CERIAS Tech Report 2011-13

Intuitive security policy configuration in mobile devices using context profiling

by Aditi Gupta, Markus Miettinen, N. Asokan
Center for Education and Research
Information Assurance and Security
Purdue University, West Lafayette, IN 47907-2086

Intuitive security policy configuration in mobile devices using context profiling

Aditi Gupta
Department of Computer Science
Purdue University, United States
aditi@purdue.edu

Markus Miettinen
Nokia Research Center
Lausanne, Switzerland
markus.miettinen@nokia.com

N. Asokan
Nokia Research Center
Radio Systems Laboratory, Finland
n.asokan@nokia.com

***Abstract*—Configuring access control policies in mobile devices can be quite tedious and unintuitive for users. Software designers attempt to address this problem by setting up default policy configurations. But such global defaults may not be sensible for all users. Modern smartphones are capable of sensing a variety of information about the surrounding environment like Bluetooth devices, WiFi access points, temperature, ambient light, sound and location coordinates. We claim that profiling this type of contextual information can be used to infer the familiarity and safety of a context and aid in access control decisions. We propose a context profiling framework and describe device locking as an example application where the locking timeout and unlocking method are dynamically decided based on the perceived safety of current context. We report on using datasets from a large scale smartphone data collection campaign to select parameters for the context profiling framework. We also describe a prototype implementation on a smartphone platform.**

***Keywords*-context awareness, access control, human computer interaction**

I. Introduction

Smartphones are fast becoming an integral part of life for many users. They are used for performing everyday tasks like emails and Internet banking that involve storing sensitive data on the device. They also contain personal data like photos and videos, communication logs, location information and logs of monetary transactions. This calls for strong protection mechanisms on mobile devices. Protection mechanisms serve their purpose only when they are configured with sensible policies for accessing and sharing data. However, managing a large number of policy configurations can be quite overwhelming and unintuitive for a user. Application and service designers attempt to tackle the usability problem by providing users with a default policy configuration. But a global default policy may not be suitable for the needs of every user. Users are therefore left with two unsatisfactory alternatives: either use one-size-fits-all default policies which may not be sensible, or, suffer through manually configuring the bulk of policies by hand which may not be intuitive or easy-to-use.

Modern smartphones are equipped with a variety of sensors capable of continuously monitoring a wide range of parameters such as location, Bluetooth and WiFi devices in the neighborhood, temperature, ambient light, noise levels etc. These observations characterize the *context* of a device, and hence of its user. We argue that by *profiling contexts* in terms of how the context parameters change over time, we can infer appropriate access and sharing policies for sensitive data on the device, which can help towards at least partially automating the process of setting sensible policies.

As an illustrative example, consider the case of *device locks*: smartphones and other mobile devices have a device lock feature similar to the screen-saver lock on PCs. When the device has been idle for a pre-defined fixed period of time, the device lock kicks in. Thereafter the user has to unlock the device before accessing the applications and data on the device. A device may support multiple unlocking methods like a slider or passcode entry but a specific unlocking method has to be chosen when the device lock feature is enabled. In an enterprise, the enterprise system administrator may force its users to use strong device lock if the device is capable of accessing enterprise data like corporate e-mail or intranet websites. Suppose a user, say Alice, finds it very inconvenient having to type in a passcode several times every day. She may decide to disable the device lock and risk the compromise of her sensitive data like e-mails, and she may opt to remove applications like corporate e-mail that mandate the use of device lock.

Alice's experience with device lock can be significantly improved by making the device lock to adapt its behavior based on the context. Instead of having a fixed pre-defined timeout for the device lock to kick-in and always using the same unlocking method, the device lock application could use dynamic configuration of these parameters depending on the device context. For example, in a safe and familiar place like her home where the likelihood of the device being stolen is low, Alice would like to have a long timeout, and a "shallow" unlocking method like a slider (that does not tax her too much), whereas in an unfamiliar place she would be willing to live with a very short timeout and a "deeper" unlocking method like passcode entry. The question then is "how can the device estimate the familiarity and safety at any given time?" We propose applying the following context profiling approach to address this question: the device periodically scans its environment for a variety of context variables like GPS readings, WiFi access points and Bluetooth devices. Based on these scans, the device can

- discover contexts which the device encounters repeatedly; these are likely personal *contexts of interest (CoIs)* for the user.
- profile the CoIs by keeping track of which WiFi and Bluetooth devices are encountered in a given CoI and the nature of those encounters. These profiles can be used to estimate the *familiarity* of a device with respect to a context. The inferred device familiarity values can then be used to estimate the familiarity of a context itself.
- use current and historically aggregated context familiarity information to estimate the safety of the current context.

This basic approach needs to be complemented by allowing the user to provide feedback about the perceived safety of a context. This is important in two respects: the user may want to speed up the learning process or may want to correct incorrectly inferred safety levels. Users may provide two types of feedback about the perceived safety of a context: *short-term* feedback should be treated so that its effect wears off quickly while *long-term* feedback should have a more long-lasting effect.

In this paper, we describe the design of a context profiling framework to intuitively infer sensible access policies without user intervention, while still allowing corrective user feedback. We use the device lock scenario as an example of applying our context profiler. We describe the implementation architecture for the context profiler. We then describe several experiments using a previously available dataset based on which we select concrete parameters for our prototype implementation of the context profiler. We then provide an evaluation of our model and discuss possible enhancements to our approach.

II. Concepts and Design

A. Context profiling

1) Detecting CoIs: In this paper, we limit our scope to geolocational contexts only. We use a grid-based clustering algorithm for GPS observations to detect CoIs, which are regions where the device has been present sufficiently often. A CoI is represented by a circular region with a fixed radius centered at the centroid of the locational observations contributing to the CoI. Once a CoI is detected, we update its centroid with every new observation that falls within the CoI.

