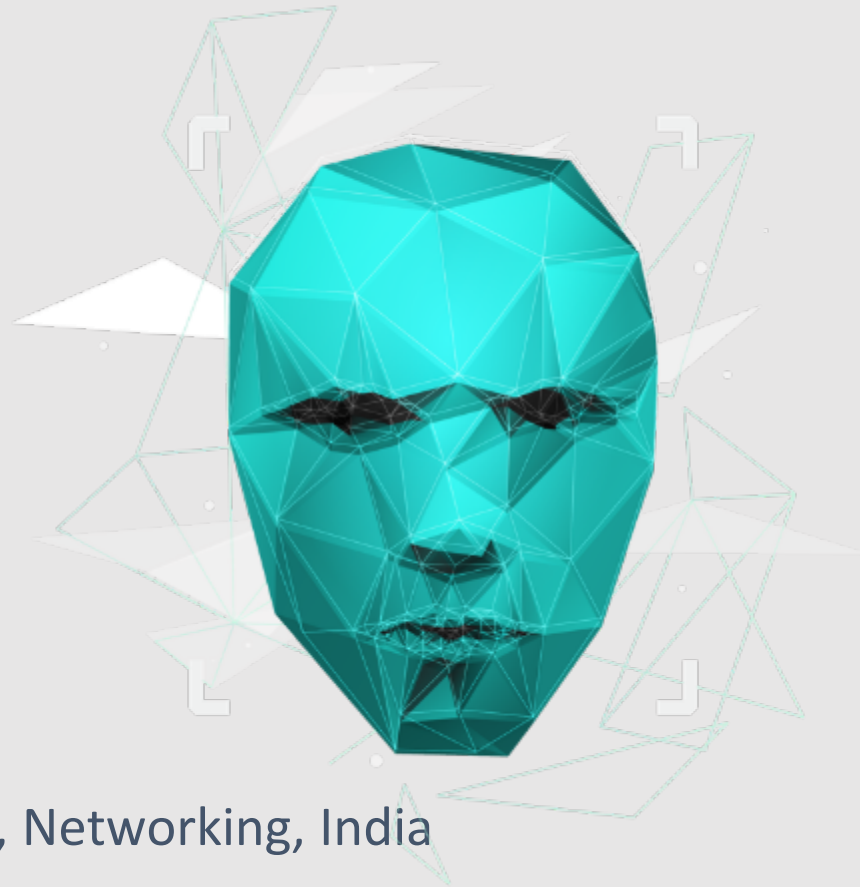


Advanced Network Security Web Security & Mobile Device Security

Dr. Yaeghoobi

PhD. Computer Science & Engineering, Networking, India
dr.yaeghoobi@gmail.com



00 | **Web Security**

01 | **Web Threats and Attacks**

02 | **Countermeasures**

03 | **Mobile Device Security**

04 | **Mobile Access Control**

05 | **Mobile Device Information Leakage**

**Web
Security**

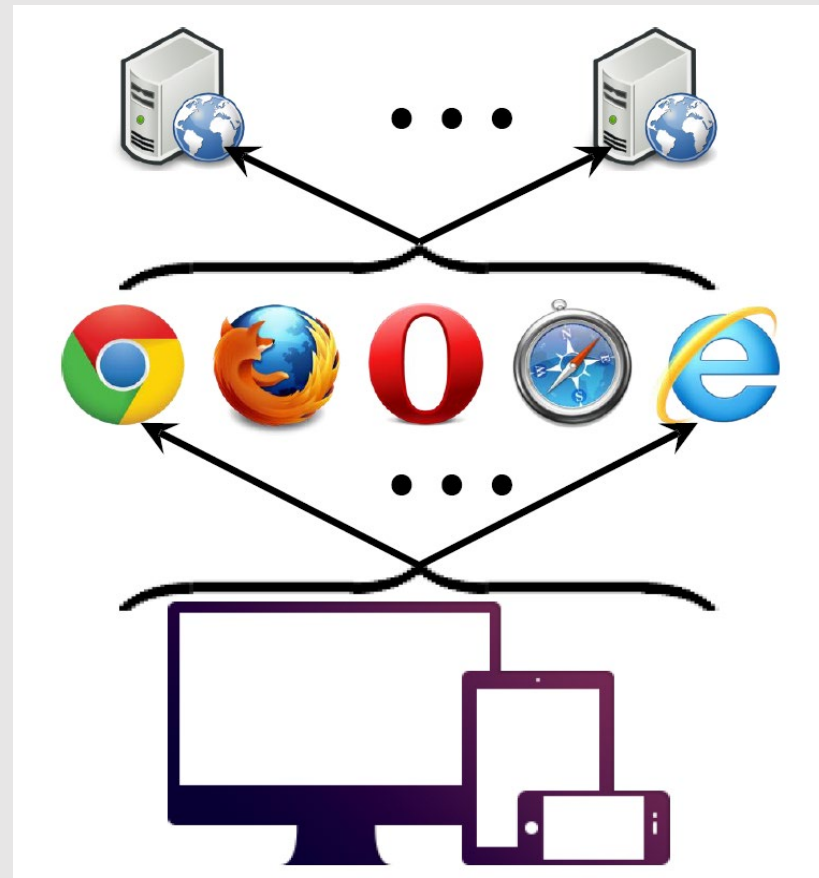
00



Web Basics

- Average user spends 32 h/month online
 - People spend much time interacting with Web, Web applications (apps)
 - Their (lack of) security has major impact
- Interaction via Web browser

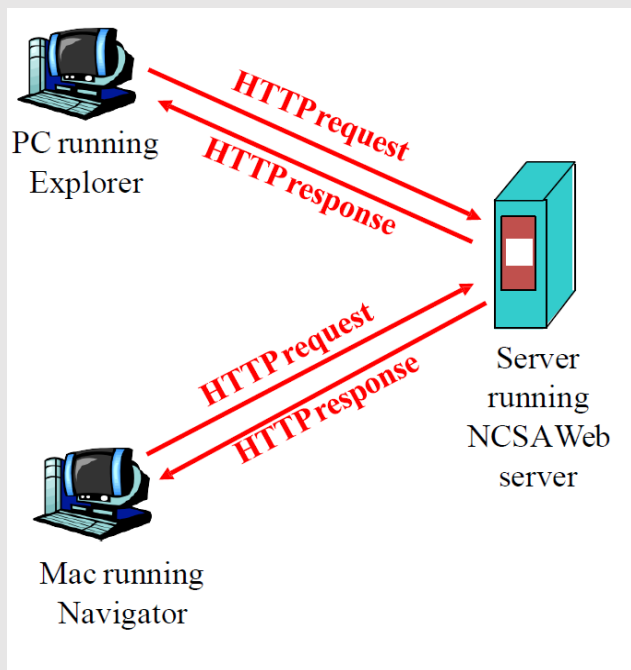
مردم وقت زیادی را صرف تعامل با وب، برنامه های وب (برنامه ها) (عدم امنیت) آنها تأثیر عمده ای دارد تعامل از طریق مرورگر وب



The Web

- Web page:
 - Consists of “objects”
 - Addressed by a URL
- Most Web pages consist of:
 - Base HTML page, and
 - Several referenced objects.
- URL has two components: host name and path name
- User agent for Web:
 - Browser
 - MS Edge
 - Chrome
 - Netscape Communicator
- Server for Web:
 - Web server
 - Apache (public domain)
 - MS Internet Information Server

The Web: the HTTP Protocol



- **HTTP: HyperText Transfer Protocol**
- Web's application layer protocol
- Client/server model
 - **Client:** browser that requests, receives, "displays" Web objects
 - مرورگری که درخواست، دریافت می کند، اشیاء وب را "نمایش" می دهد
 - **Server:** Web server sends objects in response to requests
 - وب سرور در پاسخ به درخواست ها اشیاء را ارسال می کند
- HTTP 1.0: RFC 1945
- HTTP 1.1: RFC 2068

The HTTP Protocol ...

aside

- **HTTP: TCP transport service**
 - Client initiates TCP connection (creates socket) to server, port 80
 - Server accepts TCP connection from client
 - HTTP messages (application-layer protocol messages) exchanged between browser (HTTP client) and Web server (HTTP server)
 - TCP connection closed
- **HTTP is “stateless”**
 - Server maintains no information about past client requests
 - سرور اطلاعاتی درباره درخواست های گذشته مشتری ندارد

Protocols that maintain “state” are complex!

