SPEProxy: Enforcing Fine Grained Security and Privacy Controls on Unmodified Mobile Devices

Brian Krupp
Computer Science Department
Baldwin Wallace University
Berea, OH 44145
Email: bkrupp@bw.edu

Dan Jesensky
Computer Science Department
Baldwin Wallace University
Berea, OH 44145
Email: djesensk14@mail.bw.edu

Amanda Szampias
Computer Science Department
Baldwin Wallace University
Berea, OH 44145
Email: aszampia15@mail.bw.edu

***Abstract*—Mobile applications have grown from knowing basic personal information to knowing intimate details of consumer's lives. The explosion of knowledge that applications contain and share can be contributed to many factors. Mobile devices are equipped with advanced sensors including GPS and cameras, while storing large amounts of personal information including photos and contacts. With millions of applications available to install, personal data is at constant risk of being misused. While mobile operating systems provide basic security and privacy controls, they are insufficient, leaving the consumer unaware of how applications are using permissions that were granted.**

In this paper, we propose a solution that aims to provide consumers awareness of applications misusing data and policies that can protect their data. From this investigation we present SPEProxy. SPEProxy utilizes a knowledge based approach to provide consumer's an ability to understand how applications are using permissions beyond their stated intent. Additionally, SPEProxy provides an awareness of fine grained policies that would allow the user to protect their data. SPEProxy is device and mobile operating system agnostic, meaning it does not require a specific device or operating system nor modification to the operating system or applications. This approach allows consumers to utilize the solution without requiring a high degree of technical expertise. We evaluated SPEProxy across 817 of the most popular applications in the iOS App Store and Google Play. In our evaluation, SPEProxy was highly effective across 86.55% applications where several well known applications exhibited misusing granted permissions.

I. Introduction

The pervasiveness of smartphones and tablets in consumers lives have been profound in the past decade. A Pew survey found in 2015 that 68% of Americans have smartphones and 45% have tablet computers [1]. While personal computers have experienced explosive growth in the past, smartphones and tablets have a more profound social impact where these devices sense and store troves of personal information including location data, detailed contact information, and personal photos. Additionally, amongst the two most popular mobile operating systems, Android and iOS [2] there are over 4.5 million applications that users can install on their device [3]. With millions of applications available to install, personal data is at constant risk. To address these issues, various solutions have been presented that either require a modification to the operating system or application. However, these approaches are generally not available to consumers because of hardware restrictions or other technical requirements. A solution is needed to allow consumers to become aware of how applications are using their data and allow them to enforce fine grained policies. To overcome these challenges, we propose SPEProxy (Security and Privacy Enhanced Proxy), a novel knowledge based proxy approach that allows consumers to become aware of how applications use both sense and stored data as well as implement fine grained policies. SPEProxy addresses several limitations from previous work:

- It does not require a modification to the operating system or to the application.
- It does not require a specific set of smartphones or tablets to provide protection.
- It provides protection even if an application is compromised or colludes with other applications.
- It complements existing security and privacy controls and therefore does not compromise them.
- It can be enabled or disabled without requiring the user to reinstall an application or custom operating system.

SPEProxy uses a knowledge based temporal abstraction (KBTA) approach [4] to detect misuse of consumer's data and provide fine grained security and privacy controls. Controls are implemented via policies that are created reactively, where consumers make decisions only when required. Once a decision is made, the user's policy is updated and enforced. Using this approach, consumers do not need to build a policy in advance. A companion mobile application, SPEClient was developed for both iOS and Android. SPEClient synchronizes data with SPEProxy to ensure the service can protect user's data and synchronize policy decisions made from notifications. Within the app, user's can modify existing policies or create policies manually. To ensure SPEProxy can inspect all traffic, a private certificate authority is generated for each user to establish trust between the client and proxy, and for the proxy to inspect traffic over a SSL session.

To measure the effectiveness of SPEProxy, we evaluated 817 of the most popular free applications across the iOS App Store and Google Play. In our evaluation, we measured SPEProxy's ability to enforce policy and the application's ability to function using a proxy service. From this evaluation,