2) Notions of familiarity: A user may observe certain devices more often than others in a given CoI. These devices gradually become familiar to the user's device with respect to that particular CoI. We introduce the notion of familiarity of a device in a given CoI (hereafter *device familiarity*) as a measure of how frequently and how recently a device has been observed by the user's device in a given CoI. If a familiar device stops appearing in a CoI for a long time, its device familiarity should gradually decrease. Since we do not know if the device has left the CoI permanently or is temporarily absent, the device decay should be slow and gradual. This is achieved by growing the device familiarity of device d in CoI $\mathcal{C}$ with every observation of $\mathcal{C}$ that includes d, but decaying d only if it has not been observed in N_0 successive observations of $\mathcal{C}$, where N_0 is a suitably chosen constant.

Definition 1: **Device familiarity** of a device d, with respect to CoI $\mathcal{C}$ after n observations of $\mathcal{C}$ is defined as:

$$\begin{aligned}&\text{DFam}(d, \mathcal{C}, n) = \\ &\quad \alpha_D * \text{occ}(d, \mathcal{C}, n) + (1 - \alpha_D) * \text{DFam}(d, \mathcal{C}, n-1)\end{aligned}$$

where,

$$\text{occ}(d, \mathcal{C}, n) = \begin{cases} 1 & \text{if } d \text{ is observed in } \mathcal{C} \\ & \text{in the } n^{\text{th}} \text{ sample,} \\ 0 & \text{if } d \text{ is not observed in } \mathcal{C} \\ & \text{in the } n^{\text{th}} \text{ sample, and} \\ & (n - N_{last}) \mod N_0 = 0. \\ \text{DFam}(d, \mathcal{C}, n-1) & \text{otherwise.} \end{cases} \quad (1)$$

N_{last} (defaulting to 0) is the ordinal number representing the last sample of $\mathcal{C}$ in which d was seen and $0 \leq \alpha_D \leq 1$ is a suitably chosen constant.

The selection of the smoothing factor α_D determines the weight $1 - \alpha_D$ assigned to the old device familiarity value in computing the new device familiarity value. For example, for a device present in every observation made in a CoI, higher values for α_D would imply quicker rise in the device familiarity value.

We estimate the familiarity of a CoI using two measures: *instantaneous familiarity* and *aggregate familiarity*. Instantaneous familiarity is an estimate of the familiarity of the CoI the device is currently in, in terms of the device familiarity values of the devices present in the CoI at that instant. Aggregate familiarity represents the "usual" or "typical" familiarity of a CoI over time.

Instantaneous familiarity is computed as a weighted average of the observed devices with their device familiarity values constituting the corresponding weights. The intuition is that the contribution of a device towards instantaneous familiarity of a CoI should be proportional to its device familiarity in that CoI. We compute the instantaneous familiarity separately for each class of devices and combine them by taking the average over all device classes. Currently we consider two classes: Bluetooth devices and WiFi devices.

Definition 2: **Instantaneous familiarity** of a CoI $\mathcal{C}$ at its n^{th} observation can be defined as

$$\text{instFam}(\mathcal{C}, n) = \frac{1}{c} \sum_{i=1}^{c} \text{instFam}_c(\mathcal{C}, n) \quad (2)$$

where

$$\text{instFam}_c(\mathcal{C}, n) = \frac{1}{|D_{\mathcal{C},n}|} \sum_{d \in D_{\mathcal{C},n}} \text{DFam}(d, \mathcal{C}, n)$$

and $D_{\mathcal{C},n}$ is the set of devices of class c observed in $\mathcal{C}$ at its n^{th} observation.

Aggregate familiarity of a CoI is computed as an exponential moving average of instantaneous familiarity.

Definition 3: **Aggregate familiarity** of a CoI $\mathcal{C}$ after n observations of $\mathcal{C}$ is defined as:

$$\text{aggFam}(\mathcal{C}, n) = \alpha_C * \text{instFam}(\mathcal{C}, n) + (1 - \alpha_C) * \text{aggFam}(\mathcal{C}, n-1) \quad (3)$$

where $0 \leq \alpha_C \leq 1$ is a suitably chosen constant.

The smoothing factor α_C determines how fast the aggregate familiarity should react to the changes in instantaneous familiarity. A higher value implies quicker reaction. In section IV we discuss the choice of α_D and α_C values used in equations 1 and 3.

3) Notion of Null device: We model the absence of any other device by introducing the notion of a *null device* for each class of devices. A null device is observed when no other device in that device class is observed. The device familiarity of a null device is computed in just the same way as for a real device using equation 1. Thus, in CoIs where absence of other devices is the norm, the null devices will have a high device familiarity which in turn leads to familiarity of the CoI to be high when no devices are present. On the other hand, in other CoIs, the familiarity of the null device will be low which causes the familiarity of the CoI to drop when no other devices are observed.

4) Inferring context in absence of GPS fix:

In the absence of GPS fix, we still need to infer the current context since access control has to be enforced. In such scenarios, we can utilize hints from the currently observed devices to infer the user's context. The snapshot of WiFi devices observed in a CoI is fairly static and can be used to attribute user's current position to a known CoI. We also leverage the fact that the instantaneous WiFi familiarity score of a CoI represents how familiar the current snapshot of WiFi devices is to this CoI to map a WiFi snapshot to its most familiar CoI.

Inference of user's context is done in two steps. First, we compute candidate instantaneous WiFi familiarity for the current snapshot of WiFi devices with respect to all known CoIs for the user. We use a minimum threshold for WiFi instantaneous familiarity to discard obviously incorrect CoI choices. The current position is then attributed to the CoI with maximum WiFi instantaneous familiarity score. If none of the candidate instantaneous familiarity scores exceed the minimum familiarity threshold, we use Jaccard's distance measure to compute the distance between the current snapshot of WiFi devices and the snapshot of WiFi devices corresponding to the last known observation with an associated GPS reading. If the two WiFi snapshots are close enough, we attribute the current observation to the same location.

