cybersguards.com

Exploring Alternative Smartphone OSs: Beyond Android and iOS for Security

Arash Habibi Lashkari

13–16 minutes

As part of the Understanding Cybersecurity Series (UCS) knowledge mobilization program, this article delves into seven common Smartphone OSs. Are you curious about niche smartphone OS options beyond Windows, Android, and iOS? This article explores seven alternative operating systems: **Symbian, Tizen OS, Sailfish OS, Ubuntu Touch, KaiOS, Sirin OS,** and **Harmony OS**. These players add spice to the mobile OS mix with unique offerings and approaches to security.

1 Learning Basics: The History

1.1 Symbian OS

In the ’90s, software company Psion built EPOC32, a 32-bit operating system programmed in C++. This eventually morphed into Symbian OS in collaboration with heavyweights like Nokia and Motorola by 1998. Renowned for its ironclad security (no kernel compromise in its history), Nokia snapped Symbian in 2008, turning open source before it was discontinued in 2014 [7, 9].

Symbian featured a sophisticated architecture with an integrated memory management unit (MMU) and cache, aiming for efficient memory and interrupt management. Its design allowed for the customization of user interfaces by different manufacturers [8].

Symbian OS is built on a real-time microkernel, supporting multithreading and preemptive multitasking with a request-and-callback service approach. It’s structured into four layers [9]:

1. **UI Framework Layer**: Provides the essentials for user interface development.
2. **Application Software Layer**: Offers hardware-independent services for applications, including messaging and multimedia.
3. **Middleware Layer**: Contains a mix of frameworks, servers, and libraries for OS services.
4. **Kernel and Hardware Interface Layer**: Hosts the kernel and abstracts hardware interfaces, handling device drivers.

1.2 Tizen OS history

Launched by Samsung Electronics in 2012, four years after

Android's debut, Tizen is a Linux-based OS focused on cross-platform compatibility and web technologies. Before Samsung combined its Bada operating system with Tizen, it was connected to several distributions and was supported by the Linux Foundation.

The Tizen architecture for smartphones consists of three layers:

1. **Application Layer**: Like native programs, Tizen Web applications take full advantage of the platform's capabilities.
2. **Core Layer**: Comprises Tizen Core Service and API, facilitating the creation of Web applications with standards like Khronos WebGL, W3C (including HTML5), and device-specific APIs.
3. **Kernel Layer**: Contains device drivers and the Linux kernel.

1.3 Sailfish OS History

Sailfish OS was developed by Sailfish Alliance, Mer, Jolla, and Sailfish community contributors in 2013. The OS is based on Nokia's MeeGo projects with a gesture-based "Sailfish UI" for easy navigation on smartphones, tablets, and wearables. Sailfish OS architecture includes:

1. **Mer Core**: A Linux-based core layer that provides the operating system's foundation.
2. **Middleware**: Incorporates the Qt framework to enable smooth app functioning and a unified interface experience.
3. **Silica**: Offers developers a rich set of UI elements and controls for crafting intuitive interfaces.
4. **Alien Dalvik**: A proprietary technology developed by Myriad Group enables compatibility with Android apps.
5. **Jolla Store**: The central hub for discovering and downloading apps.

1.4 Ubuntu Touch OS History

Ubuntu Touch, also known as Ubuntu Phone, was developed by Canonical Ltd. to offer a unified Ubuntu experience across devices. Launched in 2013, it aimed to bridge smartphones, tablets, and desktops but was discontinued for smartphones in 2017. Ubuntu Touch architecture includes:

1. **Ubuntu Base**: The core OS and essential system components.
2. **Mir Display Server**: Manages graphical output and input operations.
3. **Unity 8**: A user interface that provides a consistent and intuitive experience across different devices.
4. **Libertine**: Enables running traditional Linux desktop apps on Ubuntu Touch.
5. **Ubuntu Store**: The official app store for Ubuntu Touch.

