October 2021

NETWORK SANDBOX FOR CLOSED-SOURCE COMPONENTS WITH ACCESS TO SENSITIVE DATA

Eugenio Marchiori

Sergey Volnov

Thomas Hume Weedon

Wei Huang

Asela Jeevaka Ranaweera Gunawardana

NETWORK SANDBOX FOR CLOSED-SOURCE COMPONENTS WITH ACCESS TO SENSITIVE DATA

ABSTRACT

A computing device (e.g., a smartphone, a laptop computer, a tablet computer, a smartwatch, etc.) may include a system application for managing both the ability of software (e.g., an application, a program, a widget, etc.) to access a network and the type of information that can be transmitted to a computing system via the network. Rather than use a permission-based model (e.g., a model in which a user manually permits an application to access the network), which may grant the application unconstrained network access, the system application may use a dataflow model (e.g., a model in which a framework defines a policy for how an application may access the network) that results in more granular network access. In some examples, the system application may comprise a first component (e.g., an application package (APK)) that delegates all requests for network access to a second component (e.g., an application programming interface (API)) to ensure policy enforcement (e.g., limiting data exfiltration from the first component). Source code for the second component may be made available for inspection or review by anyone (e.g., open sourced) to provide a means for auditing the operation of the system application. In addition, the system application may provide a ledger to enable a user of the computing device to monitor dataflows and network usage. In this way, the system application may increase trust in applications executing at the computing device (e.g., by enabling researchers to ensure that no party is receiving preferential treatment with regards to data retention policies) and may increase transparency in how applications are using and sharing data (e.g., by allowing interested parties to verify network usage).

DESCRIPTION

FIG. 1 below is a conceptual diagram illustrating a computing device 100 that uses a system application 102 for managing both the ability of software (e.g., an application, a program, a widget, etc.) to access a network and the type of information that can be transmitted to a computing system via the network. Computing device 100 may be any mobile or non-mobile computing device, such as a cellular phone, a smartphone, a desktop computer, a laptop computer, a tablet computer, a portable gaming device, a portable media player, an e-book reader, a watch (including a so-called smartwatch), a gaming controller, and/or the like. As shown in FIG. 1, computing device 100 may include a presence-sensitive display 104, one or more processors 106, and one or more storage devices 108. Storage devices 108 may include system application 102 and one or more application(s) 116.

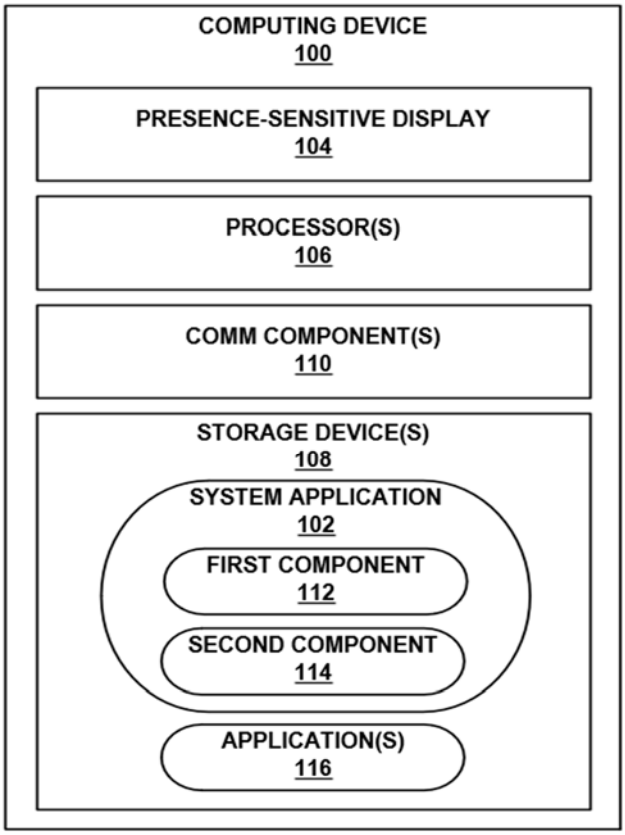

FIG. 1

Presence-sensitive display 104 of computing device 100 may be a presence-sensitive display that functions as an input device and as an output device. For example, presence-sensitive display 104 may function as an input device using a presence-sensitive input component, such as a resistive touchscreen, a surface acoustic wave touchscreen, a capacitive touchscreen, a projective capacitance touchscreen, a pressure sensitive screen, an acoustic pulse recognition touchscreen, or another presence-sensitive display technology. Additionally, presence-sensitive display 104 may function as an output (e.g., display) device using any of one or more display components, such as a liquid crystal display (LCD), dot matrix display, light emitting diode (LED) display, microLED display, organic light-emitting diode (OLED) display, e-ink, active-matrix organic light-emitting diode (AMOLED) display, or similar monochrome or color display capable of outputting visible information to a user of computing device 100.

