[Practical Paper] Inspection Method of Privacy Utilization for Rule-based Mobile Applications

Chiaki Doi†, Tomohiro Nakagawa†, Ken Ohta†, and Hiroshi Inamura†

†Research Laboratories, NTT DOCOMO, Japan
chiaki.doi.tf@nttdocomo.com

Abstract - As smartphones that offer the open development environment prevail, the number of applications is growing exponentially across the globe. Among them, context-aware mobile applications are increasing because various sensors are available to catch the user's context changes.

In order to create context-aware mobile applications, we adopt the ECA rule-based approach, which is intended for reactive applications. The application can perform functions without user's intervention, by leveraging context data such as terminal logs or sensor data as a trigger to perform predefined actions. Because context data can involve privacy data, we must ease privacy concerns before these applications will be widely accepted.

The objective of this research is to realize a method that determines how privacy data is utilized in context-aware rule-based mobile applications. The method outputs a privacy report, which is shown to users so that they can confirm if the application is acceptable or not in terms of privacy. There are three requirements that prescribe how privacy data utilization is to be described in the report.

The challenge of inspecting rule-based applications is to analyze the information flow on the implicit chain formed by independent rules. Such an implicit chain exists because there are cases in which the firing of one rule depends on another, even if the rule doesn't explicitly refer to the other.

Our privacy data inspection method is composed of a chain analysis process, a filtering process, and a summarization process. The proposed method satisfies the three requirements for privacy reporting, as well as accuracy and conciseness.

In order to evaluate the chain analysis and filtering processes, we create an evaluation model composed of two rules, which is the minimum rule chain unit. We confirm that neither false positives nor false-negatives are detected in evaluations that uses all 400 combinations of the rules generated from the model. We also confirm that the summarization process is effective in creating concise privacy reports. From this result, it is concluded that the proposed method works correctly for rule-based applications composed of straight chain rules.

Keywords: ECA rule, Information flow, Privacy data

1 INTRODUCTION

Due to the increasing number of mobile phone functions now open to application developers, it is becoming more attractive to create context-aware mobile applications. These applications leverage terminal logs such as user's location information or sensor data and behave autonomously by taking the user's context into account. For example, BreadCrumbz [1], Locale [2], and Nakamap [3] use such information to navigate people to specified places.

While the application market is expanding rapidly, concern is growing as to the security of using 3rd party applications. In recent studies of the Android platform, it has been pointed out that the Android framework faces high-risk threats[1][2], and that it is essential to determine how well applications can guarantee the Android user's security and privacy[3][9].

In order to create context-aware applications, we adopt the ECA rule, which is a suitable way to describe an application that automatically executes various functions depending on the user's situation [16].

Based on this ECA-rule based approach, we tackle the challenge of application inspection. Because the applications utilize the user's privacy data such as terminal log and sensor data, they suffer the risk of disclosing privacy information. For example, the application may directly pass the user's privacy data such as location information to another user. As another example, the application may indirectly convey the user's context to other users in the case that the user's context is utilized as the trigger to execute notification function to the other users.

In this paper, we propose a method to inspect how privacy information is utilized in ECA rule-based mobile applications. The proposed method is composed of a chain analysis process, a filtering process, and a summarization process. After these three processes conclude, the result, a privacy report, is shown to the user, the user can then confirm whether the application exhibits any harmful behavior or not before installing it.

In order to evaluate the chain analysis and the filtering processes, we created an evaluation model composed of two rules, which is the minimum rule chain unit. An evaluation result based on evaluations of all 400 combinations of the rules generated from the model shows that neither false positives nor false-negatives are detected. Thus it is confirmed that the proposed method works correctly for straight chain rules, and thus covers simple applications.

In addition, we evaluate the summarization process, and the result shows that the amount of privacy data is nearly halved in the privacy report while keeping the information

[1] BreadCrumbz http://www.bcrumbz.com/
[2] Locale http://www.twofortyfouram.com/
[3] Nakamap http://www.appbrain.com/app/nakamap-where-are-you-now/com.kayac.nakamap

needed by the user for decision making.

The paper is organized as follows. Section 2 describes the ECA rule-based application and discusses related work; it focuses on existing static analysis techniques and tools. Section 3 describes three requirements for the privacy report to be accurate and concise. Section 4 introduces our proposed method. In this section we explain how to analyze the chain of the rules, how to filter unnecessary rule chains from the candidates, and how to summarize similar items to make the privacy report concise. Section 5 describes the evaluation model for straight rule chains, i.e. no branches or loops are present. Also, the result of an evaluation of the summarization method is shown in this section. Section 6 concludes with a summary and an outlook on future work.

