itworldcanada.com

Understanding Cybersecurity on Smartphones (UCSph) Part 1 - IT World Canada

MohammadMoein Shafi and Arash Habibi Lashkari

15–19 minutes

The smartphone is one of the most remarkable inventions in contemporary human history and is currently the most widely utilized electronic device globally. Its evolution has transformed modern communication technology, allowing us to communicate efficiently and instantly across vast distances worldwide. This series delves into the historical evolution of the modern smartphone, shedding light on its significant contributions and addressing cybersecurity-related concerns associated with smartphones and their diverse applications (apps).

The previous series, entitled *Understanding Android Malware Families (UAMF)*, showcased six articles focusing on Android malware's primary categories and families, guiding readers to understand the threats' behavior and explore mitigation procedures. It presented the findings of our ongoing Android malware analysis research project initiated in 2017, which included the creation of four datasets—AAGM2017, AndMAl2017, InvestAndMAl2019, and AndMal2020. The series also encompassed related academic articles proposing solutions and

techniques for detecting and characterizing Android malware.

In this series, *Understanding Cybersecurity on Smartphones (UCSSph)*, we will conduct an in-depth analysis of various smartphone operating systems, including iPhone, Windows, Symbian, Tizen OS, Sailfish OS, Ubuntu Touch, KaiOS, Sirin OS, and Harmony OS. This five-article series aims to provide valuable insights and recommended practices for researchers, developers, and users. The series draws from the content of the recent book, *Understanding Cybersecurity on Smartphones*, published by Springer this year. The first article focuses on Apple's iOS, a global leader in mobile systems, exploring cybersecurity vulnerabilities, associated risks, malware families, attacks, and mitigation techniques.

Contents

Apple's iOS is a dominant player among mobile OSes. Despite a shrinking smartphone market in 2022, Apple increased its premium segment share from 57 per cent to 62 per cent between Q1 2021 and Q1 2022 [1].

1 iOS fundamentals

iOS, originally iPhone OS, is Apple Inc.'s Unix-based, mostly proprietary mobile OS, powering devices like iPhones and iPod Touch, and foundational for iPadOS, tvOS, and watchOS. It's primarily proprietary, with a layered design.