1.5 KaiOS History

KaiOS emerged from Mozilla's open-source B2G (Boot to Gecko) initiative in 2017. The goal was to leverage web technologies to bring smart functionalities, such as apps, internet browsing, and social media, to keypad feature phones. Various elements contribute to the KiOS architecture, including:

1. **App profile:** Defines application specifications for KaiOS, covering features, permissions, and compatibility.
2. **Web API:** Provides a set of APIs that enable web-based applications in KaiOS to access device features, such as camera, contacts, and messaging.
3. **Core:** Manages system resources, handles system-level services, and coordinates communication between different components of the OS.
4. **HAL (Hardware Abstraction Layer)**: Interfaces between the OS and device hardware, standardizing interactions with sensors, displays, and input devices.

1.6 Sirin OS history

Sirin OS, developed by Sirin Labs in 2018, prioritizes security and privacy and is the only mobile OS that is secure enough to use and store cryptocurrency. As the first to develop a blockchain smartphone, Sirin Labs integrates blockchain across their devices with SRN tokens to promote digital currency and decentralization.

Although specific information about the architecture of Sirin OS is not readily available, components commonly found in mobile operating systems are likely present, including the kernel, secure enclave, security framework, behavioral-based IPS, and application layer.

1.7 HarmonyOS History

HarmonyOS was announced by Huawei in 2019 as an innovative, distributed OS for the Internet of Everything (IoE) era. HarmonyOS's architecture is listed below:

1. **Application Layer:** The top layer of HarmonyOS where user-facing applications are executed.
2. **Framework Layer:** A comprehensive set of libraries, APIs, and tools that facilitate the development of applications on HarmonyOS.
3. **System Service Layer:** Manages core system operations like device management and resource allocation.
4. **Kernel:** Manages hardware resources, provides low-level system services, and facilitates communication between software and hardware components.

2. Getting into Cybersecurity: The Vulnerabilities and Risks

For users evaluating different mobile operating systems, it's

essential to look at how an OS handles past vulnerabilities and patches to get a sense of its security posture.

Security features themselves, ranging from encryption to secure boot, offer a lens into how seriously an OS takes protection. App security is another battleground, where the rigors of app vetting and developer oversight differ markedly across platforms. Lastly, how quickly and effectively an OS team responds to security flaws speaks volumes about their commitment to user safety.

Table 1: Mapping between architecture components and their security vulnerabilities

OS (year)	**Architecture components**	**Security Vulnerabilities**
Symbian (1998)	Four layers: User Interface framework, Application Software, Os service, and Kernel Service layers	Prone to instability with corrupted or non-standard files, easily overwritten system applications, lacks user roles and access controls in file systems [14][15][16].
Tizen (2012)	Three components: Application, Core, and Kernel	Limited third-party app support, risk of security vulnerabilities due to inadequate isolation in the web runtime, enabling apps to interact improperly or access sensitive data.
Sailfish (2013)	Five components: Mer core, middleware, silica, Alien Dalvik, and Jolla Store	Constrained app ecosystem, risks of security vulnerabilities in third-party apps due to irregular updates or patches.
Ubuntu Touch (2013)	Five components: Ubuntu Base, Mir Display Server, Unity 8, Libertine, and Ubuntu Store	Restricted app selection, security risks in Libertine include privilege escalation and code injection.
Kai (2017)	Four components: App profile, Web API, Core, and Hardware Abstraction Layer (HAL)	Limited functionality compared to full-fledged OSs. Vulnerabilities in HAL may involve buffer overflow and input validation problems.
Sirin (2018)	Five likely components: Kernel, Secure Enclave, Security Framework, Behavioral-based	Security risks in Android subsystems, dependence on the Android app ecosystem.

	IPS, and Application layer	
Harmony (2019)	Four components: Application, Framework, System Service, and Kernel	Potential vulnerabilities in the app ecosystem, lack of third-party app support

3. Dissecting Malware: The Type of Malware

This section contrasts previous chapters by focusing on the challenges of malware analysis for less widely used OSs. The limitations discussed here, do not suggest these OS are inherently insecure but highlight the challenges in analyzing malware for niche platforms.

For instance, **Symbian OS** has been notably vulnerable to various mobile malware like the Cabir worm and Skuller virus due to its design [15]. **Tizen OS**'s smaller market share translates into fewer dedicated malware research efforts. **Sailfish** and **Ubuntu Touch**, with their niche markets, also experience similar limitations, leading to potential delays in threat detection.