2 ASSUMPTION AND RELATED WORK

2.1 ECA rule-based applications

In this study we adopt the ECA rule[4][5][17][18] to describe context-aware mobile applications, which autonomously execute functions depending on the user's context. The ECA rule is composed of an event, a condition, and an action. The event triggers the processing of the rule. After the event occurs, the specified condition is checked. Only when the condition is satisfied is the action implemented. The reason why we adopt the ECA rule is that the execution of the rule is conducted without the user's explicit intervention, which is a suitable attribute for context-aware applications.

A rule-based application is composed of a set of ECA rules, which are described in XML. Each ECA rule in the application works independently, and has a different trigger as defined by the event. Thus, the rule has atomicity in that its logic is self-contained. Due to rule atomicity, a ECA-rule based application has flexibility in terms of execution, which is an additional reason why we adopt the ECA rule. For example, previous research mentioned that an ECA-based application can be customized by adding or deleting rules [6]. Also, there is research on dynamic changing rules at runtime in a transparent manner [7]. We consider that part of the ECA rules will, in the future, be transferred between a terminal and a server to realize dynamic load balancing.

In order to describe various types of mobile applications, we define several tags for the event and the condition. These tags are utilized to catch the change in the user's context through the user's operation log or sensor data. For example, an *OCCUR* tag is defined to catch the generation of a specified terminal log. As another example, a *SUM* tag is defined to evaluate if total number of terminal log generation is more or less than a specified value.

Also, various kinds of terminal logs are available as the evaluation target for these tags. There are several kinds of logs such as screen light-up or application start-up, which indicate the user's operation of the mobile phone. In addition, there are other kinds of logs such as location information or pedometer data, which indicate the user's behavior. By combining the aforementioned tags and these terminal logs, you can describe different kinds of context-aware applications.

In addition, we define a user-defined event as a kind of log. The objective of the user defined-event is to gather the user's context from their pressing a button on a dialog. Suppose the user-defined event *BEING_TIRED* is bound to a dialog button. When the button is pressed, the corresponding user-defined event is issued and recorded as user's operation log. Afterwards, the log of *BEING_TIRED* can be utilized in the same way as the normal kind of log from the event or condition by using tags such as *OCCUR* or *SUM*.

As an example of ECA-rule based applications, let's take an information distribution application which provides tourists with information depending on their location and pedometer data as captured by mobile phones. Figure 1 shows an excerpt of an ECA rule from the application. The rule defines that the event is fired when the user's location is within 1 km from Venice, which is located at latitude 45.434336 north and longitude 12.338784 east. Then, as defined in the condition, the pedometer data is checked to determine if it exceeds 5000 steps. If yes, as defined in the action, an implicit Intent is issued by using the Intent system of Android, and the browser accesses the specified URL of a web service which distributes tourist information. When the browser accesses the 3rd party's server, it transmits the user's location and pedometer data to obtain the user's context, which is utilized to personalize the delivered information.

This sample rule shows that the user's location and pedometer value is sent to the information provider. Thus, the user will want to know what kind of privacy data is being utilized and how the data is being conveyed to 3rd parties, before using the application.

```
<rule id="1">
 <event>
  <center lat="45.434336" lon="12.338784"
      kind="LOCATION_INFORMATION">
   <less_than>1000</less_than>
  </center>
 </event>
 <condition>
  <sum kind="PEDOMETER" more_than="5000">
  </sum>
 </condition>
  <action>
   <coordinate intentType="implicit"
     data="http://URL1?location=Venice&pedometer=
5000"
       action="android.intent.action.VIEW"/>
 </action>
</rule>
```

Figure 1: An example of ECA rule-based application

2.2 Rule chain by dependency between rules

As we have seen in the previous section, an ECA rule-based application is described as an aggregate of independent rules. In other words, no rule explicitly refers to any other rule.