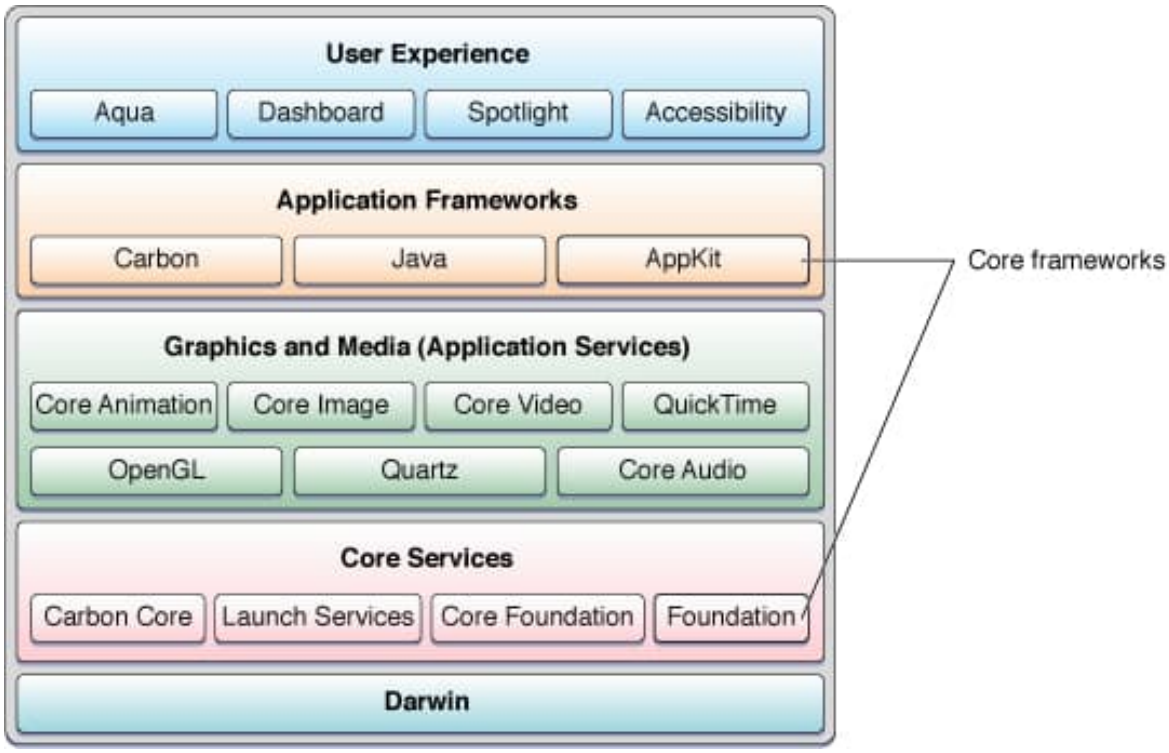

Figure 1: iOS Structure

iOS acts as an intermediary between hardware and mobile apps, using APIs for easier app development compatible with various hardware. Its Core Framework is vital, offering low-level functions. Key components include:

- **Core OS layer** – Handles kernel, file system, network, security, power management, device drivers, and libSystem library.

- **Core services layer** – Provides hardware-based services like GPS, including System Configuration and Core Location.
- **Media layer** – Supports graphics and multimedia for Cocoa Touch, including video and Core Graphics.
- **Cocoa touch layer** – Supports iOS apps, including iAd and Game Kit.
- **UIKit –** Manages user interface and behavior, like event handling.
- **Foundation layer** – Deals with object management and fundamental operations, supporting basic data types and system services.

2 Getting into cybersecurity – recognizing iOS vulnerabilities

Despite the general perception of iOS devices as more secure, they are not impervious to threats. For those entering the field of cybersecurity, a deep understanding of iOS-specific vulnerabilities is essential, especially given the popularity and extensive use of Apple products in various sectors.

This section delves into some of the more nuanced and critical vulnerabilities that are often exploited in iOS systems, illustrating the types of attacks and their potential impact:

- **Denial of service (DoS)** – Targets a system with excessive traffic to cause crashes and disrupt service.
- **Code execution** – Exploits flaws to remotely run malicious code, posing a serious threat due to its remote capabilities.
- **Overflow** – A type of code execution caused by buffer overflow,

leading to system crashes or harmful code execution.

- **Memory corruption** – Involves unintended memory alterations, often due to coding errors, enabling harmful code execution.
- **SQL injection (SQLi)** – Manipulates database queries through application gaps, often via data entry forms.
- **Cross-site scripting (XSS)** – Involves sending malicious code to users via web applications, usually as browser-side scripts.
- **Directory traversal** – Enables access to server files, risking system or application code integrity.
- **Authentication bypass** – Takes advantage of weak authentication to access the system, often using stolen credentials.
- **Information gain** – Allows attackers with initial access to acquire further authentication details for unauthorized system or database entry.
- **Privilege escalation** – Seeks unauthorized elevated access by exploiting system flaws or user errors.

Year	Types of Vulnerabilities										Total Number of Vulnerabilities
	DOS	CE	OF	MC	SQLI	XSS	DT	AB	IG	PE	
2011	✔	✔	✔	✔		✔		✔	✔	✔	83
2012	✔	✔	✔	✔		✔		✔	✔	✔	155
2013	✔	✔	✔	✔		✔	✔	✔	✔	✔	95
2014	✔	✔	✔	✔			✔	✔	✔	✔	120
2015	✔	✔	✔	✔				✔	✔	✔	386
2016	✔	✔	✔	✔		✔		✔	✔	✔	168
2017	✔	✔	✔	✔		✔		✔	✔	✔	388
2018	✔	✔	✔	✔		✔		✔	✔	✔	125