Processors 106 may implement functionality and/or execute instructions associated with computing device 100. Examples of processors 106 may include one or more of an application specific integrated circuit (ASIC), a field programmable gate array (FPGA), an application processor, a display controller, an auxiliary processor, a central processing unit (CPU), a graphics processing unit (GPU), one or more sensor hubs, and any other hardware configure to function as a processor, a processing unit, or a processing device. System application 102 may be operable by processors 106 to perform various actions, operations, or functions of computing device 100.

Storage devices 108 may include one or more computer-readable storage media. For example, storage devices 108 may be configured for long-term, as well as short-term storage of information, such as instructions, data, or other information used by computing device 100. In some examples, storage devices 108 may include non-volatile storage elements. Examples of

such non-volatile storage elements include magnetic hard discs, optical discs, solid state discs, and/or the like. In other examples, in place of, or in addition to the non-volatile storage elements, storage devices 108 may include one or more so-called "temporary" memory devices, meaning that a primary purpose of these devices may not be long-term data storage. For example, the devices may comprise volatile memory devices, meaning that the devices may not maintain stored contents when the devices are not receiving power. Examples of volatile memory devices include random-access memories (RAM), dynamic random-access memories (DRAM), static random-access memories (SRAM), etc.

Computing device 100 may include communication components 110 ("COMM components 110"). COMM components 110 may receive and transmit various types of information over a network. The network may include a wide-area network such as the Internet, a local-area network (LAN), a personal area network (PAN) (e.g., Bluetooth®), an enterprise network, a wireless network, a cellular network, a telephony network, a Metropolitan area network (e.g., WIFI, WAN, WiMAX, etc.), one or more other types of networks, or a combination of two or more different types of networks (e.g., a combination of a cellular network and the Internet).

In general, computing device 100 may include system application 102 that, to some extent, controls network access. For example, a user of computing device 100 may open application 116, which in turn may request access to a network. In examples where system application 102 uses a permission-based model, system application 102 may, in response to user input via presence-sensitive display 104, grant application 116 access to the network. However, in some cases, the access granted by system application 102 may be unconstrained or general such that application 116 may access the network for any purpose, including transmitting user

data to a computing system, such as a remote server. This possibility may be problematic because an original equipment manufacturer (OEM) of computing device 100, a third party, etc., may wish to make certain promises regarding network usage and data retention. However, verifying whether those promises are being upheld may be difficult because data transmission to and from computing device 100 via the network is generally encrypted. Thus, a transparent framework that is more granular in its approach to managing network access may be desirable.

In accordance with techniques described in this disclosure, system application 102 may use a dataflow model where system application 102 grants application 116 access to a network for particular dataflows and denies application 116 access in all other cases. In some examples, system application 102 may comprise a first component 112 (e.g., an application package (APK)) that delegates all requests for network access to a second component 114 (e.g., an application programming interface (API)) to ensure policy enforcement (e.g., limiting data exfiltration from the closed-source component).

In some examples, first component 112 of system application 102 may be proprietary software (e.g., closed source code) configured to execute routine processes (e.g., scheduling logic) unrelated to transparency and trust concerns surrounding network usage. For example, first component 112 may be an APK that downloads files from a computing system via a network. To do so, first component 112 may delegate all requests for network access to second component 114. Second component 114 may treat first component 112 in the same way it treats application 116 to ensure uniform policy enforcement (e.g., limiting data exfiltration from first component 112). That is, second component 114 may only permit first component 112 network access for particular dataflows and deny network access in all other cases. In this way, system application 102 may be configured such that first component 112 is not treated as an exception to data

retention policies, eliminating any perception that a party (e.g., the OEM of computing device 100) is receiving preferential treatment.

In contrast to first component 112, second component 114 may be open sourced such that anyone may review the source code for second component 114 to verify that second component 114 is behaving in the manner described by, for example, the OEM of computing device 100 (e.g., in accordance with promises regarding network usage and/or data retention policies). In some examples, to avoid requiring a user of computing device 100 to manually define the permissions for each dataflow of application 116, second component 114 may be one or more APIs representing a framework that enforces policies for how data may flow via the network. The policies may be defined in the open source code of second component 114 in a human-readable and machine-readable format such that an interested party may verify that second component 114 is operating in accordance with the specification of second component 114.