In spite of rule independency, there is an implicit dependency between the rules in specific situations. Such a situation occurs when the execution of a specific rule's action is a necessary condition for firing another rule's event. If there is such dependency between the rules, user's privacy data may indirectly flow out of the terminal through the continuous execution of these rules. Supporse there are two rules. The first rule's event has a SCREEN_ON tag and action has a SEND_USER_DEFINED_EVENT tag. The second rule's event has a RECEIVE_USER_DEFINED_EVENT tag and action has a SEND E-MAIL tag Rule. The second rule sent a email to specified address after the first rule fired after user touches screen. In this case, email recipent can know that the user operated the terminal. In order to clarify an output path of privacy data, we need to analyze the dependency between rules.

There are two cases of such rule dependency, which is called 'rule chain' hereafter.

In the first case, the first rule's action has a *LOG START* tag and the second rule has any kind of event tag which utilizes the same kind of log specified in the first rule. The *LOG START* tag means to start recording a kind of log. In the second case, the second rule's event has an *OCCUR* tag, which means to catch the generation of a specified log type, the second rule is fired only after the first rule is executed.

The second case of the rule chain is that the first rule's action has a *BUTTON* tag, which issues a user-defined event, and the second rule's event has any kind of event tag which utilizes the same user-defined event specified in the first rule. If the second rule's event has an *OCCUR* tag that is defined to count the number of the same user-defined event specified in the first rule, there is a chain between those rules.

2.3 Related work

For the purpose of tracking how sensitive data is handled by an application, some studies attempt to track the data at runtime[8][9]. In [8], sensitive data such as passwords or credit card numbers are tracked by simulating the whole system. TaintDroid[9] realizes a light-weight system-wide tracking system for mobile phones.

While these tracking systems have the same purpose, analyzing the flow of sensitive data, they use dynamic analysis, which assumes that the target application can be run in an emulation environment or a real system.

Our approach is to use a static analysis approach, which can analyze the information flow without executing the target application. By using static analysis, the user can notice the malformed behavior of an application before installing it.

There are several techniques for the static analysis of information flow that annotate programs handling confidential information [10][11]. Our proposal avoids such annotation. All of the privacy data is predefined as confidential information, and the flow of the privacy data is exhaustively analyzed. Then, whether the privacy data should be reported or not is judged by how they are utilized.

There are some techniques that offer the static analysis of data or programs without annotation [12][13][14]. These techniques are intended to verify the query data[12][13] or sequential execution of the code, which follows an explicit control[14].

The difference between our proposed method and existing static analysis techniques is we analyze the implicit chain formed by independent rules. This study is intended for applications that consist of independent rules. However, as described in Section 2.2, the rules can form implicit chains. Therefore, it is necessary to consider the dependence described in Section 2.2. To analyze an implicit rule chain, it is necessary to analyze the pathways of information by considering the semantic relationships between rules.

From the viewpoint of analyzing rule-based applications, a previous study described the verification of an application's convergence[15]. However, its goal was to verify whether the application's running time would converge or not, and so cannot be applied to our problem.

Existing related tools are Permission Checker Security[4] and S2Permission Checker[5] to detect applications that might be spyware applications in Android phones. The difference between these tools and our method is that those tools merely show the possibility of anomalous behavior at a rough level of granularity. For example, these tools do not show the destination of the leaked information. On the contrary, our proposed privacy report provides users with a more detailed assessment of the behavior of the application as described in Section 3.

3 PRIVACY UTILIZATION INSPECTION

3.1 Requirements

After the application is inspected, the result will be presented to the user as a privacy report as shown in Fig. 2. The privacy report is meant to be shown before the user installs the application. The user can then judge whether or not it is acceptable to utilize the application by confirming how the user's privacy data is utilized.

In order to clarify the detailed behavior of the application, there are three requirements that the privacy report must satisfy. The motivation of detailing the privacy utilization as described below is that there are various terminal logs, and a subtle change in how they are utilized may change the mind of the user.

1. Show which kind of privacy data is utilized and to which destination the information is sent
2. Details the conditions under which the privacy data was created
3. Eliminate redundancy from the privacy report so that the user can understand it easily

[4] Permission Checker Security
http://www.appbrain.com/app/application-permission-checker/jp.ne.neko.freewing.PermissionChecker
[5] S2 Permission Checker
http://www.androidzoom.com/android_applications/tools/s2-permission-checker_luge.html