- ❑ Past history (state) must be maintained
- ❑ If server/client crashes, their views of “state” may be inconsistent, must be reconciled

پروتکل هایی که “state” را ایجاد می کنند پیچیده هستند!

تاریخ گذشته state باید حفظ شود

اگر سرور / کلاینت خراب شود، نظرات

آنها در مورد “وضعیت” ممکن است

متناقض باشد، باید تطبیق داده شوند.

HTTP Example

Suppose user enters URL <http://www.someschool.edu/aDepartment/index.html>

(contains text, references to 10 JPEG images)

1a. HTTP client initiates TCP connection to http server (process) at www.someschool.edu. Port 80 is default for HTTP server.

1b. HTTP server at host www.someschool.edu waiting for TCP connection at port 80. "Accepts" connection, notifies client

2. HTTP client sends http *request message* (containing URL) into TCP connection socket


3. HTTP server receives request message, forms *response message* containing requested object ([aDepartment/index.html](http://www.someschool.edu/aDepartment/index.html)), sends message into socket

time



HTTP Example (Cont.)

4. HTTP server closes TCP connection



5. HTTP client receives response message containing HTML file, displays HTML. Parsing HTML file, finds 10 referenced JPEG objects

time 6. Steps 1-5 repeated for each of 10 JPEG objects

Non-Persistent and Persistent Connections

- **Non-persistent** غير مداوم
 - HTTP/1.0
 - Server parses request, responds, and closes TCP connection
 - 2 RTTs to fetch each object
 - Each object transfer suffers from slow start
 - **Persistent** مداوم
 - Default for HTTP/1.1
 - On same TCP connection: server, parses request, responds, parses new request, ...
 - Client sends requests for all referenced objects as soon as it receives base HTML.
 - Fewer RTTs and less slow start.
- But most browsers use parallel TCP connections.**

HTTP Message Format: Request

- Two types of HTTP messages: *request*, *response*
- **HTTP request message:**
 - ASCII (human-readable format)

**request line
(GET, POST,
HEAD commands)**

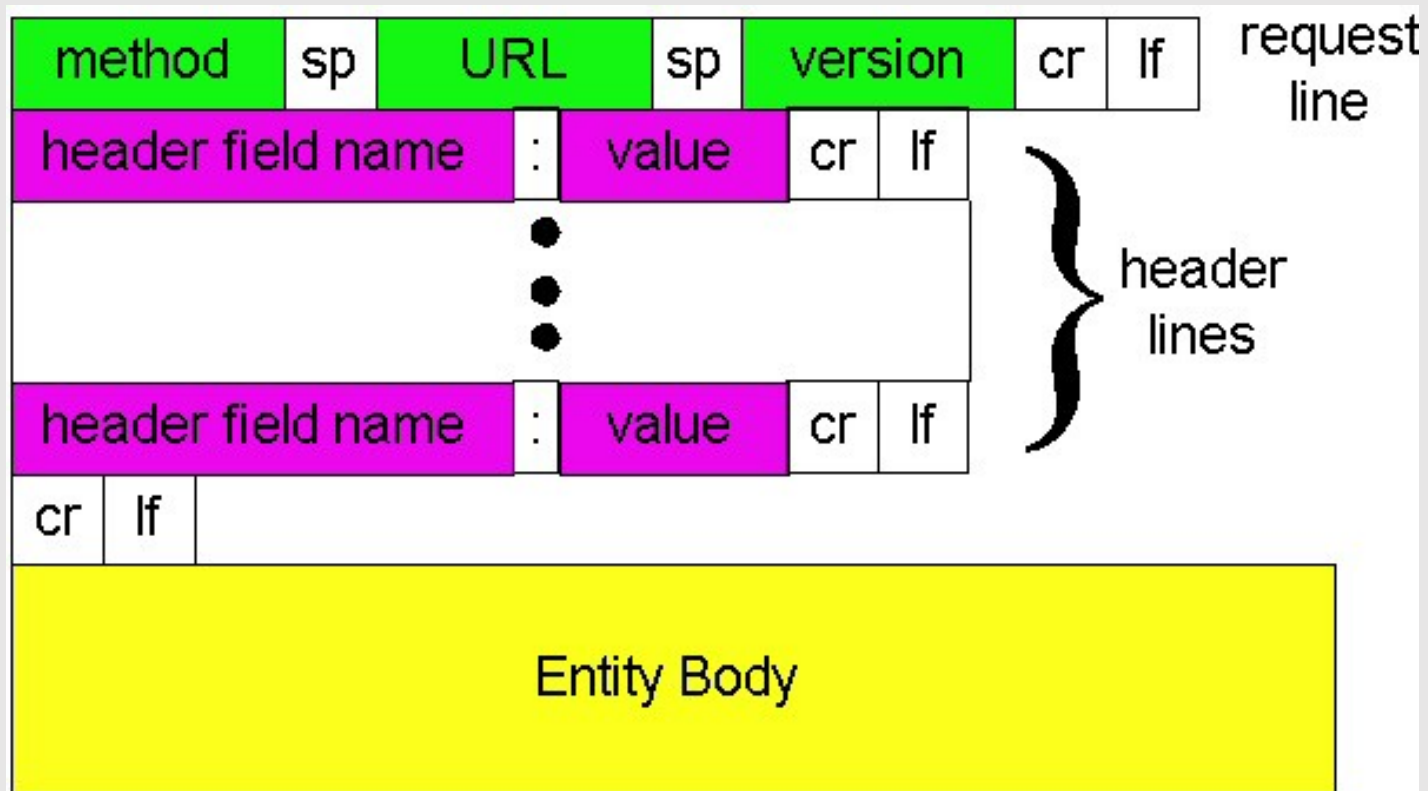
**header
lines**

```
GET /somedir/page.html HTTP/1.0
User-agent: Mozilla/4.0
Accept: text/html, image/gif, image/jpeg
Accept-language: fr
```