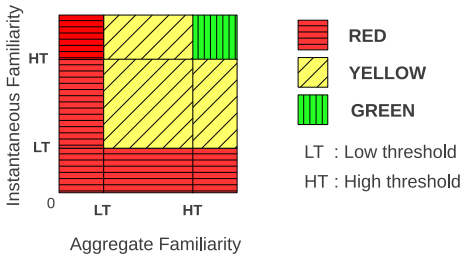

Figure 1. Familiarity-to-safety mappings

5) From familiarity to Safety: Familiarity can have different interpretations in terms of safety for different applications. How best to infer the safety level from the familiarity estimates is a difficult question. A familiar place may be considered safe by a certain application, and unsafe by some other application. For example, applications where anonymity is desired would treat a familiar place as unsafe and an unfamiliar place as safe. On the other hand, a configurable device lock mechanism would treat an unfamiliar place as unsafe.

We propose a familiarity to safety mapping for device lock and other applications with similar requirements. For device lock, we need to assess the safety level of the current context of the device so that the appropriate locking timeout and unlocking method can be enforced. We link familiarity to safety using the rationale that familiar devices can be associated with individuals who are usually present in a context and therefore do not represent a high level of risk for the user. We begin with the following intuition: a CoI that has a high familiarity both typically and currently is probably safe; as a dual, a CoI that has a low familiarity both typically and currently is probably unsafe.

We incorporate the above observations in our algorithm to estimate the safety level of the current context (Figure 1). The algorithm uses the instant and aggregate familiarity of the current CoI to estimate the safety level as one of high (GREEN), medium (YELLOW) or low (RED). To do this, we use two thresholds: a high familiarity threshold (HT) and a low familiarity threshold (LT) to delimit "high" and "low" values for familiarity (both instantaneous and aggregate). In section IV, we estimate reasonable values for these thresholds.

If the current context does not correspond to a CoI, we conclude that the safety level is low (RED). This is consistent with algorithm in Figure 1 because the aggregate familiarity of an unknown context is zero.

B. Handling user feedback

In automated access control enforcement, it is important to incorporate feedback from the user in the decision making process. Since our context profiler's safety algorithm ultimately bases its computations only on a few classes of sensor inputs, it may sometimes estimate the safety level incorrectly. User feedback is important in such cases so that the inferencing process can be tweaked to match user's expectations. Similarly, user feedback can be used to shortcut the learning process so that contexts that the user knows will become eventually familiar (like her home) can be deemed familiar more quickly.

A user can provide feedback by specifying the safety level of a context as perceived by him. The user may provide feedback on the long-term behavior of a CoI by marking it as *'Usually safe'* or *'Usually unsafe'*. Alternatively, he may want to indicate a short-term or temporary feedback like *'Now safe'* or *'Now unsafe'* for the current CoI. When a user provides 'Usually safe' feedback for the current CoI, he is also prompted to provide 'Now Safe' feedback, if appropriate. This is helpful in providing a quick boost to the short term safety value.

Feedback classification: The safety feedback provided by a user can be broadly classified into following categories:

1) *Learning phase feedback:* This feedback is provided by the user during the learning phase to shortcut learning, or to override the context profiler's estimations of the safety of a context until that context has been learned. We believe that this would be the most frequent case where user will provide feedback. A user may provide either short term or long term feedback during learning phase.
2) *Affirmative feedback:* This refers to the scenario where the user feedback matches the context profiler's perception of safety, that is the user just re-affirms the context profiler's perception. For example, when user says that certain context is *'Usually safe'* and the context profiler has already inferred the context as safe. We can safely ignore this feedback in the computation of safety scores.
3) *Corrective feedback:* This refers to the scenario when the context profiler fails to match user's perception of safety even after it has learned the context. Corrective feedback can be either short term or long term.

We base our feedback handling approach on the following two principles:

1) The effect of feedback should be immediately visible to the user. However, it should not permanently relax the safety computations, but allow for the system to react in case of sudden drops in familiarity scores.
2) When a user provides feedback, it is regarding the safety of a context and not its familiarity. Thus, the feedback handling mechanism should only tweak the familiarity to safety mapping and not the familiarity scores themselves.

Long term feedback reflects on the 'typical' behavior of a context. Our intuition is that such feedback would be provided in the learning phase to shortcut the learning process. The effect of long-term feedback should correct the safety computations until the context has been properly learned. This can be achieved by combining long term feedback and the aggregate familiarity using a dynamic weight w_{LT} that gradually fades away. We use a time decay curve to decay the value of w_{LT}.

Definition 4: **LT_Score** is the long term safety score that replaces the aggregate familiarity score in algorithm in Fig 1. It is computed as:

$$\text{LT\_ Score} = (1-w_{LT}) * \text{aggFam}(\mathcal{C}, n) + w_{LT} * \text{LT\_Feedback} \tag{4}$$

where LT_Feedback indicates long term feedback, with value either 0 (*'Usually unsafe'*) or 1 (*'Usually safe'*).

The dynamic weight w_{LT} for long term feedback is computed as:

$$w_{LT} = \begin{cases} 1 - (\frac{n_f}{N_f})^c & \text{if } n_f \leq N_f \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

where n_f is the number of observations since the long term feedback was given, N_f is the maximum number of observations after which the feedback effect should wear off and c is a constant that determines the speed of decay.

The long term feedback weight should decay slowly in the beginning so that the device has enough time to learn the context and then gradually fade away to 0. The constant N_f is decided based on the length of learning period, which depends on the α_C and α_D values (refer section II-A2).

Short term feedback reflects on the safety of current snapshot of a CoI. It indicates a temporary change in the behavior of a CoI and should fade away after a short time. We compute this score by combining short term feedback and instantaneous familiarity using a dynamic weight w_{ST}.