2019	✔	✔	✔	✔	✔	✔			✔	✔	356
2020	✔	✔	✔	✔		✔	✔	✔	✔	✔	305
2021	✔	✔	✔	✔		✔	✔	✔	✔	✔	365
2022	✔	✔	✔	✔				✔	✔	✔	43

DOS: Denial of Service CE: Code Execution OF: Overflow MC: Memory Corruption SQLI: SQL Injection	XSS: Cross-site Scripting DT: Directory Traversal AB: Authentication Bypass IG: Information Gain PE: Privilege Escalation

Table 1: a trend analysis of iOS vulnerabilities from 2011 to 2022.

For new professionals in cybersecurity, tackling the unique architecture and popularity of iOS devices involves a dual focus on technical proficiency and practical application. This includes a deep dive into iOS's operating system intricacies, such as its kernel structure and security protocols, and a hands-on approach to understanding Swift and Objective-C to identify and address app-specific vulnerabilities. Staying updated on the latest iOS exploits within both individual and enterprise contexts is key to effective risk mitigation.

3 Exploring adversarial tactics in iOS

Compared to Android, iOS may have fewer malware strategies targeting it, but the threats that do exist are sophisticated and evolving. This section delves into various strategies and vulnerabilities that have been exploited in iOS attacks. It covers a range of strategies and vulnerabilities exploited in iOS attacks, categorized into several key areas:

3.1 Propagation

This category is about how malicious software or attacks are

initially spread or delivered to the target device. This could be through direct actions like visiting a compromised website, indirect methods like tampering with software during its development or distribution, or exploiting specific device functionalities to gather information or introduce vulnerabilities.

- **Drive-by compromise** – Accessing a device via compromised websites to distribute harmful content. Examples include INSOMNIA and Stealth Mango.
- **Supply chain compromise** – Altering software tools or delivery methods to inject malicious code.
- **Network configuration discovery** – Gathering device network details, including IMEI and phone number, through network settings. DualToy is an example.
- **Software discovery** – Identifying installed apps and their configurations on a device by exploiting private iOS APIs.
- **Clipboard data** – Accessing iOS clipboard data to extract sensitive information (not applicable for iOS 14+).
- **Input capture** – Using fake dialogue prompts to phish user credentials or sensitive data. Examples include TianySpy and XcodeGhost.

3.2 Activation

This stage involves the activation or triggering of the malicious functionalities within a compromised device. It includes tactics that enable the malware to bypass security measures, hide its presence, or prepare the device for further malicious activities.

- **Downloading dynamic codes** – Bypassing static analysis to

execute dynamic code through external libraries. Examples include Windshift and ZergHelper.

- **Obfuscating files or information** – Hiding payloads to avoid detection.
- **Geofencing** – Exploiting location services for location-specific actions. eSurv is an example.
- **Software vulnerabilities exploitation** – Utilizing programming errors in apps, services, or the OS for code execution. Pegasus for iOS was a key exploitability in software.
- **Process injection** – Injecting code into processes to evade defenses.

3.3 Carrier

In this category, the focus is on the transmission and facilitation of cyberattacks through network or system manipulation. This can include intercepting and altering network communications, hiding malicious activities within normal network traffic, or using system-level functions to gather and transmit data.

- **Adversary-in-the-middle** – Intercepting network traffic via methods like VPN or DNS poisoning. KeyRaider is an example.
- **Application layer protocol** – Hiding malicious activities within regular application layer traffic.
- **Local system's files** – Searching local files or databases for sensitive data. Examples include Tangelo and Concipit1248.
- **Location tracking** – Tracking device location through standard iOS APIs.

3.4 Execution

Execution involves carrying out the intended malicious activities on a compromised device. This could include stealing sensitive information, damaging the device or data, or using the device to launch further attacks.