978-1-5386-1104-3/17/$31.00 2017 IEEE

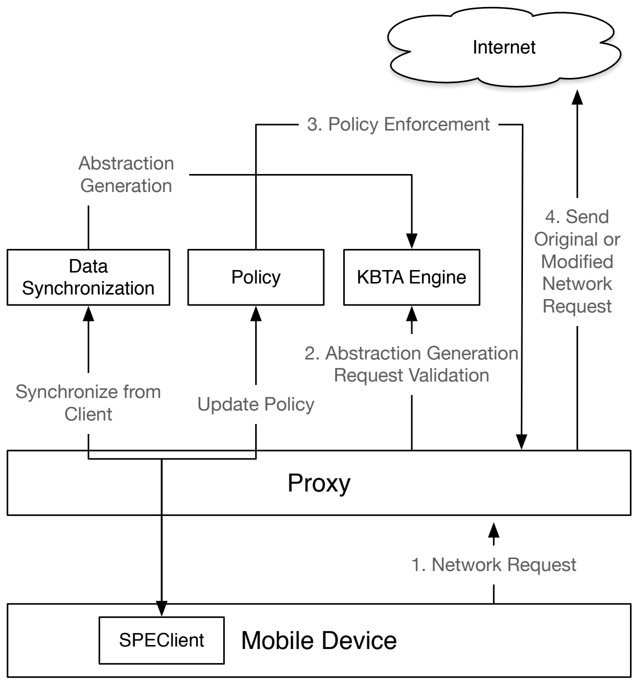


Fig. 1. High Level Design of SPEProxy

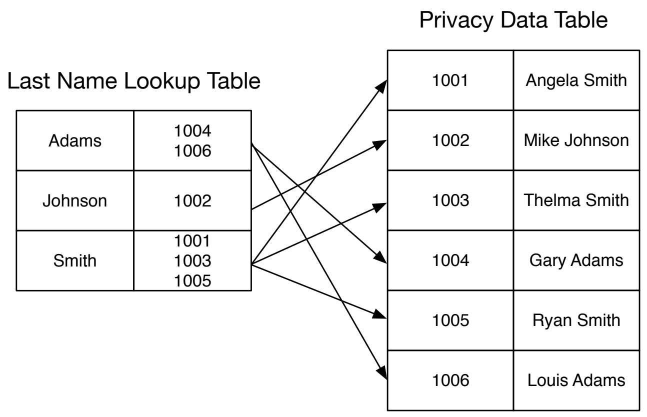


Fig. 2. Lookup to Privacy Table

SPEProxy was highly effective among 86.55% applications.

Our contributions from this research are the following:

- SPEProxy - A novel knowledge based proxy approach that provides consumer awareness and allows consumers to implement fine grained security and privacy controls without requiring a technical background to adopt.
- A solution that provides consumer awareness that is device and mobile operating system agnostic and does not break the inherent trust of the consumer's device.
- An evaluation of 817 of the most popular applications in major categories of the iOS App Store and Google Play.

The rest of the paper is organized as follows: we describe the design of SPEProxy and its related components in Section II; we discuss our approach in evaluation and provide results in Section III; we compare our work with related work in Section IV and then provide a conclusion in Section V.

II. SPEProxy Design

SPEProxy's design has two primary goals: 1) Function with any mobile device and any mobile operating system, 2) Do not require high level of technical expertise. SPEProxy contains five major components to meet these goals: Proxy Service, KBTA Engine, Data Synchronization, Policy, and SPEClient. The interaction of these components is shown in Figure 1 and are described in greater detail in the following sections.

A. *Proxy Service and Interceptor*

The proxy service contains a web proxy that is fully capable of proxying HTTP requests as well as manage SSL/TLS. SPEProxy doesn't provide web proxy capabilities directly, rather it focuses on offering security and privacy enhancements through an integration with LittleProxy [5]. However it can be tied into any proxy service that provides similar integration capabilities. The Interceptor component in SPEProxy captures all HTTP/HTTPS traffic to determine if a request originates from SPEClient or a mobile application. SPEClient's primary goal is to synchronize sensed and stored data from the the user's device, synchronize policy decisions, and handle notifications from SPEProxy. If a request is sent from SPEClient, it does not need to go through an approval process, otherwise it will be inspected within Request Approver.

B. *Request Approver*

The Request Approver performs the following tasks for each HTTP request:

1) Inspect properties of the request that could contain personal data.
2) Generate knowledge abstractions from inspected properties and forward abstractions to KBTA Engine.
3) Request KBTA Engine to validate abstractions (described in further detail in Section II-D)
4) If KBTA validation fails, enforce policy defined by user.

Within the HTTP request, the headers, body, and request string are inspected for personal data which can exist in one of three categories: sensed data, textual data, or multimedia. The process for inspecting each property is based on the category. For example, to identify location coordinates (sensed data), SPEProxy requires previous knowledge of sensed location coordinates. Without this knowledge, floating point numbers in the range of valid latitude and longitude values could be falsely flagged as location data. To ensure SPEProxy is able to protect location data, location coordinates are periodically synchronized from the client.