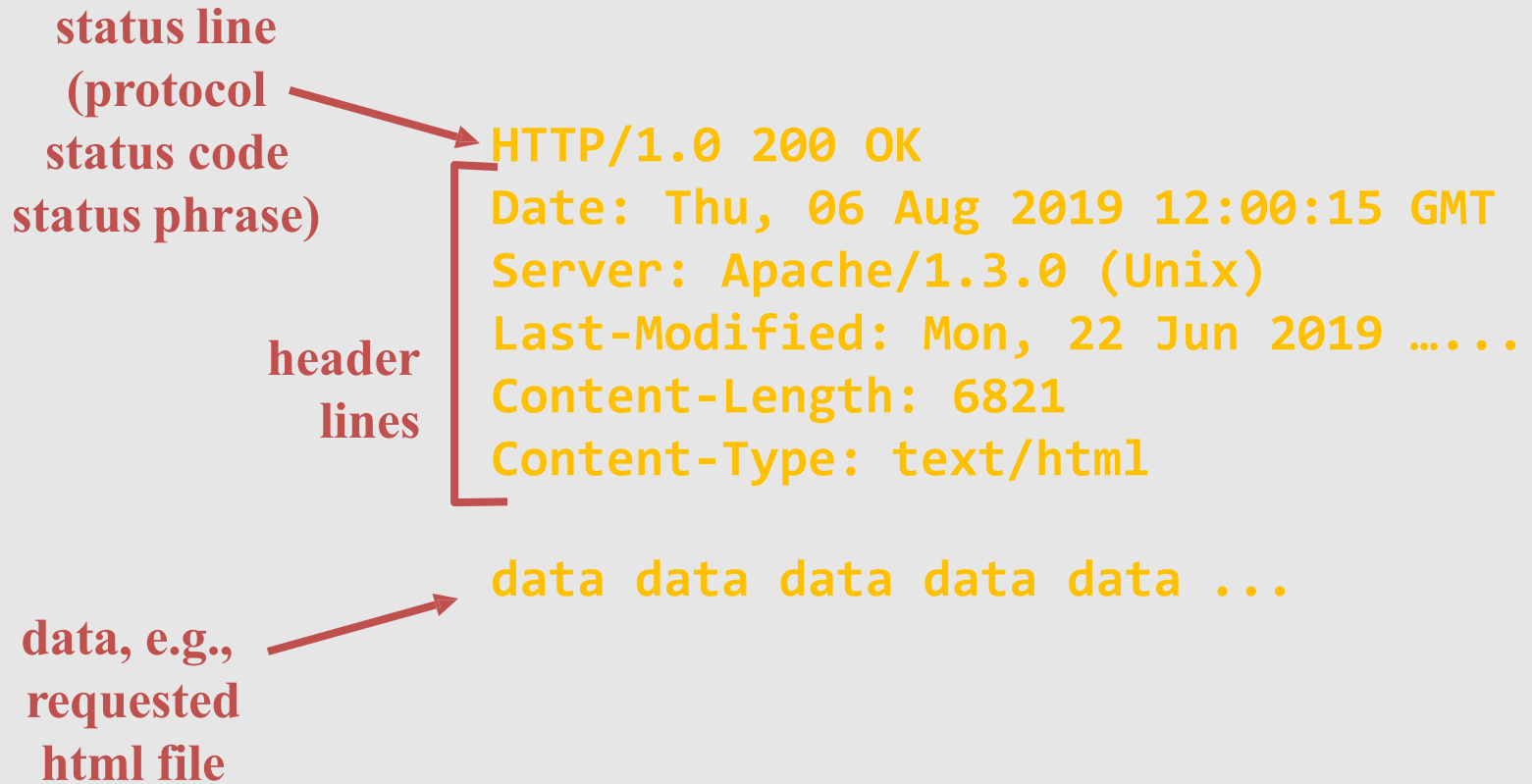
**Carriage return,
line feed
indicates end
of message**

(extra carriage return, line feed)

HTTP Request Message: General Format



HTTP Message Format: Response



HTTP Response Status Codes

- In first line in server→client response message. A few sample codes:
- **200 OK**
 - – request succeeded, requested object later in this message
 - درخواست موفق، پیام هدف ادامه درخواست
- **301 Moved Permanently**
 - –requested object moved, new location specified later in this message (Location:)
 - نقل مکان شیء درخواست شده، مکان جدیدی در این پیام مشخص شده است (مکان:)
- **400 Bad Request**
 - – request message not understood by server
 - درخواست پیام توسط سرور قابل درک نیست
- **404 Not Found**
 - – requested document not found on this server
 - سند درخواست شده در این سرور یافت نشد
- **505 HTTP Version Not Supported**

Try HTTP (Client Side) for Yourself

1. Telnet to your favorite Web server:

```
telnet www.cse.ohio-state.edu/ 80
```

Opens TCP connection to port 80 (default HTTP server port) at www.cse.ohio-state.edu. Anything typed in sent to port 80 at www.cse.ohio-state.edu

2. Type in a GET HTTP request:

```
GET /~xuan/index.html HTTP/1.0
```

By typing this in (hit carriage return twice), you send this minimal (but complete) GET request to HTTP server

3. Look at response message sent by HTTP server!

HTTP Versions 2, 3

- Max. webpage latency: 250–300 msec (lower is better!)
• حداکثر تأخیر صفحه وب: ۲۵۰-۳۰۰ msec (پایین تر بهتر است!)
- HTTP 2, 3 designed for security, performance
• HTTP 2، 3 برای امنیت، کارایی طراحی شده است

- HTTP 2:

Supports encryption as “first-class citizen”

پشتیبانی از رمزگذاری بعنوان "شهروند درجه یک"

More info: I. Gregorik, *High Performance Browser Networking*,
O'Reilly, 2017; <https://hpbn.co/http2/> (Chapter 12)

- HTTP 3:

Uses Google's QUIC transport protocol (UDP) for lower latency

از پروتکل QUIC transport protocol (UDP) برای تأخیر کمتر استفاده می کند

More info: <http://www.chromium.org/quic>;

**Web
Threats and
Attacks**

01

Information Leakage

- **Sensitive information can be leaked via Web:**
 - All files accessible under a Web directory can be downloaded via GET requests

– از طریق درخواست GET تمام پرونده ها در زیر فهرست وب قابل دسترسی است

– Example 1:

- <http://www.website.com/secret.jpg> publicly accessible
- <http://www.website.com/index.html> has no link to secret.jpg

Attacker can still download secret.jpg via GET request!

مهاجم هنوز می تواند secret.jpg را از طریق درخواست GET بارگیری کند!

– Example 2: searching online for “proprietary confidential” information

– جستجوی آنلاین برای اطلاعات "محرمانه اختصاصی"

Misleading Websites

- Cybersquatters can register domain names similar to (trademarked) company, individual names

• Cyberquatters می توانند نام دامنه مشابه با (مارک تجاری) شرکت، نامهای شخصی را ثبت کنند

- Example: <http://www.google.com> vs. <http://gogle.com>

vs. ...

- Practice is illegal *if* done “in bad faith” غیرقانونی
- Arbitration procedures available for name reassignment (ICANN)

• روشهای داوری موجود برای انتصاب مجدد نام (ICANN)

XSS and CSRF

- Cross-site scripting (XSS): inject JavaScript from external source into insecure websites

جاوا اسکریپت را از منبع خارجی به وب سایت های ناامن تزریق کنید

- Example: `input <script type="text/javascript"> <!--evil code> </script>`

- Cross-site request forgery (CSRF): force victim browser to send request to external website → performs task on browser's behalf

مرورگر قربانی را مجبور به ارسال درخواست به وب سایت خارجی می کند -
وظیفه ای را از طرف مرورگر انجام می دهد

- Example: force load ``

SQL Injection

- Common vulnerability (~71 attacks/hour) آسیب پذیری شایع (۷۱ حمله در ساعت)
- Exploits Web apps that:
 - Poorly validate user input for SQL string literal escape characters, معتبر بودن ناکافی برای ورودی های SQL کاربر
 - Example:
 - "SELECT * FROM users WHERE name = '' + userName + ''";
 - If userName is set to ' or '1'='1, the resulting SQL is
SELECT * FROM users WHERE name = '' OR '1'='1';
 - This evaluates to SELECT * FROM users ⇒ displays all users

Malicious Shellcode

- Shellcode is *non-self-contained* binary executable code
 - Distinct from malware that executes on its own
 - Shellcode can only execute after injection into a running process's virtual address space
- Most shellcode written in Intel IA-32 assembly language (x86)

Shellcode کد اجرایی باینری غیر خود شامل است

متمایز از بدافزارهایی است که به خودی خود اجرا می شود

Shellcode فقط پس از تزریق در فضای آدرس مجازی یک فرایند در حال اجرا قابل اجرا است

بیشترین کدک برای زبان مونتاژ Intel IA-32 نوشته شده است (x86)

Malicious Shellcode ...

