمئن از کاربر یا منابع خارجی
	عدم پاک‌سازی خروجی‌ها یا جلوگیری از تزریق
	ضعف در اعتبارسنجی پارامترها در API یا فیلترهای سمت سرور
	امکان دستکاری ورودی‌ها برای نفوذ یا خراب‌کردن منطق برنامه

بررسی چند حمله نوین

Zero Click Attacks – No interaction exploits

- بهره‌جویی از آسیب‌پذیری‌هایی که نیازی به اقدام کاربر ندارند

آسیب‌پذیری‌های زنجیره تأمین و کتابخانه‌های شخص ثالث

- مثلاً کتابخانه‌های SDKناامن یا مخرب
- مهاجمان با آلوده کردن پکیج‌های متن‌باز یا SDKهای تبلیغاتی محبوب، کد مخرب را به اپلیکیشن‌های بسیاری تزریق می‌کنند.
- XcodeGhost

Welcome to X

```
<string>OSX.XcodeGhost.A</string>
<key>LaunchServices</key>
<dict>
        <key>LSItemContentType</key>
        <string>com.apple.application-bundle</string>
</dict>
<key>Matches</key>
<array>
```

متدهای دفاع

ابزارهای تشخیص

- ابزارها

FRIDA

آزمایشگاه ها

- NowSecure
- یکی از آزمایشگاه های پیشگام و ارائه دهنده خدمات تخصصی ارزیابی امنیتی و ریسک برنامک موبایلی
- سازوکار مبتنی بر OWASP
- ارائه دهنده راهکارهای ارزیابی امنیتی خودکار برنامک های موبایلی
- دارای راهکارهای صنعت محور
  - سلامت
  - مالی
  - خرده فروشی
  - حاکمیتی
  - تکنولوژی و فین تک
- بهره مندی و به کار گیری سرویس های هوش مصنوعی در ارزیابی امنیتی نرم افزار

آزمایشگاه ها

- App Defense Alliance – ADA
- همکاری مشترک Google، ESET، Lookout و Zimperium
- تضمین امنیت گوگل پلی و همچنین اکوسیستم های گسترده‌تر.
- سازوکار مبتنی بر OWASP
- ارائه خدمت آزمایشگاه ارزیابی امنیتی هم به اکوسیستم برنامک های موبایلی و هم اکوسیستم ابری
- ارائه دهنده گواهی امنیتی در دو سطح مجزا: Verified Self Attest و Lab Eval
- ناظر بر ۶ آزمایشگاه امنیتی دیگر
  - Bishop Fox
  - DEKRA
  - Leviathan
  - NCC Group
  - NowSecure
  - TAC

آزمایشگاه ها

- Google Play Protect
- سرویس ارزیابی امنیت برنامک های موبایلی بارگذاری شده بر روی Google Play
- اطلاعاتی مبنی بر چارچوب تست را منتشر نکرده است.
- امکان اسکن پیش از نصب
- قابلیت شناسایی رفتارهای غیرمجاز در پس‌زمینه، دسترسی به اطلاعات حساس بدون مجوز، و بهره‌برداری از آسیب‌پذیری‌های امنیتی
- بهره وری از سرویس های هوش مصنوعی در زمینه ارزیابی امنیتی
- توسعه الگوریتم های تشخیص ویروس
- توسعه فناوری های تحلیل رفتاری

