Privacy control in smart phones using semantically rich reasoning and context modeling

Dibyajyoti Ghosh, Anupam Joshi, Tim Finin and Pramod Jagtap
{dg9, joshi, finin, pramod1}@cs.umbc.edu
University of Maryland, Baltimore County

***Abstract*—We present our ongoing work on user data and contextual privacy preservation in mobile devices through semantic reasoning. Recent advances in context modeling, tracking and collaborative localization have led to the emergence of a new class of smartphone applications that can access and share embedded sensor data. Unfortunately, this also means significant amount of user context information is now accessible to applications and potentially others, creating serious privacy and security concerns. Mobile OS frameworks like Android lack mechanisms for dynamic privacy control. We show how data flow among applications can be successfully filtered at a much more granular level using semantic web driven technologies that model device location, surroundings, application roles as well as context-dependent information sharing policies.**

***Index Terms*—context awareness, mobile, android, semantic web, Jena**

I. Introduction

The automatic determination of a user's social context is a desirable functionality for the next generation of adaptive, personalized mobile phone applications. Everyday smart phone devices generate tremendous amount of data on user preferences, on device intercommunication on user context. By exploiting built-in sensors, smart phones have become attractive options for large-scale sensing of human and social behavior [1], [2].

A wide array of novel applications has been built using geo-location information from GPS. Applications like Instagram allow taking pictures and sharing with social networks, while Foursquare allows location tagging. Certain commercial entities like Jawbone [3] have come up with hardware synchronized with Apple's iOS to monitor user's sleeping patterns or number of footsteps taken in a day by the user. The accuracy of the reported data for such systems is beyond the scope of this literature. What is pertinent is that each of these successful mobile applications has moved beyond geo-location awareness and has given a new meaning to user context.

Compared to the security of traditional computing platforms, the security of mobile devices faces more challenges [4] because they possess many unique features, including personalization, mobility, pay-for-service and limited resources. Mobile operating systems like Android [5], a Google-led open source mobile platform, adopts a series of security mechanisms such as UIDs, permission label, application signing and sandboxing to enhance its security [6], [7], [5]. The existing controls in context-aware systems are based on the static information and are predetermined. In most of the cases user is asked to make decision to share sensor information such as location at application install time.

These controls are not adequate for context-aware systems, since context is dynamic and determinative of what data can be shared. User should be in control of the release of user's personal information at different levels of granularity, from raw sensed data to high-level inferred context information. They can be understandably sensitive about how sensor data is captured and used, especially if it is used to reveal their location, speech, images, or video. Mobile applications such as the Audio Loop [8], which continuously record raw audio, also raise concerns and introduce issues about how (or even whether) to obtain consent to be recorded from others whose data might be captured by the user's device [9].

This paper discusses a semantic policy based system that reasons over a user's dynamic context and constrains the information flow among applications by extending the open source Android framework. Previous work of Chen et al. [10] presented an ontology to represent various types of contextual information in pervasive computing environments, specifically, smart meeting rooms.

Jagtap et al. [11] described context-aware system that decided whether and how to report a user's location based on the user's context and information about the requester. The knowledge based was implemented using the Semantic Web language OWL and reasoning was done on the mobile device using a subset of the Jena software framework. A user-specified information sharing policy was generated from a choices made via a graphical interface and expressed via a combination of OWL-DL axioms and Jena rules. We have validated this implementation in an on-campus context-aware prototype system that aggregates information from a variety of sensors on the phone, infers dynamic user context and accordingly overrides data flow from hardware sensors to the applications.

Extending the popular open-source Android operating system to incorporate extended security mechanisms is not new. Mockdroid [12] renders device resources like location, network connectivity etc. unavailable at run time with user intervention. But it does not uses semantic web reasoning to infer from current context. Enck et al. extended the Android operating system to include a policy based application mediation logic in Saint [13]. The policies allow application behavior to change at run time. The focus of their work has primarily been to include additional operating security policies to oversee inter-application communication. What makes our

© 2012, Dibyajyoti Ghosh. Under license to IEEE.
DOI 10.1109/SPW.2012.27

IEEE computer society

approach novel is that semantic web driven policies are more expressive, allowing the modeling of complex user context and its inferencing capability. Context-aware services provided by smart phone frameworks can facilitate handling of multiple scenario where image upload feature of certain applications will automatically turn on/off, camera or microphone can be enabled or disabled based on device context.