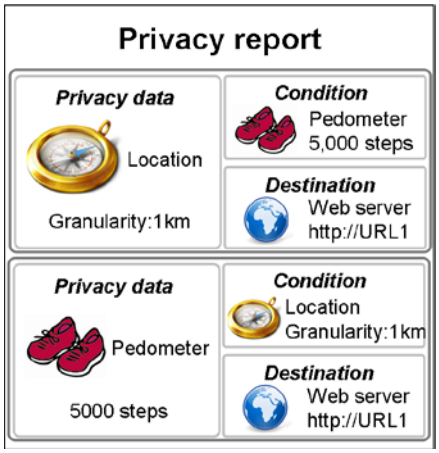

Figure 2: An example of the privacy report

Requirement 1 specifies that the privacy data such as user's location and pedometer data should be displayed jointly with the destination. The destination is specified for every notification method that uses the action. We defined eight kinds of actions such as e-mail transmission, Intent invocation, and location sharing. Existing tools specify only the privacy information that could be output. The addition of the destination makes the privacy report much more useful.

Requirement 2 specifies that the condition under which the action is taken should be included in the privacy report. We consider that the conditions may influence the user's judgment on whether or not to utilize the application. This is because the user should weigh the benefits and the privacy risks of the application in making the judgment. For example, suppose it is unacceptable for user A to share location information anytime and anywhere with person B. By limiting the time to share the location to around the time of rendezvous with B, the same application can turn to be an attractive one for the same user A.

Requirement 3 specifies that the privacy data described in the privacy report should be concise, eliminating items of the same sort. This is necessary to make the report readable.

3.2 Proposed inspection method

By adopting the ECA rule-based approach, we can describe a context-aware application flexibly by combining various kinds of tags and terminal logs. We designed an application programming language composed of seven events, three conditions, and eleven actions. In addition, 54 terminal logs can be utilized from the event and condition tags.

In order to generate accurate privacy reports, we need to clarify the implicit rule chain formed by any combination of the rules freely defined by utilizing these tags and logs. If there was any false recognition or oversight of a rule chain, the privacy report might contain false positives or false negatives.

In this section and the following subsections, we describe the proposed method, which is composed of the three processes shown in Fig. 3. The proposed method processes the ECA rules and generates a privacy report as its output.

The chain analysis process and second filtering process are designed to satisfy requirements 1 and 2, by clarifying the sequence in which the privacy data is utilized. The summarization process is designed to satisfy requirement 3, by aggregating the same kind of items in the privacy report.

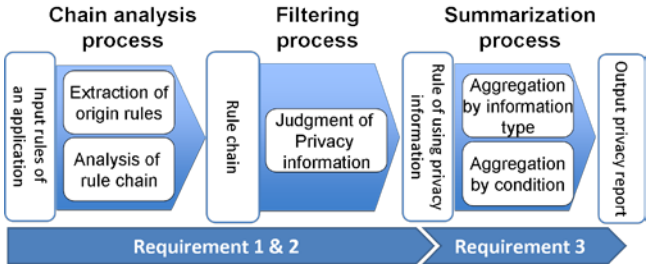

Figure 3: Processes of the proposed method

3.2.1. Chain analysis process

The rule chain analyzed in this process starts from an origin rule and the chain is formed by the dependency of the rules as described in Section 2.2.

The origin rule is defined as a source of the privacy data utilization. In other words, it is the rule that utilizes the user's privacy data in its event, condition, or action tags. For example, a rule that includes *OCCUR* tag or a *SUM* tag is regarded as an origin rule. On the other hand, if a tag doesn't utilize any privacy data, a rule that uses only that tag is not an origin rule. For example, *TIME* tag is fired at the designated time regardless of the user's situation. That's why a rule that includes *TIME* tag is not an origin rule.

Note that the origin rule is defined by not only the event tag, but also the condition and action tags. The reasons are as follows. The condition tag has a function of judgment by taking the user's context data as a parameter, just as the event tag does. The action tag doesn't have such function, as it either directly or indirectly transmits the user's privacy data. An example of direct transmission is a *LOCATION* tag, which has the function of sending location information to a server to share it with designated agents/persons. An example of indirect transmission is a *BUTTON* tag, which has the function of issuing a user-defined event. In this case, the user's button operation can be transmitted via a rule chain.

Figure 4 shows a flowchart of the chain analysis process. This procedure is performed in two steps.

