2014 IEEE 6th International Conference on Cloud Computing Technology and Science

STOVEPipe: Observable Access Control of User Data for Untrusted Applications on Mobile Devices

Jiaqi Tan, Utsav Drolia, Rolando Martins, Rajeev Gandhi, Priya Narasimhan
Dept. of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, USA
Email: {tanjiaqi, utsav, rolandomartins}@cmu.edu, rgandhi@ece.cmu.edu, priya@cs.cmu.edu

***Abstract*—The rapid growth in mobile devices will give rise to the trend of the leasing out of compute and data resources on mobile devices to third-parties for applications to be run on multiple mobile devices. However, these third-party applications running on leased mobile devices are typically written by unknown entities, and cannot be trusted by mobile device owners. Current mobile device platforms (e.g. Android) have permissions and access control systems designed for mobile apps that are written by reputable developers and vetted by authoritative app stores, and they are not suitable for untrusted apps. We propose STOVEPipe, an observable access control system for user data on mobile devices for untrusted third-party applications. STOVEPipe ensures that untrusted code is isolated and cannot directly access system data, and performs all data accesses on behalf of untrusted apps. This enables STOVEPipe to observe all data accessed by untrusted apps, implement content-based access control, perform accounting and auditing on accessed data easily, and perform privacy-preserving data transformations.**

I. Introduction

The number of personal mobile devices in the world has grown rapidly in recent years. In 2014 alone, more than 2 billion smartphones and tablets were sold [1]. Just as the rapid growth in enterprise-level compute resources such as servers gave rise to the leasing out of enterprise compute resources to third-parties in cloud computing, this rapid growth in personal mobile devices will lead to the leasing out of the compute and data resources on personal mobile devices to third-parties. Systems have been proposed ([2], [3], [4], [5]) which enable third-parties to run programs which make use of both the compute resources and data on the personal mobile devices of individuals. However, users who lease out their mobile devices to third-parties have to run applications from unknown entities on their devices, and users may not be able to trust these applications. While mobile device platforms such as Android have permissions systems to provide access control to private user data and platform functionalities such as access to hardware sensors and their data, these permissions systems were designed for mobile apps which have been vetted by authoritative app stores such as the Google Play Store, and determined to be safe for users to run. Thus, the permissions systems of mobile platforms such as Android were designed for the "reputable developer" trust model, for apps from known developers, which have additionally been vetted by a technically competent authority to be safe. Thus, these mobile platform permissions systems are inappropriate and insufficient for providing access control for untrusted applications from unknown sources. A different access control model is required for users to protect the private data on their mobile devices from the untrusted apps which they would run when leasing the resources on their mobile devices to third-parties.

In this paper, we propose STOVEPipe, an access control system which provides strict and observable access control for user data on mobile devices. STOVEPipe is designed to provide access control for untrusted apps which need to access user data, and is intended to address shortcomings in current mobile platform permissions systems when used for untrusted apps. In addition, STOVEPipe can provide additional access control features which are useful for common mobile resource leasing scenarios, such as participatory sensing [6] and crowd-sourced mobile applications [2]. The key feature of the STOVEPipe design is that untrusted apps are not allowed to directly access any user data or sensor data on mobile devices. Instead, STOVEPipe provides a runtime system which retrieves data on behalf of the untrusted apps, and supplies it to the untrusted app as input. This design has two advantages: (i) it provides STOVEPipe with a single point at which to enforce access control policies, and (ii) because STOVEPipe retrieves all data on behalf of untrusted apps, it is possible to observe all data that is made available to the untrusted app, making it possible for users to audit, inspect, or even transform the data being accessed by untrusted apps. A key building block required for STOVEPipe's strict and observable access control, is the ability to prevent untrusted apps from directly accessing data on the mobile device. In concurrent work [7], we describe the STOVE Data and Execution Models for untrusted applications, in which the STOVE Execution Model includes a static verifier [8] which proves that untrusted apps built using the STOVE model are isolated and are unable to directly access system data. The STOVEPipe access control system is an implementation of the STOVE Data Model [7], and we elaborate on the design rationale and features of STOVEPipe in this paper.