Definition 5: **ST_Score** is the short term safety score that replaces the instantaneous familiarity score in algorithm in Fig 1. It is computed as:

$$\text{ST\_ Score} = (1-w_{ST}) * \text{instFam}(\mathcal{C}, n) + w_{ST} * \text{ST\_Feedback} \tag{6}$$

where ST_Feedback indicates short term feedback, with value either 0 (*'Now unsafe'*) or 1 (*'Now safe'*).

The short term dynamic weight w_{ST} should depend on the time elapsed and the change in the snapshot of observed devices since the feedback was given.

$$w_{ST} = \begin{cases} 1 - \max\{\frac{t-t_0}{t_{max}-t_0}, \text{Dist}(S, S_r)\} & \text{if } t \leq t_{max} \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

where t_0 is time at which short term feedback was given, t is the current time, t_{max} is time after which short term

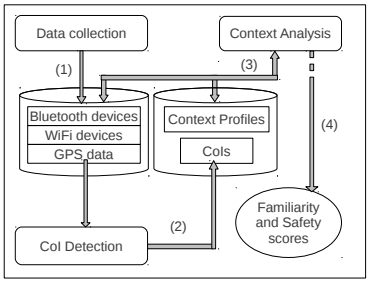

Figure 2. System components: (1) Data collection module collects GPS, Bluetooth, WiFi data; (2) GPS data is clustered to detect CoIs; (3) Context analysis module updates context-specific information and (4) computes familiarity and safety scores for the current context.

effect should wear off (we use $t_{max} = 60$ mins.), S_r is the snapshot of devices at time t_0, S is snapshot of devices at time t and Dist() is the distance metric, the definition of which is based on the following rationale:

- Familiar devices in S_r, but not S should increase the distance measure
- Unfamiliar devices in S but not in S_r should increase the distance measure
- Unfamiliar transient devices S_r, but not in S should not increase the distance measure
- Familiar devices in S but not in S_r should not increase the distance measure

Let S correspond to the n^{th} (i.e., current) observation and $\text{occ}(d, S1, S2) = 1$ if device $d \in (S1 - S2)$ and 0 otherwise. Then we define[1]

$$\text{Dist}(S_r, S) = \frac{\left\{ \begin{array}{l} \sum_{d_i \in S_r} \text{DFam}(d_i, \mathcal{C}, n) * \text{occ}(d_i, S_r, S) \\ + \sum_{d_i \in S} (1 - \text{DFam}(d_i, \mathcal{C}, n)) * \text{occ}(d_i, S, S_r) \end{array} \right\}}{|S_r \cup S|} \quad (8)$$

The effective safety level is inferred using Figure 1 where `ST_Score` and `LT_Score` will serve the purpose of instantaneous and aggregate familiarity respectively.

III. Implementation Architecture

The system architecture for the context profiler software is described in Figure 2. It consists of three main modules:

- **Data Collection** module is responsible for continuously sensing the current context and collecting raw data about various context variables
- **CoI Detection** module periodically clusters the location data collected by the *data collection module* to detect CoIs for the user, based on their significance to the user which is determined by the amount of time the user spends in a particular place.
- **Context Analysis** module is responsible for analyzing the raw data and infer familiarity and safety scores for the current context. For each CoI, it maintains a context profile to keep track of the devices that are observed in a CoI and their familiarity scores with respect to that CoI. Based on the current snapshot of the CoI, it computes instantaneous and aggregate familiarity scores using equations 2 and 3 respectively. These familiarity scores are used to infer the safety of the context as discussed in section II-A5.

In our current implementation [1], the *data collection module* scans the environment every five minutes to record the GPS co-ordinates (if available) as well as the currently visible Bluetooth devices and WiFi access points. This information is stored in a database on the device itself and is used by other modules to identify and analyze CoIs. This module can be extended to sense other kinds of context variables.

For CoI detection, we used a simple grid-based clustering algorithm with a grid cell width of 250 meters. We required a cluster to have at least 1% of all observations within a time window of 30 days which corresponds to 8640 observations at our current rate of sampling. Consequently, the detection threshold of 1% ($\approx$ 86 observations) would correspond to roughly an equivalent of seven hours of observations of a place in the GPS trace data for the place to become identified by our clustering algorithm as a CoI. Using these clusters as the CoIs, we associated Bluetooth and WiFi observations having an associated GPS fix within 100 meters from a cluster's centroid as belonging to that CoI.

The *context analysis module* periodically generates the familiarity and safety scores for the current context. These values can be used by applications to automatically configure access policies that depend on the current context. In the device locking use case, the safety scores are used to dynamically configure the unlocking method and the locking timeout of the device.

IV. Determining Parameters

In order to determine suitable parameters for the Context Profiling framework described in Section II-A, we ran several experiments using traces from the Lausanne Data Collection Campaign, a large-scale data collection experiment focusing on mobile device users' behavioural and contextual data traces [2], [3]. The dataset contained the GPS location traces and regular scans of the WiFi and Bluetooth radio environments of a large number of users.

To match our device implementation as closely as possible, we filtered the dataset to include one Bluetooth and WiFi scan observation per five-minute observation window, if available. Each of these Bluetooth/WiFi observations was matched with the closest GPS fix within the time window, if available. By applying our CoI identification algorithms, we

[1]We could define $Dist()$ simply as the Jaccard distance $J_\delta(S_r, S)$, but that will not distinguish devices based on familiarity.

identified a total of 167 CoIs for 37 users, giving on average 5.22 CoIs per user (median 5 CoIs).

In the device lock scenario, the context profiler effects visible to the end user are (a) how long does it take for a safe CoI to be recognized as such by the context profiler and (b) how volatile is the safety labeling of a safe CoI. As a guiding principle, we want the context profiler to learn safe CoIs within two days. At our current sampling frequency of every five minutes, a day consists of 288 observations. We conjectured that a user is likely to spend about a third of a day in a given safe CoI. Thus we need safe CoIs to be deemed safe in about 200 observations. We set this as our approximate target.