II. Related Work

Context-aware systems have been studied for a long time, though the focus has been mainly on the location and activity inference. Norman Sadeh [14] discussed semantic web driven context aware mobile applications. MyCampus [15] is a mobile application that involved a collection of customizable agents capable of (semi-) automatically discovering and accessing Intranet and Internet services during the process of assisting their users in carrying out various tasks. In a later article, Sadeh [16] talked about semantic web driven reconciliation of privacy and context awareness.

Research on privacy controls in these systems has received significant attention in the last five years. AnonySense [17] is a privacy-aware architecture for collaborative pervasive applications that use mobile sensing. The Aware Home project [18] captures, processes and stores data collected by sensors about home residents and their activities. It uses role-based access control (RBAC) by defining environment roles similar to the subject roles of typical RBAC database applications. Context Privacy Service (CoPS) [19] describes the design and implementation of a privacy service that controls how, when and to whom you could disclose a user's context information. However, it does not handle context-dependent privacy policies, which can be specified by users on dynamic context data.

During the past decade a body of work on rule-based policy frameworks and access control systems has emerged. Rei [20] is a policy language designed for pervasive computing applications. It has been used to build a security framework [21] that addresses the issues of security for web resources, agents and services in the Semantic Web. Rein (Rei and N3) [22] is a distributed framework for describing and reasoning over policies in the Semantic Web. It supports N3 rules [23] for representing interconnections between policies and resources. Khalil et al. [24] studied the feasibility of context-aware telephony and examined context sharing patterns with an objective of improving design of context aware applications and services. Taintdroid [25] does a noteworthy job of tracking sensitive data flow inside the device across layers but as the authors have pointed out Taintdroid follows passive approach to prevention. CRePE [26] introduces a policy based Android extension but the user context model and the user level CRePE assumes is simplified when one considers the extent of granularity of user context and user role possible in real life circumstances.

III. Semantic Web and Policy Description

We adopt description logic (DL), specifically OWL (Web Ontology Language), and associated inferencing mechanisms to develop a model of context and policies. In our ontology model [27], the actions are in general lower level tasks and have no associated role. The activities are introduced as means to abstract multiple actions and further, to associate roles to the sets of actions. Places can be defined in terms of the activities that occur there. Ambiance includes concepts describing the environment of the principal (e.g., noise level, ambiance light, and temperature).

```
[ShareGPSRule:
    (?requester ex:requestTime ?localTime)
    (?user ex:systemUser ?true)
    (?localTime time:dayOfWeek ?day)
    ge(?day, 1) le(?day, 6)
->
    (?requester ex:canAccessGPSCoordinates "True") ]
```

Fig. 1. This simple policy rule permits sharing GPS coordinates on weekdays.

Using this ontology, each device contains a declarative knowledge base with semantically rich information about user's information, activities, inferences, and further contextual information. The knowledge base aligns with the context ontology which defines the key context concepts used for making access control decisions. The ontology supports generalization of context information by having hierarchical models for different aspects of context viz. activity and location. The following section describes location generalization and activity generalization in detail. The policy rule shown in Figure 1 allows the sharing of GPS sensor information on weekdays. Consider another scenario where requester is currently inside BuildingXYZ and the user does not want any application to know that she is present at BuildingXYZ.

IV. Prototype Implementation

The prototype implementation has two major components: a privacy control module and a device operating system. A detailed discussion on the privacy module has been provided in an earlier paper[11]. The privacy control module aims to protect user privacy by performing reasoning over the context. It deals with the resource to be protected, the owner of a resource and the requester who wants to access it. More abstractly, it accepts an RDF triple (U, C, Q), where U is the identity of the requester, C is the requester's context (expressed as RDF triples in our ontology), and Q is the query pertaining to context information. The module has access to owner's profile information and the group information along with specified privacy policies. It enforces owner's privacy policies using static information about the owner as well as dynamic information observed and inferred from her context. It consists of (i) a set of ontologies for describing activities/-context, policies and access requests, (ii) the knowledge about the owner, (ii) the privacy preferences, and (iv) a reasoning engine that accepts requests and performs the reasoning.

