PHD research proposal

Enrico Steffinlongo

May 27, 2014

1 Background

In many application we have a typical scenario where the software have to interact with many elements that can be both sensitive or untrusted. This can be very dangerous because an attacker coming from the untrused outer world pose very harmful threat to the sensitive elements treated by the program. To avoid this, is often used a privilege separated architecture that divide the application in various components giving to each component only limited permission and makes the communication of the various component using a Message Passing Interface. To avoid privilege escalation attacks, where a component compromised by an attacker can obtain permission that it doesn't has originally, usually the permission are given statically before the execution of the application. Moreover, according to the permission system, there is a strict punctual mapping between components and permission in order to reduce the possible exploits of the attacker.
To maintain full functionality, the components can communicate via message passing in order to exchange information and such channel between components could be centralized or distributed and can pass different kind of messages: from simple strings to complex object with pointer and methods. Even this message passing, is an attack surface because it lead to privilege escalation attacks. In fact a compromised component can send messages asking to another for the execution of a privilege that is requested by the attacker.
These choice, as said, are used in a lot of different system architecture such for example in Android each application has a limited set of permission that are given statically and it can communicate with other apps using a IPC message passing interface. As discussed in [4, 6, 7, 5, 8] even this architecture suffers various attacks like privilege escalation. Moreover sometime applications that are written by programmers that are not security experts can expose a large attack surface since they can be over privileged or they cal leak some capability for example not checking properly the inbound messages.
Another case of Privilege separated architecture that uses a simple Message passing interfaces is the Google Chrome Extension framework. In this architecture, used to extend functionality of the browser, as shown in [2, 3], the choice is to separate every app in two sets of components: Backgrounds that is the part of the application that interact with sensible data as password, cookies and with the browser core and it is isolated from the page on which the extension is running and the Content Scripts that has no permission except the one used to send messages to the Background. They have access to the page on which they are injected. Moreover to avoid delegation of privilege, messages can only be

strings. This restrict the attack surface indeed an attacker can't send function to be executed in the privileged Background environment, but reduce also the control on messages so an infected Content Script can send arbitrary messages to the Background that not ever is able to understand if a message is benign or not.

This architecture unfortunately suffer several limitations that affects both functional and non functional requirements. For example since permission are given statically often the programmer tends to give to each component the maximum of the permission required even if a certain privilege is used very rarely. Some architecture (e.g. Chrome Extension [1]) mitigate this behavior letting the user to request and release permission dynamically up to an upper bound that is given statically. Another problem of such architecture is that restricting too much the kind of messages (e.g. Chrome Extension) to avoid injection of code can limit the expressiveness of Object oriented programming paradigm.

2 Proposal

In this scenario interesting research could examine deeper

References

[1] Chrome extension overview
https://developer.chrome.com/extensions/overview, May 2014.

[2] Adam Barth, Adrienne Porter Felt, Prateek Saxena, and Aaron Boodman. Protecting browsers from extension vulnerabilities. Technical Report UCB/EECS-2009-185, EECS Department, University of California, Berkeley, Dec 2009.

[3] Nicholas Carlini, Adrienne Porter Felt, and David Wagner. An evaluation of the google chrome extension security architecture. In *Proceedings of the 21st USENIX Conference on Security Symposium*, Security'12, pages 7–7, Berkeley, CA, USA, 2012. USENIX Association.

[4] Erika Chin, Adrienne Porter Felt, Kate Greenwood, and David Wagner. Analyzing inter-application communication in android. In *Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services*, MobiSys '11, pages 239–252, New York, NY, USA, 2011. ACM.

[5] Lucas Davi, Alexandra Dmitrienko, Ahmad-Reza Sadeghi, and Marcel Winandy. Privilege escalation attacks on android. In *Proceedings of the 13th International Conference on Information Security*, ISC'10, pages 346–360, Berlin, Heidelberg, 2011. Springer-Verlag.

[6] Adrienne Porter Felt, Erika Chin, Steve Hanna, Dawn Song, and David Wagner. Android permissions demystified. In *Proceedings of the 18th ACM Conference on Computer and Communications Security*, CCS '11, pages 627–638, New York, NY, USA, 2011. ACM.

[7] Adrienne Porter Felt, Erika Chin, Steve Hanna, Dawn Song, and David Wagner. Android permissions demystified. In *Proceedings of the 18th ACM Conference on Computer and Communications Security*, CCS '11, pages 627–638, New York, NY, USA, 2011. ACM.

[8] Adrienne Porter Felt, Steven Hanna, Erika Chin, Helen J. Wang, and Er Moshchuk. Permission re-delegation: Attacks and defenses. In *In 20th Usenix Security Symposium*, 2011.