Table 1: Example of privacy data in an origin rule

Category	Example of privacy data
User data	Phonebook entry added/modified
	Schedule added/deleted/modified
Terminal state	Battery power changed
	Silent mode set/canceled
Operation log	Screen light on/off
	Button operation
Behavioral information	User's location
	Pedometer data

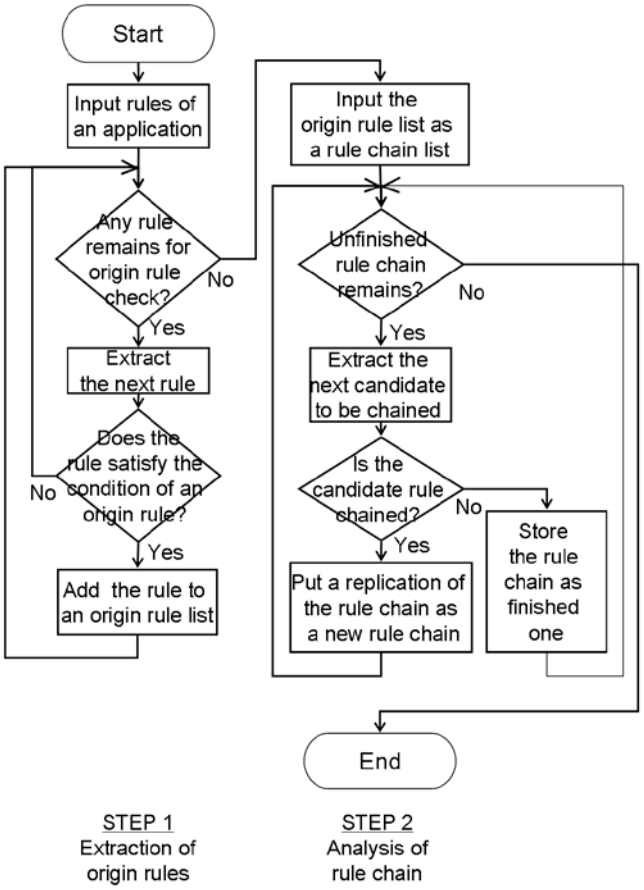

Figure 4: Flowchart of chain analysis process

The first step is to extract all origin rules, which is realized by pattern matching of the tags that utilize privacy data. There are four classes of privacy data: user data, terminal state, operation log, and behavioral information (see Table 1). User data is defined as data input by a user, such as schedule information or phonebook entries. Terminal state is defined as data that shows the status of the mobile phone, such as remaining battery power or silent mode. These data can be used to infer the current state of the user's situation. Operation is defined as the record of terminal operation by a user, such as screen light on and off, or pressing a button displayed on a dialog. The operation log makes it possible to determine whether there is a user's operation or not. Behavioral information is defined as data that shows user's activity in the real world, such as user's location or pedometer count. With this information, the system can infer where and what users are doing. These categories are utilized in the filtering process, as described in the next subsection.

The second step is to determine the rule chains starting from the origin rules extracted in the first step. This determines the pathway on which the user's privacy data is output. To extract all rule chains, whenever a new branch is found, the original rule chain is replicated. Loops are avoided by excluding rules that already exist in the rule chain.

The tricky part of the second step is that in some exceptional cases we need to define a rule chain as invalid, considering the semantics of the tags. For example, because *LOCATION* tag directly outputs user's location information by itself, a rule with the tag must be regarded as a finished rule chain, and so cannot be connected to any other rule.

3.2.2. Filtering process

At the end of the rule chain analysis, we get a set of rule chains, which are filtered to extract the items essential for the privacy report. The motivation of the filtering process is to remove trivial items from the report. Even if a rule chain starts from privacy data utilization, the privacy data need not be present in the privacy report as long as the data is utilized internally or in a normal way.

For filtering the privacy data, we focused on the relationship between privacy data in the origin rules and how the data is finally utilized in end rules. The end rule is defined as a rule that includes an action tag with an output function.

Table 2: Criteria for privacy data filtering

	Utilization purpose		
	Network output	Display output	Internal processing
User data	○	-	-
Terminal state	○	-	-
Operation log	○	○	-
Behavioral information	○	○	-

○ : Described in the privacy report
- : Omitted from the privacy report

The criteria for the filtering process are shown in Table 2. The intersection of the categories of privacy data and utilization purpose shows whether the privacy data should be described in the privacy report or not. The utilization purpose is categorized as network output, display output, and internal processing. In our filtering algorithm, all privacy data that is output via the network is reported to the user. On the other hand, only operation log and behavioral information are reported when these privacy data are output via the mobile's display. This is because user data and terminal data can be displayed for other applications, and these cases are regarded as relatively insignificant.