- **Scheduled task/job** – Utilizing iOS task scheduling for malicious code execution.
- **Interpreter abuse** – Executing harmful scripts or commands via interpreters.
- **Subvert trust controls** – Modifying code signing policies to allow unofficial application execution. Examples include XLoader for iOS and YiSpecter.
- **Exfiltration over alternative protocol** – Sending data over protocols different from the command-and-control channel.

3.5 Persistence

Persistence is about maintaining access or control over a compromised device over time. The focus here is on ensuring that the malware or attacker's access remains undetected and uninterrupted, even after reboots, updates, or attempts to remove the malware.

- **Compromise client software binary** – Modifying software binaries for persistent access.
- **Hijack execution flow** – Installing malicious apps via physical connections like USB. Wirelurker is an example.
- **Credentials from the password store** – Extracting keychain data

for credential access.

4 Analyzing iOS Malware Varieties & Tools

Android malware often overshadows iOS in security research, mainly due to iOS's closed-source nature. To address this gap, this section offers an examination of iOS malware types, categories, and tool sources.

Malware Family	Category
TRacer, YiSpecter, wirelurker	Spyware
iOS adware Cydia, zerghelper	Adware
Inception APT, Inception Whatsapp	Inception
iPhone click fraud	Clickfraud
KeyRaider, AppBuyer, Xsser	Stealer

Table 2: Types of iOS Malware

Historically, the majority of iOS malware has predominantly affected jailbroken devices. Jailbreaking is the process of bypassing Apple's restrictions to gain root access to the operating system.

Additionally, state-sponsored actors have occasionally targeted iOS devices through sophisticated malware campaigns. Notable examples include the Pegasus spyware, developed by the NSO Group, which has been used to target journalists and activists, and the XcodeGhost incident, where a counterfeit version of Apple's development tool led to the distribution of infected apps through the App Store.

To grasp the extent of iOS malware, refer to the iOS taxonomy in Figure 3.2, based on tool types found in the wild. Cybercriminals

have learned how to monetize iOS devices using four primary types of tools:

1. **Tools for sale to the public** – Targeting users with tools like keyloggers, spyware, and RATs. Examples include 1mole, FlexiSpy, iKeyMonitor keylogger, and StealthGenie.
2. **Research-based tools** – Developed as proofs of concept by security researchers, such as iSAM, Instastock, and NeonEggShell.
3. **Government-used tools** – Employ backdoors and spyware to surveil targets, such as activists and politicians. Examples include FinSpy mobile, Pegasus, and CIA Vault 7.
4. **Tools found in the wild** – Targeting the general public with botnets, RATs, and adware, including iKEE, WireLurker, and YiSpecter.

