We have you by the gadgets

A Security Analysis of the Microsoft Windows Sidebar Gadget Platform

Mickey Shaktov
Toby Kohlenberg

Abstract

Windows 7 is the most widely used operating system in the world, reportedly used on 52.3% of all computers worldwide *(ref.1)*. Included within the operating system is a built-in framework passed on from Vista known as the "Windows sidebar gadget platform" which enables the user to run simple applications known as "gadgets" on the Windows desktop.

This paper provides a short overview of the Windows Gadget platform, the security features of this platform and how it can be leveraged by attackers to compromise the operating system.

Gadget Overview

Introduction

Microsoft provides detailed explanations of the Gadgets functionality and architecture. Rather than duplicate their materials here, we will provide summaries of the key topics along with links to the relevant source materials. Full references are included in the material but the majority can be found at these two links:

- Windows Sidebar: http://msdn.microsoft.com/en-us/library/windows/desktop/aa965850(v=vs.85).aspx
- Gadgets for windows sidebar security: http://msdn.microsoft.com/en-us/library/windows/desktop/ff486358(v=vs.85).aspx

What are gadgets?

Gadgets should be thought of as essentially being a website that is run from the Windows desktop with some advanced capabilities and additional APIs being made available to increase functionality.

The gadget is distributed as a renamed .cab or .zip file; <example>.gadget. The archive contains the combination of files and images necessary to define the gadget and allow the sidebar process to render and present the gadget. Most of the time, a gadget is a combination of HTML, XML, CSS and Javascript however a gadget can be created using other languages such as Silverlight or WPF.

When opened for the first time, the gadget is imported into one of three directories:

- %systemdrive%\Program Files\Windows Sidebar\Shared Gadgets
- %systemdrive%\Program Files\Windows Sidebar\Gadgets
- %systemdrive%\Users\%user%\AppData\Local\Microsoft\Windows Sidebar\Gadgets

The first two can only be modified by members of the Administrator group, the third is where all user-installed gadgets are installed.

When a user loads a gadget through the Desktop Gadget Gallery, the Sidebar.exe process is started and loads the specified gadgets. As of Windows 7 all gadgets are run in a single sidebar process. The Sidebar process can be configured to start (or not) at logon and any gadgets specified in the Settings.ini file in the gadgets directory will be started.

Gadget Security Model

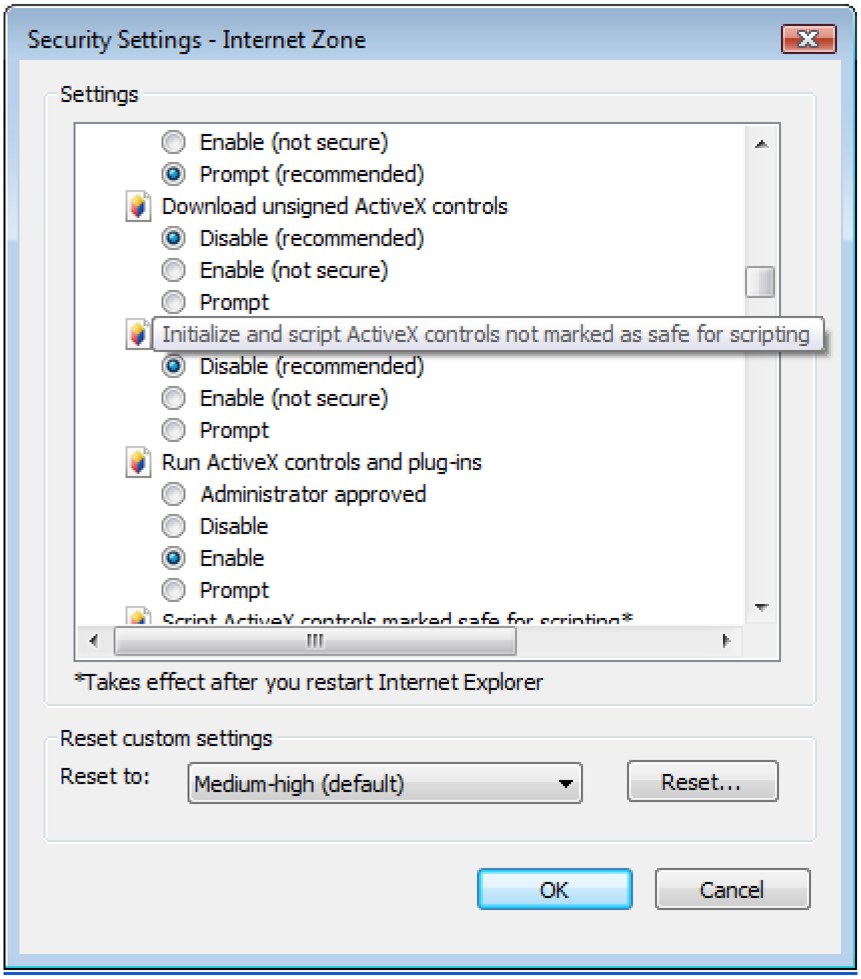
For this section we have included content from the Microsoft Gadgets for Windows Sidebar Security*(ref.3)* paper where the available explanations didn't require and summarization or :

