Ontology-based privacy protection for smartphone: A firewall implementation

Johanne Vincent, Christine Porquet, Idriss Oulmakhzoune

ONTOLOGY-BASED PRIVACY PROTECTION FOR SMARTPHONE: A FIREWALL IMPLEMENTATION

Johann Vincent, Christine Porquet and Idriss Oulmakhzoune

GREYC Laboratory, ENSICAEN – CNRS, University of Caen-Basse-Normandie, 14000 Caen, France

During the past few years, smartphones have tended to replace feature phones in the user's pocket. They allow users to access various applications and services that aim at making their life easier. For that purpose, some applications access private data such as the user's location and do not always ask for the user's consent. Data collection is a growing issue for the user's privacy and the current protection mechanisms have proved to be inefficient [1]. In a previous work [2], we have proposed a real-time firewall that aims at preventing malicious applications from accessing private data on an Android smartphone. The firewall relies on an ontology that represents privacy protection concepts and the digital identity of a smartphone's user. Our firewall is based on a set of privacy rules and applies description logic reasoning to decide whether access to private data by a given application may be authorized or not. However, the use of semantic technologies can be an issue when implemented on devices that offer low computation capabilities. The contribution of the present work is an optimized implementation of the firewall that offers good performances as well as good privacy protection. This paper is organized as follows: the first part is a description of our optimized implementation; the second one details the criteria we have chosen to evaluate the implementation and is followed by the results of our tests on this implementation. Finally, the conclusion of the paper highlights the interest of the approach and proposes future improvements.

Our first proposition consists in a local application implemented on an Android device. The application relies on the JENA API [3][4] to handle OWL [5] ontologies. The ontology file is first loaded as a JENA object that will be used by the firewall. The application simulates a request made by a specific application to access some data and creates the corresponding individuals within the ontology. The individuals can be viewed as instances of concepts. The firewall then launches reasoners [6] in order to classify the ontology regarding a set of privacy rules. Finally the firewall takes its decision. The first effort conducted to optimize the decision time was to try different reasoning modes; in the JENA API three inference modes are available: forward chaining, backward chaining and mixed chaining. None of them led to significant performance gain and were able to reduce the loading time of the firewall. To improve the loading time, using a local database instead of the OWL ontology file was also tried. This solution was also unsatisfactory as there is an incompatibility between JENA and SQLite on Android. The current JENA API does not support binary serialization and serializing the ontology as a JENA object is not possible either.

As our attempts to optimize the local firewall proved unsuccessful, an alternative approach is proposed in this paper: the use of a distant server. This server stores the ontology and takes its decisions while only a local database is maintained to store the most recent results on the smartphone. With this approach, the decision time is only function of the connection bandwidth. The overall architecture proposed is presented on Figure 1. The server side is implemented as a JAVA servlet accessible via HTTPS. Obviously, the choice of a secure protocol is important as the requests can also leak privacy. The server is in charge of creating individuals in the ontology upon reception of requests. It also classifies the ontology regarding a set of privacy rules and takes the decision to authorize the request or not. It then sends back its answer to the application on the phone. The local firewall application stores the answer of the server in a SQLite database as well as a time to live (TTL) for the request. The

use of the TTL is here to prevent the phone from making too many requests to the server as well as to keep a minimal protection while disconnected.