Inspection of textual data includes attributes such as a contact's phone number or street address. Regular expressions containing keys from lookup tables are used to identify data of interest. Lookup tables are constructed from recently synchronized data from the SPEClient, where the key in the table

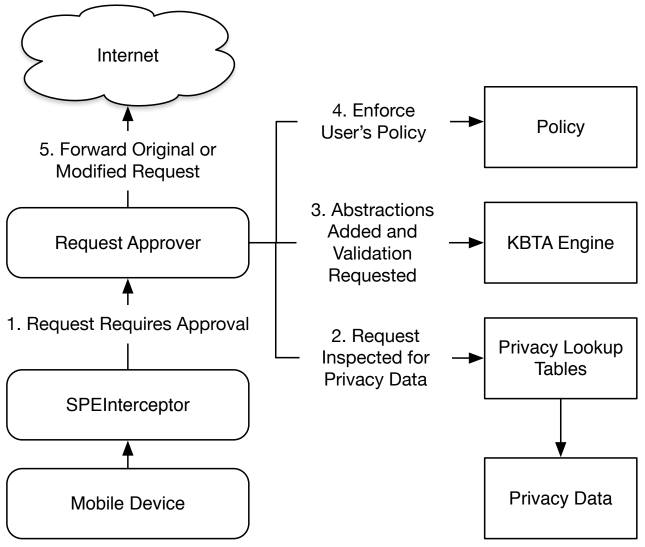

Internet
5. Forward Original or Modified Request
Request Approver
1. Request Requires Approval
SPEInterceptor
Mobile Device
4. Enforce User's Policy
Policy
3. Abstractions Added and Validation Requested
KBTA Engine
2. Request Inspected for Privacy Data
Privacy Lookup Tables
Privacy Data

Fig. 3. Request Approval Process

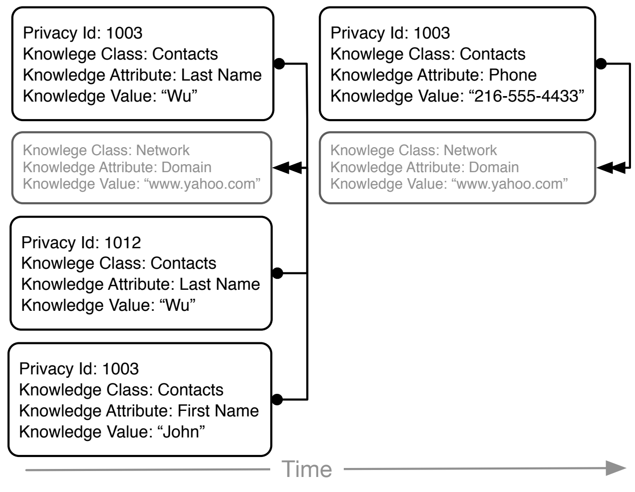


Fig. 4. KBTA Vector Including Knowledge Abstractions

corresponds to a list of identifiers that uniquely identify a given element such as a contact. This relationship is depicted in Figure 2. Any match from a key in the lookup table generates a knowledge abstraction that is added to the KBTA Engine, where rules determine if the abstraction represents personal data. After abstractions are added to the KBTA Engine, validation is performed. If validation succeeds, the request is forwarded to the destination as is. If validation fails, the user's policy is enforced and Request Approver will block or make modifications to the request based on policy definitions. This process is depicted in Figure 3.

C. Certificate and Key Management

To ensure SSL and TLS traffic can be inspected, trust is established between the mobile device and the SPEProxy by generating a private certificate authority (CA) for the user. The CA generate certificates based on the destination's common name (CN). The CA is imported conveniently via SPEClient, requiring no technical expertise from the user. Frequently used keys and certificates are cached by SPEProxy to reduce on demand generation which can be resource intensive.

D. KBTA Engine

The KBTA Engine is a critical component in the SPEProxy's ability to detect personal data, enforce policy, and eliminate false positives. It creates, manages, and validates knowledge based temporal abstractions that are used for detecting meaningful events such as contact information being leaked. KBTA was initially proposed in the medical domain [6] where it could be used to diagnose patients. KBTA provides a formal method for modeling events, context, and parameters to create abstractions based on predefined goals which can reduce the complexity of inferring knowledge [4]. Other research has proposed using KBTA to provide an intrusion detection system in Android [7]. In SPEProxy, KBTA is used to create inferred knowledge by using defined rules that contain criteria for matching and correlating knowledge abstractions.

Using a knowledge based approach, data found in requests and synchronized from the SPEClient creates knowledge abstractions. Together, these abstractions generate inferred knowledge that is used to detect personal data without false positives. As an example, suppose a request contains the last name "Wu". This finding alone may not represent personal data and if reported could represent a false positive that blocks or alters a legitimate request. However, if other knowledge abstractions contain elements of a contact such as the phone number or address, then it can be inferred that personal data is in a request. This example is depicted in Figure 4. Abstractions are categorized by the following types: *Network Activity*, *Existing Knowledge*, and *Inferred Knowledge*. KBTA Rules generate inferred knowledge and are described in detail in Section II-D2. KBTA rules also allow SPEProxy to detect collusion among applications [8] by examining requests across more than one application in a given period. For example, if applications use their combined permissions to misuse user's data, knowledge abstractions are generated regardless of application, type, and destination. Using a KBTA approach, SPEProxy can detect the attempt and enforce a user's policy.