The runtime

The gadget runtime can be compared to the Internet explorer runtime, the gadgets are configured similarly to HTML applications (HTAs) and are configured with specific set of permissions.
Gadgets configuration sets are different from common web pages and other HTAs in several ways:

ActiveX

Gadgets can instantiate any installed ActiveX objects when the option “Initialize and script ActiveX controls not marked as safe for scripting” is enabled in Internet explorer (see image below). Without this option enabled many known 3rd party gadgets will not work properly. As a result it is enabled by default.



Security Settings - Internet Zone

Settings

○ Enable (not secure)
● Prompt (recommended)

Download unsigned ActiveX controls
● Disable (recommended)
○ Enable (not secure)
○ Prompt

Initialize and script ActiveX controls not marked as safe for scripting
● Disable (recommended)
○ Enable (not secure)
○ Prompt

Run ActiveX controls and plug-ins
○ Administrator approved
○ Disable
● Enable
○ Prompt

Script ActiveX controls marked safe for scripting*

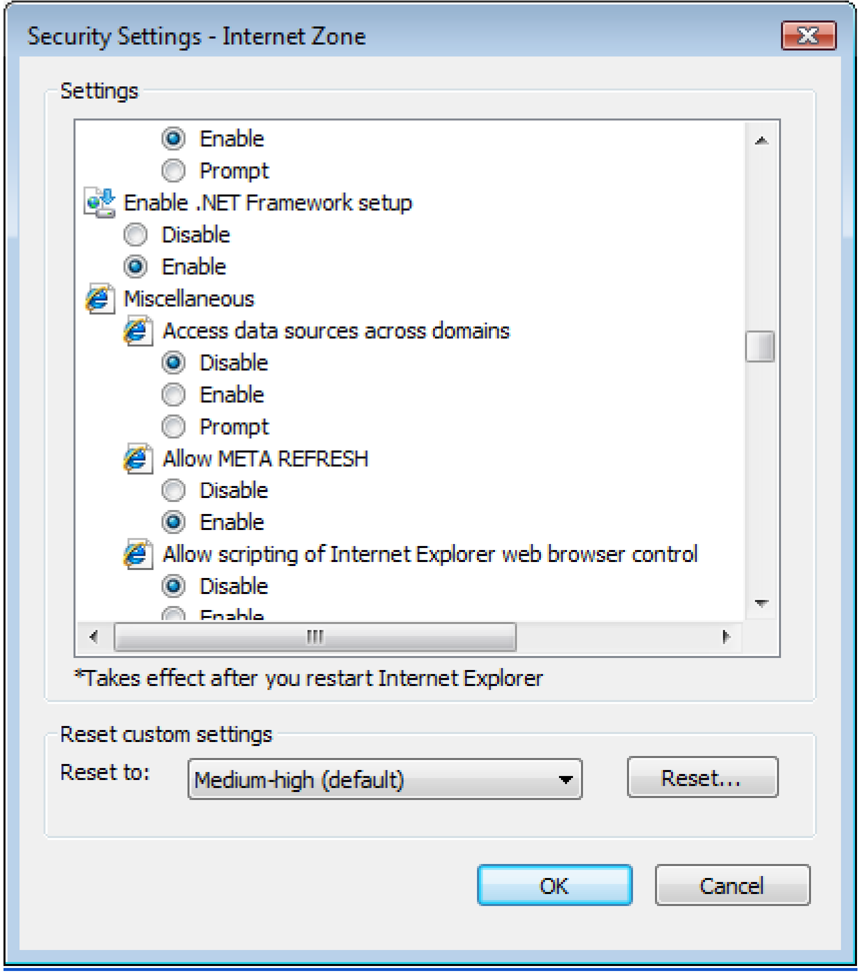
*Takes effect after you restart Internet Explorer

Reset custom settings

Reset to: Medium-high (default)

Cross domain access

Since gadgets must be able to aggregate data from various locations, the option “Access data sources across domains” is enabled in Internet explorer.