- آزمایشگاه ها

ITRC
وزارت ارتباطات و فناوری اطلاعات
پژوهشگاه ارتباطات و فناوری اطلاعات

راهبردها و رهنمودهای

OWASP

MASTG

Mobile Application Security Testing Guide

Sven Schleier
Bernhard Mueller
Carlos Holguera
Jeroen Willemsen

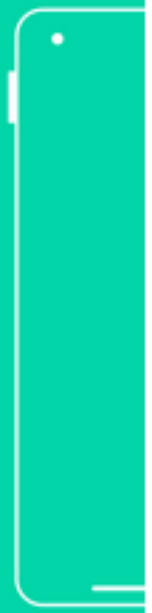
OWASP

- Mobile Application Security Testing Guide (MASTG)
- نسخه 1.7
- سند جامع و راهبردی برای الزامات امنیتی جهت تست برنامک های موبایلی
- تست های جامع مشتمل بر ۸ دسته کلی
- ابزار محور
- تمرکز عملیاتی و تحلیل عمیق
- همگام با MASVS

```java
import javax.crypto.spec.*;
import javax.crypto.*;
import java.security.*;
import android.util.*;

public class Encrypt
{
    private static byte[] keyBytes;

    static {
        Encrypt.keyBytes = new byte[] { 7, 3, 4, 5, 6, 7, 8, 9, 16, 17, 18, 9, 20, 21, 15, 1, 10, 11, 12, 13, 14,
    }

    public static String decrypt(final String s) throws Exception {
        final SecretKeySpec secretKeySpec = new SecretKeySpec(Encrypt.keyBytes, "AES");
        final Cipher instance = Cipher.getInstance("AES");
        instance.init(2, secretKeySpec);
        return new String(instance.doFinal(Base64.decode(s.getBytes(), 0)));
    }

    public static String encrypt(final String s) throws Exception {
        final SecretKeySpec secretKeySpec = new SecretKeySpec(Encrypt.keyBytes, "AES");
        final Cipher instance = Cipher.getInstance("AES");
        instance.init(1, secretKeySpec);
        return new String(Base64.encode(instance.doFinal(s.getBytes()), 0));
    }
}
```

MASVS

Mobile Application Security Verification Standard



Sven Schleier
Bernhard Mueller
Jeroen Beckers
Carlos Holguera
Jeroen Willemsen

OWASP

- Mobile Application Security Verification Standard (MASVS)
- هشت حوزه و دسته بندی همگام با MASTG
- قابلیت تطابق با مدل تهدید برنامک ها
- سطح بندی امنیتی
- استاندارد معماری، توسعه محور و ارزیاب پذیر

MASVS–Storage

- ارزیابی امنیت ذخیره سازی داده ها در برنامه های موبایلی
- آیا داده‌های حساس مانند اطلاعات شخصی، رمزهای عبور و داده‌های مالی به‌طور ایمن در دستگاه موبایل ذخیره می‌شوند یا خیر
- آزمون‌ها شامل بررسی نحوه مدیریت کلیدهای رمزنگاری، شیوه‌های ذخیره‌سازی ایمن، و تضمین این که داده‌ها به‌طور امن از برنامه به سرور و بالعکس منتقل می‌شوند

MASVS–Crypto

- ارزیابی استفاده صحیح از رمزنگاری در برنامه‌های موبایلی
- آیا برنامه از الگوریتم‌های رمزنگاری قوی و استاندارد برای محافظت از داده‌های حساس استفاده می‌کند
- آیا مدیریت کلیدهای رمزنگاری به‌صورت امن انجام می‌شود

MASVS–Auth

- آزمون‌های مرتبط با احراز هویت و مدیریت نشست‌ها در برنامه‌های موبایلی
- بررسی فرآیندهای ورود به سیستم و مدیریت نشست‌ها
- حصول اطمینان از پیاده سازی مکانیزم‌های احراز هویت و مدیریت نشست به‌طور ایمن و مؤثر

MASVS–Network

- امنیت ارتباطات شبکه‌ای برنامه‌های موبایلی
- آیا داده‌ها به‌طور امن از طریق شبکه‌ها منتقل می‌شوند یا خیر

MASVS–Platform

- ارزیابی نحوه تعامل برنامه با سکوی سیستم عامل موبایل iOS یا Android
- آیا برنامه به‌طور ایمن از منابع سیستم‌عامل، APIها و دسترسی به اطلاعات سیستم استفاده می‌کند
- آیا از قابلیت‌های امنیتی سیستم‌عامل به‌درستی استفاده شده است

MASVS–Code

- تحلیل کیفیت کد و پیاده‌سازی تکنیک‌های جلوگیری از بهره‌برداری در برنامه‌های موبایلی
- آیا از بهترین شیوه‌های برنامه‌نویسی برای جلوگیری از آسیب‌پذیری‌هایی مانند حملات تزریق کد، حملات سرریز بافر و حملات XSS استفاده شده است یا خیر

MASVS–Resilience

- ارزیابی توانایی برنامه در مقاومت در برابر مهندسی معکوس و دستکاری کد
- که آیا برنامه از روش‌های امن برای جلوگیری از مهندسی معکوس استفاده می‌کند
- ارزیابی می‌شود که آیا برنامه در برابر حملات مانند دستکاری داده‌ها و تزریق کد مقاوم است یا خیر

