TY BTech Semester-V (AY 2023-2024)

Computer Science and Engineering

Disclaimer:

a. Information included in these slides came from multiple sources. We have tried our best to cite the sources. Please refer to the references to learn about the sources, when applicable.

b. The slides should be used only for preparing notes, academic purposes (e.g. in teaching a class), and should not be used for commercial purposes.

CET4004B: Wireless and Mobile Device Security

Examination Scheme:

Continuous Assessment: 50 Marks End Semester Examination: 50 Credit: 3+1

Course Objectives:

- To understand wireless networks technologies and applications
- To study Ad-Hoc networks architecture and challenges
- To know Sensor networks architecture and applications
- To understand basic security needs and issues in wireless networks

Course Outcomes:

After completion of this course students will be able to:

- Compare different wired and wireless technologies
- Simulate wireless and Ad-Hoc networks
- Setup and configure Wi-Fi access point for security in wireless networks
- Understand practical aspects security wireless networks with programing and Tools

Pre-requisites

- Basics of Computer Networks
- Basics of Information Security

UNIT-V Mobile Device Security

Unit: V	**Mobile Device Security** Introduction to Device management and security, Android device architecture, Android Security Model, Device Permissions, Details of Device Security: Controlling OS Boot-Up and Installation, Verified Boot, Disk Encryption, Screen Security, Secure USB Debugging, device Backup, NFC Secure Elements.	9 Hrs

Introduction to Device management and security

Mobile device management features:

- The developers of mobile operating systems and manufacturers of mobile devices control what MDM software can and can't do on their devices through their APIs.
- As a result, mobile device management has become a commodity, with most vendors offering a similar set of core capabilities.
- MDM vendor differentiation comes by integrating mobile device management servers with other enterprise software.
- Common mobile device management features include:
  - device inventory and tracking;
  - app distribution and/or an enterprise app store;
  - remote wipe;
  - password enforcement;
  - app whitelisting and blacklisting; and
  - data encryption enforcement.

Mobile device management policies

So, what are mobile device management policies?

MDM policies answer questions about how organizations will manage mobile devices and govern their use.

To configure and publish their policies and processes, enterprises will ask questions, such as:

– Do devices need passcode protection?

– Should cameras be disabled by default?

– Is wi-fi connectivity important?

– What customization options will the device provide?

– Do certain devices need to be geo-fenced?

Components of mobile device management tools

Device tracking

- Each device enrolled with or issued by an enterprise can be configured to include GPS tracking and other programs.
- The programs allow an enterprise's IT professionals to monitor, update and troubleshoot the device in real time.
- They can also detect and report high-risk or non-compliant devices and even remotely lock or wipe a device if lost or stolen.

Mobile management

- IT departments procure, deploy, manage and support mobile devices for their workforce, such as troubleshooting device functionality.
- These departments ensure each device comes with the needed operating systems and applications for their users — including applications for productivity, security and data protection, backup and restoration.

Components of mobile device management tools

Application security

- Application security can involve app wrapping, in which an IT administrator applies security or management features to an application.
- Then that application is re-deployed as a containerized program.
- These security features can determine whether user authentication is required to open an app; whether data from the app can be copied, pasted or stored on the device; and whether the user can share a file.

Identity and access management (IAM)

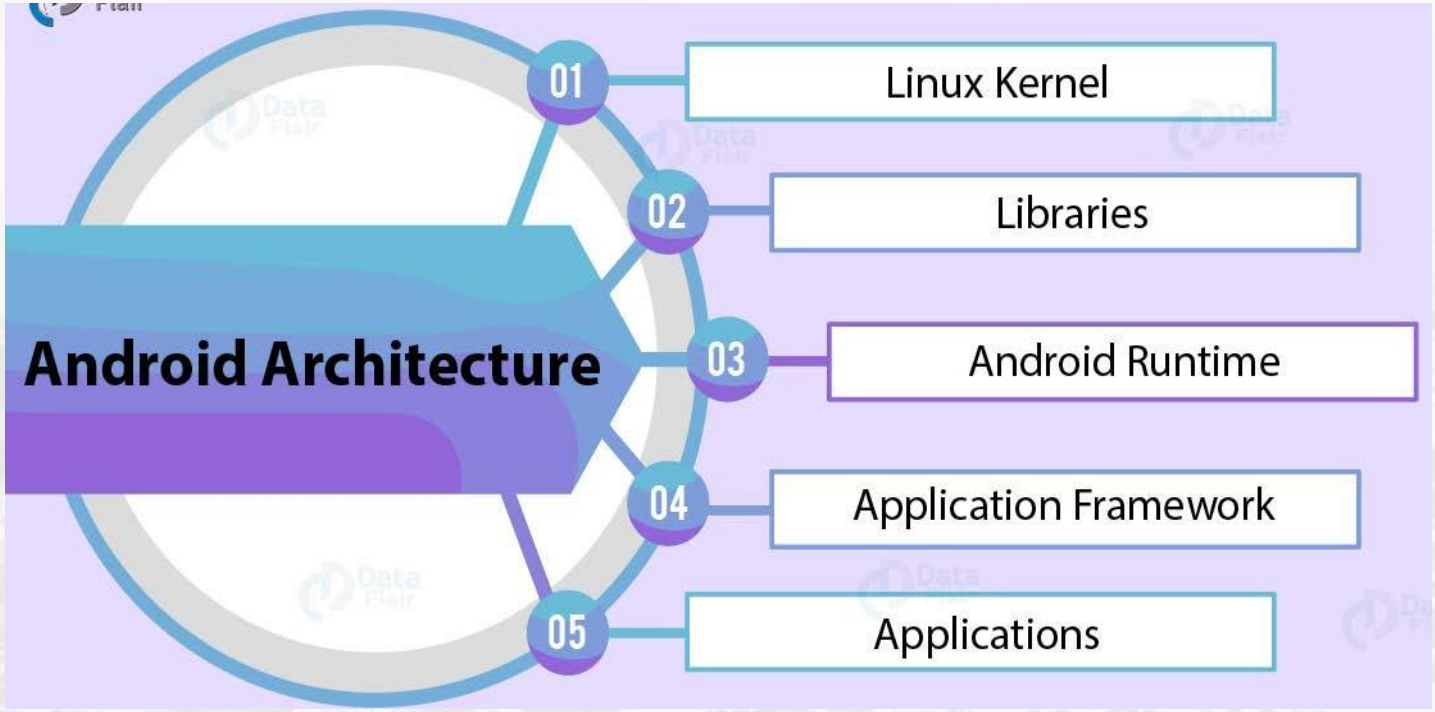
- Secure mobile management requires strong identity and access management (IAM).
- IAM allows an enterprise to manage user identities associated with a device.
- Each user's access within an organization can be fully regulated, using such features as single sign-on (SSO), multifactor authentication and role-based access.