The reasoner runs as a system service. It is built on top of Androjena v0.5 [28] which is Android port of Jena Semantic Framework [29]. The policy files reside on device storage.

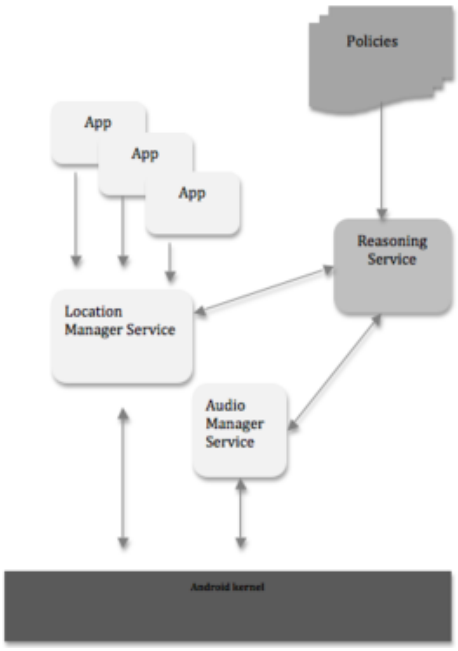

Fig. 2. In this sketch of our prototype architecture, arrows denotes the steps involved and the information flow among the components.

On receiving reasoning request from Android framework or more specifically the services e.g. LocationManagerService, AudioManagerService etc. the reasoning service consults policy files and based on the current context returns the decision asynchronously to the requesting service. The context string is forwarded to the reasoner by the requesting service upon receiving certain resource request on the device. Androjena loads policy files in-memory for every incoming decision request which is expensive because of the disk I/O involved. For smaller set of policies, lazy loading approach can slow down the reasoner response time as opposed to in-memory one time prefetch of the policies since effective memory footprint of policies will be minimal.

We used CyanogenMod-7.2.0-RC0-N1-KANG [30] for gingerbread [31] with kernel build 2.6.37.6-cyanogenmod-01509-g8913be8shade@toxygene#1 for google nexus one. Preliminary progress includes modification of LocationManagerService and AudioManagerService to mediate user land application's request to access geo location and phone ringer mode information. The architecture of the prototype is shown in Figure 2. The arrows in the diagram denotes the steps involved and the information flow among the components.

The Location Manager typically allows harnessing location information from multiple sources including GPS, network services and passive mode. Passive mode is a special piggybacking mode which allows an application requesting geocoordinates to snoop for the location information requested by some other application running on the device. Restricting access to locational information under passive mode is complicated by the fact that the requesting applications do not directly ask for the information from the LocationManagerService.

```
[ShareMockGPSSimple:
    (?user ex:systemUser ?someValue)
->
    (?requester ex:shareMockGPSCoordinates "True") ]
```

Fig. 3. Share actual or false location depending on requester

```
[ShareMockGPSComplex1:
    (?user ex:systemUser ?someValue)
    (?someActivity platys:occurs_at ?userPlace)
    (?userPlace platys:has_location ?userLocation)
    (?userLocation platys:part_of ?userBuilding)
    (?userBuilding rdf:type platys:Building)
     equal(?userBuilding, platys:BuildingABC)
->
    (?requester ex:shareMockGPSCoordinates "True") ]
```

Fig. 4. Policy to share a false location if user is inside BuildingABC

The AudioManagerService exposes a *setRingerMode()* API for programmatically setting device under normal, vibrate or silent modes. Thus modification of AudioManagerService allowed us interception of user's request and acting according to the device context. The prototype implements a simple work flow based on inferencing of users context by the semantic policy driven reasoning framework.

In our experiments, we asked the reasoning service to infer the current location information to be broadcasted across the Android framework. The reasoning service was supplied with two different facts representative of two different applications running on the device: one with privileges to access current location and the other one without the required privileges. The first requester is returned a false location in the San Francisco area while the other requester is provided with the actual device location of Baltimore.