- When injected into JS code, shellcode executes
 - Hijacks browser process
 - Can totally control target process or system
- Shellcode: attack vector for malicious code execution on target systems (e.g., Conficker worm)
 - Usually, browser downloads JS code containing shellcode
 - JS code executes, controls target process/system

• هنگامی که به کد JS تزریق می شود، shellcode اجرا می شود

◦ فرآیند مرورگر ربودن

◦ می تواند فرآیند یا سیستم مورد نظر را کاملاً کنترل کند

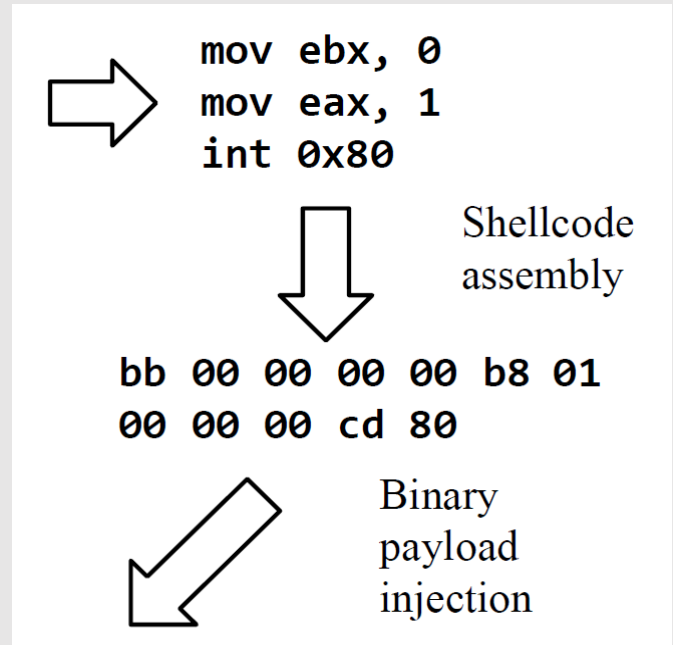
• Shellcode برای اجرای کد مخرب بر روی سیستم های هدف (به عنوان مثال ، کرم Conficker)

◦ معمولاً مرورگر کد JS حاوی shellcode را بارگیری می کند

◦ کد JS اجرا می کند ، فرآیند / سیستم هدف را کنترل می کند

A Toy Shellcode

- Shellcode for `exit()` system call
 - Store `0` into register `ebx`
 - Store `1` into register `eax`
 - Execute instruction `int 0x80`
- Assembled shellcode injected into JS code



Disguised as normal data; injected into target processes' address spaces; compromises target processes' security

Countermeasures

02

HTTPS (HTTP Secure)

- HTTPS uses cryptography with HTTP
 - Alice, Bob have public, private keys; public keys accessible via certificate authority (CA)
 - Alice encrypts message with Bob's public key, signs message with her private key
 - Bob decrypts message with his private key, verifies message using Alice's public key
 - Once they "know" each other, they can communicate via symmetric crypto keys
- HTTPS provides greater assurance than HTTP

• HTTPS از رمزنگاری با HTTP استفاده می کند

• آلیس ، باب دارای کلیدهای عمومی و خصوصی هستند. کلیدهای عمومی از طریق مجوز (CA) قابل دسترسی است

• آلیس با کلید عمومی باب پیام را رمزگذاری می کند، و با کلید خصوصی خود پیام را امضا می کند

• باب با کلید خصوصی خود پیام را رمزگشایی می کند، با استفاده از کلید عمومی آلیس پیام را تأیید می کند

• هنگامی که یکدیگر را می شناسند ، می توانند از طریق کلیدهای رمزنگاری متقارن ارتباط برقرار کنند

• HTTPS اطمینان بیشتری نسبت به HTTP فراهم می کند

TLS/SSL

- HTTPS uses Transport Layer Security (TLS), Secure Sockets Layer (SSL), for secure data transport

- Data transmitted via client- server “tunnel”
- Much harder to compromise than HTTP

(HTTPS از Transport Layer Security (TLS)، لایه سوکت ایمن (SSL) برای حمل و نقل ایمن داده استفاده می کند)

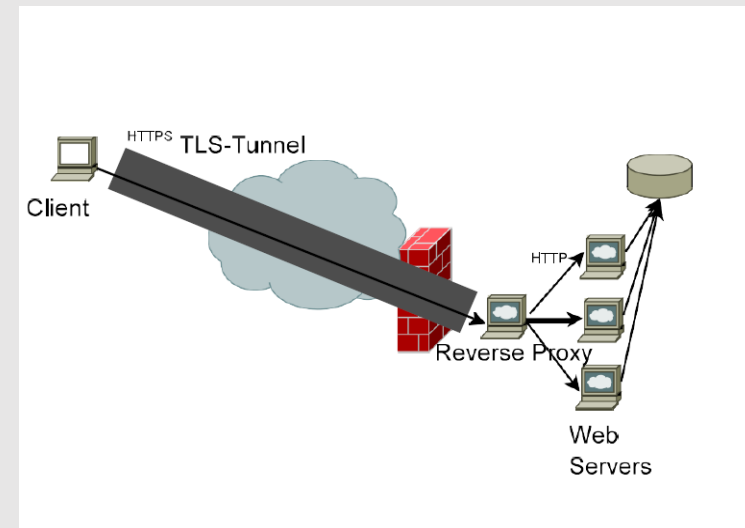
داده های منتقل شده از طریق سرویس دهنده "تونل" سرویس دهنده
سازش بسیار سخت تر از HTTP

- Problems: چالش ها و مسائل

- Relies on CA infrastructure integrity
- Users can make mistakes (blindly click “OK”)

- به یکپارچگی زیرساخت CA متکی است

- کاربران می توانند اشتباه کنند (کورکورانه بر روی "OK کلیک کنید)



HTTPS Example

- User visits website via HTTPS, e.g., <https://gmail.com>
- Browser sends TLS/SSL request, public key, message authentication code (MAC) to gmail.com; gmail.com does likewise;
 - TLS/SSL encrypt entire connection; HTTP layered stop it
 - Both parties verify each other's identity, generate symmetric key for following communications:
- Browser retrieves public key certificate from gmail.com signed by certificate authority (Equifax)
 - Certificate attests to site's identity
 - If certificate is self-signed, browser shows warning
- Browser, gmail.com use symmetric key to encrypt/decrypt subsequent communications

HTTPS مثال

- کاربر از طریق HTTPS، به عنوان مثال ، `https://gmail.com` از وب سایت بازدید می کند

- مرورگر درخواست TLS / SSL، کلید عمومی ، کد احراز هویت پیام (MAC) را به `gmail.com` ارسال می کند. `gmail.com` به همین ترتیب عمل می کند؛

- TLS / SSL کل اتصال را رمزگذاری می کند. HTTP لایه باز آن را متوقف کند
- هر دو طرف هویت یکدیگر را تأیید می کنند ، کلید متقارن را برای ارتباطات زیر ایجاد می کنند:

مرورگر گواهی کلید عمومی را از `gmail.com` بازپایی می کند امضا شده توسط مرجع صدور گواهی نامه (Equifax)

گواهی هویت سایت را تأیید می کند

در صورت امضای گواهی بصورت شخصی، مرورگر هشدار را نشان می دهد

مرورگر، `gmail.com` از کلید متقارن برای رمزگذاری / رمزگشایی ارتباطات بعدی استفاده میکند

Blacklist Filtering

- **Misleading websites:** Register domain names similar trademarks, e.g., www.google.com, gogle.com, etc.
وب سایت های گمراه کننده: علائم تجاری مربوط به نام دامنه را به عنوان مثال، ثبت www.google.com، gogle.com
- **XSS:**
 - Validate user input; reject invalid input ورودی نامعتبر را رد کنید
 - Blacklist offending IP addresses لیست سیاه که آدرس های متخلف
- **CSRF:**
 - Use random “token” in web app forms استفاده از از "نشانه" تصادفی در فرم های برنامه وب
 - If token is replayed, reject form (blacklist IP addresses) در صورت پخش مجدد توکن، رد فرم (blacklist IP addresses)
- **SQL injection:**
 - Validate user input to databases, reject invalid input ورودی کاربر را به پایگاه داده تأیید کنید، ورودی نامعتبر را رد کنید
 - Blacklist IP addresses آدرسهای لیست سیاه

Blacklist Filtering ...

- Helpful browser extensions:
 - NoScript/NotScripts/... (stop XSS)
 - Adblock (can stop malicious scripts in ads)
 - SSL Everywhere (force HTTPS)
 - Google Safe Browsing, etc.

برنامه های افزودنی مرورگر مفید:

NoScript / NotScripts / ... (توقف XSS)

Adblock می تواند اسکریپت های مخرب را در تبلیغات متوقف کند

SSL Everywhere (نیروی HTTPS)

مرور ایمن Google و ...