1) Knowledge Classes and Attributes: Abstractions are represented by four main *Knowledge Classes*. These include: *Contact*, *Photo*, *Location*, and *Network*. Within each class, *Knowledge Attributes* specify additional details about the abstraction. As an example, the Contact class would contain the following attributes: *FirstName*, *LastName* and *Email*. For each attribute detected, separate abstractions are created and stored in a time sorted vector within the KBTA Engine.

2) KBTA Rules: KBTA Rules provide a mechanism to create inferred knowledge to detect personal data and enforce policy. KBTA Rules contain the following properties: *Rule*

```
$ValidationResult$ = Empty Result
for all $Rule$ in $KBTAEngine.Rules$ do
    $MatchingAbstractions$ = Empty Array
    for all $Abstraction$ in $KBTAEngine.abstractions$ do
        if $Abstraction$ Satisfies $Rule.MatchCriteria$ then
            Add $Abstraction$ to $MatchAbstractions$
        end if
    end for
    if $MatchAbstractions$ Satisfies $Rule.CorrelationCriteria$
    then
        $ValidationResult$ = Inferred Knowledge from Criteria
    end if
end for
return $ValidationResult$
```

Fig. 5. Algorithm for Evaluating KBTA Rules

Name, *Temporal Span*, *Match Criteria*, and *Correlation Criteria*. Temporal Span specifies a time span where abstractions within given Match and Correlation Criteria must exist in to create inferred knowledge.

Match Criteria provides a mechanism for querying generated abstractions. As such, they have properties that correspond to properties of abstractions: *Matching Knowledge Item*, *Matching Knowledge Class*, *Matching Knowledge Attribute*, and *Matching Knowledge Attribute Value*. Matching Knowledge Item provides a unique identifier for a personal data element such as a contact or photo. Matching Knowledge Class will only return abstractions for a given class such as Photo or Contact. However, all properties are optional; if no properties are defined all abstractions will be returned.

Correlation Criteria provides a mechanism for identifying relationships among abstractions. When rules are evaluated, if Match Criteria returns a non-empty set, then Correlation Criteria is evaluated across the returned abstractions. Correlation Criteria contain the following properties: *Min Number of Correlations Required*, *Require Same Item Id*, *Require Same Knowledge Attribute Value*, and *Require Different Attribute Values*. All properties except *Min Number of Correlations Required* are boolean properties.

KBTA Rules are evaluated using the algorithm defined in Figure 5. Additionally, rules ensure abstractions do not grow indefinitely as abstractions are removed that exist outside the maximum temporal span defined by an active rule.

E. Data Synchronization

Periodically, data from the SPEClient must be synchronized with SPEProxy. This allows SPEProxy to protect data that may be transmitted and ensure data that is no longer on the device is not inspected. Additionally, policy decisions from notifications are synchronized with SPEProxy so they take effect immediately. To synchronize location updates, SPEClient only captures major location updates to reduce power consumption, a similar approach used for *geofencing*. Photos and contacts however only need to be synchronized with SPEProxy when there are updates. To reduce synchronization time, SPEClient initializes synchronization with a *syncStart* message containing a list of unique identifiers. SPEProxy determines from this list any elements it requires and returns a list of elements to SPEClient, which the client will synchronize with SPEProxy. This approach allows the proxy to remove artifacts that no longer exist on the device.

F. Policy

SPEProxy utilizes an ontology developed specifically for security and privacy policies on mobile devices using a knowledge based approach [9]. This policy allows users to specify fine grained security and privacy controls. As an example, a user can specify a policy that allows a subset of contacts to be transmitted to a domain without allowing certain attributes of the contacts. As another example, a user can define a policy that allows anonymized location data to be transmitted to a domain that provides a legitimate service however randomize location data destined to advertisement servers. There are two main categories of policies that can be defined: *Privacy Policies* and *Domain Policies*. Privacy Policies can have one to many domain policies associated with them. These policies are described in further detail in the following sections:

1) Privacy Policies: Privacy Policies have the following properties: *privateDataType*, *applyToAllElementsInClass*, *privacyIds*, *applyToAllDomains*, *domainPolicies*, *permitRawData*, and *denyPrivacyAction*. The *privateDataType* property defines which Knowledge Class the policy is for (Photos, Contacts, etc). The flag *applyToAllElementsInClass* and array *privacyIds* provides the ability to apply the policy to all elements of a set or restrict access to elements in a set.

If *permitRawData* is true, then no modification to the request will occur. However if it is false, than one of the *denyPrivacyActions* will be applied. *denyPrivacyAction* is an enum with the values *FuzzRandom*, *FuzzAnonymous*, *FuzzGeneralized*, and *BlockRequest*. The fuzzing properties allow the user to permit data but not in its original state. For example with location data, FuzzRandom produces a random location with valid latitude and longitude coordinates. FuzzAnonymous produces a random point within a given radius of the last detected location. FuzzGeneralized produces a fixed point within a given radius of the last detected location.