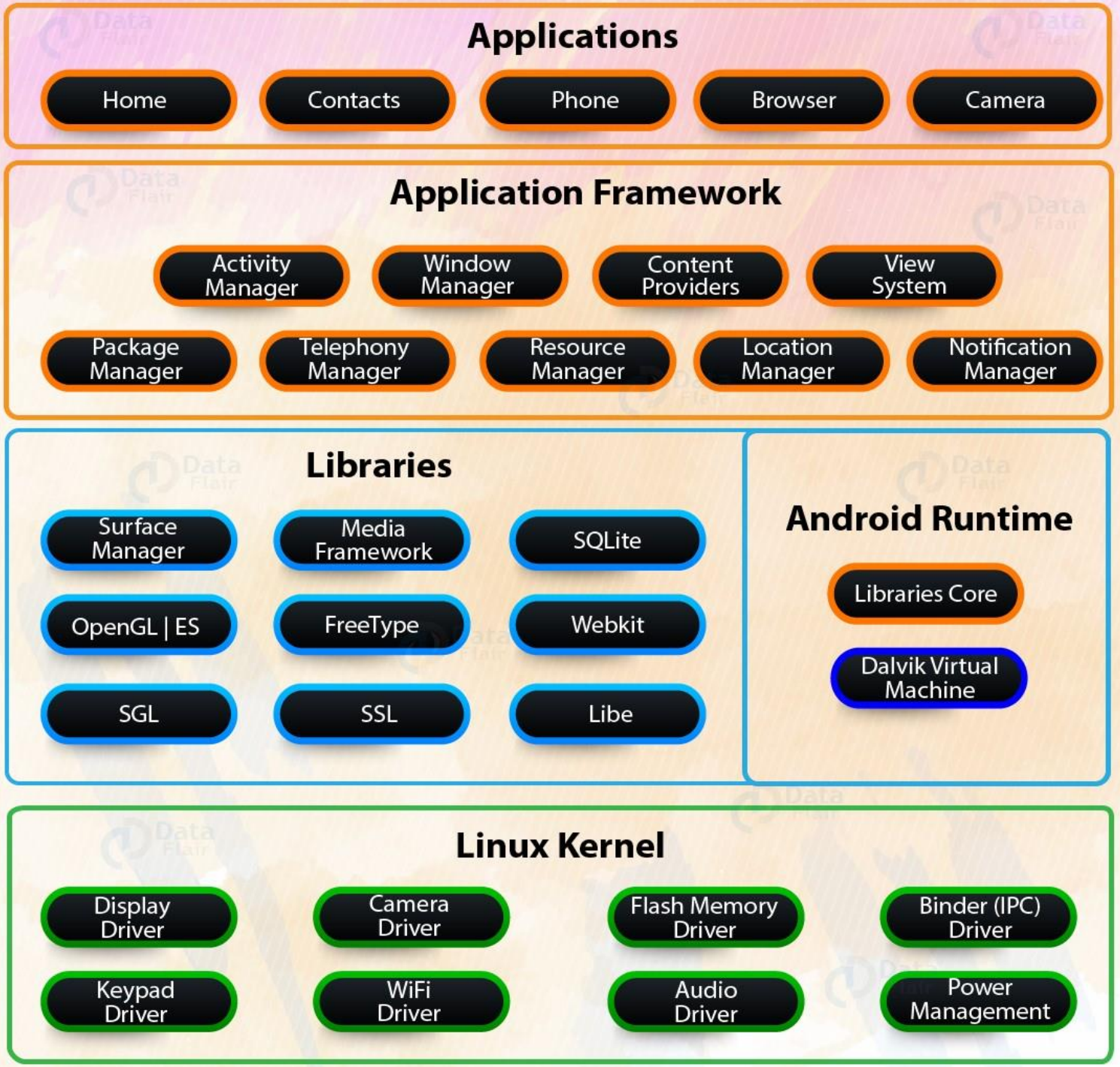
Endpoint security

- Endpoint security encompasses all devices that access a corporate network, including wearables, Internet of Things (IoT) sensors and non-traditional mobile devices.
- Endpoint security can include standard network security tools such as antivirus software and network access control and incident response, URL filtering and cloud security.

Android Architecture

Android is a mobile OS developed by Google that is based on the Linux kernel and written in Java.




Applications
Home
Contacts
Phone
Browser
Camera
Application Framework
Activity Manager
Window Manager
Content Providers
View System
Package Manager
Telephony Manager
Resource Manager
Location Manager
Notification Manager
Libraries
Surface Manager
Media Framework
SQLite
OpenGL | ES
FreeType
Webkit
SGL
SSL
Libe
Android Runtime
Libraries Core
Dalvik Virtual Machine
Linux Kernel
Display Driver
Camera Driver
Flash Memory Driver
Binder (IPC) Driver
Keypad Driver
WiFi Driver
Audio Driver
Power Management

1. Linux Kernel

It provides features such as:

- Security
- Process management
- Memory management
- Device management
- Multitasking

- It is also responsible for a level of abstraction between device hardware and upper layers of Android architecture. It consists of device drivers like camera, flash memory, Display, keypad, Wifi etc.

2. Libraries

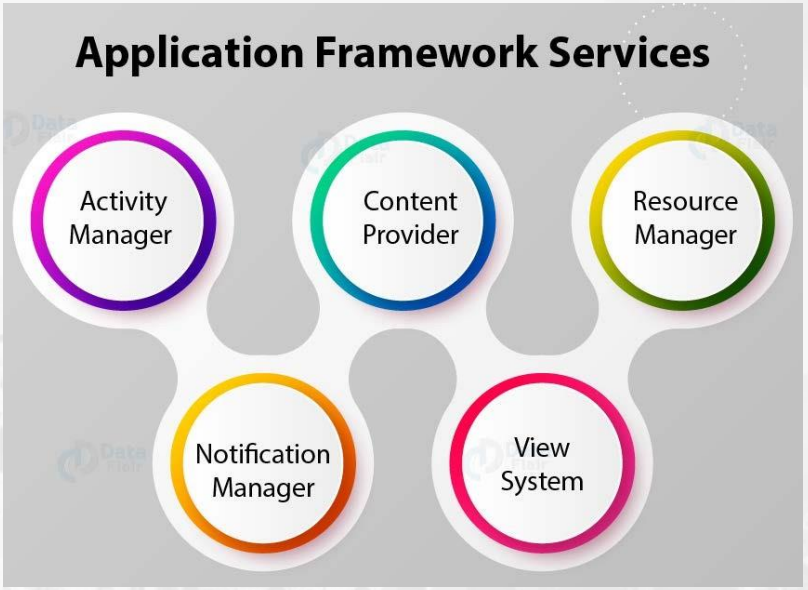
- This layer consists of a set of Libraries and Android Runtime. The Android component is built using native codes and require native libraries, which are written in C/C++ and most of the libraries are open source libraries.
- Also, this layer handles data that is specific to the hardware. Some of the native libraries are SSL, SQLite, Libc, OpenGL, media framework, FreeType and Surface Manager.