Defending Against Shellcode

- Two main detection approaches: دو روش اصلی تشخیص
 - 1. Content Analysis تجزیه و تحلیل محتوا
 - Checks objects' contents before using them
 - Decodes content into instruction sequences, checks if malicious
 - مطالب اشیاء را قبل از استفاده بررسی می کند
 - محتوا را در توالی های دستورالعمل رمزگشایی می کند ، اگر مخرب باشد ، بررسی می کند
 - 2. Hijack Prevention پیشگیری از ربودن
 - Focuses on preventing shellcode from being fully executed
 - Randomly inserts special bytes into objects' contents, raises exception if executed
 - Can be thwarted using several short "connected" shellcodes
 - تمرکز خود را در جلوگیری از اجرای کامل shellcode متمرکز می کند
 - بطور تصادفی بایت های خاص را در محتوای اشیاء وارد می کند ، در صورت اجرا استثنا را ایجاد می کند
 - با استفاده از چندین کد "متصل" shellcode می توان جلوگیری نمود

Content Analysis

1. Static Analysis تجزیه و تحلیل استاتیک

- Uses **signatures, code patterns** to check for malicious instructions
- از امضاها، الگوهای کد برای بررسی دستورالعمل های مخرب استفاده می کند
- **Advantage:** Fast سریع
- **Disadvantages:** Incomplete; can be thwarted by obfuscation techniques
اقص؛ با استفاده از تکنیک های انسداد می توان خنثی نمود

2. Dynamic Analysis تجزیه و تحلیل پویا

Detects a malicious instruction sequence by emulating its execution

یک دنباله دستورالعمل مخرب را با شبیه سازی اجرای آن تشخیص می دهد

Advantages: Resistant to obfuscation; more complete than static analysis
مقاوم در برابر ابهامات؛ کاملتر از آنالیز استاتیک

Disadvantage: Slower کندتر

- Focus on dynamic analysis (greater completeness)

Dynamic Analysis

- **Approaches assume self-contained shellcodes**
- Analyses' shellcode emulation:
 - Inefficiently uses JS code execution environment information
 - All memory reads/writes only go to emulated memory system
 - Detection uses GetPC code
 - از اطلاعات محیط اجرای کد ناکارآمد JS استفاده می کند
 - تمام خواندو/نوشتن حافظه را به سیستم حافظه شبیه سازی شده می فرستد
 - از کد GetPC برای تشخیص استفاده می کند
- **Current dynamic analysis approaches can be fooled:**
 - Shellcode using JS code execution environment info
 - Shellcode using target process virtual memory info
 - Shellcode not using GetPC code
- To detect all malicious shellcodes, we need a better approach

JSGuard (1)

- Use dynamic analysis to detect malicious JS objects
 - Create a virtual execution environment for detection

Leveraging: (1) target processes' virtual memory information; (2) target systems' context information in detection

NOT a whole-system emulator
 - Facilitate multiple-level redundancy reduction

Stack frames: check origins of JS code being interpreted

Native methods: check if native methods to be called originate from JS interpreter or external components

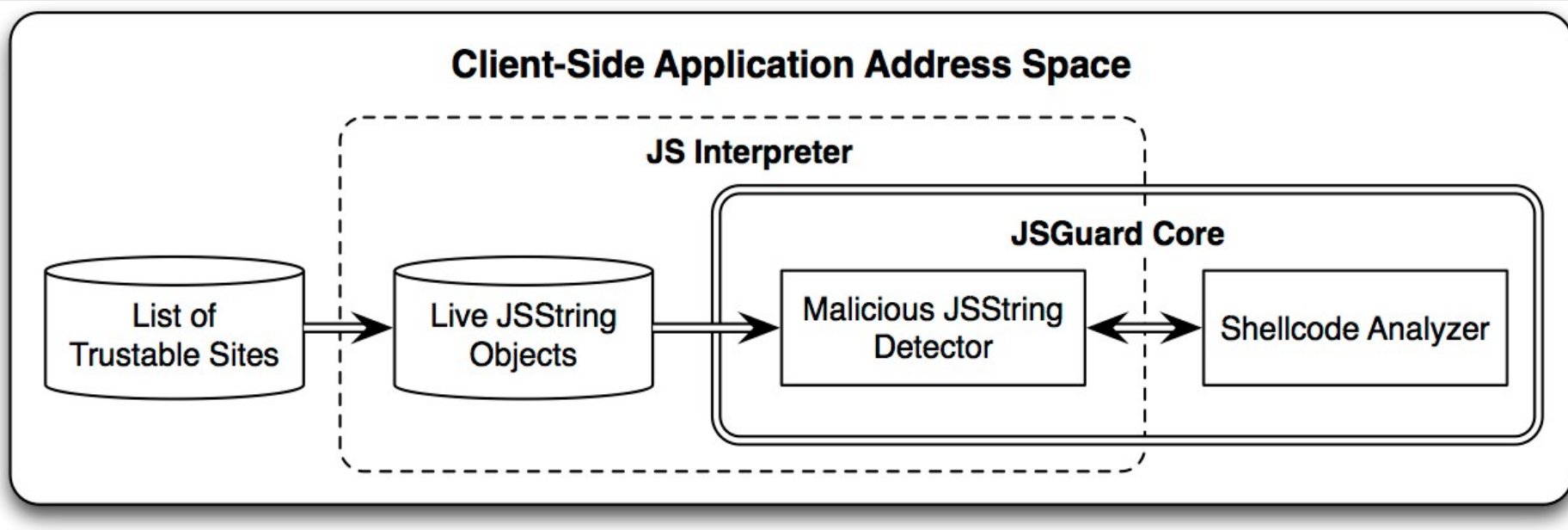
Objects' properties
- Assume: JS interpreter's (native) methods have no memory errors

JSGuard (2)

- It's hard to fool JSGuard method:
 - Shellcode can use JS code execution environment information to fool other dynamic analysis approaches
 - JSGuard design leverages system's context information
 - Shellcode can use target process's virtual memory information to fool other dynamic analysis approaches
 - JSGuard design uses target processes' virtual memory information
 - Shellcode can avoid GetPC code to fool other dynamic analysis approaches
 - JSGuard method does not rely on GetPC code for detection. It leverage real virtual memory content to decode instructions and emulate their execution

JSGuard (3)

- Architecture runs in client-side application's address space



Summary

- Web based on plaintext HTTP protocol (stateless)
- Web security threats include information leakage, misleading websites, and malicious code
- Countermeasures include HTTPS, blacklist filtering mechanisms, and malicious code detection

- وب مبتنی بر پروتکل HTTP متن ساده (بدون تابعیت)
- تهدیدات امنیتی وب شامل نشت اطلاعات ، وب سایت های گمراه کننده و کد مخرب است
- اقدامات متقابل شامل HTTPS ، مکانیسم های فیلتر لیست سیاه و تشخیص کد مخرب است