Privacy policies also allow the ability to specify *spatialRestrictions* and *temporalRestrictions*. These restrictions allow data to be blocked if the user is present in a given location or during certain times of the day. For specific knowledge classes such as contacts, extended attribute exist which include the following: *allowEmailAccess*, *allowPhoneAccess*, *allowAddressAccess*. These properties are boolean types and if a property is set to *false*, SPEProxy replaces data in the request with arbitrary values. These additional controls allow applications that expect data to still function while protecting the user from compromising their privacy.

2) Domain Policies: Domain Policies provide a mechanism for permitting or denying traffic to specific domains and have the following properties: *domainPattern*, *uriPattern*, *requireSSL*, and *permitData*. The *domainPattern* and *uriPattern* properties allow policies to be defined for specific endpoints

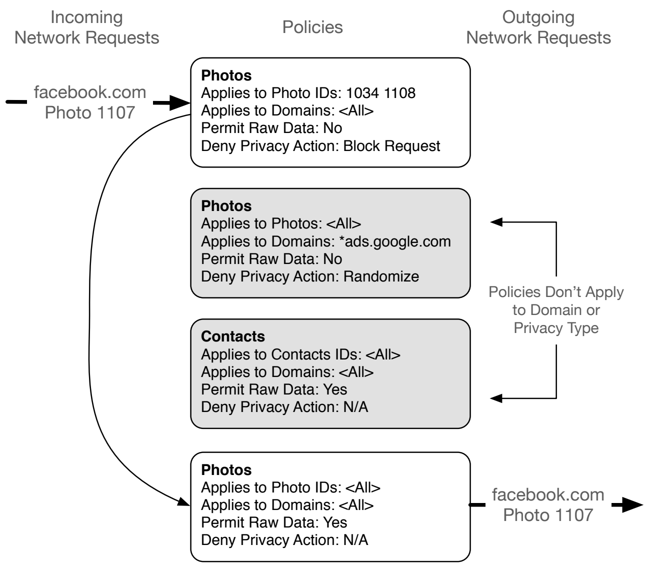
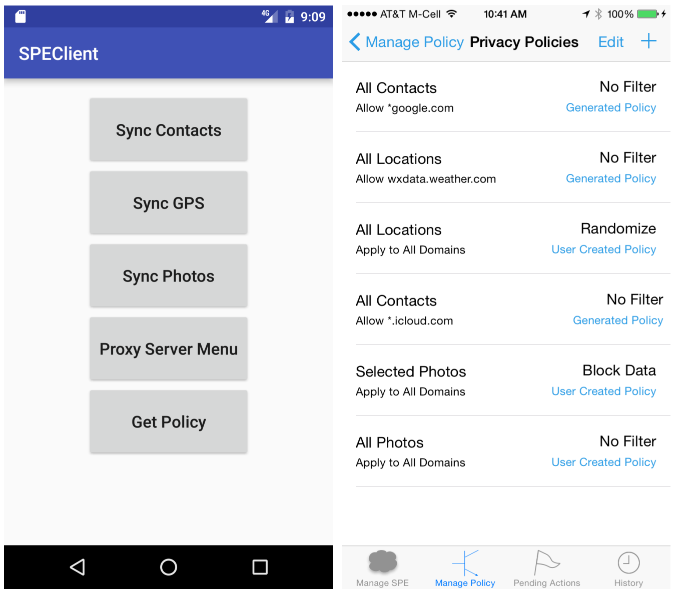

Incoming Network Requests
Policies
Outgoing Network Requests
facebook.com Photo 1107
Photos
Applies to Photo IDs: 1034 1108
Applies to Domains: <All>
Permit Raw Data: No
Deny Privacy Action: Block Request
Photos
Applies to Photos: <All>
Applies to Domains: *ads.google.com
Permit Raw Data: No
Deny Privacy Action: Randomize
Policies Don't Apply to Domain or Privacy Type
Contacts
Applies to Contacts IDs: <All>
Applies to Domains: <All>
Permit Raw Data: Yes
Deny Privacy Action: N/A
Photos
Applies to Photo IDs: <All>
Applies to Domains: <All>
Permit Raw Data: Yes
Deny Privacy Action: N/A
facebook.com Photo 1107

Fig. 6. Policy Evaluation Top Down



All Contacts Allow *google.com	No Filter Generated Policy
All Locations Allow wxdata.weather.com	No Filter Generated Policy
All Locations Apply to All Domains	Randomize User Created Policy
All Contacts Allow *.icloud.com	No Filter Generated Policy
Selected Photos Apply to All Domains	Block Data User Created Policy
All Photos Apply to All Domains	No Filter User Created Policy

Fig. 7. Screenshots of SPEClients

and can use wildcard patterns. The *requireSSL* attribute ensures requests will only be sent if SSL/TLS is being used. Additionally, *permitData* is a flag that allows the user to block data to any domain (such as advertisement servers).