Selecting α_D: From Equation 1 we see that higher values for α_D will imply that the device familiarity $DFam$ will grow quickly if a device continues to appear in successive samples in a CoI. Given our rough target of recognizing a safe CoI within 200 observations, we decided to select α_d so that a device that appears in about 20 consecutive samples of a CoI would have a $DFam$ reaching 0.9. Using Equation 1, we compute this value of α_D to be 0.1. This is in line with the standard practice of choosing a smoothing factor between 0.05 and 0.3 for processes that are locally constant (Chapter 8 of [4]).

Selecting N_0: To select the value of N_0 in equation 1, we reasoned that the familiarity of a device should decay if it did not show up even once in consecutive samples spanning a day. Again, based on the assumption that a user may spend about a third of a day ($\approx$ 96 observations) in a given safe CoI, we chose N_0 to be 100.

Selecting α_C: In Equation 3 the smoothing factor α_C affects the lag time of the smoothing applied to the aggregate familiarity scores. The lag time determines the number of observations required for the aggregate familiarity score to react to changes in the trend of the instantaneous familiarity scores. Consequently, the choice of α_C will impact both the user-visible effects discussed above.

We presume that most users have at least two frequently visited CoIs (e.g. their home and workplace). We further assume that the majority of such CoIs can be presumed to be 'familiar' places for the users. We denote the set of the top-two most frequently observed CoIs of each user as the set of *frequent CoIs*. We studied how different choices of α_C affects the evolution of the aggregate familiarity score in frequent CoIs over time. Figure 3 shows the result: the y-axis on the left shows the average aggregate familiarity scores for frequent CoIs; the y-axis on the right shows the average of the standard deviation of the aggregate familiarity scores of the same, calculated over the latest 100 observations at each point.

We observed the following from Figure 3:

- values of α_C greater than 0.05 have little impact in the behavior of the average aggregate familiarity score.
- the "knee" in the graph near the 200th observation implies that most of the frequent CoIs reach a steady state after this point.
- the average standard deviation of aggregate familiarity scores is reasonably small (less than 5%) for all values of α_C less than 0.05 beyond the steady state.

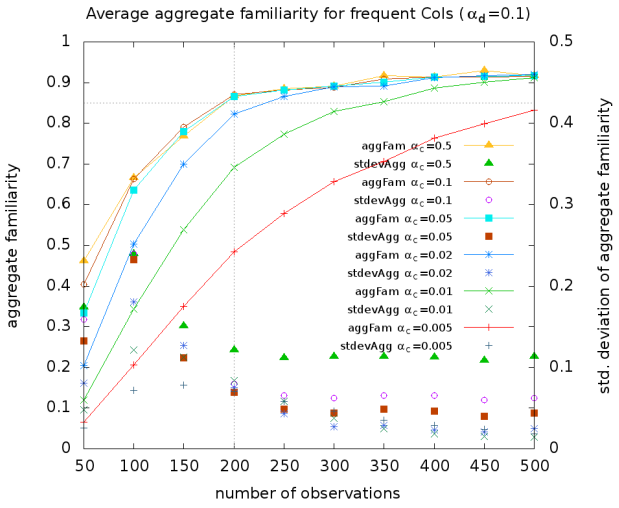

Figure 3. Behavior of aggregate familiarity score in frequent CoIs

Based on these results, we chose 0.05 as the value for α_C.

Selecting N_f: The number of observations for frequent CoIs to reach steady state (200) is a suitable value for N_f in equation 5.

Selecting HT **and** LT: In Figure 1, a natural value for HT is the point reached by the average aggregate score of frequent CoIs at the steady state. From Figure 3, this is 0.85. To choose the value of the low threshold LT, we used the following rationale. We expect that for most users, a familiar CoI like home will exhibit stable behavior in the long-term. Thus we can choose LT such that the aggregate familiarity score of most familiar CoIs will be above this value. We resort to an 90-10 rule of thumb to assume that 90 percent of the set of frequent CoIs are likely to be stable. Figure 4 shows the the aggregate familiarity score of the CoI at the lowest tenth percentile for a given number of observations. From the graph, we can see that 0.4 appears to be a good choice for LT because at all times after reaching the steady state (refer to Figure 3), all frequent CoIs in the set above the 10^{th} percentile have aggregate familiarity scores above this value.

V. Validation of the Model

A. Comparison to ground truth

Once the parameters were determined, we applied our familiarity and safety algorithms to the observation data related to the frequent CoIs of each user. Figure 5 shows the distributions of the classifications of individual observations in all CoIs. These are sorted in descending order of number

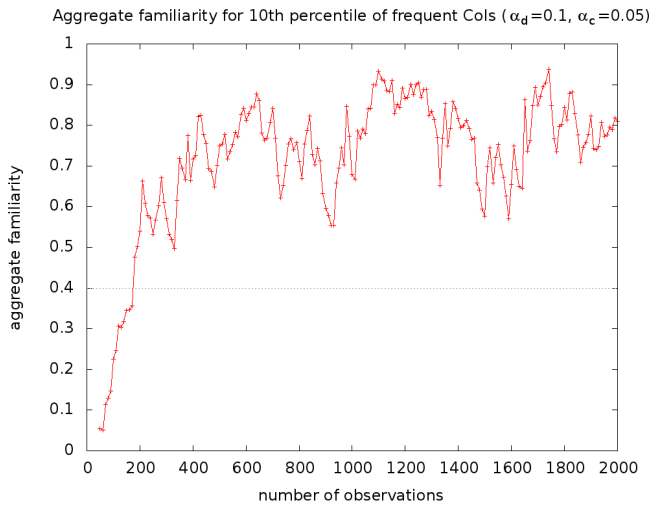

Figure 4. Determining the low threshold

of observations associated with the CoI. As can be seen, the familiarity scores of CoIs which do not have enough observations are pre-dominantly flagged as "unsafe". As the number of observations grow an increasingly greater proportion of the observations are flagged as "safe".