3. Android Runtime

- It comprises of DVM (Dalvik Virtual Machine).
- Just like JAVA uses JVM, Android uses DVM to optimize battery life, memory and performance.
- The byte code generated by the Java compiler has to be converted to .dex file by DVM, as it has its own byte code.
- Also, multiple class files are created as one .dex file and the compressed .jar file is greater than the uncompressed .dex file.

4. Application Framework

- **Activity Manager:** The method in this class uses testing and debugging methods.
- **Content provider:** It provides data from application to other layers.
- **Resource Manager:** It provides access to non-code resources.
- **Notification Manager:** The users get notification about all the actions happening in the background.
- **View System:** It acts as a base class for widgets and is responsible for event handling.



Applications

- It is the top-most layer of Android architecture.
- This layer consists of native Android applications and third-party installed apps.
- They are bundled in an Android package and all the applications that are to be installed are written in this layer only such as contacts, games, settings, and messages.

Details of Device Security: Android Security Model

Android's Five Key Security Features:

1. **Security at the operating system level through the Linux kernel**
2. **Mandatory application sandbox**
3. **Secure interprocess communication**
4. **Application signing**
5. **Application-defined and user-granted permissions**

Android Security: System-Level Security Features

- The Linux kernel provides Android with a set of security measures.
- It grants the operating system a user-based permissions model, process isolation, a secure mechanism for IPC, and the ability to remove any unnecessary or potentially insecure parts of the kernel.
- It further works to prevent multiple system users from accessing each other's resources and exhausting them.

Details of Device Security: Android Security Model

Android Application Security Features

- This user-based protection allows Android to create an “Application Sandbox.” Each Android app is assigned a unique user ID, and each runs as a separate process.
- Therefore, each application is enforced at the process level through the Linux kernel, which does not allow applications to interact with one another, and gives them only limited access to the Android operating system.
- This gives the user permission-based access control, and he/she is presented with a list of the activities the Android application will perform and what it will require to do them, before the app is even downloaded.
- The same goes for filesystem permissions – each application (or user) has its own files, and unless a developer explicitly exposes files to another Android application, files created by one application cannot be read or altered by another.

Details of Device Security: Android Security Model

Android Application Security Scans

When building and testing the security of Android apps, developers should follow Android security best practices and keep the following in mind when performing security tests:

- Inbound SMS listeners (command and control)
- Unsafe file creation
- Improper database storage
- Unsafe use of shared preferences
- Storage of sensitive data on mass storage device
- Content provider SQL injection
- APN or proxy modification

The Basics of Mobile Security

What is Mobile Security?

- Mobile security is essential to protect mobile devices, such as smartphones, tablets, and laptops, from threats like viruses, malware, and unauthorized data leakage.
- These devices can contain critical personal and financial information which makes them a tempting target for cybercriminals.
- To fight these cyber threats, Android and iOS deploy technologies to secure data transmission, applications, and intrusions. The configuration, implementation, and accessibility of these systems could differ.
- Poorly secured devices face the chance of being hacked, and data extracted or encrypted for a ransom later.

iOS and Android:
A Brief Overview Brief introduction to iOS and Android

- iOS, created by Apple, is an operating system primarily used on iPhone and iPad devices.
- Built on Unix, it is renowned for its reliable performance and robust security.
- In the ever-growing iOS vs Android security war, iOS users have fewer customization options due to its “closed” environment, also known as the “walled garden.”
- All apps on an iOS device must be downloaded from the Apple App Store.
- Android is an open-source OS powered by Linux Kernel.
- As such, it can be altered and utilized by other device manufacturers.
- This has been popularized by Google, and it’s often seen as a more user-friendly, customizable system, boasting the most available apps (though not all come from the official Play Store).

iOS Security

Pros of iOS Security

iOS is renowned for its security, owing to its closed-source code and “walled garden” approach. Some benefits that set it apart in the iOS vs Android security debate include:

- It creates a stable, secure environment, minimizes the risk of malware, and only allows vetted applications into the Apple App Store.
- Data encryption is enabled by default and biometric security features like FaceID and TouchID provide strong user authentication.
- Apple also provides quick fixes for any vulnerabilities via timely patches, due to the control it has over update distribution.
- Furthermore, data sharing between devices is also tightly limited, providing an extra layer of privacy.

Cons of iOS Security

Unfortunately, iOS is not immune to security vulnerabilities despite its high standards. Some pitfalls include:

- With all updates processed solely by Apple, any significant deviations or malfunctions may delay rectification.
- Centralized control limits customization options and restricts data transfer between devices.
- Additionally, a security breach at Apple could affect all iOS devices, though this is highly improbable.
- The reliance on a single App Store amplifies the possibility of a single point of failure.

Note on the threat level

- The iOS environment is generally secure due to Apple's tight control over the apps available in the App Store, making it far less susceptible to malware than Android.
- Nonetheless, caution should still be taken when accessing links from untrusted sources, avoiding jailbreaking and only downloading app from legitimate App stores to ensure optimal security.

Android Security

Pros of Android Security

Android stands out from iOS in the iOS vs Android security debate due to its open-source platform, which allows for a diverse community of users to review and patch vulnerabilities to enhance security. Some benefits include:

- Custom ROMs can be installed on devices to provide additional features.
- Many quality antivirus apps are available for Android users.
- Google Play scans for harmful software and alerts users, and UVFS (Linux Vault File System) encrypts user data.
- Furthermore, the diversity of Android devices brings various security features from different manufacturers.

Android Security

Cons of Android Security

Android's open-source nature exposes it to various security vulnerabilities. Its large market share makes it an attractive target for attackers.

- Unfortunately, updates to the Android OS are often stalled by manufacturers, leading to an increased risk of security breaches.
- Additionally, users have the freedom to customize their devices, which can lead to them unintentionally downloading malware.
- Furthermore, fragmentation in the Android ecosystem makes detecting surveillance challenging.
- Finally, third-party app stores typically have insufficient review processes, which can result in malware-laden applications.