II. Approach and Design

The main goal of STOVEPipe is to provide strict and observable access control of user data on mobile devices for untrusted applications. STOVEPipe is not intended to replace existing permissions systems on mobile platforms, but instead enables specifying additional fine-grained access control policies, and auditing data accesses of untrusted apps.

A. Goals, Non-goals, Threat Model

We aim to meet the following goals in our design and implementation of STOVEPipe: (1) not require any changes to mobile platforms (e.g. Android) or operating systems, (2) provide access control for mobile data access by untrusted apps, (3) enable users to make fine-grained access control decisions, (4) provide users with high confidence that untrusted

978-1-4799-4093-6/14 $31.00 © 2014 IEEE
DOI 10.1109/CloudCom.2014.118

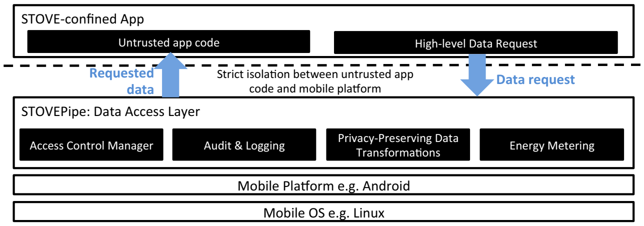
STOVE-confined App
Untrusted app code
High-level Data Request
Requested data
Strict isolation between untrusted app code and mobile platform
Data request
STOVEPipe: Data Access Layer
Access Control Manager
Audit & Logging
Privacy-Preserving Data Transformations
Energy Metering
Mobile Platform e.g. Android
Mobile OS e.g. Linux

Fig. 1. Overall architecture of STOVEPipe showing how data is provided to untrusted apps isolated by the STOVE [7] model.

apps are not making unauthorized accesses to their personal data, and (5) allow users to keep track of the privacy impact of data accesses. STOVEPipe addresses only data access by untrusted apps, and it is not the goal of STOVEPipe to provide general access control for all user apps on mobile devices. Instead, STOVEPipe requires untrusted apps to be written using the STOVE Data and Execution model [7].

In STOVEPipe, our threat model consists of arbitrarily malicious attackers who may try to steal user data or harm the user's mobile device. However, because untrusted apps are provably prevented from directly accessing data from the mobile device or interacting with any other running process (based on the execution isolation provided by the STOVE model [7]), and untrusted apps need to specify the data they wish to access, untrusted apps are unable to directly harm the system through data access. The main threat to the privacy of the user's data on the mobile device is in unauthorized data access, and because STOVEPipe collects all data on behalf of the untrusted app and allows users to observe all data accessed by the untrusted app, STOVEPipe's access control mechanism alone is sufficient to mitigate the threat of arbitrarily malicious attackers against unauthorized data access. Nonetheless, untrusted apps can still mount denial-of-service attacks on the mobile device indirectly by requesting data at high frequencies, resulting in high battery consumption. STOVEPipe mitigates this by allowing users to specify an energy budget for each untrusted app, and monitoring the energy consumed when retrieving the data requested by an untrusted app.

B. *STOVEPipe Design*

Figure 1 shows the overall architecture of STOVEPipe. STOVEPipe consists of: (i) a runtime which accepts data requests from untrusted apps and retrieves this data from the mobile device, (ii) an access control monitor which checks if the requested data is allowed and logs data transferred, and (iii) a policy manager for users to view and edit access control policies, and view data access logs. STOVEPipe provides untrusted apps with access to the following data on mobile devices: (i) stored photos and videos, (ii) the mobile user's contacts (i.e. the address book), (iii) the mobile user's calendar and appointments. In addition, STOVEPipe provides untrusted apps with access to data from sensors (if present on the device) such as: (i) WiFi and cellular state, (ii) GPS and location data, (iii) mobile device battery and power status, (iv) accelerometer and gyroscope, (v) camera(s), (vi) microphone.