MASVS–Privacy

- ارزیابی حریم خصوصی در برنامه‌های موبایلی
- ارزیابی نحوه جمع‌آوری، ذخیره‌سازی، و اشتراک‌گذاری داده‌ها

سپاس از توجه و نگاه ارزشمند شما

محمد علی زمانی
m.zamani@itrc.ac.ir

بخش دوم: بررسی حمله‌ی تزریق کد در برنامه‌های اندرویدی

ارائه دهنده: محمدحسین هاتفی

مقدمه

- درباره‌ی چه چیزی صحبت می‌کنیم؟
  - وارد کردن و اجرای کد مخرب داخل اپلیکیشن برای دسترسی غیرمجاز یا تغییر رفتار برنامه
- سناریوهای رایج
  - دکس لودینگ
    - بارگذاری کد مخرب از حافظه یا شبکه
    - پویا و در زمان اجرا
    - دشواری تشخیص توسط آنتی‌ویروس
  - دستکاری اسمالی و ری‌پک برنامه
    - تزریق کد مخرب در فایل برنامه
    - ایستا و پیش از نصب
    - آسیب به برند و اعتماد کاربر

مقدمه

- در این کارگاه
  - آشنایی با حملات پیشین تزریق کد
  - بررسی حمله‌ی تزریق کد پویا
    - آشنایی با روند کامپایل برنامه‌های اندرویدی
    - درباره‌ی دکس
    - بررسی روال تزریق کد پویا از طریق دکس لودینگ
  - بررسی حمله‌ی تزریق کد ایستا
    - آشنایی با روند مهندسی معکوس برنامه‌های اندرویدی
    - آشنایی با اسمالی
    - بررسی یک سناریوی دستکاری اسمالی
    - آشنایی با روش‌های پیشگیری

مقدمه

- نمونه‌های پیشین حمله

Joker –

- حمله‌ی دکس لودینگ
- کشف در سال ۲۰۲۰
- فقط در موج اول ۴۷۲,۰۰۰ نصب از پلی‌استور
- دی‌کد و بارگذاری ماژول‌های مخرب از حافظه یا شبکه

Agent Smith –

- یک کمپین بدافزاری تزریق کد ایستا
- کشف در سال ۲۰۱۹
- آلوده کردن نزدیک به ۲۵ میلیون دستگاه
- پچ و ری‌پک اپ‌های کاربر و نصب با استفاده از آسیب‌پذیری janus

Anubis –

- حمله‌ی دکس لودینگ
- کشف در سال ۲۰۱۹
- بیش از ۱۷,۴۹۰ اپ شناسایی شده
- دریافت ماژول‌های مخرب از شبکه و بارگذاری