Note on the threat level

- Android security has come a long way in recent years, yet it is still at higher risk due to its open-source environment and popularity.
- To stay secure, users should make sure to keep their software updated, avoid downloading apps from untrustworthy sources, not alter security settings unnecessarily, and use antivirus protection. Bad user habits are more likely to be targeted by attackers than the system itself.

iOS and Android: Security Comparison

- Both Android and iOS possess strengths and weaknesses in security.
- While iOS impresses with its “walled garden” approach and solid commitment to security updates, it carries the risk of having a single point of failure.
- Android, on the other hand, with all its adaptability and open sources, becomes a dual-edged sword, where customization poses both protection advantages and serious hazards.
- The system-wide security is as strong as individual user habits.
- To draw a conclusion, iOS vs Android security isn’t so much a matter of the operating system you’re using, as it is of your behaviors and alertness as a user — regardless of whether on Android or iOS.
- In the comparison, provided you are using the latest software updates and practicing cybersecurity best practices, both platforms have improved dramatically in security from their inception.
- Personal responsibility combined with operating systems engineering leads us toward a future of more secure mobile experiences.

iOS vs Android security: Vulnerability Comparison

- From a vulnerability standpoint, both operating systems have had histories of exploiting but have also shown prompt commitment toward addressing and resolving these issues.
- iOS has had a strong track record for fixing security vulnerabilities yet attracts its share of high-value security threats given its popularity with a premium-end user base.
- Cases such as the XCodeGhost and WireLurker compromise, and the consistent discovery of zero-day exploits underscore this.
- On the other hand, Android faces more threats owing to its extensive user base and open-source platform.
- This features malware such as CopyCat, which affected millions, as well as ransomware like Charger inserting via Google Play.
- Cases continue to multiply, partly influenced by many users operating on outdated versions.
- As such, instead of the iOS vs Android security debate, secure user behaviors –— such as regularly updating the system, avoiding public and insecure Wi-Fi networks, and downloading apps strictly from recognized App stores — are crucial across both platforms.

Device Permissions

- You can allow some apps to use various features on your device, such as your camera or contacts list.
- An app will send a notification to ask for permission to use features on your device, which you can Allow or Deny.
- You can also change permissions for a single app or by permission type in your device's settings.
- Android app permissions are designed to protect users' privacy and security by controlling what an app can access and do on a device.

App permissions help support user privacy by protecting access to the following:

- **Restricted data**, such as system state and users' contact information
- **Restricted actions**, such as connecting to a paired device and recording audio

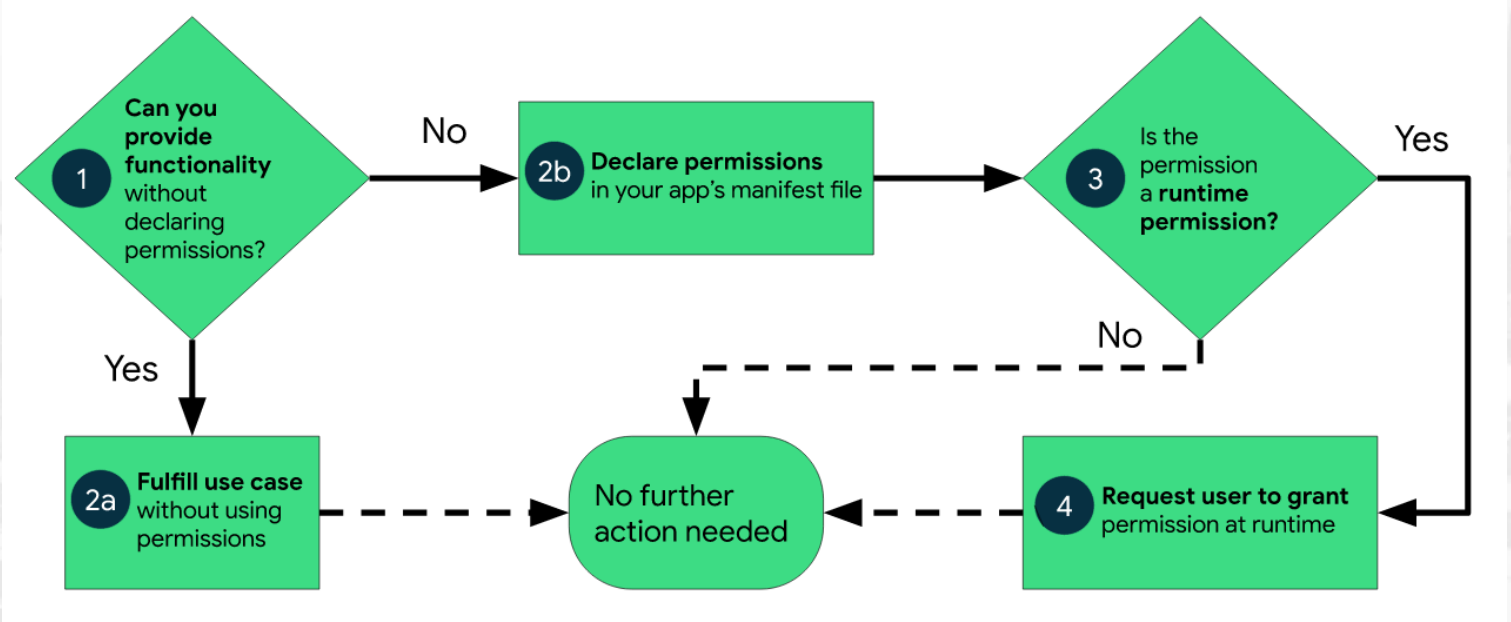
Device Permissions

Workflow for using permissions

If your app offers functionality that might require access to restricted data or restricted actions, determine whether you can get the information or perform the actions without needing to declare permissions. You can fulfill many use cases in your app, such as taking photos, pausing media playback, and displaying relevant ads, without needing to declare any permissions.

If you decide that your app must access restricted data or perform restricted actions to fulfill a use case, declare the appropriate permissions. Some permissions, known as install-time permissions, are automatically granted when your app is installed. Other permissions, known as runtime permissions, require your app to go a step further and request the permission at runtime.

Figure illustrates the workflow for using app permissions:



Device Permissions

Types of permissions

Android categorizes permissions into different types, including install-time permissions, runtime permissions, and special permissions. Each permission's type indicates the scope of restricted data that your app can access, and the scope of restricted actions that your app can perform, when the system grants your app that permission. The protection level for each permission is based on its type and is shown on the permissions API reference page.

Disk Encryption

Securing Mobile Devices with Mobile Encryption

- Full-disk encryption is the process of encoding all user data on an Android device using an encrypted key.
- Once a device is encrypted, all user-created data is automatically encrypted before committing it to disk and all reads automatically decrypt data before returning it to the calling process.
- Encryption ensures that even if an unauthorized party tries to access the data, they won't be able to read it. Android has two methods for device encryption: file-based encryption and full-disk encryption.
- With the rise in mobile devices, it makes sense that more businesses are using mobile devices to process, store, and transmit card data.
- But with the rise in technology comes the rise in all sorts of security issues. One common issue is stolen or lost devices.
- Say you have a tablet that has sensitive information on it, such as card data, personal information, etc.
- If that tablet is stolen, all that data is now in the wrong hands. So how do you secure that data? Things like physical security and mobile device policies are good at protecting the device itself, but one way to protect the data on the device is encryption.

