An Inline Monitoring System for .NET Mobile Devices*

Nicola Dragoni, Fabio Massacci and Katsiaryna Naliuka

Abstract Users of mobile devices are increasingly requesting a controlled way to exit from the sandboxing model in order to exploit the full computational power of the device. Porting in-line security monitors to mobile devices requires to solve both theoretical and practical challenges. Current security monitors provided solutions only for the desktop and either monitor a single instance of an application. In this demonstration we show an inline monitoring system for .NET mobile devices developed in the context of the EU-FP6-IST-STREP-S3MS project[2]. The demonstration consists of two different demos with real mobile devices, highlighting the usage model and the main features of the system.

1 Usage Model

The goal of the inline monitoring system ([1] for details) is to give to the user the opportunity of downloading a completely untrusted program on its mobile device and then to run it safely. The inline monitoring system is used to enforce the user policies in a reliable and autonomous way. Obviously, in order to control the execution of the program the monitor needs to be notified about the relevant actions of the program. At first we could rewrite the entire virtual machine but this would require Microsoft or Sun preliminary agreement, and this effort alone would dwarf any technical difficulty. In-line monitors [2, 3, 4] looked more promising.

For this reason, we offer the possibility to rewrite the downloaded program inserting (*in-lining*) hooks that notify the monitor before and after each security-relevant

Nicola Dragoni and Fabio Massacci and Katsiaryna Naliuka
DISI - University of Trento, Via Sommarive 14, POVO (TN) - Italy, I-38100. e-mail: name.surname@disi.unitn.it

* Research partly supported by the projects EU-FP6-IST-STREP-S3MS, EU-FP6-IP-SENSORIA, EU-FP7-IP-MASTER.

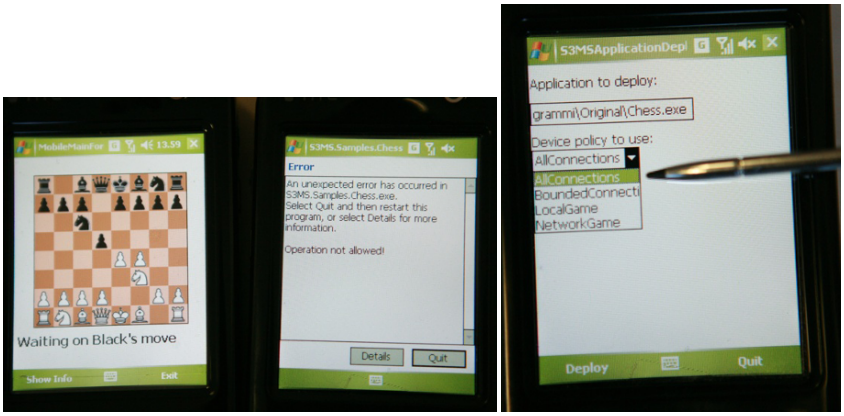
[2] http://www.s3ms.org

Please use the following format when citing this chapter:

Dragoni, N., Massacci, F. and Naliuka, K., 2008, in IFIP International Federation for Information Processing, Volume 263; *Trust Management II*; Yücel Karabulut, John Mitchell, Peter Herrmann, Christian Damsgaard Jensen; (Boston: Springer), pp. 363–366.

event. The in-lining is performed directly on the device, so that we do not need to rely on any third party for that. As the result of the in-lining each security-relevant method is executed only if it has been allowed by the policy.

Example 1. Figure 1(a) shows two HTC P3600 smartphones playing chess by sending each other SMSs. The smartphone on the left has no monitoring framework installed, so the game is allowed to send as many SMSs as it will. The monitoring system of the phone on the right detected that the user-defined limit of the sent messages was exceeded. For this reason the game is terminated through the security exception.



(a) Policy violation and error message (b) User selects a policy

Fig. 1 Basic Usage Examples of the Inline Monitoring System

While the in-lining of the downloaded programs is performed on-device, the policies need to be managed off-line. The security policies are written in our policy language 2D-LTL and then automatically converted in the form suitable for monitoring. This representation of the policy is deployed to the device, where it is used by the in-liner to extract the method calls that are relevant to the policy and by the monitor itself.

As writing the policies in the logic-based language is likely to appear to complex for an average user, this task might be undertaken by the mobile service provider as the added-value service. At the end of the day the user simply selects the "right" policy from a pull-down menu (See Fig. 1(b)).