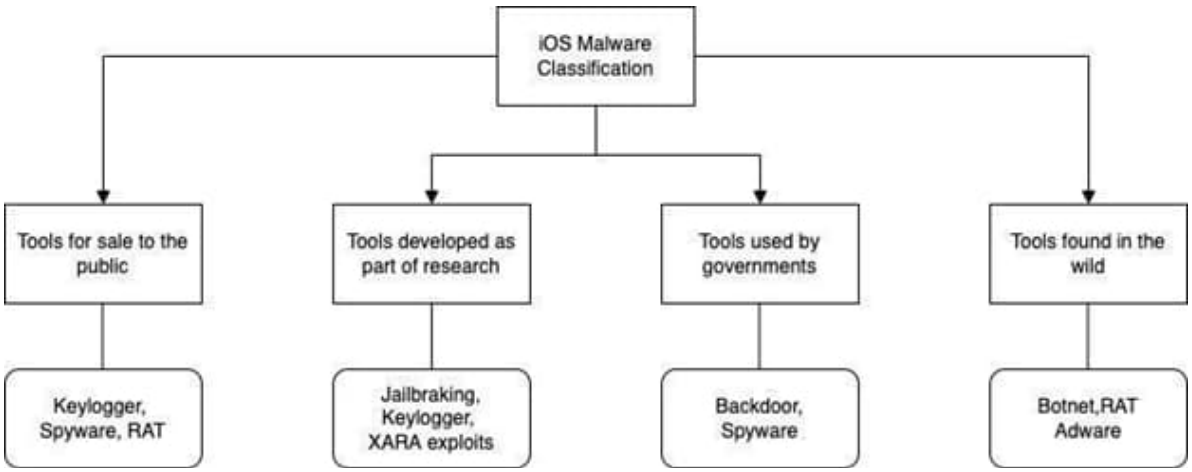

Figure 2: iOS Malware Taxonomy

5 Embracing iOS services – current trends

Apple’s creative influence is driving the latest trends in iOS app development. Key trends for 2023 were wearable technology, mobile wallets, augmented and virtual reality, voice assistants, and

enhanced app security.

- **Wearable technology** – Wearable tech, like Apple's smartwatches, reached a milestone of over 7.6 million units sold as of 2020. The market's growing appetite suggests a sustained strong demand for wearables.
- **Mobile wallets** – Apple Pay's popularity is soaring thanks to its straightforward and secure payment system. With just a tap, it uses NFC technology to make transactions quick and hassle-free. This growing preference for Apple Pay signals a broader move towards simpler, digital payment solutions in our daily lives.
- **Augmented Reality (AR) and Virtual Reality (VR)** – AR and VR are revolutionizing iOS apps by merging real and virtual environments. They're more than just technological advancements; they're transforming user experiences, providing immersive and engaging interactions that take app usage to a whole new level.
- **Voice assistants** – Siri, since its 2007 debut, has transitioned from a unique feature to a core element of the iOS experience, helping users with a variety of tasks and queries. Its integration into apps through SiriKit marks a major shift, paving the way for voice-activated features across diverse iOS applications
- **iOS app security** – The security of iOS apps stands out in an era of escalating cyber threats. Apple's commitment to robust security protocols protects user data and reinforces the trust and reliability users place in iOS applications.
- **iOS HomeKit** – Apple HomeKit, or Apple Home, is revolutionizing the smart home scene. It allows users to manage a wide array of home devices via their iOS gadgets, symbolizing the fusion of

technology with daily life.

6 What's next

This article has explored iOS vulnerabilities in-depth, tracking their development over time and highlighting the growing attention to iOS malware research.

The next article of the series, entitled *Understanding Cybersecurity on Smartphones (UCSSPh): Introduction to Windows Phone*, will delve into the history, evolution, and unique features of Microsoft's Windows Phone, from its early beginnings as Windows Mobile to its latest updates and innovations as Windows Phone.

References:

[1] Claud Xiao, WireLurker: A New Era in OS X and iOS Malware, 2014.

MohammadMoein Shafi and Arash Habibi Lashkari

*** Moein Shafi is a graduate student at York University with a keen interest in Cybersecurity, Computer Networks, IoT, Machine Learning, and Network Analysis. As a Research Assistant at the esteemed Behavior-Centric Cybersecurity Center (BCCC), he actively contributes to advancements in digital security and technology innovation. *** Dr. ARASH HABIBI LASHKARI is a Canada Research Chair and an Associate Professor in Cybersecurity at York University. As the founder and director of the Behavior-Centric Cybersecurity Center (BCCC), he has over 25

years of academic and industry experience. He has received 15 awards at international computer security competitions - including three gold awards - and was recognized as one of Canada's Top 150 Researchers for 2017. He also is the author of ten published books and more than 100 academic articles and 12 books on a variety of cybersecurity-related topics. In 2020, he was recognized with the prestigious Teaching Innovation Award for his personally created teaching methodology, the Think-Que-Cussion Method. He is the founder of the Understanding Cybersecurity Series (UCS), an ongoing research and development project culminating with a varied collection of online articles and blogs, published books, open-source packages, and datasets tailored for researchers and readers at all levels. His research focuses on cyber threat modeling and detection, malware analysis, big data security, internet traffic analysis, and cybersecurity dataset generation.