Just as in the case of the chain analysis process, we need to be careful of slight differences in the semantics of tags in the filtering process. We need to disable an end tag when the origin rule doesn't output user's context in a single rule. For example, a *BUTTON* tag is an origin tag which shows user's button operation. Even a rule including the *BUTTON* tag has also an action tag with output function, the button operation should be omitted from the privacy report. This is because the *BUTTON* tag is a part of an action tag . Only if the rule with the *BUTTON* tag is chained to other rules and the

chained rule includes an output function, should the button operation be described in the privacy report.

3.2.3. Summarization process

In the summarization process, we eliminate the redundancy of the information presented to the user to satisfy requirement 3. This process aggregates similar rule chains by classifying them into the same category. These rule chains are aggregated when the same operation and behavior information are output by the same function.

We implemented two methods for the summarization process. Table 3 shows the items to be considered in the summarization process. The first method aggregates the rule chains by considering the conditions of privacy information. The second method does not consider these conditions.

Table 3: Items considered in the summarization process

	Origin rule		End rule
	Privacy data	Condition	
Method 1	○	○	○
Method 2	○	—	○

○ : Considered , — : Not Considered

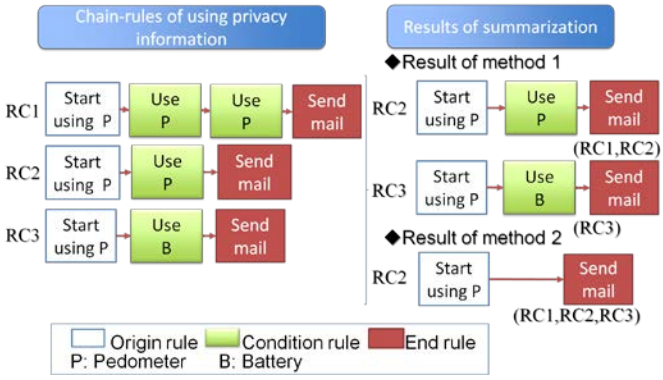

Figure 5: Operation of summarization process

Figure 5 shows the operation of the summarization process.

In method 1, judgment of whether the pedometer data meets the condition is performed for the second and the third rule. The second and third rules set the same condition; therefore, these rules are placed in the same category. RC1and RC2 are placed in one category if they have the same combination of privacy information and condition. As a result of summarization process, RC1 is aggregated into RC2 because RC2 has fewer rules than RC1. RC2 is output as the result of the summarization process. RC3 has different condition in the second rule, so RC3 is also output as an independent result.

In method 2, rule chains are aggregated without consideration of the conditions. Only the origin rule and the end rule are used for aggregation, so the amount of information output by summarization is reduced. The result of applying this filtering process to RC1, CR2 and RC3, is that RC1 and RC3 are aggregated into RC2 which has common origin rules and end rules.

The summarization process is stronger with method 2 than with method 1. The latter can show details that include the conditions of using privacy information.

4 EVALUATION

4.1 Filtering and chain analysis processes

In order to evaluate the filtering process and the chain analysis process, we created an evaluation model composed of two rules, which is the minimum rule chain unit. Figure 6 shows the evaluation model proposed here. The rules in the evaluation model are composed of only event and action, i.e. they do not include any condition. The reason for this omission is that event and condition have the same mechanism for extracting privacy data.

For the sake of model completeness, the tags utilized in the evaluation model are selected so that they encompass all the categories that can impact the proposed processes. The result of the filtering process is influenced by the category; therefore we chose a representative tag from each category. We neglect the privacy data utilized in the tag because the difference in privacy data has no influence on the result of the chain analysis process.

Table 4 shows the tags used in this model. The event in the first rule includes four tags to cover all the categories of the privacy data which we defined in Table 1. The action in the first rule includes two tags which chain to other tags and three tags that output tags which include privacy information. The event in the third rule includes three tags which chained to other tags and two tags are chained to other tags. The action in the fourth rule includes three tags which output privacy information and a tag which finishes recording the log.