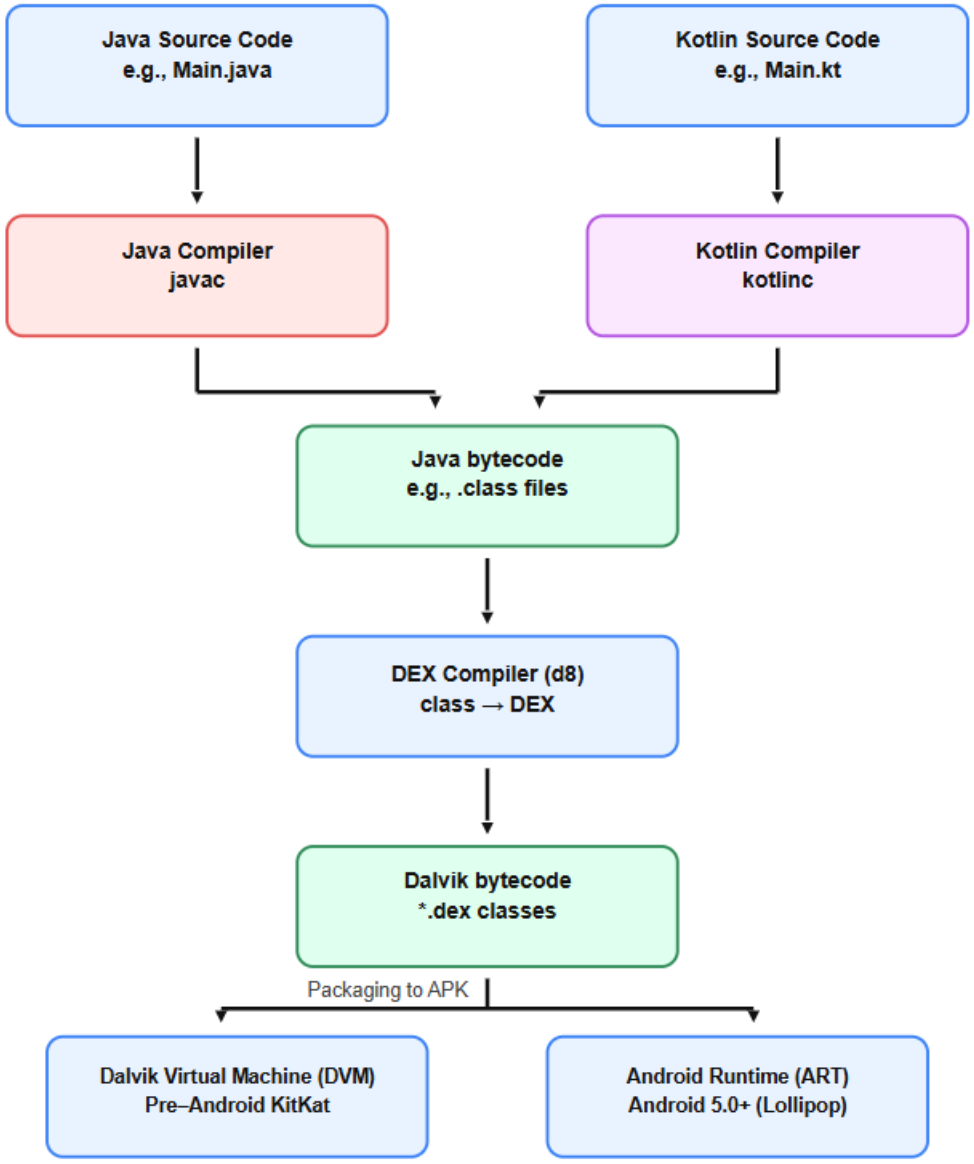
WhatsApp/Telegram تروجان‌شده –

- حمله‌ی تزریق کد ایستا
- کشف در سال ۲۰۲۳
- توزیع نسخه‌های غیررسمی و دستکاری‌شده در وب/کانال‌ها
- جابه‌جا کردن آدرس کیف‌پول رمزارز با آدرس مهاجم

تزریق کد پویا

حمله از طریق دکس لودینگ

تزریق کد پویا



- روند کامپایل برنامه‌های اندرویدی
  - کد جاوا – کاتلین
  - بایت‌کد جاوا
  - فایل دکس
  - محیط اجرای اندروید (ART) جایگزین jvm