One interesting observation is that the current implementation does not take care of cases when subsequent randomized coordinates are too far apart to raise suspicion of the intended recipient of information. For example if the device location is shown as Hong Kong and Chicago in two different readings taken few minutes apart then the recipient will clearly figure out that device location is masked. Two figures listed below shows the results rendered on the mobile screen. The policy used by the reasoner in this case is a fairly simple one which uses only the requester identity for resource sharing decision making. The reasoner maps user identity to a predefined group and infers from the groups allocated privileges. The policy used in the experiment is shown in Table 3. But the current implementation can be augmented to handle other more complex policies such as "share a false location to the requester if the device is in BuildingABC". The policy for such a scenario is shown in the Table 4.

V. Conclusion

Our implementation does not reduce the Android security. For each requested access by an application to system service or system resources, our implementation only introduces further checks depending on the active policies. However, each access that is not denied by the reasoner is passed on to the Android Permission Check and not influenced by the current implementation anymore. As a result, our implementation can only reduce the number of accesses allowed, not reducing the security. Finally, we underline that an adversary (either a user or an application) cannot skip the enforcement put forward by the implementation. We remind that our implementation is essentially an extension of Android and it runs with the privileges of the Android Middleware. The only part of our implementation that lies outside the middleware is the reasoner. The adversary cannot influence the context activation since the values considered for all context collection operation (e.g. current time) are taken directly from the underlying Android system. In order to avoid the adversary modifying the operating system of the phone (drivers and the prototype included) Trusted Computing mechanisms leveraging Trusted Platform Module can be used. However, the discussion of these mechanisms is outside the scope of this paper.

Thus in this paper we show how by embedding semantically rich policies based on device context in the smartphone' framework, user privacy can be protected at runtime as opposed to the current generation of smart phones where application's runtime privileges are decided on the basis of install time user input. We intend to test and improve the prototype to handle complex policies similar to the one we touched upon in the prototype architecture discussion and we plan to make other underlying services on smart phone context aware to facilitate finer degree of privacy preservation.

Acknowledgment

This research was partially supported by the National Science Foundation (award 0910838) and the Air Force Office of Scientific Research (grant FA550-08-0265).

References

[1] E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing, "Mixed membership stochastic blockmodels," *J. Mach. Learn. Res.*, vol. 9, pp. 1981–2014, Jun. 2008.

[2] D. Ashbrook and T. Starner, "Using GPS to learn significant locations and predict movement across multiple users," *Personal Ubiquitous Computing*, vol. 7, no. 5, pp. 275–286, Oct. 2003.

[3] "Jawbone with MotionX technology," http://content.jawbone.com/static/www/pdf/press-releases/up-press-release-110311.pdf.

[4] C. R. Mulliner, "Security of smart phones - masters thesis," Master's thesis, Department of Computer Science, University of California, Santa Barbara, 2006.

[5] Google, "Android reference. developers guide." http://developer.android.com/guide/index.html.

[6] Google.com, "Android security reference," http://source.android.com/tech/security.

[7] B. Jesse, "Android security reference," http://www.blackhat.-com/presentations/bh-usa-09/BURNS/BHUSA09-Burns-AndroidSurgery-PAPER.pdf.

[8] G. R. Hayes, S. N. Patel, K. N. Truong, G. Iachello, J. A. Kientz, R. Farmer, and G. D. Abowd, "The personal audio loop: Designing a ubiquitous audio-based memory aid," in *Proc. Sixth International Symposium on Mobile Human-Computer Interaction*. Springer Verlag, 2004, pp. 168–179.

[9] G. Iachello, K. N. Truong, G. D. Abowd, G. R. Hayes, and M. Stevens, "Prototyping and sampling experience to evaluate ubiquitous computing privacy in the real world," in *Proc. SIGCHI conference on Human Factors in computing systems*. New York, NY, USA: ACM, 2006, pp. 1009–1018.

[10] H. Chen, F. Perich, T. Finin, and A. Joshi, "SOUPA: Standard Ontology for Ubiquitous and Pervasive Applications," in *First International Conference on Mobile and Ubiquitous Systems: Networking and Services*, Boston, MA, August 2004.

[11] P. Jagtap, A. Joshi, T. Finin, and L. Zavala, "Preserving privacy in context-aware systems," in *Proc. 2011 IEEE Fifth International Conference on Semantic Computing*. Washington, DC, USA: IEEE Computer Society, 2011, pp. 149–153.