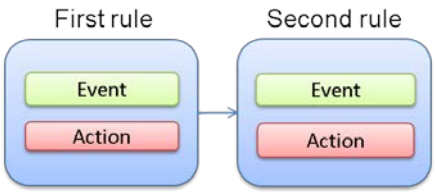

Figure 6: Evaluation model

Our evaluation model effectively reduces the number of samples needed for the evaluation. Our model needs only 400 samples generated by the combinations of the representative tags. Without our model, 1,171,800 combinations would have to be evaluated.

An evaluation using all 400 combinations of the rules generated from the model confirms that neither false positives nor false negatives were detected. We covered applications which chain linearly in this evaluation model. In other words, applications that include branches and loops are not covered. It is a future work to build a general evaluation model to cover such applications.

Table 4: Tags in evaluation model

Rule		Function	Tags used in the evaluation model
The first rule	Event	Tags which utilize user's privacy data	<sum kind= "21">
			<sum kind= "43">
			<sum kind= "40">
			<sum kind= "28">
	Action	Tags which chain to other tags	Send user-defined event
			Start to record the privacy data
		Tags which output privacy information	<location>
			<dialog>
			<timer>
The second rule	Event	Tags which chain to other tags	receive user-defined event
			Utilize the privacy data
		Tags which don't chain to other tags	<occur kind="43">
			<occur kind="40">
			<occur kind="28">
	Action	Tags which don't chain	<log stop>
		Tags which output privacy information	<location>
			<dialog>
			<timer>

4.2 Summarization process

Our evaluation of the summarization process used the 18 kinds of tags shown in Table 4. We generated 25 applications, each of which had a set of five rules. The rule sets were generated by randomly choosing tag from among the candidate tag group with equal probability. The average number of the privacy data in the privacy report was 25 for the original data before the summarization process. After applying our summarization method, the number of the privacy data was reduced to 13 for method 1, and 11 for method 2. Thus, we confirmed that the summarization process is effective in reducing the number of private data, making it possible for users to confirm the privacy report more easily.

4.3 Processing time

We used Java to implement our proposed method. Table 5 shows the execution environment in which we evaluated our proposal.

Table 5: Execution environment

Execution environment	
CPU	**IntelCore2 6400@2.13GHz**
OS	**Windows XP Professional SP3**
RAM	**2GB**

Figure 7 shows the average processing time of the proposed method. The total processing time is the time required for filtering process, chain analysis process, and summarization process. The test data were the 25 applications utilized in the evaluation of the summarization process.

The average total processing time of the proposed method is 41 ms for method 1, and 37 ms for method 2. The result ranged from 32 ms to 44 ms for methods 1 and 2. This result confirms that either method 1 or 2 can be selected depending on the user's requirement.

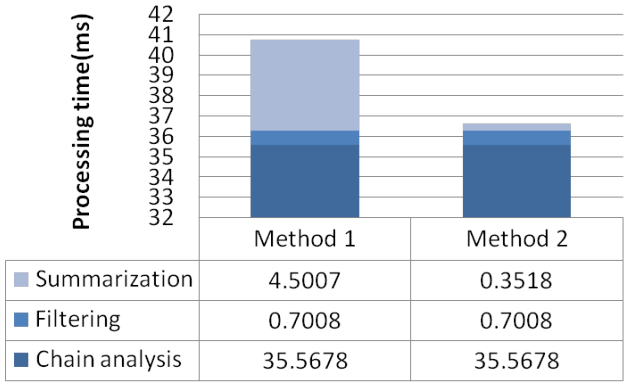

Figure 7: Processing time of the proposed method

5 CONCLUSION

We proposed an application inspection method for rule-based applications that is composed of a chain analysis process, a filtering process, and a summarization process. We confirmed the validity of the proposed method by using an evaluation model with minimum rule chain configuration. The evaluation results show that the proposed method works correctly for applications that are composed of straight rule chains. We also evaluated two summarization methods and confirmed that our method cut the amount of information present on the privacy report.

As a future work, we plan to evaluate the chain analysis and filtering processes by a more generalized model. In such a model, we need to include branches and loops of the rule chain to cover complicated applications. It is also a future work to improve the clarity of the privacy report in case there is many kinds and granularity of privacy data as well as several ways of notification to other users. As the severity of privacy risk depends on the combination of these elements, it is a challenge how to create concise privacy report that includes required alert a user.

REFERENCES

[1] A. Shabtai, ``Google Android: A Comprehensive Security Assessment,'' IEEE Security & Privacy Magazine, pp. 35-44 (2010).

[2] W. Enck, ``Understanding Android Security, Security Assessment,'' IEEE Security & Privacy Magazine, pp. 50-57 (2009).