3) Policy Management: Fine grained privacy and security controls can be complex. Therefore, SPEProxy utilizes a reactive based policy mechanism that provides the ability to create policy through a menu of options, only when action is required. By default, all personal data is blocked unless a policy indicates that it can proceed to the provided domain in its current state or in a modified state. Using this approach, users can choose to allow data to proceed through the proxy using the following options:

- Allow Domain(s) to Receive All *Knowledge Class*
- Allow Domain(s) to Receive Detected *Knowledge Class*
- Block Domain(s) from Receiving All *Knowledge Class*
- Block Domain(s) from Receiving Detected *Knowledge Class*

Once the user selects one of these options, policy is synchronized and enforced with SPEProxy. However, the user can further modify the policy within SPEClient.

4) Policy Evaluation: Policy is evaluated in top-down ordering where users can organize policies created reactively through notifications or created manually. This ordering allows more restrictive policies to be evaluated before more permissive policies at the bottom. Only policies that correspond to the appropriate knowledge class and network domain are evaluated. This evaluation is depicted in Figure 6.

G. SPEClient

SPEClient is a critical component of SPEProxy, providing consumer awareness of personal data being utilized. Screenshots of the SPEClients are shown in Figure 7. SPEClient has four primary responsibilities:

- Synchronize personal and sensed data from SPEClient to SPEProxy.
- Alert user on personal data being utilized via notifications and allow user to make a policy decision.
- Allow manual configuration of policy and synchronize policy with SPEProxy.
- Allow configuration required for SPEProxy and show current protection status.

III. Analysis and Evaluation

We evaluated SPEProxy with 817 of the most popular applications in the iOS App Store and Google Play. In our evaluation, we focus on the ability of SPEProxy to detect personal data and protect it using a user defined policy. We also evaluated the ability to enforce various attributes of policies including modifying data in requests. Lastly, we measured the ability of an application to function with SPEProxy being utilized. To evaluate SPEProxy, we deployed the service on an internally hosted Ubuntu Virtual Machine and on several Linux virtual machines we received through Google's Cloud Platform Education Grants. To evaluate iOS applications, we utilize an iPhone 6 and iPad Mini 3. To evaluate Android we utilized a Samsung Tab 4 and Nexus 6p.

We evaluated the top 20 free applications across major categories in iOS and top 10 applications across major categories for Android. The total number of applications evaluated includes 817 applications where several application existed in multiple categories and were therefore only evaluated once. The following procedures were used to evaluate each application:

- Download target application to mobile device.

Platform	Category	Application Name	Misuse Identified
iOS	Weather	Weather Channel	Sent location data to advertisement servers.
iOS	Business	iScanner	Sent location data to advertisement servers.
iOS	Food	Domino's Pizza	Sent location data to advertisement servers.
iOS	Lifestyle	Realtor	Sent location data to advertisement servers.
iOS	News	CBS	Sent location data to advertisement servers.
iOS	Reference	Dictionary.com	Sent location data to advertisement servers.
iOS	Social	LinkedIn	Sent contact email, phone number, and street address.
iOS	Social	Pinterest	Sent contact email and phone number.
iOS	Utilities	Calculator Pro	Sent location data to advertisement servers.
Android	Art & Design	IBIS Paint	Sent location data to advertisement servers.
Android	Dating	FastMeet	Sent location data to advertisement servers.
Android	Reference	LifeChurch	Sent location data to advertisement servers.
Android	Reference	Merriam Webster	Sent location data to advertisement servers.
Android	Music	Go Music Player	Sent location data to advertisement servers.
Android	Social	TextFree	Sent contact information to multiple advertisement servers.

TABLE I
SUBSET OF POPULAR APPLICATIONS THAT DEMONSTRATED MISUSE OF DATA

- Initiate synchronization of location from SPEClient.
- Open application and approve any permission requests from the operating system.
- Test application, approve permission requests, and monitor notifications and proxy log files for personal data.
- Evaluate ability to detect and protect personal data.
- Close application, monitor log file for transmission of data, and save log file.

Of the 817 applications evaluated, 744 were able to be evaluated using the proxy. 73 applications could not be fully tested due to various circumstances such as requiring paid accounts, specific hardware, or requiring an account that could not be created through an online service. From the 744 applications that were evaluated, 644 of them were fully functional with SPEProxy, an 86.55% success rate. Of these applications, 283 communicated with advertisement services where personal data was shared and 43 applications misused data beyond their stated intent when requesting a permission.

When evaluating personal data usage, we checked if the application requested the permission through the native operating system permission system. We granted the appropriate permission then monitored notifications and logs within the proxy to determine if personal data was transmitted. Even though more iOS applications (489) were evaluated than Android applications (328), there were 160 iOS permission requests compared to 381 Android permission requests. Misuse of personal data is defined as an application misrepresenting their stated intent to request access to a resource. In our evaluation, the common misuse case was an application sending location data to advertisement servers. However, misuse also included using photo or contact information without explicitly notifying the user how that information was used. For example, *LinkedIn* requested access to contacts to suggest connections in the application. However, the application transmitted the email address, phone numbers, and street address. Highlights from 43 cases of confirmed misuse are included in Table I.