In access control terminology, STOVEPipe provides Discretionary Access Control (DAC) for data items to untrusted apps. The access matrix model [9] describes authorizations in a matrix, which specifies *objects*, which are items to be protected, and *subjects*, which execute *actions* and make requests on the objects. Then, each subject is assigned a row in the matrix, and each object a column. Each cell in the matrix specifies the actions that a subject is allowed to make on an object. Hence, the above data items make up the objects which STOVEPipe provides access control for. Each untrusted app is a subject which desires access to protected data items. In access control terminology, each subject is allowed only the "read" action for each object, i.e. untrusted apps are only allowed to read (if the user allows it) a data item, and untrusted apps are not allowed to modify any data items. Also, in STOVEPipe, only mobile device owners, i.e. the users of the system, are allowed to grant or revoke permissions of each subject for each object, i.e. subjects are not allowed to modify the access control matrix [9].

High-Level Data Request: STOVEPipe allows untrusted apps to describe the mobile device data they wish to access using a high-level language, such as XML. Untrusted apps will supply a Data Manifest, which is a declaration of the data items they wish to access. For each category of data item, apps will be able to specify additional relevant attributes of the desired data. For instance, for stored user data (photos/videos, contacts, calendar), untrusted apps can specify whether they wish to access a subset of the data as defined by certain criteria (e.g. photos/videos taken within a certain time period), or if they wish to access all available data. For sensor data, untrusted apps can specify attributes such as the number and frequency of sensor samples, and times at which to collect sensor samples. Hence, STOVEPipe's data request facilities enable isolated untrusted apps to specify the source and type of data they wish to access although these apps do not have programmatic access to the mobile platform API.

Access Control Policies: STOVEPipe allows users to specify per-app and global access control policies which specify (i) individual per-app permissions for whether each untrusted app is able to access each of the possible mobile device data items, and (ii) global conditions under which each data item is allowed (or not allowed) to be accessed by all untrusted apps. STOVEPipe will allow simple access control decisions (e.g. deny access to all photos, contacts, and calendar items) as well as more complex decisions based on fine-grained attributes. STOVEPipe will also allow access control policies to be specified based on: (i) context, such as time of day and location, (ii) properties of individual data items and sensors, and (iii) aggregate data transferred to untrusted apps.

STOVEPipe allows users to specify environmental contextual conditions under which to allow or deny access to certain data or sensors by untrusted apps. For instance, users can specify certain locations at which to block access to cameras by untrusted apps when the user is at his home or workplace. STOVEPipe also allows users to specify access control decisions based on the specific data items themselves by using content-based techniques. For instance, for photos, STOVEPipe can make use of content-based access control systems such as CHIPS [10] to allow users to specify that only photos not containing certain specified faces be made available to untrusted apps. For sensor data, STOVEPipe allows users to specify the maximum allowed frequency of sampling and the maximum number of samples that can be collected in each execution. Finally, because STOVEPipe performs all data accesses on behalf of untrusted apps, STOVEPipe can log the total amount of each type of data that has already been transferred to an untrusted app. This will allow users to specify the maximum aggregate amount of data from a particular source an untrusted app is allowed to receive.

Logic-based Access Control: STOVEPipe allows users to specify both per-app and global access control policies, and it allows users to specify access control policies based on the context of the mobile device, and the content of individual data items. Hence, the final access decision of whether to grant an untrusted app access to a single piece of data item can be complex as it must take into account all the above policies. To ensure that access control decisions are correct, we plan to model the various aspects of STOVEPipe's access control policies using logic. We can model rudimentary access control decisions by using an `may-access` boolean relation [11], augmented with extra arities for contextual and environmental parameters to model context-aware access control. We can also use more complex logics such as the Dependency Core Calculus (DCC) [12] to model more complex scenarios such as delegation between principals. Finally, we can store proofs of access control decisions to help users understand which policies contributed to each decision [13].