Second component 114 may enforce one or more policies allowing application 116 to access a network. For example, second component 114 may permit application 116 to download a machine learning model from a remote server but deny network access for the sending of any unique parameters (e.g., the name of the user of computing device 100, the model of computing device 100, etc.) to a remote server. As another example, second component 114 may specify that user data (e.g., image data, sound data, text data, etc.) may be sent by application 116 across a network in certain cases (e.g., when the data has been anonymized and aggregated to maintain privacy) but not in other cases (e.g., when the number of users is insufficient to maintain privacy). Thus, second component 114 may facilitate, for example, federated learning that complies with network usage promises and data retention policies and, following training of the

machine learning model, send the train machine learning model to first component 112 to install and download at computing device 100.

System application 102 may enable a user of computing device 100 to selectively permit network access to send and/or receive data on a per application and per dataflow basis. For example, if application 116 (e.g., an e-commerce application, a navigation application, a music streaming application, etc.) lacks permission to send user data to a remote computing system via a network, second component 114 may output a notification to the user requesting the required permission. Depending on the user input, application 116 may be granted or denied network access. Second component 114 may store user inputs as preferences regarding network access and enforce the preferences with respect to future requests for permission.

In some examples, second component 114 may provide a ledger that logs network usage (e.g., within the last 14 days). For example, the ledger may display each instance application 116 requested network access, the reason application 116 requested network access, the time and date when application 116 requested network access, and so on. As such, the ledger may provide a user of computing device 100 the ability to audit how system application 102 is managing network access.

In some examples, system application 102 may monitor and control more than just data sent and received over a network. For example, system application may monitor and control actions involving any API. Second component 114 may mediate, on behalf of computing device 100, calls to an API as well as responses to an API call. As one example, a camera application may request access to a camera API that controls a camera in computing device 100. Second component 114 may intercept the call to the camera API and determine whether to authorize the camera API call based on previously granted permissions (e.g., a location-based permission, a

time-based permission, a one-time permission, an audio-based permission, a cloud-storage permission, a contacts permission, a handover permission, etc.). If none of the previously granted permissions apply to the current component (e.g., the camera application and/or the camera API) of computing device 100, second component 114 may deny network access and output a notification to the user of computing device 100 requesting the required permission. On the other hand, if one or more of the previously granted permissions apply to the current component, second component 114 may authorize the API call. In this way, second component 114 may more granularly restrict the ability of first component 112, applications 116, etc., to access a network.

One or more advantages of the techniques described in this disclosure include enabling more granular network access by configuring an APK of a computing device to provide permissions for particular dataflows for software executing at the computing device. That is, by separating a system application into a first component with a relatively low-level of privilege and a second component with a relatively high-level of privilege (in this way, creating a network sandbox), the second component may act as a broker that mediates, in accordance with an API framework and on a per dataflow basis, access to the network on behalf of the first component and/or applications executing at the computing device. Thus, the system application may intelligently permit and deny the sharing of different types of information.

In addition, the techniques described in this disclosure provide a way for a user of the computing device, a researcher, etc., to verify that the system application is using data in a manner consistent with network usage promises and/or data retention policies. For example, the source code for the second component may be made available for inspection or review by anyone (e.g., open sourced) to provide a means for auditing the operation of the system

application. As a result, the techniques may increase trust in applications executing at the computing device (e.g., by enabling researchers to ensure that no party is receiving preferential treatment with regards to data retention policies) and transparency in how applications are using and sharing data (e.g., by allowing interested parties to verify network usage).

It is noted that the techniques of this disclosure may be combined with any other suitable technique or combination of techniques. As one example, the techniques of this disclosure may be combined with the techniques described in U.S. Patent Application Publication No. 2012/0017213A1. In another example, the techniques of this disclosure may be combined with the techniques described in U.S. Patent Application Publication No. 2020/0228561A1. In yet another example, the techniques of this disclosure may be combined with the techniques described in Küçük et al., "Managing confidentiality leaks through private algorithms on Software Guard eXtensions (SGX) enclaves: Minimised TCB on secret-code execution with Early Private Mode (EPM)," EURASIP Journal on Information Security, 2019, September 5, 2019. In yet another example, the techniques of this disclosure may be combined with the techniques described in Xing et al., "Cracking App Isolation on Apple: Unauthorized Cross-App Resource Access on MAC OS~X and iOS," In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS '15), 2015.