**Mobile
Device
Security**

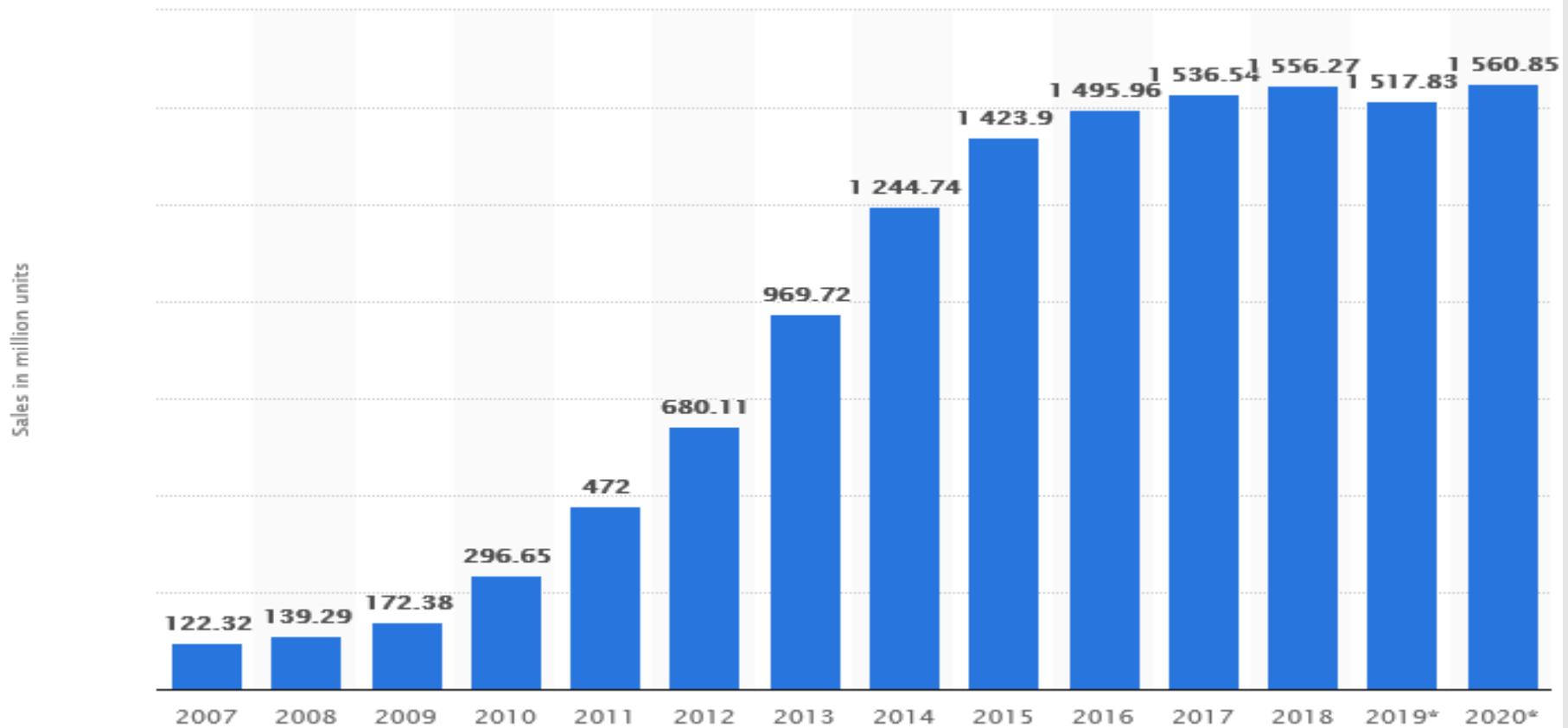
03

Overview of Mobile Devices

- Mobile computers:
 - Mainly smartphones, tablets
 - **Sensors:** GPS, camera, accelerometer, etc.
 - **Computation:** powerful CPUs (≥ 1 GHz, multi-core)
 - **Communication:** cellular/4G, Wi-Fi, near field communication (NFC), etc.
- Many connect to cellular networks: billing system



Number of smartphones sold to end users worldwide from 2007 to 2020 (Statista 2020)



Mobile Threats and Attacks

- Mobile devices make attractive targets:
 - People store much personal info on them: email, calendars, contacts, pictures, etc.
 - Sensitive organizational info too...
 - Can fit in pockets, easily lost/stolen
 - Built-in billing system: SMS/MMS (mobile operator), in-app purchases (credit card), etc.
 - Many new devices have near field communications (NFC), used for contactless payments, etc.
 - Your device becomes your credit card
 - Location privacy issues
- NFC-based billing system vulnerabilities

تهدیدات و حملات موبایل

• دستگاه های تلفن همراه اهداف جذاب را ایجاد می کنند:

• افراد اطلاعات شخصی زیادی را روی آنها ذخیره می کنند: ایمیل ، تقویم ها ، مخاطبین ، تصاویر و غیره.

• اطلاعات سازمانی حساس نیز ...

• می تواند از جیب، به راحتی سرقت شود

• سیستم صورتحساب داخلی: SMS / MMS (اپراتور تلفن همراه)، خرید درون برنامه (کارت اعتباری) و غیره

• بسیاری از دستگاه های جدید دارای ارتباطات میدانی نزدیک (NFC) هستند که برای پرداخت های بدون تماس و غیره استفاده می شوند.

• دستگاه شما به کارت اعتباری شما تبدیل می شود

• مشکلات حریم خصوصی موقعیت

• آسیب پذیری سیستم صورتحساب مبتنی بر NFC

Mobile Device Loss/Theft

- Many mobile devices lost, stolen each year

• بسیاری از دستگاه های تلفن همراه هر سال گم می شوند، سرقت می شوند

- 113 mobile phones lost/stolen every minute in the U.S.
- 56% of us misplace our mobile phone or laptop each month
- Lookout Security found \$2.5 billion worth of phones in 2011 via its Android app
- Symantec placed 50 “lost” smartphones throughout U.S. cities

- 96% were accessed by finders

- 80% of finders tried to access “sensitive” data on phone

• ۹۶٪ توسط یابندگان قابل دسترسی بودند

• ۸۰٪ از یافتگان سعی کردند به داده های "حساس" از طریق تلفن دسترسی پیدا کنند

Device Malware

- iOS malware: very little
- Juniper Networks: Major increase in Android malware from 2010 to 2011
- Android malware growth keeps increasing (\$\$\$)
- Main categories:
 - Trojans
 - Monitoring apps/spyware
 - Adware
 - Botnets
- We'll look at notable malware examples

Device Search and Seizure

- *People v. Diaz*: if you're arrested, police can search your mobile device without warrant

- اگر دستگیر شوید ، پلیس می تواند دستگاه تلفن همراه شما را بدون ضمانت جستجو کند

- Rationale: prevent perpetrators destroying evidence

- Quite easy to break the law (overcriminalization)

- Crime severity: murder, treason, etc. vs. unpaid citations

- “Tens of thousands” of offenses on the books

- Easy for law enforcement to extract data from mobile devices (forensics)

- استخراج داده ها از دستگاه های تلفن همراه برای اجرای قانون آسان است

Location Disclosure

- MAC, Bluetooth Addresses, IMEI, IMSI etc. are globally unique
- Infrastructure based mobile communication
- Peer-to-Peer ad hoc mobile communication

- MAC، آدرسهای بلوتوث، IMEI، IMSI و غیره در سطح جهان همانند یکدیگر نیستند

- ارتباطات سیار مبتنی بر زیرساخت ها

- ارتباطات همراه و همکار همراه با همتا

**Mobile
Access
Control**

04

Countermeasures (Mobile Access Control)

- Very easy for attacker to control a mobile device if he/she has physical access
 - Especially if there's no way to authenticate user
 - Then device can join botnet, send SMS spam, etc.
- اگر دسترسی فیزیکی ایجاد شود، کنترل یک دستگاه تلفن همراه برای مهاجم بسیار آسان است
 - به خصوص اگر هیچ راهی برای تأیید اعتبار کاربر وجود ندارد
 - سپس دستگاه می تواند به botnet بپیوندد ، اسپم پیامکی ارسال کند ، و غیره
- Need access controls for mobile devices
 - احراز هویت ، مجوز ، پاسخگویی **Authentication, Authorization, Accountability**
 - روش تأیید اعتبار: Authentication workflow
 - درخواست دسترسی Request access
 - درخواست (user provides identity, e.g., John Smith) Supplication
 - احراز هویت (system determines user is John) Authentication
 - مجوز (system determines what John can/cannot do) Authorization

Authentication: Categories

- Authentication generally based on:
 - Something supplicant knows
 - Password/passphrase
 - Unlock pattern
 - Something supplicant has
 - Magnetic key card
 - Smart card
 - Token device
 - Something supplicant is
 - Fingerprint
 - Retina scan
- تأیید هویت به طور کلی بر اساس:
 - چیزی که درخواست کننده می داند
 - رمز عبور / کلمه عبور
 - الگوی باز کردن
 - چیزی که درخواست کننده دارد
 - کارت کلید مغناطیسی
 - کارت هوشمند
 - دستگاه توکن
 - چیزی درخواست کننده است
 - اثر انگشت
 - اسکن شبکیه