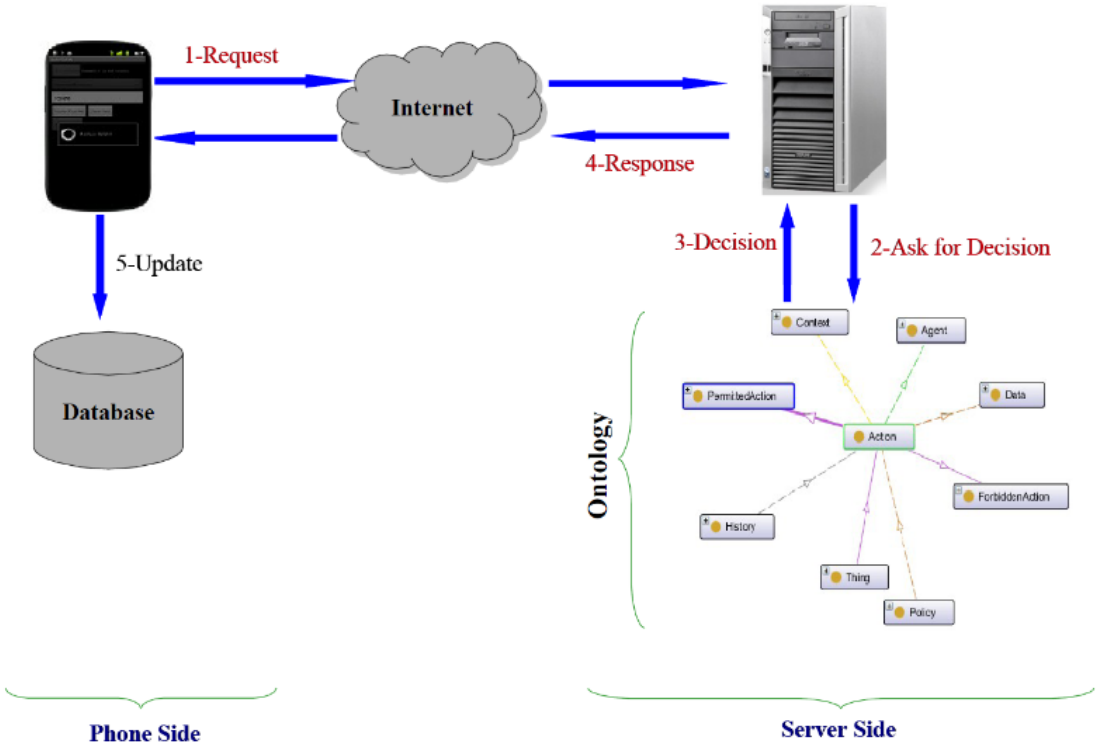

Figure 1: Architecture of the privacy firewall implementation

To evaluate the firewall performance the first criterion to be considered is obviously the decision time. As the firewall is active on the phone, it is important to keep its intervention on requests as unnoticeable to the user as possible. The second criterion is the battery consumption; in fact the firewall acts as a service and is always active on the smartphone, it is therefore important to be sure it has no severe impact on the battery. To verify such a hypothesis, the firewall has been stressed to answer one request every second for one hour. As we have introduced a network component we also have measured the requests and answer sizes. We have conducted the tests on two Android smartphones of two separate generations: a Samsung Galaxy (Android 1.6, 528 MHz and 192 MB of RAM) and a Samsung Nexus S (Android 2.3, 1 GHz and 512 MB of RAM). Tests were conducted using a complete ontology with 67 concepts and 94 properties that weights 86 KB.

The results of our experimentations are shown on Table 1 and 2. The first thing to note is that the initialization time has been greatly reduced even considering the delay introduced by the connection to a server. The initialization takes around 4 s for the ontology while using a distant server instead of 31 s locally on the Galaxy and 5 s on the Nexus S. The decision time is also quicker while using the distant server: between 300 ms and 400 ms for both phones instead of 9 s on the Galaxy and 1.5 s on the Nexus S. It is also interesting to note that when relying on the results stored in the local database the firewall response takes around 10 ms. During our tests on power consumption, we have measured a battery total discharge of around 7%. However, other applications were also running on the phone and we have noticed that our firewall accounts for only 4% of these 7%, that is 0.28% of total power consumption. As stated before, message size is an important matter for our solution to be viable. We have divided the messages into two types: configuration messages and request messages. We have measured that the configuration messages from the phone weight around 1.2 KB and that the answers

from the server weight around 1.5 KB. The request messages weights between 0.861 KB (Galaxy) and 1 KB (Nexus) and that the answer weights between 1.9 KB and 1.3 KB. The Table 2 also shows the results we had with the HTTP protocol, but it is important to note that this solution is not secure.

	Samusung Galaxy			Samusung Nexus S		
	Server		local	Server		local
	HTTP	HTTPS		HTTP	HTTPS	
Total loading time	4349	5572	24121	3553	4354	4477
Decision time	274	401	9584	333	323	1560

Table 1: Decision time (ms) with a complete ontology (86KB)

Data	Samusung Galaxy				Samusung Nexus S			
	HTTP		HTTPS		HTTP		HTTPS	
	P to S	S to P	P to S	S to P	P to S	S to P	P to S	S to P
Config	0.596	0.308	1.201	1.72	0.580	0.395	1.127	1.369
Request	0.454	0.583	1.061	1.934	0.388	0.518	0.861	1.315

Table 2: Exchanged data size (in KB) between phone and server (P to S) and server and phone (S to P)

In this paper an efficient and realistic privacy protection mechanism for smartphones is proposed. Also it was tested on Android devices; the interest of our solution is that it is not bound to any specific brand. In fact, every connected device can benefit from our approach as it only needs to implement an HTTPS client. Thanks to an extensive ontology and complex privacy rules, our solution can handle new privacy threats such as the data collection from two or more application. Future works will mainly focus on the expression of specific rules that match these new privacy threats.

1. Lookout. The app genome project (2010), http://blog.mylookout.com/2010/07/introducing-the-app-genome-project/
2. Vincent, J., Porquet, C., Borsali, M. and Leboulanger, H.: Privacy Protection for Smartphones: An Ontology-Based Firewall. In Information Security Theory and Practice. Security and Privacy of Mobile Devices in Wireless Communication, pp. 371-380 (2011)
3. Carroll, J.J., Dickinson, I., Dollin, C., Reynolds, D., Seaborne, A., Wilkinson, K.: Jena: implementing the semantic web recommendations, pp. 74–83 (2004)
4. Androjena. Jena android porting (2010), http://code.google.com/p/androjena/
5. McGuinness, D.L., Van Harmelen, F., et al. Owl web ontology language overview. W3C recommendation, 10:2004–03 (2004)
6. Sirin, E., Parsia, B., Grau, B.C., Kalyanpur, A., Katz, Y.: Pellet: A practical owl-dl reasoner. Web Semantics: Science, Services and Agents on the World Wide Web 5(2), 51–53 (2007)