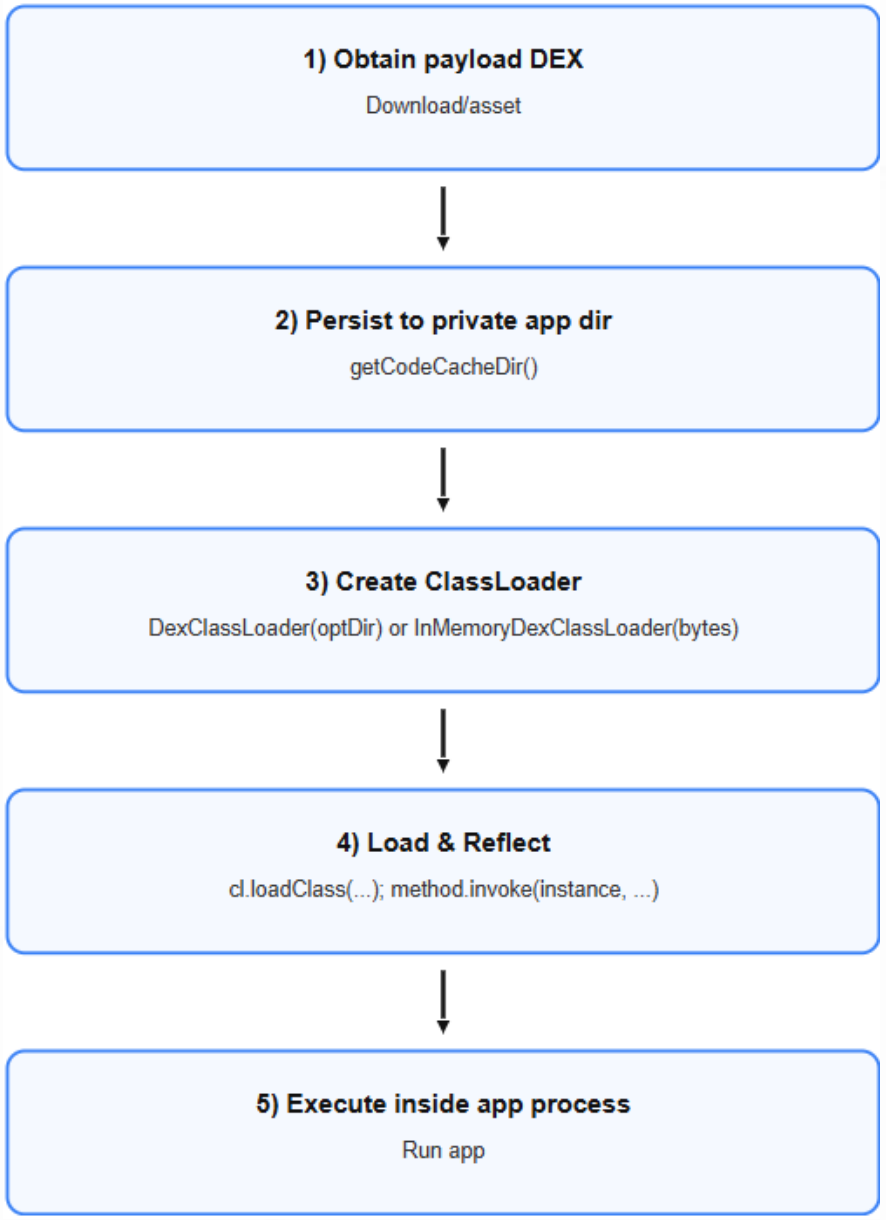
تزریق کد پویا

- درباره‌ی دکس
  - Dalvik Executable
  - فرمت بایت‌کد اندروید برای اجرای کد جاوا/کاتلین روی محیط اجرای اندروید
  - داخل بسته‌ی اپ به صورت فایل classes.dex
  - در حالت مولتی‌دکس به صورت:

    classes2.dex, classes3.dex, ...

- چرا دکس؟
  - داده‌های اشتراکی (string/type/method ids) برای کاهش حجم
  - لود سریع‌تر به علت mmap قسمت‌هایی از فایل
  - پشتیبانی از مولتی‌دکس

تزریق کد پویا



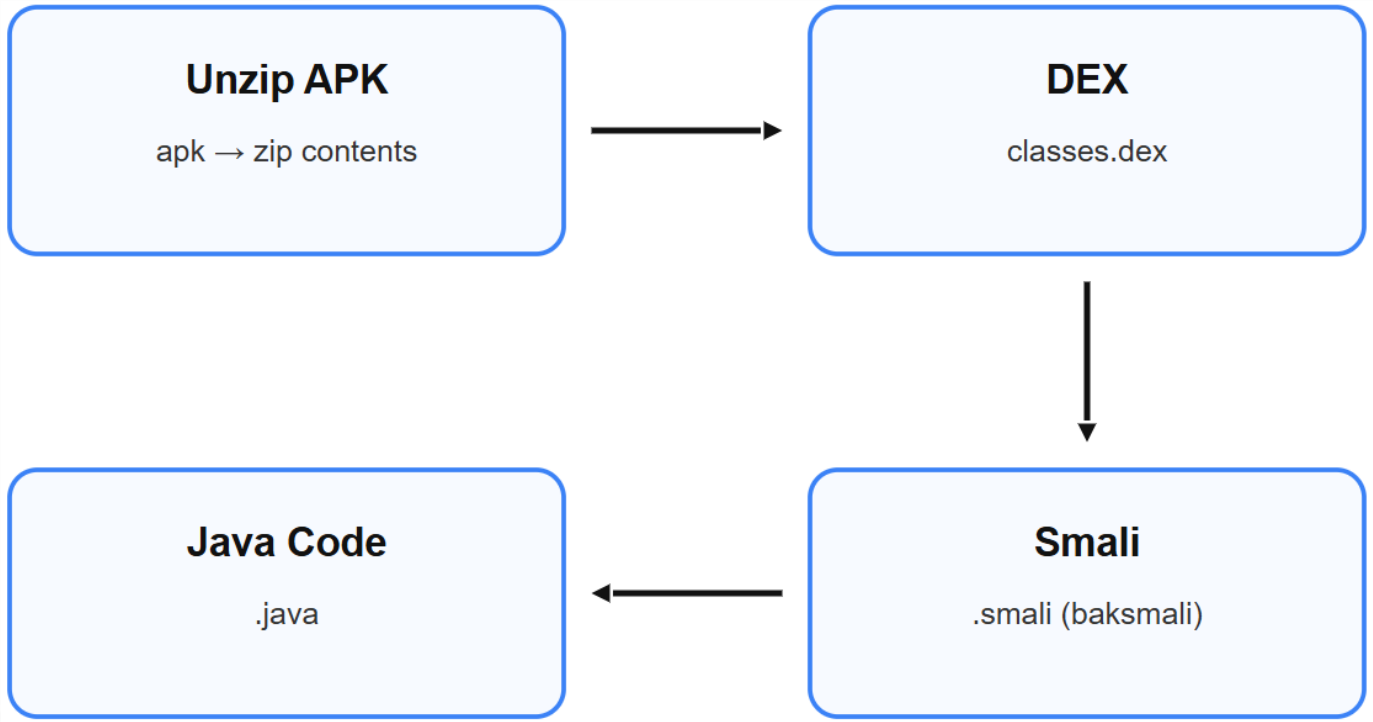
- دکس لودر
  - مکانیزمی برای بارگذاری دکس در زمان اجرا
  - افزودن کلاس/متد جدید به اپ بدون انتشار نسخه‌ی جدید
  - حالت اول: از مسیر دلخواه (فایل/دایرکتوری)
  - حالت دوم: از بایت‌ها در حافظه
  - سوءاستفاده‌ی بدافزارها برای بارگذاری پی‌لود آلوده