Disk Encryption

Securing Mobile Devices with Mobile Encryption

What is encryption?

- Full disk encryption is basically encryption on a hardware level.
- It automatically converts data on a hard drive into something that can't be deciphered without the key.
- Without the right authentication key, the data is inaccessible, even if a hard drive is removed and placed in another machine. Full disk encryption (FDE) encrypts all the data on your storage device.
- The idea is to protect your data from falling into the wrong hands, should someone get ahold of a mobile device.
- What's nice about FDE is it's automatic, so it requires no special action from the user other than providing a key. As data is written, it's automatically encrypted, and as it's read, it's automatically decrypted.
- Mobile devices like smartphones and tablets have encryption options that will also provide protection of storage.
- In this case, it's not typically a disk but is still just storage that's encrypted and accessed using some key. It's usually just a matter of enabling the appropriate options and an extra step to provide a key.

Disk Encryption

Securing Mobile Devices with Mobile Encryption

Why should I use encryption?

- If your organization deals with a lot of mobile devices that carry critical data, it's a good idea to make sure none of that data falls into the wrong hands.
- Using encryption is another step to properly securing your data.
- Taking this extra step in security can help many organizations.
- This can also protect you from liability.
- If a device is lost or stolen, and it was fully encrypted, organizations don't have to report a breach.

What should I use encryption for?

- Encryption is useful for laptops and other smaller devices that can be physically stolen/lost. This ensures that should a laptop, phone, USB, etc. is stolen or lost, the data is still secured.
- While it may be true that encrypting mobile devices is not required by all government or financial mandates, taking this extra step in security can help many organizations.
- You should consider encryption for any mobile device that is storing sensitive data.

Disk Encryption

Securing Mobile Devices with Mobile Encryption

What type of encryption should I get?

- There are many different types of encryption software and tools. Some come with other security elements included. Many computers and software already come with options like full disk encryption.
- But the problem is this software is usually available on most devices, but many businesses don't realize it hasn't been implemented. Fortunately, it's fairly easy to activate encryption on devices.
- Check if your current software offers storage encryption. If not, there are plenty of tools that offer encryption.

How secure is encryption?

- Keep in mind that encryption doesn't guarantee the security of your data. Encryption keys can still be stolen.
- With full disk encryption, cold boot attacks can be used where keys are stolen by cold booting a machine, then dumping the contents of its memory before the data disappears.
- Some best practices are to secure the encryption key properly, employ a strict password policy, and limit access to these keys.
- If your business uses a lot of mobile devices, implementing encryption is a great security tool to protect your data.

Screen Security

- With the growth in smartphone usage around the world, issues surrounding mobile security have grown as well. It is more important than ever to arm your mobile devices with protective software.
- The largest target for attacks on mobile devices is Google's Android operating system, due to the rapid expansion and market penetration of Android smartphones.

Secure Your Lock Screen

- Smartphones contain a wealth of your personal information, ranging from personal messages and photos, to bank information.
- In the event of your mobile device being lost or stolen, the first line of defense is locking it securely. Smartphones offer several locking options including pins, passwords and biometric methods.

Screen Security

Pattern

- Uncheck the "make pattern visible" option in the settings. This makes it more difficult for people around you to see your pattern.
- Use six or more nodes.
- Don't use a simple or common pattern. 40% of patterns start in the top left corner, and 77% start in one of the four corners.

PIN

- Use at least a six digit PIN.
- Avoid simple number sequences (123456), simple patterns of numbers (147147), and just repeating the same number (11111).
- Don’t use a significant date like a birthday or anniversary.
- Don’t use any part of your address as your PIN number.

Screen Security

Passwords

- Long alphanumeric passwords are stronger than either a PIN or a pattern.
- A common mistake is using passwords like **123456** or **password**.
- Always remember that the longer the password, the more secure it is.

Face ID and Touch ID

- There is a 1 in 50,000 chance that someone else's fingerprint will unlock your phone.
- There is a 1 in a million chance that someone else's face will unlock your phone.
- Combine this with a strong pattern, PIN or password for the best security.

Secure USB Debugging

What is USB debugging?

- USB debugging is often used by developers or IT support people to connect and transfer data from an Android device to a computer. While this feature is useful, a device isn't as secure when connected to a computer. So that's why some organizations require you to turn this setting off.

Where is USB debugging security settings?

- Enable USB debugging in the device system settings under Developer options.
- You can find this option in one of the following locations, depending on your Android version: Android 9 (API level 28) and higher: Settings > System > Advanced > Developer Options > USB debugging.

Is USB debugging safe for Android?

- Trustwave recommends that mobile devices should not be set to USB Debugging mode. When a device is in USB Debugging mode, a computer connected to the device can read all data, run commands, and install or remove apps.
- The security of the device settings and data could be compromised.

Web Link: https://www.embarcadero.com/starthere/xe5/mobdevsetup/android/en/enabling_usb_debugging_on_an_android_device.html

Secure USB Debugging

- USB debugging on Android allows developers to connect their devices to a computer and access advanced debugging and development features.
- By enabling USB debugging, developers can install and debug apps directly on the device, access device logs for troubleshooting, and use development tools for performance analysis and testing.
- It's a crucial setting for developers to interact with their Android device through a USB connection and perform various tasks related to app development, testing, and debugging.

Web Link: https://www.scaler.com/topics/usb-debugging-on-android/

What Is USB Debugging Mode on Android?

- A key feature of Android devices is USB Debugging, which enables connectivity with a computer running the Android Software Developer Kit (SDK) and allows for more complex operations and testing of Android applications. The SDK is a crucial tool for Android app development since it gives programmers the tools and frameworks they need to create reliable applications. In combination with the SDK, Android Studio, a development environment for Android apps,` offers several crucial tools, including a visual editor and a debugger for troubleshooting issues.
- Even while Android devices come with a tonne of capability by default, developers still need extra tools to make operations like transferring files across devices, executing instructions, and carrying out complex tasks more efficient. A set of tools are available for developers to carry out these tasks through Android Studio and the Android SDK`. You must enable USB Debugging on your Android smartphone to use these tools.
- The Android SDK may be installed separately from Android Studio, which is sometimes required for advanced operations like rooting. Your Android smartphone can connect with a PC if USB Debugging is enabled, enabling you to fully utilize the tools offered by the Android SDK and Android Studio.

Web Link: https://www.scaler.com/topics/usb-debugging-on-android/

Is USB Debugging Safe?