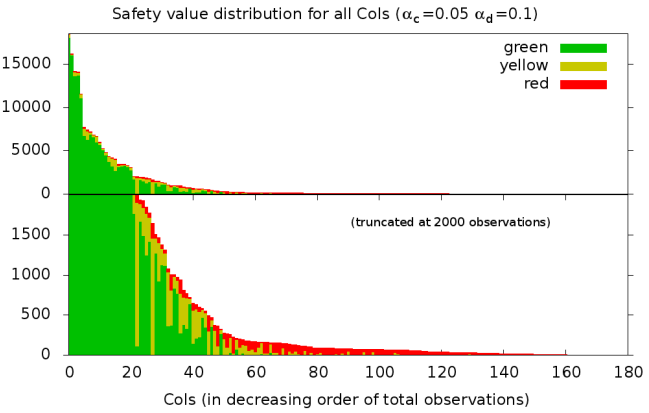

Figure 5. Distribution of safety classifications of observations for different CoIs

Ideally, the evaluation of the model would be based on ground truth information indicating the user's perception of the safety of a CoI over time. Unfortunately the dataset we used did not have ground truth information at this granularity. However, it did have information where the users have labeled locations using one of several pre-defined labels such as "My home", "My main work place", "Shop" etc. We grouped these labels into "safe" and "unsafe" as shown in Table I. We ignored locations with labels whose safety classification from a user's perspective is unclear (e.g., labels such as "Home of a friend").

Safe	*Unsafe*
My home	Holiday resort or vacation spot
My freetime home	Shop or shopping center
My main workplace	Location related to transportation (bus stop)
	Place for indoor sports (e.g. gym)
	Place for outdoor sports (e.g. walking)
Unclassified	
Home of a friend	My main school or college place
My other work place	Other
I don't know	

Table I
CLASSIFICATIONS OF PLACE LABELS IN GROUND TRUTH DATA

Making the simplifying assumption that the CoIs identified by the users as "safe" or "unsafe" in the ground truth data are *always* safe or unsafe respectively, we estimated the effectiveness of the context profiler with the parameters selected above as follows. We identified the sets as in Table II.

Sets in ground truth data		#
Observations in "Safe" CoIs	G_{safe}	51446
Observations in "Unsafe" CoIs	G_{unsafe}	2607
Observations in Unclassified CoIs	G_{UC}	10119
Sets identified by Context Profiler		#
"Safe" observations	C_{GREEN}	55234
"Unsafe"observations	C_{RED}	2862
Neither	C_{YELLOW}	6076
Set intersections		#
True "Safe" obs.	$\lvert G_{safe} \cap C_{GREEN} \rvert$	47197
Other "Safe" obs.	$\lvert \{G_{unsafe} \cup G_{UC}\} \cap C_{GREEN} \rvert$	8037
True "Unsafe" obs.	$\lvert G_{unsafe} \cap C_{RED} \rvert$	889
Other "Unsafe" obs.	$\lvert \{G_{safe} \cup G_{UC}\} \cap C_{RED} \rvert$	1973

Table II
SETS USED IN VALIDATION

We then calculated the following figures of merit for recognizing "safe" situations:

	Formula	*value*
Precision	$\frac{\lvert G_{safe} \cap C_{GREEN} \rvert}{\lvert C_{GREEN} \rvert}$	0.854
Recall	$\frac{\lvert G_{safe} \cap C_{GREEN} \rvert}{\lvert G_{safe} \rvert}$	0.917
Fallout w.r.t. "unsafe"	$\frac{\lvert G_{unsafe} \cap C_{GREEN} \rvert}{\lvert G_{unsafe} \rvert}$	0.152
Fallout w.r.t. "unclassified"	$\frac{\lvert G_{UC} \cap C_{GREEN} \rvert}{\lvert G_{UC} \rvert}$	0.755

and analogously the following for recognizing "unsafe" situations:

	Formula	*value*
Precision	$\frac{\lvert G_{unsafe} \cap C_{RED} \rvert}{\lvert C_{RED} \rvert}$	0.311
Recall	$\frac{\lvert G_{unsafe} \cap C_{RED} \rvert}{\lvert G_{unsafe} \rvert}$	0.341
Fallout w.r.t "safe"	$\frac{\lvert G_{safe} \cap C_{RED} \rvert}{\lvert G_{safe} \rvert}$	0.019
Fallout w.r.t "unclassified"	$\frac{\lvert G_{UC} \cap C_{RED} \rvert}{\lvert G_{UC} \rvert}$	0.096

The precision and recall of recognizing safe situations are sufficiently high. The fallout value reflecting the likelihood of unsafe CoIs receiving 'safe' classifications is slightly higher than desirable (15%), but still in acceptable range.