Misuse generally occurred at the beginning of evaluating an application. Weather and utility apps abused permissions most with 9 out of the top 20 iOS weather applications asking for location data for a legitimate use case only to later transmit location data to advertisement servers. This included the most popular application in that category: *The Weather Channel*. Most applications that requested access to photos did not attempt to transmit photos. Applications that requested access to contacts frequently sent the email address and phone number of a contact. This can be justified as a mechanism to uniquely identify a user, however often the user was unaware that this data was being transmitted based on the application's stated intent. Misuse is detected by SPEProxy when data is transmitted from the device. Once the server has this data, it may be shared with other services. However, even if an application uses data legitimately, SPEProxy is able to protect the data. For example, SPEProxy can ensure that precise location information is not shared unless absolutely required as in a navigation application.

IV. Related Work

To provide additional security and privacy controls for mobile OS, there have been two primary approaches: modify the operating system or application.

Modify Operating System Approaches have been proposed by modifying the operating system to provide protection. We we will briefly enumerate them here. One of the more popular solutions TaintDroid detected privacy leakage using dynamic taint analysis [10]. ProtectMyPrivacy is another solution that targeted iOS devices that were jailbroken [11]. Other solutions that required modification include TISSA [12], Android Runtime Security Policy Enforcement Framework [13], MockDroid [14], RiskMon [15], Context-Based Access Control Systems for Mobile Devices [16], PSiOS [17] DefDroid [18] and RecDroid [19].

Modify Application As with modifying the operating system, approaches have been proposed to modify the application. One example is a method for integrating a framework in iOS applications by modifying application behavior at runtime [20]. Other examples that include a modification to the application include PrivacyCapsules [21] and CASE [22].

While these approaches were successful providing fine grained security and privacy controls, these primary approaches are unsustainable. In modifying the operating system, each minor update to the source OS requires an update to the modified OS. Additionally, this approach cannot be used on iOS as it is not open source. For Android, this approach is limited to the Nexus and Pixel family of devices. In modifying the application, similar to modifying the OS, application updates are required any change to the framework. Additionally, application producers need an incentive to adopt a framework and a third party needs to verify framework adoption.

V. Conclusion

We presented SPEProxy, a novel knowledge based proxy approach utilizing a formal KBTA process to provide consumers awareness of how their personal data is being utilized by mobile applications and apply fine grained policies on their personal data. This approach does not require a specific operating system or device, allowing consumers to utilize the solution with a simple network configuration setting. SPEProxy can be installed by any user on a personal computer, microcomputer, or in a private cloud service. We created an implementation of SPEProxy and mobile clients for both Android and iOS. We evaluated 817 applications across Google Play and the iOS App Store, focusing on the top ranked applications across major categories. From this evaluation, we found SPEProxy to be highly effective across 86.55% of the applications. Additionally, we found 43 confirmed cases of applications misusing personal data. SPEProxy was highly effective in protecting the user's data and also reducing false positives by utilizing a formal KTBA process. SPEProxy achieves its primary goal in providing a generally available solution to consumers that provide awareness in how applications use their data and fine grained policies. Future work for this research includes evaluating dynamic modification to KBTA rules to ensure successful detection of personal data and enforcement of policy. Dynamic modification can be incorporated using AI or through user recommendations. Additionally, future work includes evaluating SPEProxy on microcomputer devices such as Raspberry Pi for consumers to adopt locally.

References

[1] Monica Anderson, "Technology Device Ownership: 2015," Oct. 2015. [Online]. Available: http://www.pewinternet.org/2015/10/29/technology-device-ownership-2015/

[2] "IDC: Smartphone OS Market Share," Dec 2016. [Online]. Available: http://www.idc.com/prodserv/smartphone-os-market-share.jsp

[3] S. V. President and BCG, "App stores: number of apps in leading app stores 2016," Dec 2016. [Online]. Available: https://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/

[4] Y. Shahar, "A framework for knowledge-based temporal abstraction," *Artificial Intelligence*, vol. 90, no. 1, pp. 79 – 133, 1997. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0004370296000252

[5] A. Fisk, "adamfisk/LittleProxy," 04 2015. [Online]. Available: https://github.com/adamfisk/LittleProxy

[6] Y. Shahar and M. A. Musen, "Knowledge-based temporal abstraction in clinical domains," *Artificial intelligence in medicine*, vol. 8, no. 3, pp. 267–298, 1996.

[7] A. Shabtai, U. Kanonov, and Y. Elovici, "Intrusion detection for mobile devices using the knowledge-based, temporal abstraction method," *J. Syst. Softw.*, vol. 83, no. 8, pp. 1524–1537, Aug. 2010. [Online]. Available: http://dx.doi.org/10.1016/j.jss.2010.03.046

[8] A. M. Memon and A. Anwar, "Colluding apps: Tomorrow's mobile malware threat," *IEEE Security and Privacy*, vol. 13, no. 6, pp. 77–81, 2015.