As long as you utilize USB debugging carefully, it's typically safe to enable it on your Android smartphone. Here are a few things to think about:

- **Higher security risks:** Enabling USB debugging gives your device more freedom to connect with other devices, including potentially hostile PCs. If you connect your device to an unreliable computer, this might raise the danger of infection or data theft.
- **Accidental actions:** When USB debugging is enabled, it is simpler to make decisions that might damage or erase crucial data from your device. For instance, if you use the command line to push files to your device, you can unintentionally destroy a crucial file.
- **Limitations of developer mode:** In some circumstances, turning on USB debugging necessitates turning on developer mode on your device, which may restrict some security features. For instance, turning on developer mode might eliminate some certificate checks, which makes it simpler for attackers to take advantage of flaws in your device.
- **Insecure connections:** It's crucial to have a secure connection while utilizing USB debugging to connect your device to a computer. The data being exchanged between your device and the computer might be intercepted if you use a public or insecure Wi-Fi network.

Web Link: https://www.scaler.com/topics/usb-debugging-on-android/

MIT-WPU
॥ विश्वशान्तिर्ध्रुवं ध्रुवा ॥

USB Debugging
Security Risks and Precautions

While USB Debugging is a helpful feature for developers and experienced Android users, if used carelessly, it also poses certain security hazards. Here are some safety measures you may do to reduce the hazards connected with USB debugging:

- **Only allow USB Debugging when necessary:** Only enable USB Debugging when essential for certain tasks like app development or software upgrades. Your device becomes more exposed to assaults if USB Debugging is left turned on constantly.
- **Keep your device updated:** Make sure that the software on your Android smartphone is updated often to prevent security flaws.
- Use a passcode or pattern lock to secure your device to prevent unauthorized access to your data if someone has physical access to it.
- **Only allow trusted computers:** Only allow trusted computers when connecting your device to a computer. Avoid using insecure or public computers since they can include harmful software that jeopardizes the security of your device.
- **Use reliable apps:** Install software only from respected stores, such as the Google Play Store. Malware installation on your smartphone is more likely to occur when you install apps from unidentified sources.
- **Use a VPN:** By encrypting your internet connection, a virtual private network (VPN) makes it more challenging for hackers to steal information being transferred to and from your device.
- **Use antivirus software:** To shield your device from malware and other security risks, use a reliable antivirus program.
- **When not in use, turn off USB Debugging:** Be sure to turn off USB Debugging after using it. This lowers the possibility of someone breaking into your device.

Web Link: https://www.scaler.com/topics/usb-debugging-on-android/

Overview of various Mobile Malware

- Mobile malware is malicious software specifically written to attack mobile devices such as smartphones, tablets, and smart watches.
- These types of malware rely on exploits of particular mobile operating systems and mobile phone technology.
- Although mobile malware is not as pervasive as malware that attacks traditional workstations, it is a growing threat to consumer devices.
- Mobile malware is becoming a challenge to the mobile security industry as attacks increase in frequency and strength.
- Mobile malware developers, also called cybercriminals, may have one or several objectives, including stealing data, signing users up for services and charging them fees for services they did not agree to or locking a device or data and demanding money for its release.

Web Link:

https://www.techtarget.com/searchmobilecomputing/definition/mobile-malware#:~:text=Types%20of%20mobile%20malware,%2C%20ransomware%2C%20spyware%20and%20Trojans.

https://www.crowdstrike.com/cybersecurity-101/malware/mobile-malware/

https://infinitysol.com/mobile-malware-attacks-and-defense/

Overview of various Mobile Malware

Types of mobile malware

The most common mobile malware attacks include viruses, worms, mobile bots, mobile phishing attacks, ransomware, spyware and Trojans. Some mobile malware combines more than one type of attack.

Mobile viruses

These are adapted for the cellular environment and designed to spread from one vulnerable phone to another.

Computer worm

This is a type of malware that infects other devices while remaining active on infected systems. Cybercriminals can transmit worms through short message service (SMS) or Multimedia Messaging Service (MMS) text messages and typically do not require user interaction to execute commands.

Mobile bot

This is a type of malware that runs automatically once a user installs it on a device. It gains complete access to the device and its contents and starts communicating with and receiving instructions from one or more command and control servers. A cybercriminal called a botmaster adds and manages the infected devices to a network of mobile bots (botnet).

Mobile phishing

These attacks often come in the form of email or SMS text messages. SMS phishing, sometimes called SMiShing, uses text messaging to convince victims to disclose account credentials or to install malware. The attack masquerades as a reputable entity or person and distributes malicious links or attachments that can extract login credentials or account information from victims.

Overview of various Mobile Malware

Ransomware

This is a type of malware that locks the data on a victim's device or the device itself, typically by encryption, and demands payment before the data or device is decrypted and access returned to the victim. Unlike other types of attacks, the victim is usually notified that an exploit has occurred and is given instructions on how to recover the data. Cybercriminals often demand payment in a Cryptocurrency such as Bitcoin, so that the cybercriminal's identity remains unknown.

Spyware

These attacks synchronize with calendar apps, passwords, email accounts, notes and other sources of personal data, collects that data and sends it to a remote server.

It is often attached to free software downloads or to links clicked by users. Peer-to-peer (P2P) file sharing has increased the amount of spyware and the ramifications. Adware is a type of spyware.

Trojan horse

This virus type requires users to activate it. In mobile devices, cybercriminals typically insert Trojans into non-malicious executable files or apps on the device. The user activates the Trojan virus when he or she clicks or opens a file. Once activated, Trojans can infect and deactivate other applications or the device itself and paralyze the device after a certain period of time or a certain number of operations. Banking Trojans target both international and regional banks by using fake versions of legitimate mobile apps or through phishing campaigns.

Wireless Application Protocol (WAP) clickers are Trojan viruses that use WAP billing to charge fees directly to a user's mobile phone bill. Mobile network operators use WAP billing for paid services or subscriptions. This form of payment charges fees directly to the user's service account, avoiding the need to register a credit card or set up an account. A WAP clicker covertly subscribes to a cybercriminal's services and charges the mobile device owner's account.

Mobile malware prevention

- Anti-malware software for mobile devices can minimize the risks, but administrators should be proactive to reduce attacks. Anti-malware software can come in two forms: apps that users can download to their devices, and mobile threat defense, which administrators can incorporate into an Enterprise Mobility Management (EMM) strategy and then deploy to their mobile device fleet.
- Businesses and mobile administrators can reduce mobile attacks by upgrading to the latest security updates and OS updates for iOS and Android. Administrators should keep up to date about mobile threats so they can blocklist and allowlist apps, which prevents users from downloading certain applications onto a device. Administrators can also perform jailbreak/rooting and unlocked bootloader detection, disallow untrusted sources and third-party app stores, and require complex passcodes.
- User training is also important. Users must know what they should and should not do with their devices. Mobile device management (MDM) and unified endpoint management (UEM) systems can also help protect both personal and company-owned devices and ensure that admins have the proper visibility to keep things in check.
- Consumer mobile users should keep their devices up to date with the latest OS updates and educate themselves on emerging threats.