تزریق کد ایستا

دستکاری اسمالی و ری‌پک

تزریق کد ایستا

- روند مهندسی معکوس برنامه‌های اندرویدی
  - استخراج محتوای اپ
  - دریافت فایل دکس
  - دی‌اسمبل فایل دکس و ساخت فایل اسمالی
  - بازسازی کد جاوا



تزریق کد ایستا

- درباره‌ی اسمالی
  - ابزار متن‌باز برای اسمبل/دی‌اسمبل کردن دکس
  - پروژه‌ی شخصی Ben Gruver (jesusfreke)
  - یک استاندارد در مهندسی معکوس اندروید
  - بک‌اسمالی دی‌اسمبلر: DEX → Smali
  - خروجی smali.
  - اسمالی اسمبلر: Smali → DEX
- چرا اسمالی؟
  - اسمالی، نمای مستقیم دکس
  - عدم اطمینان‌پذیری جاوای دی‌کامپایل‌شده
  - نیاز به پروژه‌ی کامل برای بیلد جاوا
  - رفت‌وبرگشت ۱:۱ DEX ⇄ Smali

تزریق کد ایستا

- نمونه کد جاوا:

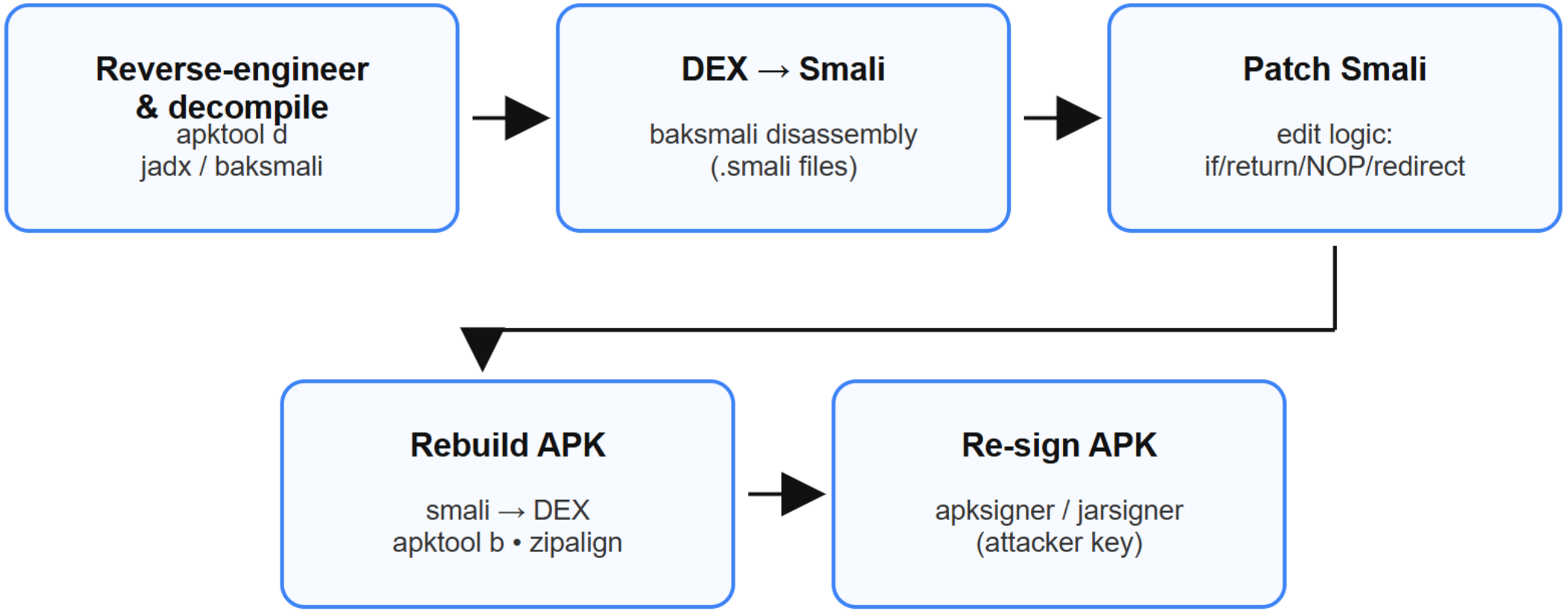
```java
public boolean login(int pin) {
    return pin == 1234;
}
```

- نمونه کد اسمالی:

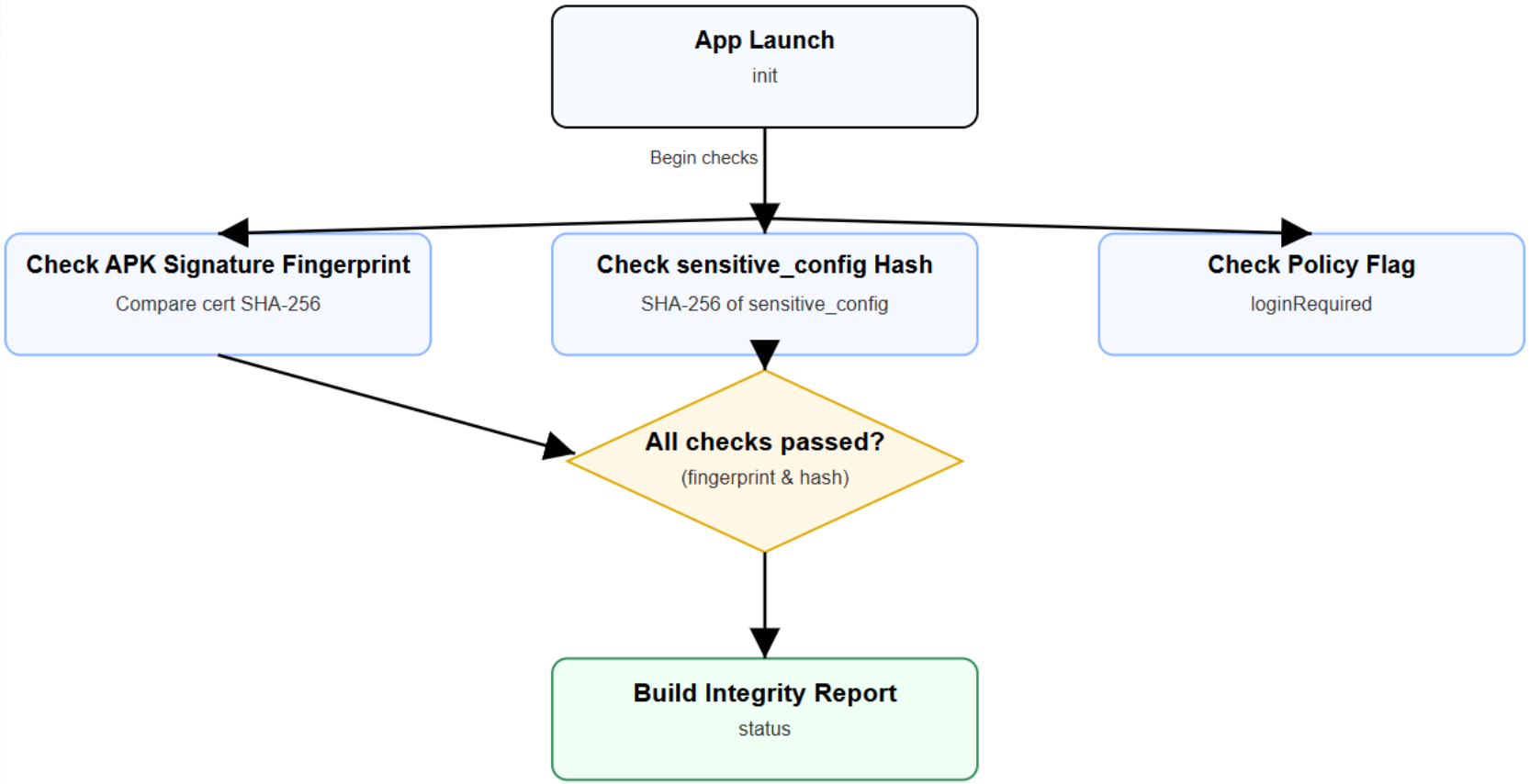
```
.method public login(I)Z
    .locals 1
    const/16 v0, 0x4d2   # 1234
    if-ne p1, v0, :no
    const/4 v0, 0x1      # true
    return v0
  :no
    const/4 v0, 0x0      # false
    return v0
.end method
```