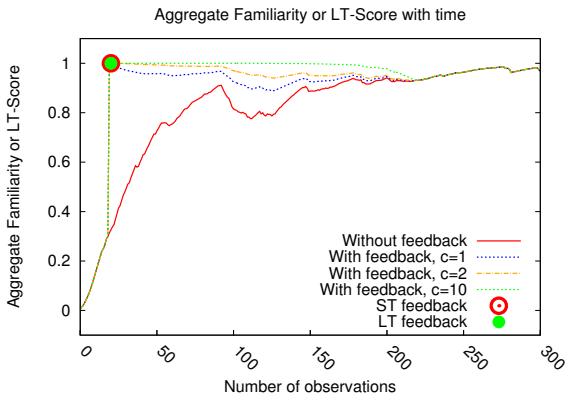

Figure 6. Effect of user feedback during learning

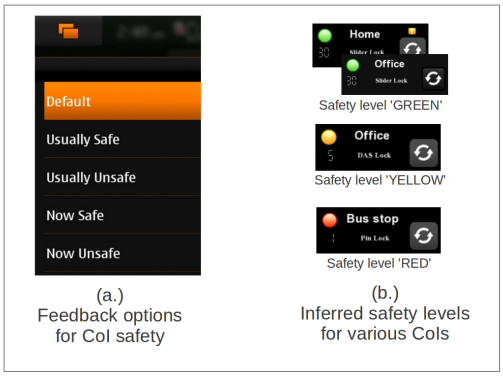

Figure 7. Device implementation: feedback options and inferred safety

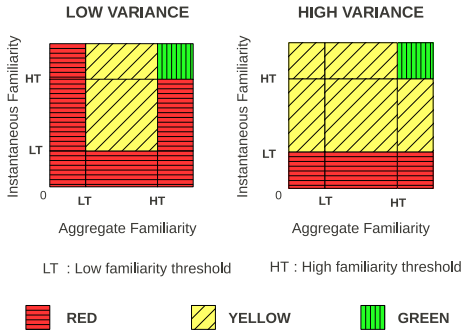

Figure 8. Safety algorithm with variance

The fallout with regard to 'unclassified' CoIs is remarkably high (75%). This may be caused by the fact that a major fraction of the CoIs in the 'unclassified' set G_{UC} actually represent places that are familiar to the user (e.g. 'Home of a friend',or, 'My other work place' might be such places). The precision of recognizing unsafe situations is low, but acceptable as it errs on the safe side. The recall is low, implying that the context profiler recognized only a third of the unsafe observations as such. However, among the 6076 YELLOW observations made by the context profiler (the set C_{YELLOW}), 1321 were in locations labeled as "unsafe" in the ground truth data. If we combine this set with C_{RED}, then the recall figure climbs up to 0.848. This suggests that the YELLOW safety level should not be considered significantly safer than RED. Overall, the figures of merit validate the choice of parameters.

B. Implementation

We implemented a prototype of the context profiler [1] with the chosen parameters on Nokia N900 Linux-based smartphones. We also implemented three different unlocking methods (passcode, draw-a-secret, and slider) which were linked to the RED, YELLOW, and GREEN safety levels respectively. The three safety levels also corresponded to three different default timeout values of 1 minute, 5 minutes and 30 minutes respectively. We use a *low watermark* approach to decide the unlocking method: if a device is locked in a safe context, a change in context can lock it deeper (i.e., requiring a stronger unlocking method), but the converse is not true. The unlocking method will correspond to the safety of the least safe context encountered since the device was locked.

C. Effect of user feedback

We studied the effect of user feedback using our prototype context profiler. The user can provide feedback about a CoI's safety at any time to modify its behavior using a GUI as shown in Figure 7(a.). Figure 6 shows the effect of 'Usually safe' feedback that is provided by the user for his 'Home' context when the context profiler is still in learning phase.

The graph shows the effect of using different c values in equation 5. While a bigger value of c provides a steady behavior until CoI has been learnt, it also reduces the CoI's tolerance to genuine drops in instantaneous familiarity. To address this tradeoff and from the behavior of the LT_Score in Figure 6, we decide to use $c = 2$.

VI. Discussion

Alternate safety algorithm

The safety algorithm discussed in Fig. 1 can be further strengthened by incorporating volatility of the CoI as a factor. CoIs that are stable (less volatile) should be less tolerant to changes in instantaneous familiarity. Even small changes should severely affect the perceived safety of a stable CoI. Similarly, CoIs with high variance should be more tolerant to fluctuations. The variance of instantaneous familiarity can be an indicator for the volatility of a CoI. A context can be deemed volatile if the variance is above a certain threshold. To incorporate volatility of a CoI in the safety algorithm, we use its modified version as shown in Fig 8. This algorithm is not used when the user feedback is in effect, since the volatility of the CoI cannot be reliably determined in such cases.

Privacy considerations

Collection of user's contextual data by different services usually raises *privacy* concerns. However, in our approach this data collection is used to help users in intuitive enforcement of access control and never leaves the device's storage.

Energy considerations

Continuous context profiling comes at a cost of increased *battery consumption*. This limitation can be overcome by using intelligent sampling techniques. For example, instead of performing frequent GPS scanning, one could use accelerometer triggered scanning so that GPS is turned on only when motion is detected. Another technique to conserve battery could be to use WiFi access points to detect geolocation instead of GPS. Our initial prototype does not incorporate these enhancements yet. However, intelligent sampling would be highly desirable in a usable product.

Security considerations

The *security* requirements of context profiling depends on the application. An attacker who can fake Bluetooth or WiFi addresses can influence the estimated familiarity scores. This can be addressed by revising the familiarity calculations by giving a greater weights to devices whose identities are cryptographically verifiable based on existing security associations with those devices. For the device lock application this is not a significant concern because we target users like Alice described in Section I who do not use any device lock in the first place. Compared to this starting point, if the use of the context profiler improves the perceived usability of device lock for such users, it can only improve the security!

Unknown contexts

In the current implementation, as discussed in Section II-A5, an unknown context is treated as unsafe. This is logical because there is no notion of aggregate familiarity for an unknown context. However, this approach may be too pessimistic: one can plausibly make the argument that the familiarity of an unknown context where all devices present are highly familiar should be high. Since we already keep track of the number of times a device has been seen in all contexts from which we can estimate the *global familiarity* of a device and use those to estimate the familiarity of unknown contexts. The exact formulation is left as future work.

Intelligibility

A common concern in context-aware systems is "intelligibility"[5]: they should be able to explain to the users the bases and implications of the inferences they make. We have taken some steps towards intelligibility of the context profiler (like showing inferred safety level and the familiarity scores used in the inference), we need a more thorough analysis of how to make the context profiler more intelligble.

VII. Related work

Location, WiFi and Bluetooth traces provide rich context information and have been utilized for several other applications as well. The Jyotish framework [6] utilizes the joint WiFi and Bluetooth traces for predicting the movement of users. It clusters the WiFi access point information to detect locations and uses Bluetooth traces to predict the most likely future contacts. Our work uses WiFi and Bluetooth traces to estimate context familiarity and safety.