[9] W. Z. Brian Krupp, Nigamanth Sridhar, "An ontology for enforcing security and privacy policies on mobile devices," in *Proceedings of the 6th International Conference on Knowledge Engineering and Ontology Development (KEOD'14)*, 2014.

[10] W. Enck, P. Gilbert, B.-G. Chun, L. P. Cox, J. Jung, P. McDaniel, and A. N. Sheth, "Taintdroid: an information-flow tracking system for realtime privacy monitoring on smartphones," in *Proceedings of the 9th USENIX conference on Operating systems design and implementation*, ser. OSDI'10. Berkeley, CA, USA: USENIX Association, 2010, pp. 1–6. [Online]. Available: http://dl.acm.org/citation.cfm?id=1924943.1924971

[11] Y. Agarwal and M. Hall, "Protectmyprivacy: detecting and mitigating privacy leaks on ios devices using crowdsourcing," in *Proceeding of the 11th annual international conference on Mobile systems, applications, and services*, ser. MobiSys '13. New York, NY, USA: ACM, 2013, pp. 97–110. [Online]. Available: http://doi.acm.org/10.1145/2462456.2464460

[12] Y. Zhou, X. Zhang, X. Jiang, and V. W. Freeh, "Taming information-stealing smartphone applications (on android)," in *Proceedings of the 4th international conference on Trust and trustworthy computing*, ser. TRUST'11. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 93–107. [Online]. Available: http://dl.acm.org/citation.cfm?id=2022245.2022255

[13] H. Banuri, M. Alam, S. Khan, J. Manzoor, B. Ali, Y. Khan, M. Yaseen, M. N. Tahir, T. Ali, Q. Alam, and X. Zhang, "An android runtime security policy enforcement framework," *Personal Ubiquitous Comput.*, vol. 16, no. 6, pp. 631–641, Aug. 2012. [Online]. Available: http://dx.doi.org/10.1007/s00779-011-0437-6

[14] A. R. Beresford, A. Rice, N. Skehin, and R. Sohan, "Mockdroid: Trading privacy for application functionality on smartphones," in *Proceedings of the 12th Workshop on Mobile Computing Systems and Applications*, ser. HotMobile '11. New York, NY, USA: ACM, 2011, pp. 49–54. [Online]. Available: http://doi.acm.org/10.1145/2184489.2184500

[15] Y. Jing, G. J. Ahn, Z. Zhao, and H. Hu, "Towards automated risk assessment and mitigation of mobile applications," *IEEE Transactions on Dependable and Secure Computing*, vol. 12, no. 5, pp. 571–584, Sept 2015.

[16] B. Shebaro, O. Oluwatimi, and E. Bertino, "Context-based access control systems for mobile devices," *IEEE Transactions on Dependable and Secure Computing*, vol. 12, no. 2, pp. 150–163, March 2015.

[17] T. Werthmann, R. Hund, L. Davi, A.-R. Sadeghi, and T. Holz, "PSiOS: Bring Your Own Privacy & Security to iOS Devices," in *Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security*, ser. ASIA CCS '13. New York, NY, USA: ACM, 2013, pp. 13–24. [Online]. Available: http://doi.acm.org/10.1145/2484313.2484316

[18] P. Huang, T. Xu, X. Jin, and Y. Zhou, "Defdroid: Towards a more defensive mobile os against disruptive app behavior," in *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '16. New York, NY, USA: ACM, 2016, pp. 221–234. [Online]. Available: http://doi.acm.org/10.1145/2906388.2906419

[19] B. Rashidi, C. Fung, and T. Vu, "Recdroid: A resource access permission control portal and recommendation service for smartphone users," in *Proceedings of the ACM MobiCom Workshop on Security and Privacy in Mobile Environments*, ser. SPME '14. New York, NY, USA: ACM, 2014, pp. 13–18. [Online]. Available: http://doi.acm.org/10.1145/2646584.2646586

[20] B. Krupp, N. Sridhar, and W. Zhao, "Spe: Security and privacy enhancement framework for mobile devices," *IEEE Transactions on Dependable and Secure Computing*, vol. PP, no. 99, pp. 1–1, 2015.

[21] R. Herbster, S. DellaTorre, P. Druschel, and B. Bhattacharjee, "Privacy capsules: Preventing information leaks by mobile apps," in *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '16. New York, NY, USA: ACM, 2016, pp. 399–411. [Online]. Available: http://doi.acm.org/10.1145/2906388.2906409

[22] S. Zhu, L. Lu, and K. Singh, "Case: Comprehensive application security enforcement on cots mobile devices," in *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '16. New York, NY, USA: ACM, 2016, pp. 375–386. [Online]. Available: http://doi.acm.org/10.1145/2906388.2906413