تزریق کد ایستا

- روند حمله



تزریق کد ایستا



- نمونه‌ای از برنامه‌ی هدف
  - بررسی امضای برنامه
  - بررسی صحت فایل کانفیگ برنامه
  - بررسی نیاز به لاگین

تزریق کد ایستا

دو سناریو:

- **تشخیص دستکاری (Outcome A — Mismatch)**
  - تغییر فقط یک یا چند سیاست (مثلا loginRequired=false) توسط مهاجم
  - فعال بودن حداقل یک چک یک‌پارچگی
  - پیامد: تشخیص دستکاری و انجام اقدامات در سطح کلاینت و سرور
- **دورزدن چک‌ها (Outcome B — Forced Pass)**
  - پچ‌کردن چک‌های یک‌پارچگی علاوه بر تغییر سیاست‌ها
  - نتیجه چک‌ها هر چه باشد غیرفعال می‌شود.
  - پیامد: حمله‌ی موفق

تزریق کد ایستا

- چک‌های سطح کلاینت
  - تطبیق هش اپ و فایل‌های حساس مانند با سرور
  - تطبیق Fingerprint امضای اپ با سرور
- مبهم‌سازی کد
  - مجموعه‌ای از تکنیک‌ها برای تغییرنمای ظاهری کد بدون عوض‌کردن رفتار
  - استفاده از ابزارهایی مانند R8/ProGuard
- سخت‌سازی جریان کنترل
  - مجموعه‌ای از تکنیک‌ها برای پیچیده‌تر کردن مسیر اجرای کد
    - اضافه‌کردن شرط‌هایی با نتیجه ثابت
    - اضافه‌کردن مسیرهای اضافی بی‌اثر
    - استفاده از ابزارهایی مانند DexGuard

تزریق کد ایستا

تصمیم نهایی در سرور؛ کلاینت فقط اطلاع می‌دهد.

تزریق کد ایستا

- **ارتباط امن**
  - جلوگیری از دستکاری ترافیک و القای پاسخ جعلی از طریق حمله MITM
- **اعتبارسنجی سمت سرور**
  - ارسال اطلاعات بررسی‌شده در کلاینت و نتایج بررسی به سرور و اعتبارسنجی مجدد آن‌ها
  - مقایسه‌ی نتیجه اعتبارسنجی سرور با نتیجه اعتبارسنجی کلاینت
- **تنظیمات فروشگاه**
  - فعال‌سازی App Signing by Play سرویسی که در آن گوگل کلید نهایی انتشار اپ را نگه می‌دارد و هنگام توزیع، برنامه را با همان کلید رسمی امضا می‌کند. (اعتبارسنجی قبل از نصب)
  - فعال‌سازی Play Integrity API که با هر درخواست یک توکن معتبر ایجاد و ارسال می‌کند و کلاینت باید آن را برای سرور خود بفرستد. توکن در سرور اعتبارسنجی می‌شود. (اعتبارسنجی در زمان اجرا)

پرسش و پاسخ