[12] A. R. Beresford, A. Rice, N. Skehin, and R. Sohan, "Mockdroid: trading privacy for application functionality on smartphones," in *Proc. 12th Workshop on Mobile Computing Systems and Applications*. New York, NY, USA: ACM, 2011, pp. 49–54.

[13] M. Ongtang, S. McLaughlin, W. Enck, and P. McDaniel, "Semantically rich application-centric security in android," in *2009 Annual Computer Security Applications Conference*. IEEE, 2009.

[14] N. M. Sadeh, "A semantic web environment for context-aware mobile services," in *Proc. Wireless World Research Forum*, 2001.

[15] N. M. Sadeh, T.-C. Chan, L. Van, O. Kwon, and K. Takizawa, "A semantic web environment for context-aware m-commerce," in *Proc. Fourth ACM Conference on Electronic commerce*. New York, NY, USA: ACM, 2003, pp. 268–269.

[16] F. Gandon, "Semantic web technologies to reconcile privacy and context awareness," *Web Semantics Science Services and Agents on the World Wide Web*, vol. 1, pp. 241–260, 2004.

[17] M. Shin, C. Cornelius, D. Peebles, A. Kapadia, D. Kotz, and N. Triandopoulos, "AnonySense: A system for anonymous opportunistic sensing," *Journal of Pervasive and Mobile Computing*, vol. 7, no. 1, pp. 16–30, February 2011.

[18] C. D. Kidd, R. Orr, G. D. Abowd, C. G. Atkeson, I. A. Essa, B. MacIntyre, E. Mynatt, T. E. Starner, and W. Newstetter, *The Aware Home: A Living Laboratory for Ubiquitous Computing Research*. Springer, 1999, pp. 191–198.

[19] V. Sacramento, M. Endler, and F. Nascimento, "A privacy service for context-aware mobile computing," in *First International Conference on Security and Privacy for Emerging Areas in Communications Networks*. New York, NY, USA: ACM, 2005, pp. 182–193.

[20] L. Kagal, T. Finin, and A. Joshi, "A policy language for a pervasive computing environment," in *Proc. 4th IEEE International Workshop on Policies for Distributed Systems and Networks*. Washington, DC, USA: IEEE Computer Society, 2003, pp. 63–.

[21] ——, "A Policy Based Approach to Security for the Semantic Web," in *Second International Semantic Web Conference*, September 2003.

[22] L. Kagal and T. Berners-Lee, "Rein : Where policies meet rules in the semantic web," Massachusetts Institute of Technology, Tech. Rep., 2005.

[23] T. Berners-Lee and D. Connolly, "Notation3 (N3): A readable RDF syntax," W3C, Tech. Rep., 2008.

[24] A. Khalil and K. Connelly, "Context-aware telephony: privacy preferences and sharing patterns," in *Proc. 2006 20th anniversary conference on Computer supported cooperative work*. New York, NY, USA: ACM, 2006, pp. 469–478.

[25] 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 *Proc. 9th USENIX conference on Operating systems design and implementation*. Berkeley, CA, USA: USENIX Association, 2010, pp. 1–6.

[26] M. Conti, V. T. N. Nguyen, and B. Crispo, "Crepe: Context-related policy enforcement for android," in *ISC*, ser. Lecture Notes in Computer Science, M. Burmester, G. Tsudik, S. S. Magliveras, and I. Ilic, Eds., vol. 6531. Springer, 2010, pp. 331–345.

[27] L. Zavala, R. Dharurkar, P. Jagtap, T. Finin, and A. Joshi, "Mobile, Collaborative, Context-Aware Systems," in *Proc. AAAI Workshop on Activity Context Representation: Techniques and Languages*, AAAI. AAAI Press, August 2011.

[28] A. D. Team, "Androjena - Jena Android port," http://code.google.com/p/androjena/.

[29] Apache.org, "Java framework for building semantic web applications," http://incubator.apache.org/jena/.

[30] cyanogenmod.com, "Nexus one: Full update guide," http://wiki.cyanogenmod.com/wiki/Nexus_One:_Full_Update_Guide.

[31] android.com, "Android 2.3 platform," http://developer.android.com/sdk/android-2.3.html.