Accounting and Auditing: By modeling access control decisions in logic, each access control decision can be justified by a proof of the access decision. This will provide users with confidence that the access control decisions are correct, and serves as an auditing tool. Users will be able to inspect how a particular access control decision is arrived at, based on all the access control policies specified. Also, as STOVEPipe performs all data accesses on behalf of untrusted apps, STOVEPipe can log all the data accesses made by each untrusted app for later inspection.

Privacy-preserving Data Transformations: As STOVEPipe collects all data on behalf of untrusted apps, it is easy for STOVEPipe to apply privacy-preserving data transformations on the data before passing the data to the untrusted app. For instance, STOVEPipe can add noise to data from sensors such as GPS, and for photos and videos, STOVEPipe can attempt to automatically detect faces and pixelate them.

Energy Metering: To mitigate denial-of-service attacks from untrusted apps, STOVEPipe prevents untrusted apps from consuming too much energy when the STOVEPipe runtime is retrieving data on behalf of a given untrusted app. STOVEPipe allows users to specify energy budgets for each untrusted app within a given duration (e.g. no more than 5% of battery power in 24 hours), and STOVEPipe can measure the energy consumed when collecting data for a given untrusted app. Then, when the untrusted app's energy budget has been fully consumed, STOVEPipe will stop collecting data for the app.

III. Discussion: Design Rationale

A. Why Observable Access Control

No need to intercept or modify data access mechanisms: Many current techniques for fine-grained and context-aware access control for mobile device data propose changes to mobile platforms such as Android [14], [15], [16], [17]. Some mobile platforms such as Android might be well-documented and well-structured, lending themselves to modification for implementing different types of access control systems. However, to fully implement a new access control mechanism for mobile device data, it is necessary to intercept and modify every possible way in which the particular data item can be accessed, which can be challenging given the size and complexity of mobile platforms. On the other hand, by using the STOVE model [7] to restrict untrusted apps to be fully isolated from the mobile device, STOVEPipe does not need to be concerned with fully mediating every data access mechanism to provide access control, and we can focus on the STOVEPipe access control mechanisms without worrying about completely mediating all data access points in mobile platforms.

Provides high assurance for untrusted apps: As mobile users cannot trust unknown apps, users need a high degree of confidence that these untrusted apps in a mobile resource leasing scenario will not make unauthorized access to the user's personal data on their mobile device. By using the STOVE model, untrusted apps are provably isolated from the mobile device. This gives mobile users the confidence that STOVEPipe is the single point through which any data access by the untrusted app can happen, providing users with high assurance. In addition, because STOVEPipe's access control mechanism is implemented in a single location, rather than spread out across different parts of the mobile platform (e.g. when the mobile platform is retrofitted to implement new access control measures), STOVEPipe's design is much simpler, and can give users the high degree of confidence they need that STOVEPipe's access control mechanism is correct. Also, as STOVEPipe's access control mechanisms are independent of the mobile platform, it is much simpler to use model-checking and other program verification techniques to check the correctness of STOVEPipe's access control mechanisms, whereas checking the correctness of an access control mechanism embedded in a mobile platform will require considering the behavior of the mobile platform, which can be complex.

Enabling privacy-preserving data transformations: As STOVEPipe retrieves all data on behalf of untrusted apps, STOVEPipe effectively materializes all of a user's mobile device data being accessed by untrusted apps before passing it to the untrusted app. This provides a convenient location to implement content-specific access control checks, and privacy-preserving data transformations. In contrast, an access control mechanism which is directly embedded in the mobile platform, such as [14], which interposes on permissions checks in the mobile platform, is unable to actually observe the data being transferred to the app which is accessing the data. Hence such access control mechanisms will not be able to implement content-specific access control checks or transform the data.

