University of Padova

Computer and Network Security

insert-title-here

Students:	*Registration number:*
Davide Trevisan	1070686
Andrea Multineddu	MATRICOLA

Thursday 25 August 2016

1 Stealth Screenshot Extension

The source code of the extension we developed is downloadable here

1.1 Scope of the extension

the scope of the stealth screenshot extension is to create a small prototype of an extension that catch screenshot in a regular interval with the minimum possible user interaction and obviously giving it back.

1.2 How the extension work

1.2.1 Activation and screenshot collection

The extension uses the relatively new tab APIs given by chrome too take screenshot. The extension makes use of the "storage", "tabs", "all_urls;", "unlimitedStorage", "activeTab" permissions in his manifest file. The extension works in a simple way: it triggers on a browser event, in this case the click on the icon of the extension, but there are other possibilities: we tested that there are no simple ways to take screenshot without no interaction, because the security policy of the API doesn't allow to take a scrrenshot without some kind of events, presuming that the user should be aware of what is happening; not triggering any event returns only null. Once triggered, the extension schedule all the future screenshot through the Javascript setTimeout method: this allows to take screenshot for hours, with only the need for a click to start. The screenshot are stored in an array visible to all the extention methods.

1.2.2 Screenshot retrival

The screenshot can be collected in two ways in the wxtension we developed: through a combination of key for retriving all screenshots (we programmed it on Ctrl+Shift+Y) or writing "show" in the omnibox, pressing "tab" and writing in the omnibox the number of screenshot to show (starting from the last one).

1.2.3 Performance

We tested that the extension is able to take screenshot with a interval of 1 minute for more than 2 hours without losing any screenshot. The application occupy less of 50MB of RAM for an hour of screenshots taken every minute. The impact on the CPU is negligible. Screenshot retrival through the kay combo however can crash chrome, although it never happened in our PC because of the great performance (we had an Intel i5 6400, with 8GB of DDR4 memory and SSD), but it got freezed for some seconds.

1.3 Limitations

The major limitation of this extension is that the screenshot only lives until the extension is active: this implies that closing all the windows of chrome completely deletes the screenshot taken. This is a consequence of how the API and the chrome sandbox works. We at the moment found no way to get around it, but we are pretty confident that is possible to save those screenshots, but for our lack of knowledge we are not able at the moment to demonstrate it.

1.4 Future work

Possible future works will focus on find a way to save those screenshot. The application should also be rewritten without the tab API, to avoid of the limitation of it. Javascript inject the code for the screenshot or use it for simulate the right events should do the work, but the time and the train needed to do it made impossible for us to test it for this paper.

2 conclusions

All the solution we have presented in thispaper makes use of the most used chrome permission, as already stated in the HULK paper[2]

Rank	Top 10 types of permissions	# ext.
1	tabs	16,787
2	notifications	12,011
3	unlimitedStorage	9,424
4	storage	5,725
5	contextMenus	4,774
6	cookies	2,872
7	webRequest	2,849
8	webRequestBlocking	2,102
9	webNavigation	1,623
10	management	1,533

Figure 1: The top 10 permissions found in the manifest files for all extensions we ran. Extensions can include more than one permission.

As shown in the screenshot extension, taking screenshot without user consensus should not be allowed to the API (just a popup should be enough)

References

[1] Eric Zhang,
ChromeLogger,
A keylogger and form grabber for Google Chrome that runs as an extension.
ChromeLogger
referring site

[2] *Hulk: Eliciting Malicious Behavior in Browser Extensions*
Alexandros Kapravelos, Chris

Grier, Neha Chachra, Christopher Kruegel Giovanni Vigna, Vern Paxson, UC Santa Barbara, UC Berkeley, UC San Diego and International Computer Science Institute	*23rd USENIX Security Symposium.* Paper