KaiOS, despite its growing popularity among feature phones, and **Sirin OS**, with its focus on privacy and security, both encounter hurdles in thorough malware analysis due to their unique architectures and limited research focus.

HarmonyOS, being relatively new, is in the early stages of developing specific malware analysis tools, facing a scarcity of data for comprehensive threat analysis.

4. Mitigating Attacks: The Current Solutions

Each OS incorporates unique security features to protect user data and devices against vulnerabilities. By analyzing how these systems map their security measures against potential weaknesses, we gain insights into the efficacy of each OS's security strategies.

- **Symbian**: Faces stability issues with corrupted files and lacks user roles and access controls. It counters these with kernel separation and restrictions on device drivers and ROM-based plug-ins.
- **Tizen**: Struggles with limited third-party app support and web runtime vulnerabilities. Security measures include secure boot, SELinux, and app sandboxing.
- **Sailfish**: Has a limited app ecosystem and potential vulnerabilities in third-party apps. Offers encrypted data storage, device encryption, and user permission controls.
- **Ubuntu Touch**: Sees limited app availability and risks in Libertine related to privilege escalation and code injection. Protects with app confinement, secure boot, and system updates.
- **KaiOS**: Limited functionality compared to comprehensive operating

systems but enhances security through app verification, updates, and data encryption.

- **Sirin**: Risks include vulnerabilities in Android subsystems and dependency on the Android app ecosystem. Features secure elements for crypto transactions and hardware security.
- **Harmony (Hongmeng)**: Faces app ecosystem vulnerabilities and limited third-party app support, with defenses like a Trusted Execution Environment, sandboxing, and device virtualization.

5. Utilizing Services: The Trend Now

As modern mobile OSs like Android and iOS have dominated the market, other mobile operating systems discussed in this chapter have been phased out or repurposed. Here are some notable applications of these OSs:

- **Tizen OS** powers all Samsung Smart TVs, offering an open-source, Linux-based platform for developers to create apps for smart TVs and connected devices.
- **Ubuntu Touch OS**, although discontinued by its original developers, has been kept alive by the UBPorts community as Meizu Pro 5. It offers a unique mobile experience and supports a wider range of phones [10].
- **KaiOS** targets extremely low-cost “smart feature phones” in emerging markets such as India, Nigeria, and Indonesia, providing internet access to the next billion users without the complexities and distractions of standard smartphones [11].
- **Sirin OS** is still available for cryptocurrency users. It features a built-in cold storage wallet and advanced security protocols, making it a go-to for those prioritizing cryptocurrency safety on mobile devices [12].
- For users, **HarmonyOS** forms a super device that connects various smart devices, enables them to share data with each other, and facilitates cross-device collaboration [13].

This article’s overview of alternative mobile OSs highlights the dynamic nature of mobile computing. Each OS introduces unique features, catering to different user needs and devices. Their future impact on mobile computing as the industry progresses remains exciting.

References:

[1] https://developer.tizen.org/zh-hans/tizen-architecture

[2] https://sailfishos.org/content/uploads/2018/11/Sailfish_Architecture.jpg

[3] https://fr.m.wikipedia.org/wiki/Fichier:Architecture_Ubuntu_Touch.png

[4] https://phone.docs.ubuntu.com/en/devices/porting-new-device

[5] https://developer.kaiostech.com/docs/

[6] https://www.huaweicentral.com/huawei-publishes-january-2023-harmonyos-security-patch-details/

[7] D. Muthukumaran, S. Anuj, J. Schiffman, B. M. Jung, and T. Jaeger, “Measuring Integrity on Mobile Phone Systems,” in the 13th ACM symposium on Access control models and technologies, Estes Park, 2008.

[8] All About Symbian, http://www.allaboutsymbian.com/

[9] R. Kumar, “Symbian Os,” 2023. [Online]. Available: https://www.codingninjas.com/codestudio/library/symbian-os#:~:text=in%20the%20article.-,Symbian%20OS%20architecture,kernel%20also%20performs%20pre [Accessed 2023].

[10] C. Cawley, “How to Install Ubuntu Touch on Your Mobile Phone,” Make use of, 2022. [Online]. Available: https://www.makeuseof.com/how-to-install-ubuntu-touch-on-your-mobile-phone/. [Accessed 2023].