Zhou at al. [7] and Nurmi et al. [8] use the location traces along with other information to identify meaningful places like home and work for their user. These meaningful places have several applications in location based services. We also exploit similar facts to identify points of interest and build up a context familiarity profile for these places.

The Familiar Stranger project [9] studies the properties and phenomenon of Familiar Stranger relationships. A *familiar stranger* is a stranger that the user repeatedly encounters but never interacts with. It uses a notion of device familiarity that is derived from the number of encounters with the stranger's device. The degree of familiarity is used to visualize the number of familiar strangers present at a specific place to the user. Unlike this work, we tie the notion of device familiarity to a given place and use it to estimate the familiarity and safety of a context.

Greenstadt and Beal [10] propose that mobile devices can utilize cues from user behavior to identify the users and make security decisions on their behalf. Jakobsson et al. [11] emphasize on the need for authentication techniques on mobile device with no or very limited user involvement. They utilize cues from user behavior like phone activity, mobility etc. to implicitly authenticate the user to the device and to provide addition assurance in sensitive transactions. Our primary focus is not on the method for user authentication, but on how to select one out of many authentication methods (with varying usability and strength) based on the safety of current context.

In [12], Danezis discusses how various social contexts can be automatically inferred for users from the social graphs around them. Privacy settings for these social contexts can be extracted based on the policy that content generated in a social context should be accessible only in that context. We focus on using device's context to configure access policies.

Conti et al. [13] propose a framework for enforcing context-related policies for smartphones that requires manual configuration of policies. Our system profiles the user's context to estimate its familiarity and automatically infer policies. Our system can be integrated with the Crepe framework to allow a user to specify policies based on context familiarity as a logical sensor in addition to other sensor values.

Kelley et al. [14] introduce the notion of *user-controllable policy learning* where the user and system refine a common

policy model in an incremental manner. Their system benefits from user feedback to gradually learn and identify policy improvements. Our model also incorporates user feedback to improve the decision making process.

Edwards et al. [15] highlight the pitfalls of automating access control where the control over security decisions is removed from the user's hands and given to the system. In our approach, we do not take away the control from a user. Instead, we assist the user by suggesting policy decisions and also incorporating user feedback.

VIII. Conclusion

We described a context profiler which uses location traces to detect places of interest for a user and profiles the Bluetooth and WiFi devices in such places to estimate the familiarity of a place. We showed how familiarity can be used to infer safety and use this safety score to make access control decisions. Our context profiler incorporates user feedback to shortcut learning and temporarily modify the behavior of our system. We chose parameters of the context profiler by running experiments using a large dataset and evaluated the effectiveness of our approach using ground truth data from the dataset. We have prototyped the context profiler on Nokia N900 Linux-based smartphones and plan to conduct user studies next.

References

[1] A. Gupta, M. Miettinen, and N. Asokan, "Using context-profiling to aid access control decisions in mobile devices," in *9th IEEE International Conference on Pervasive Computing and Communications (Demo paper)*, 2011, pp. 659–661.

[2] N. Kiukkonen, J. Blom, O. Dousse, and J. Laurila, "Towards rich mobile phone datasets: Lausanne data collection campaign," in *ICPS 2010: The 7th International Conference on Pervasive Services*, 2010.

[3] "Lausanne Data Collection Campaign," 2011, [referenced: 2011-09-23]. [Online]. Available: http://research.nokia.com/page/11367

[4] R. G. Brown, *Smoothing, Forecasting and Prediction of Discrete Time Series*. Dover Phoenix Edition, 2004.

[5] V. Bellotti and K. Edwards, "Intelligibility and accountability: Human considerations in context-aware systems," *Human-Computer Interaction*, vol. 16, pp. 193–212, 2001.

[6] L. Vu, Q. Do, and K. Nahrstedt, "Jyotish: A novel framework for constructing predictive model of people movement from joint wifi/bluetooth trace," in *9th IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 2011.

[7] C. Zhou, D. Frankowski, P. Ludford, S. Shekhar, and L. Terveen, "Discovering personally meaningful places: An interactive clustering approach," *ACM Trans. Inf. Syst.*, vol. 25, July 2007.

[8] P. Nurmi and S. Bhattacharya, "Identifying meaningful places: The non-parametric way," in *Proceedings of the 6th International Conference on Pervasive Computing*, ser. Pervasive '08. Springer-Verlag, 2008, pp. 111–127.

[9] E. Paulos and E. Goodman, "The familiar stranger: anxiety, comfort, and play in public places," in *CHI '04: Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 2004, pp. 223–230.

[10] R. Greenstadt and J. Beal, "Cognitive security for personal devices," in *Proc. of AISec'08*. ACM, Oct 2008, pp. 27–30.

[11] M. Jakobsson *et al.*, "Implicit authentication for mobile devices," in *Proceedings of the 4th Usenix Workshop on Hot Topics in Security (HotSec '09)*. Usenix, Aug 2009.

[12] G. Danezis, "Inferring privacy policies for social networking services," in *Proc. of AISec'09*. ACM, Nov 2009, pp. 5–9.

[13] M. Conti, V. T. N. Nguyen, and B. Crispo, "Crepe: context-related policy enforcement for android," in *Proceedings of the 13th international conference on Information security*, ser. ISC'10. Springer-Verlag, 2011, pp. 331–345.

[14] P. G. Kelley *et al.*, "User-controllable learning of security and privacy policies," in *Proceedings of AISec'08*. ACM, Oct 2008, pp. 11–18.

[15] W. K. Edwards, E. S. Poole, and J. Stoll, "Security automation considered harmful?" in *NSPW '07: Proceedings of the 2007 Workshop on New Security Paradigms*. ACM, 2008, pp. 33–42.