Identity theft

What is identity theft?

Identity theft happens when someone uses your sensitive data to pose as you or steal from you. Identity thieves may drain your bank and investment accounts, open new credit lines, get utility service, steal your tax refund, use your insurance information to get medical treatments, or give police your name and address when they are arrested.

Frequent data breaches mean your information may already be exposed. In this new reality, it's smart to take steps to prevent malicious actors from using your personal information and ruining your financial life.

Web Link:

https://www.nerdwallet.com/article/finance/how-to-prevent-identity-theft

Ways to prevent identity theft

1. Freeze your credit

Freezing your credit with all three major credit bureaus — Equifax, Experian and TransUnion — restricts access to your records so new credit files cannot be opened. It's free to freeze your credit and unfreeze when you want to open an account, and it provides the best protection against an identity thief using your data to open a new account.

2. Safeguard your Social Security number

Your Social Security number is the master key to your personal data. Guard it as best you can. When you are asked for your number, ask why it is needed and how it will be protected. Don't carry your card with you. Securely store or shred paperwork containing your Social Security number.

3. Be alert to phishing and spoofing

Scammers can make phone calls appear to come from government entities or businesses, and emails that appear to be legitimate may be attempts to steal your information. Initiate a callback or return email yourself, working from a known entity such as the official website, rather than responding to a call or email. And be wary of attachments — many contain malware.

4. Use strong passwords and add an authentication step

Use a password manager to create and store complex, unique passwords for your accounts. Don't reuse passwords. Adding an authenticator app can reduce your risk. Don't rely on security questions to keep your accounts safe; your mother's maiden name and your pet's name aren't hard to find. Think carefully about what you post on social media so you don't give away key data or clues about how you answer security questions.

5. Use alerts

Many financial institutions will text or email when transactions are made on your accounts. Sign up so that you know when and where your credit cards are used, when there are withdrawals or deposits to financial accounts and more.

Ways to prevent identity theft

6. Watch your mailbox

Stolen mail is one of the easiest paths to a stolen identity. Have your mail held if you're out of town. Consider a U.S. Postal Service-approved lockable mailbox. You can also sign up for Informed Delivery through the USPS, which gives you a preview of your mail so you can tell if anything is missing.

7. Shred, shred, shred

Any credit card, bank or investment statements that someone could fish out of your garbage shouldn't be there in the first place. Shred junk mail, too, especially preapproved offers of credit.

8. Use a digital wallet

If you're paying online or in a store, use a digital wallet, an app containing secure, digital versions of credit and debit cards. You can use it to shop online or at a compatible checkout terminal. Transactions are tokenized and encrypted, which makes them safer. In addition, contactless transactions have fewer health risks.

9. Protect your mobile devices

Use passwords on your electronic devices. Use a banking app rather than a mobile browser for banking.

10. Check your credit reports regularly

The three major credit reporting bureaus give consumers access to free credit reports weekly, accessible by using AnnualCreditReport.com. Check to be sure that accounts are being reported properly and watch for signs of fraud, like accounts you don't recognize. You can also sign up for a free credit report and score from NerdWallet to receive alerts when there are changes.

11. Monitor financial and medical statements

Read financial statements. Make sure you recognize every transaction. Know due dates and call to investigate if you do not receive an expected bill. Review "explanation of benefits" statements to make sure you recognize the services provided to guard against health care fraud.

IoS Security- iOS security overview-pairing

iOS and iPadOS use a pairing model to control access to a device from a host computer. Pairing establishes a trust relationship between the device and its connected host, signified by public key exchange. iOS and iPadOS also use this sign of trust to enable additional functionality with the connected host, such as data syncing. In iOS 9 or later, services:

- That require pairing can't be started until after the device has been unlocked by the user
- Won't start unless the device has been recently unlocked
- May require the device to be unlocked to begin (such as with photo syncing)

The pairing process requires the user to unlock the device and accept the pairing request from the host. In iOS 9 or later, the user is also required to enter their passcode, after which the host and device exchange and save 2048-bit RSA public keys. The host is then given a 256-bit key that can unlock an escrow keybag stored on the device. The exchanged keys are used to start an encrypted SSL session, which the device requires before it sends protected data to the host or starts a service (iTunes or Finder syncing, file transfers, Xcode development and so on). To use this encrypted session for all communication, the device requires connections from a host over Wi-Fi, so it must have been previously paired over USB. Pairing also enables several diagnostic capabilities. In iOS 9, if a pairing record hasn't been used for more than 6 months, it expires. In iOS 11 or later, this time frame is shortened to 30 days.

Certain diagnostic services, including com.apple.mobile.pcapd, are restricted to work only over USB. Additionally, the com.apple.file_relay service requires an Apple-signed configuration profile to be installed. In iOS 11 or later, Apple TV can use the Secure Remote Password protocol to wirelessly establish a pairing relationship.

A user can clear the list of trusted hosts with the Reset Network Settings or Reset Location & Privacy options.

Introducing app security

Potential Security risks in iOS

Data leak — By using application user mostly enters their private data and storing that data in an unsecured manner that creates risk of this data being leaked if device got unauthorized hands.

Man in middle attack — Intracting http/https requests and responses is relatively easy to do when it comes to iOS apps. Using tools like Charles Proxy, even an amateur can get to know our app requests, corresponding server responses, and manipulate network traffic by sending doctored requests. Unfortunately, SSL is not enough to make your app secure.

How to make our app secure?

in this article, we will discuss mistakes that developers make towards app security and how to taking care while developing iOS app.

User data protection (Keychain vs. NSUserDefaults for storing sensitive data)

As per researched on multiple apps from the Store and a lot of them are doing the same mistake, storing sensitive data where they do not belong.

If you are storing sensitive data in **UserDefaults**, then you are risking your application’s information.

Introducing app security

UserDefaults get stored simply as a property list file that is located inside Preferences folder of your app. They get saved in our app without being encrypted .

Basically, by using a third party mac application like **iMazing** without even having to Jailbreak your device, you can easily view **UserDefaults** data for any app downloaded from the AppStore.

These mac apps are simply designed to allow you to explore and manage third-party application files that are on your iPhone. And you can easily explore **UserDefaults** of any app.

Examples are **Access Tokens, subscription flags, email ,password** ,etc.

All this data can be easily retrieved and altered and make damage to apps, from free usage of paid features to hacking network layer and much more.

You should always keep in mind one thing **UserDefaults** is designed only to save a small amount of data like preferences of a user inside the app, like stuff that is completely insensitive.