Authentication: Passwords

- Cheapest, easiest form of authentication

• ارزان ترین ، ساده ترین شکل تأیید اعتبار

- Works well with most applications

• با بیشتر برنامه ها خوب کار می کند

- Also the weakest form of access control ضعیف ترین شکل کنترل دسترسی

- Lazy users' passwords: *1234, password, letmein, etc.*
- Can be defeated using dictionary, brute force attacks

- Requires administrative controls to be effective

• مستلزم کنترل برای مؤثر بودن است

- Minimum length/complexity
- Password aging
- Limit failed attempts

Authentication: Smart Cards/ Security Tokens

- More expensive, harder to implement اجرای گران تر، سخت تر
- **Vulnerability:** *prone to loss or theft* آسیب پذیری: مستعد ضرر و سرقت
- Very strong when combined with another form of authentication, e.g., a password
- هنگامی که با شکل دیگری از تأیید اعتبار، به عنوان مثال، یک رمز عبور همراه باشد، بسیار قوی است
- Does not work well in all applications در همه برنامه ها خوب کار نمی کند
- Try carrying a smart card in addition to a mobile device!
علاوه بر دستگاه تلفن همراه ، یک کارت هوشمند نیز حمل کنید!

Authentication: Biometrics

- More expensive/harder to implement

• اجرای گران تر / سخت تر

- Prone to error:

- **False negatives:** not authenticate authorized user

- **False positives:** authenticate unauthorized user

– منفی های کاذب: کاربر مجاز را تأیید نکنید

– مثبت کاذب: تأیید کاربر غیرمجاز

- **Strong authentication** when it works

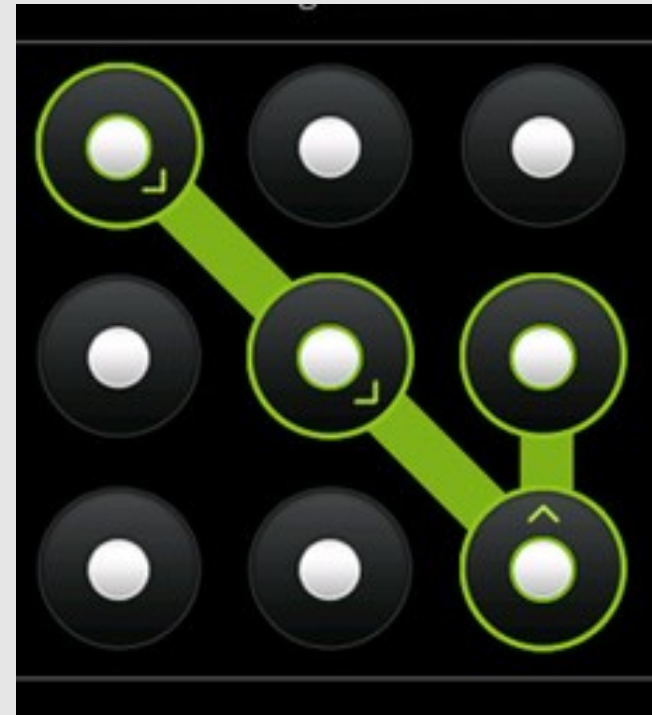
- Does not work well in all applications

• اگر کار نماید قوی است

• بر روی همه نرم افزارها خوب کار نمیکند

Authentication: Pattern Lock

- Swipe path of length 4–9 on 3 x 3 grid
- Easy to use, suitable for mobile devices
- استفاده آسان، مناسب برای دستگاه های تلفن همراه
- Problems:
 - 389,112 possible patterns; (456,976 possible patterns for 4-char case-insensitive alphabetic password!)
 - **Attacker can see pattern from finger oils on screen**



Authentication: Comparison

	Passwords	Smart Cards	Biometrics	Pattern Lock
Security	Weak	Strong	Strong	Weak
Ease of Use	Easy	Medium	Hard	Easy
Implementation	Easy	Hard	Hard	Easy
Works for phones	Yes	No	Possible	Yes

Deeper problem: mobile devices are designed with single-user assumption...

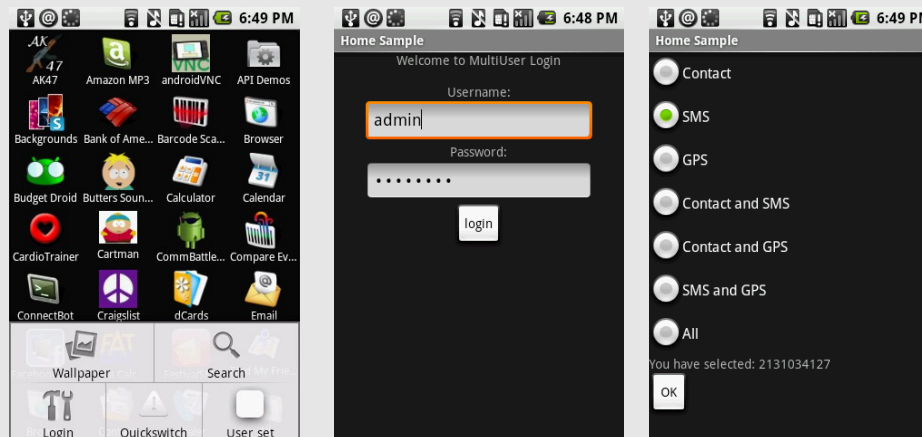
DiffUser

- Current smartphone access control focus: 1 user (admin)
- Hard to achieve *fine-grained*
- Mobile device management:
 - Control app installation/gaming
 - Parental controls
 - Lend phone to friend
- DiffUser, differentiated user access control model
 - Different users use smartphone in different contexts
 - User classification: admin, normal, guest

Smartphone Privilege		Admin	Normal	Guest
Personal	SMS	✓	✓	X
Info	Contacts	✓	✓	X
Resource Access	WiFi	✓	✓	Limit!
	GPS	✓	✓	Limit!
	Bluetooth	✓	✓	Limit!
Apps	App Install	✓	Limit	X
	Sensitive Apps	✓	Limit	X

DiffUser ...

- Implement DiffUser system on Android using Java
- Override Android's "Home" Activity for multi-user authentication, profile configuration



From left to right: "normal" user screen; user login and authentication; user profile configuration.

**Mobile
Device
Information
Leakage**

05

Mobile Device Information Leakage

- Types of mobile device **information sources**
 - Internal to device (e.g., GPS location, IMEI, etc.)
 - External sources (e.g., CNN, Chase Bank, etc.)

• انواع منابع اطلاعات دستگاه تلفن همراه

• داخلی دستگاه (به عنوان مثال ، مکان GPS ، IMEI و غیره)

• منابع خارجی (به عنوان مثال CNN ، Chase Bank و غیره)

Mobile Device Information Leakage ...

- **Third-party mobile apps can leak info to external sources**
 - Send out device ID (IMEI/EID), contacts, location, etc.
 - Apps ask permission to access such info; users can ignore!
 - Apps can intercept info sent to a source, send to different destination!

• برنامه های تلفن همراه شخص ثالث می توانند اطلاعات را به منابع خارجی انتقال دهند

○ شناسه دستگاه (IMEI / EID)، مخاطبین، مکان و غیره را ارسال کنید.

○ برنامه ها اجازه دسترسی به چنین اطلاعاتی را می دهند. کاربران می توانند نادیده بگیرند!

○ برنامه ها می توانند اطلاعات ارسال شده به یک منبع را رهگیری کنند، به مقصد مختلف ارسال کنند!