[3] M. Butler, ``Android: Changing the Mobile Landscape,'' IEEE Pervasive Computing, Vol. 10, No. 1, pp. 4-7 (2011).

[4] J. Bailey, ``An event-condition-action language for

XML," Proceeding of the 11th international conference on World Wide Web, pp. 486-495 (2002).

[5] G. Papamarkos, ``Event-condition-action rule languages for the semantic web, '' Workshop on Semantic Web and Databases, pp. 855-864 (2003).

[6] M. Miyama, ``An event-driven navigation platform for wearable computing environments," Proceeding of the 9th IEEE International Symposium on Wearable Computers, pp. 100-107 (2005).

[7] S. Chakravarthy, ``Dynamic Programming Environment for Active Rules," The 7th International Baltic Conference on Database and Information Systems, pp. 3-16 (2006).

[8] J. Chow, ``Understanding Data Lifetime via Whole System Simulation," USENIX Security Symposium, pp. 321-336 (2004).

[9] W. Enck, ``TaintDroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones," Proceeding of the 9th USENIX Symposium on Operating Systems Design and Implementation, pp. 1-15 (2010).

[10] E. Dorothy, ``A lattice model of secure information flow," Communications of the ACM, Vol.19, No.5, pp.236-243 (1976).

[11] A. C. Myers, ``JFlow: Practical Mostly-Static Information Flow Control," Proceeding of the 26th ACM Symposium on Principles of Programming Languages, pp. 228-241 (1999).

[12] P. Geneves, ``XML Reasoning Made Practical," The 26th International Conference on Data Engineering, pp. 1169-1172 (2010).

[13] S. Boettecher, ``Finding the Leak: A Privacy Audit System for Sensitive XML Databases," Proceeding of the 22nd International Conference on Data Engineering Workshops, p. 100 (2006).

[14] C. Kirkegaard, ``Static Analysis of XML Transformation in Java," IEEE Transaction on Software Engineering, Vol.30, No.3, pp181-192 (2004).

[15] A. Cheng, ``Fast Static Analysis of Real-Time Rule-Based Systems to Verify Their Fixed Point Convergence," Proceedings of the 5th Annual Conference on Software Safety and Process Security, pp. 46-56(1990).

[16] P. Moore, B. Hu and M. Jackson, ``Rule Strategies for Intelligent Context-Aware Systems," Proceedings of the 2011 International Conference on Complex, Intelligent, and Software Intensive Systems, pp. 9-16 (2011).

[17] T. Hu and B. Li, ``Research on the Mechanism of Tourism Information Change Management Based on ECA Rules," Proceedings of the 2010 13th IEEE International Conference on Computational Science and Engineering, pp. 388-392, (2010).

[18] P. Moore, M. Jackson and B. Hu, ``Constraint Satisfaction in Intelligent Context-Aware Systems," Proceedings of the 2010 International Conference on Complex, Intelligent and Software Intensive Systems, pp 75-80, (2010).

(Received February 13, 2012)
(Revised February 28, 2013)

Chiaki Doi joined NTT DOCOMO, Inc., in 2009. Her research interests include mobile computing, system security, and computer graphics. She received B.E. degree from Iwate Prefectural University and M.E degree from Keio University, Japan. She is a member of Information Processing Society of Japan.

Tomohiro Nakagawa is a research engineer at NTT DOCOMO, Inc. His research interests include mobile computing and distributed systems. He received his B.E. and M.E. from the University of Tokyo, Japan in 1998, and 2000, respectively. He is a member of IPSJ.

Ken Ohta received the BE, ME, and DE degrees from Shizuoka University, Japan in 1994, 1996, and 1998, respectively. In 1999, he joined NTT Mobile Communications Network Inc. (NTT DOCOMO). His research interests include mobile computing, distributed systems, and system security. He is a member of the Information Processing Society of Japan and of the Institute of Electronics, Information and Communication Engineers.

Hiroshi Inamura joined NTT DOCOMO, Inc. in 1999. His research interests include system research on mobile device and distributed system. Before DOCOMO, he was a research engineer in NTT labs since 1990.From 1994 to 1995, he was a visited researcher in the Department of Computer Science, Carnegie Mellon University. He received B.E., M.E. and D.E. degree in Keio University, Japan. He is a member of IPSJ, IEICE, and ACM.