MIT-WPU
॥ विश्वशान्तिर्ध्रुवं ध्रुवा ॥

Introducing app security

Keychain API

In Order to save our apps sensitive data, we must use Security services provided by Apple.
Keychain services API helps you solve these problems by giving your app a way to store the small amount of user data in an encrypted database called the **keychain**.
here some below is sample for store and retrieve values
for save data we need to query with kSecClass , kSecAttrAccount , kSecAttrServer , kSecValueData as below on key

```
class func save(key: String, data: NSData) -> OSStatus {
    let query = [
        kSecClass as String       : kSecClassGenericPassword as String,
        kSecAttrAccount as String : key,
        kSecValueData as String   : data ] as [String : Any]

    SecItemDelete(query as CFDictionary)

    return SecItemAdd(query as CFDictionary, nil)

}
```

Introducing app security

For Get data from keychain by key which was used at save data like below

```
class func load(key: String) -> NSData? {
    let query = [
        kSecClass as String       : kSecClassGenericPassword,
        kSecAttrAccount as String : key,
        kSecReturnData as String  : kCFBooleanTrue,
        kSecMatchLimit as String  : kSecMatchLimitOne ] as [String : Any]

    var dataTypeRef:AnyObject? = nil

    let status: OSStatus = SecItemCopyMatching(query as CFDictionary, &dataTypeRef)

    if status == noErr {
        return (dataTypeRef! as! NSData)
    } else {
        return nil
    }
}
```

Web Link: https://medium.com/@ankurvekariya/ios-app-security-tips-and-tricks-42cdf9301181

Introducing app security

Below are example for saving string value with keychain and retrieve it:
Example: save (here “XYZ ” value stored with key “**TestString**”)

```
//save data example
let data = KeyChain.stringToNSDATA(string: "XYZ")
let status = KeyChain.save(key: "TestString", data: data)
print("status = \(status)")
```

save value

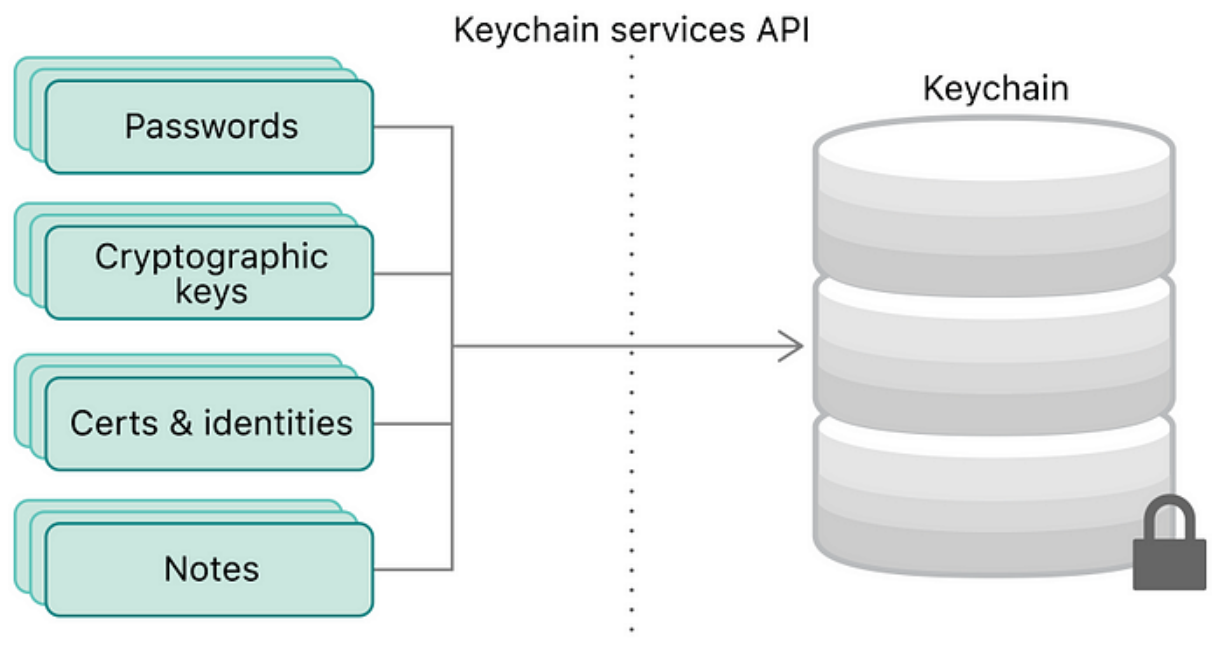
Get Value (here getting stored value from “TestString”)

```
    //get data example
    if let receivedData = KeyChain.load(key: "TestString") {
        let result = KeyChain.NSDATAtoString(data: receivedData)
        print("Result = \(result)")
    }
}
```

receiving data from keychain

Introducing app security

In the **keychain**, you are free to save **passwords and other secrets** that the user explicitly cares about, such as credit card information or even short sensitive notes.



Few useful Keychain Wrappers:

1. **SwiftKeychainWrapper by Jason Rendel(jrendel) for Swift.** https://cocoapods.org/pods/SwiftKeychainWrapper or https://github.com/jrendel/SwiftKeychainWrapper
2. **SAMKeychain by Sam Soffes for Objective C.** https://cocoapods.org/pods/SAMKeychain
3. **Locksmith by Matthew Palmer for Swift.** (Check out the video tutorial) https://github.com/matthewpalmer/Locksmith

Introducing app security

Enable Application Transport Security:

With the launch of iOS 9 and OS X El Capitan, Apple has introduced App Transport Security, which enforces developers to use secure network connections. This change implies that every connection the application makes must use HTTPS protocol and TLS 1.2.

In other words, our application cannot communicate with a server using a non-secure connection, such as HTTP, unless it is explicitly indicated. As this was a breaking change, Apple provided an easy way to master this new requirement by adding exceptions or disabling it in the plist file.

Nevertheless, it is strongly recommended not to bypass this restriction, and instead, use secure connections in our apps to avoid potential (and easy) attacks.

SSL Pinning:

Once ATS is enabled, the second step to increase our apps security consists in **enabling SSL Pinning**.

SSL Pinning is a technique that allows us to deal with an attack called *Man in the Middle*. SSL is based on the certificate's "chain of trust". When the communication starts, the client checks if the received server's SSL certificate is trusted by any **SSL Certificate Authority.**

We can use SSL Pinning to ensure that the app communicates only with the designated server itself. This is done by saving the target server's SSL certificate inside the app bundle.

SSL Pinning has a visible disadvantage, not related to the security itself: the app must be updated whenever the server's SSL key is changed, due to expiration and other reasons.

MIT-WPU

॥ विश्वशान्तिर्ध्रुवं ध्रुवा ॥

Thank You….