B. Challenges with Observable Access Control

Complex manipulation of sensor hardware: STOVEPipe performs all data accesses on behalf of untrusted apps, based on the Data Manifest supplied by untrusted apps to STOVEPipe (§II-B). Hence, untrusted apps are restricted to accessing sensor data in the ways supported by STOVEPipe. As a result, untrusted apps may not be able to use complex operations of the sensor hardware on the mobile device, such as manually controlling the camera on the mobile device.

Performance considerations: As STOVEPipe performs all data accesses on behalf of untrusted apps, this will incur performance overheads. The STOVEPipe access control system will also have to be efficiently implemented to minimize the performance costs of processing access control rules.

IV. Related Work

Various techniques have been proposed to improve the access control and permissions systems of mobile platforms such as Android. A number of techniques provide developer tools [18] or mobile app rewriting and repackaging methods [19], [20] to introduce fine-grained resource permissions or resource usage prompts. These techniques need to ensure that their developer tools and repackaging capture all methods by which sensitive resources can be accessed in the mobile platform for completeness, which can be challenging. In contrast, STOVEPipe relies on execution isolation from the STOVE model [7] to prevent untrusted apps from accessing sensitive data directly. AppFence [21], MockDroid [22] and Apex [17] all modify the Android mobile platform to provide privacy controls for existing unmodified Android apps. AppFence allows users to choose to return dummy data in place of sensitive data by modifying the Android platform, and must mediate sensitive data accesses at all points in the Android framework, whereas STOVEPipe is not concerned with where in the mobile platform data accesses can occur. Apex and MockDroid allow for runtime-revocable app permissions for Android, and their systems do not observe the actual privacy-sensitive data being exchanged, unlike STOVEPipe. ipShield [23] provides privacy-preserving data obfuscations, which is similar to STOVEPipe's privacy-preserving data transformations. However, ipShield modified the Android framework to place their data obfuscations, while STOVEPipe does not need to modify the Android framework, so the correctness of their design can be difficult to check automatically due to the size and complexity of Android.

V. Conclusion and Future Work

We have presented STOVEPipe, an access control system for mobile device data for untrusted applications on mobile devices. STOVEPipe is designed for mobile devices and their rich data sources such as stored photos, videos, documents, as well as sensor data. STOVEPipe relies on the STOVE Data and Execution models [7] to ensure that untrusted apps are isolated and cannot directly access any data on the mobile device. This allows STOVEPipe to retrieve all data on behalf of untrusted apps, rendering all data accessed by untrusted apps observable. This "observable" property of STOVEPipe's access control enables us to easily implement features, such as content-based access control policies, accounting and auditing of accessed data, and privacy-preserving data transformations. As STOVEPipe's access control is not embedded in an underlying mobile platform (e.g. Android), we envision that STOVEPipe's implementation can also be easily checked, providing users with high confidence in the access control provided. We plan to implement STOVEPipe and evaluate its security and performance for mobile resource leasing systems.

Acknowledgements

This research is supported in part by the CMU-SYSU Collaborative Innovation Research Center and the SYSU-CMU International Joint Research Institute.

References

[1] "Gartner Says Worldwide Traditional PC, Tablet, Ultramobile and Mobile Phone Shipments to Grow 4.2 Percent in 2014," Jul 2014, http://www.gartner.com/newsroom/id/2791017.

[2] G. Chatzimilioudis, A. Konstantinidis, C. Laoudias, and D. Zeinalipour-Yatzi, "Crowdsourcing with Smartphones," *IEEE Internet Computing*, Sep/Oct 2012.

[3] T. Yan, V. Kumar, and D. Ganesan, "CrowdSearch: Exploiting Crowds for Accurate Real-time Image Search on Mobile Phones," in *ACM MobiSys*, 2010.