Example 2. Fig. 1(b) shows how a user of HTC P3600 might choose between four pre-set policies allowing any kind of connections, a bounded number of connections (max SMS sent and max MB downloaded), some access to GPRS or a purely local game where no connection is allowed.

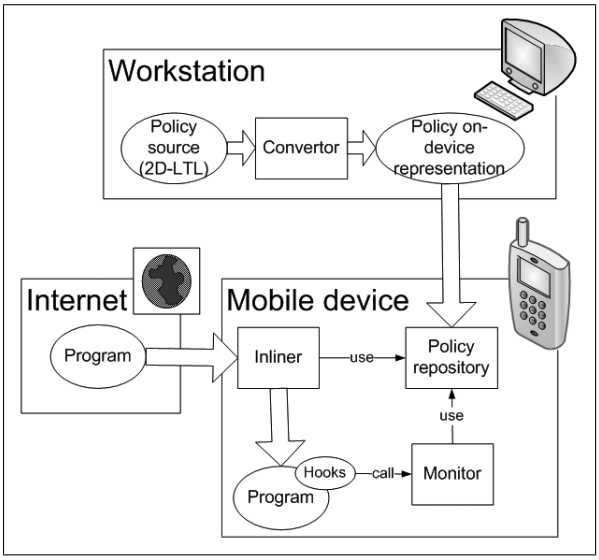


Fig. 2 Usage Model

Summing up Fig. 2 show our basic usage model that is further described below.

Scenario 1 *The user designs his security policies or, if he or she is not competent enough, picks the predefined one from the operator. The policy is formalized in the logical language 2D-LTL, which is described below. Then the policies are automatically compiled in the form suitable for the runtime monitoring. The compiled policies are deployed at the device. When the application is downloaded and installed at the device it might be in-lined with any of the user policies. After the user selects the policy the application is rewritten and the calls to the monitor are inserted before and after each call to the method mentioned in the policy. Then when the application is run the monitor checks whether its execution satisfies the policy, and is the violation is detected a security exception is raised in the application. If the application does not handle the security exceptions then it terminates. Otherwise if the developers took the security exceptions into consideration the application may be able to proceed (or at least to terminate gracefully, notifying the user about the situation). But in any case the execution that violates the policy will not be possible.*

2 Demonstration Description

The demonstration of the inline monitoring system consists of the following two demos.

Demo 1 *Bob and Anna want to play with a m-MMPORG game using their HTC P3600 smart phones. After they have downloaded the game on their devices, they*

select the same policy and they start to play. Unfortunately, Anna downloaded a malicious version of the mobile game: in the background it tries to connect to the remote server and to download tons of data. Thanks to the inline monitoring system the "good" version downloaded by Bob works properly and the malicious version downloaded by Anna crashes and is not allowed to execute.

Demo 2 *Now Bob wants to play a downloaded chess game with Anna. As shown in Figure 1(b) he might choose between four pre-set policies allowing any kind of connections, a bounded number of connections (max SMS sent and max MB downloaded), some access to GPRS or a purely local game where no connection is allowed. He chooses the second policy and he runs the mobile chess application. Figure 1(a) shows the two HTC P3600 smart phones playing chess by sending each other SMSs. The smart phone of Anna (on the left) has no monitoring framework installed, so the game is allowed to send as many SMSs as it will. The monitoring system of the phone of Bob (on the right) detected that the user-defined limit of the sent messages was exceeded. For this reason the game is terminated through a security exception.*

References

1. Massacci, F., Naliuka, K.: Beyond Sandboxing: an Effective In-line Monitoring System for Mobile Code. Available at: http://www.ing.unitn.it/∼massacci/mass-nali-08-implementation.pdf.
2. Erlingsson, U., Schneider, F. B.: IRM Enforcement of Java Stack Inspection. In Proceedings of SSP-00, Washington, DC, USA, 2000.
3. Erlingsson, U.: The Inlined Reference Monitor Approach to Security Policy Enforcement. Department of Computer Science, Cornell University, Technical Report 2003-1916, 2003.
4. Hamlen, K.W., Morrisett, G., Schneider, F.B.: Certified In-lined Reference Monitoring on .NET. In Proceedings of PLAS '06, Ottawa, Ontario, Canada, ACM Press, 2006.