Information Flow Tracking (IFT)

- IFT tracks each information flow among internal, external sources
- Each flow is tagged, e.g., “untrusted”
- Tag propagated as information flows among internal, external sources
- Sound alarm if data sent to third party

• IFT هر جریان اطلاعات را بین منابع داخلی و خارجی ردیابی می کند

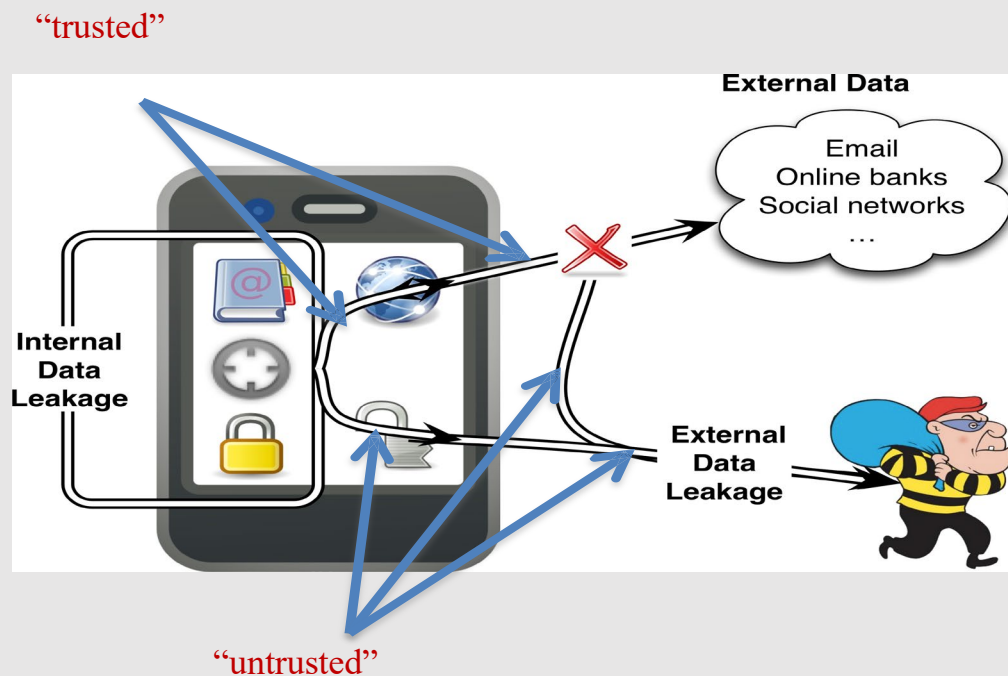
• هر جریان برچسب گذاری شده است، به عنوان مثال ، "غیر قابل اعتماد" ، "قابل اعتماد"

• برچسب به عنوان جریان اطلاعات در میان منابع داخلی و خارجی

• در صورت ارسال داده به شخص ثالث، زنگ هشدار

Information Flow Tracking (IFT) ...

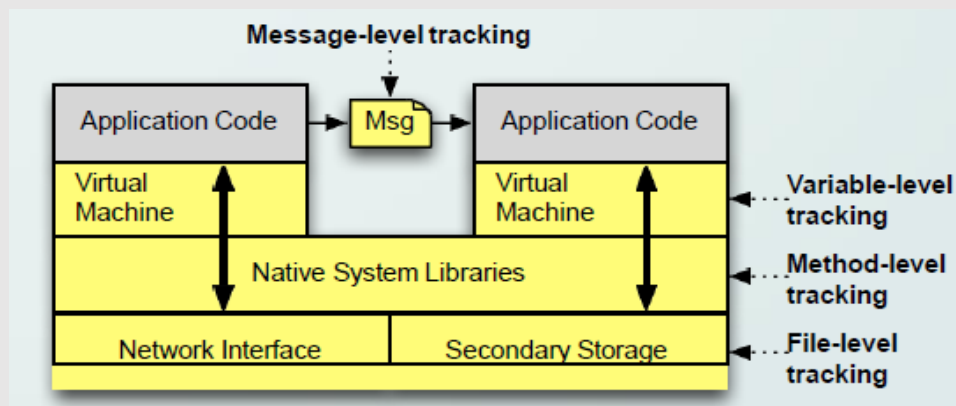
- چالش ها Challenges
 - Reasonable runtime, space overhead سرریز فضا ، زمان معقول زمان اجرا ،
 - Many information sources فراوانی از منابع اطلاعاتی



Information leakage on mobile devices

TaintDroid

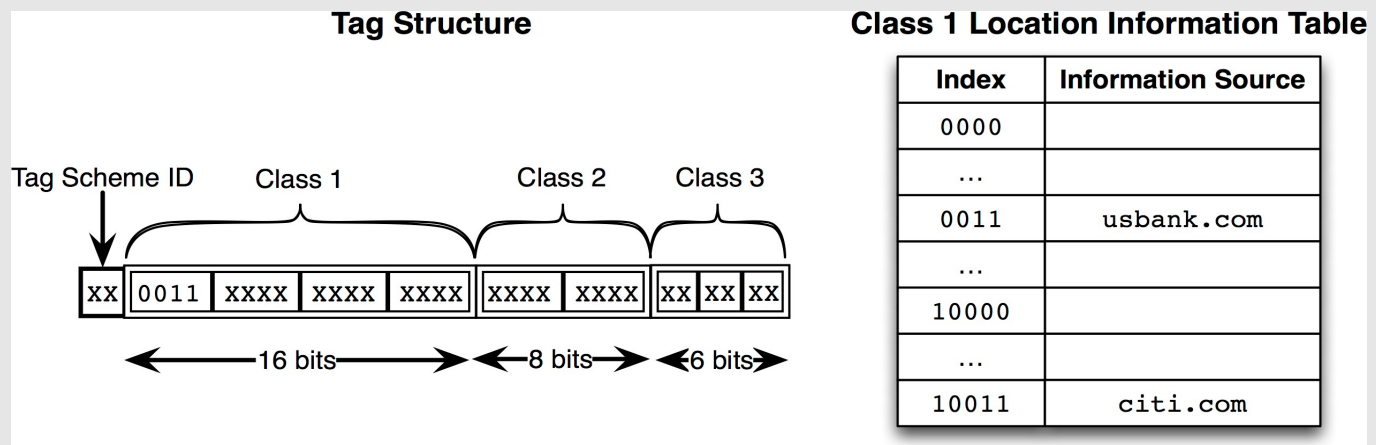
- Enck et al., OSDI 2010
- IFT system on Android 2.1



- System firmware (*not* app)
- Modifies Android's Dalvik VM, tracks info flows across methods, classes, files
- **Tracks the following info:**
 - Sensors: GPS, camera, accelerometer, microphone
 - Internal info: contacts, phone #, IMEI, IMSI, Google acct
 - External info: network, SMS
 - Notifies user of info leakage

D2Taint

- D2Taint uses fixed length tag (32 bits)
 - Tag includes segments corresponding to classes
 - Each segment stores *representations* of information sources in its class
 - Representation: info source's class table index
- Note: source table grows over time
 - Information source representation does *not* uniquely ID source



D2Taint ...

- D2Taint implemented on Android, Nexus One smartphones
- Evaluate D2Taint: 84 popular free apps from Google Play
- D2Taint has overhead similar to TaintDroid's

Location Privacy Protection

- Strong regulation مقررات شدید
 - Corporate
 - Individual
- Dynamic MAC and Bluetooth addresses و بلوتوث MAC آدرس های پویا
 - Collision
 - How often to change?
- Proxy-based communications ارتباطات مبتنی بر پروکسی
 - Dummy device as proxy
 - Group communications

Summary

- Mobile devices are increasingly popular
- There are many threats and attacks against mobile devices, e.g., loss/theft, sensitive information leakage, and location privacy compromise
- Mobile access control, information leakage protection, and location privacy protection, etc.

- دستگاه های تلفن همراه به طور فزاینده ای محبوب هستند
- تهدیدها و حملات زیادی علیه دستگاه های تلفن همراه ، به عنوان مثال ، از دست دادن / سرقت، نشت اطلاعات حساس و سازش حریم خصوصی مکان وجود دارد
- کنترل دسترسی به موبایل، محافظت در برابر نشت اطلاعات و محافظت از حریم خصوصی مکان و غیره

Thanks for your Attention.