[4] P. Simoens, Y. Xiao, P. Pillai, Z. Chen, K. Ha, and M. Satyanarayanan, "Scalable Crowd-Sourcing of Video from Mobile Devices," in *ACM MobiSys*, 2013.

[5] Y. Lee, Y. Ju, C. Min, S. Kang, I. Hwang, and J. Song, "CoMon: Cooperative Ambience Monitoring Platform with Continuity and Benefit Awareness," in *ACM MobiSys*, 2012.

[6] T. Das, P. Mohan, V. Padmanabhan, R. Ramjee, and A. Sharma, "PRISM: Platform for Remote Sensing using Smartphones," in *ACM MobiSys*, 2010.

[7] J. Tan, R. Gandhi, and P. Narasimhan, "STOVE: Strict, Observable, Verifiable Data and Execution Models for Untrusted Applications," in *IEEE CloudCom Doctoral Symposium*, 2014.

[8] J. Tan, U. Drolia, R. Gandhi, and P. Narasimhan, "Poster: Towards Secure Execution of Untrusted Code for Mobile Edge-Clouds," in *ACM WiSec*, 2014.

[9] B. Lampson, "Protection," in *Proc. 5th Princeton Conf. on Information Sciences and Systems*, 1971.

[10] J. Tan, U. Drolia, R. Martins, R. Gandhi, and P. Narasimhan, "Short Paper: CHIPS: Content-based Heuristics for Improving Photo Privacy for Smartphones," in *ACM WiSec*, 2014.

[11] M. Abadi, "Logic in Access Control," in *IEEE LICS*, 2003.

[12] ——, "Access Control in a Core Calculus of Dependency," in *ENTCS*, 2007.

[13] J. Vaughan, L. Jia, K. Mazurak, and S. Zdancewic, "Evidence-based Audit," 2008.

[14] M. Conti, V. Nguyen, and B. Crispo, "CRePE: Context-Related Policy Enforcement for Android," in *Information Security Conference (ISC)*, 2010.

[15] S. Bugiel, S. Heuser, and A. Sadeghi, "Flexible and Fine-grained Mandatory Access Control on Android for Diverse Security and Privacy Policies," in *USENIX Security*, 2013.

[16] M. Miettinen, S. Heuser, W. Kronz, A. Sadeghi, and N. Asokan, "ConXsense - Context Profiling and Classification for Context-Aware Access Control," in *ASIACCS*, 2014.

[17] M. Nauman, S. Khan, and X. Zhang, "Apex: Extending Android Permission Model and Enforcement with User-defined Runtime Constraints," in *ASIACCS*, 2010.

[18] B. Livshits and J. Jung, "Automatic Mediation of Privacy-Sensitive Resource Access in Smartphone Applications," in *USENIX Security*, 2013.

[19] J. Jeon, K. Micinski, J. Vaughan, A. Fogel, N. Reddy, J. Foster, and T. Millstein, "Dr. Android and Mr. Hide: Fine-grained Permissions in Android Applications," in *SPSM*, 2012.

[20] R. Xu, H. Saidi, and R. Anderson, "Aurasium: Practical Policy Enforcement for Android Applications," in *USENIX Security*, 2012.

[21] P. Hornyack, S. Han, J. Jung, S. Schechter, and D. Wetherall, "These Arent the Droids Youre Looking For: Retrofitting Android to Protect Data from Imperious Applications," in *ACM CCS*, 2010.

[22] A. Beresford, A. Rice, N. Skehin, and R. Sohan, "MockDroid: Trading privacy for application functionality on smartphones," in *Workshop on Mobile Computing Systems and Applications (HotMobile)*, 2011.

[23] S. Chakraborty, C. Shen, K. Raghavan, Y. Shoukry, M. Millar, and M. Srivastava, "ipShield: A Framework for Enforcing Context-Aware Privacy," in *NSDI*, 2014.