Security Settings - Internet Zone

Settings

● Enable
○ Prompt

Enable .NET Framework setup
○ Disable
● Enable

Miscellaneous

Access data sources across domains
● Disable
○ Enable
○ Prompt

Allow META REFRESH
○ Disable
● Enable

Allow scripting of Internet Explorer web browser control
● Disable
○ Enable

*Takes effect after you restart Internet Explorer

Reset custom settings

Reset to: Medium-high (default)

Code Signing

While gadgets do support code signing, it is not required by default. In our research we found only a few gadgets that were digitally signed while the majority were not.

The fact that most of the gadgets we encountered were unsigned can be related to the fact that digitally signing a gadget is not an easy task, mostly because of the fact that gadgets are not executables but rather archives.

UAC

Gadgets run with standard user privileges with the Administrator Approval mode. In addition, gadgets cannot directly raise UAC elevation prompts. This prevents them from directly attempting to escalate privileges. Microsoft provides the following example:

“if a gadget attempts to delete a file in the System32 directory, the delete operation would not succeed and no elevation prompt would be shown to the user. This failure happens because most critical files cannot be modified by standard users”(ref.3)

However a gadget is able to launch an instance of a locally installed application and that application is able to raise a UAC prompt. This presents an obvious opportunity for abuse.

Enterprise Controls

There are a number of enterprise controls that are available to enable greater restrictions through the use of GPOs. There are GPOs to restrict gadgets in the following ways:

- Turn off Windows Sidebar
  - This can also be done permanently using Microsoft Security Advisory (2719662) which disables the Windows Sidebar and all gadget functionality.
- Disable automated installation of unsigned gadgets
  - This does not prevent the manual installation of unsigned gadgets and does not remove already installed gadgets.
- Prevent users from installing gadgets
  - This specifically prevents the running of gadgets placed into the user directory (%systemdrive%\Users\%user%\AppData\Local\Microsoft\Windows Sidebar\Gadgets) and prevents them from being displayed in the Gadget Gallery
- Remove the “Get more gadgets online” link
  - This is irrelevant as of the publication of this whitepaper due to Microsoft’s decision to shutdown their online gadget store.

Overview of attack surface

Risks exposed by gadgets

As explained in the introduction section, gadgets are granted significant permissions to execute code roughly equivalent to an HTML Application.
As a result, the risks that gadgets are exposed to are the same as those faced by any web-based application, e.g. Man-In-The-Middle or code injection. Similar issues existed in earlier versions of most web browsers but modern browsers have specifically implemented controls to attempt to mitigate many of these issues. These controls have not be implemented in the Gadgets platform, leaving them vulnerable to well-known and thoroughly discussed attacks.

Risks posed by gadgets

A windows sidebar gadget can be developed in a way that will allow it to spread like a worm and infect other machines while performing malicious operations on user’s machines.
Considering the tools that the gadget framework gives gadgets to help make them more robust in functionality, it is easy to see how gadgets can be made to serve a malicious intent.
Gadgets can be developed in a way that they are almost identical to traditional software (compiled binaries) giving attackers a “way in” to an unsuspecting user’s operating system.

Common vulnerabilities in gadgets

As has been discussed by other researchers, the simplicity of certain programming languages and development frameworks can increase the likelihood that inexperienced developers will release applications that are insufficiently secured. Different languages and frameworks tend to encourage or discourage specific kinds of mistakes.

Gadgets that were created by developers who did not follow the security guidelines to the letter or

did not fully understand the implications of code injection are exposing the user's operating system to arbitrary code execution via various vectors.
This risk can be demonstrated by 2 examples, the first being a very old reported vulnerability by Aviv Raff *(ref 4)* in which a windows vista sidebar gadget was vulnerable to XSS and as a result to code execution. The second example is that the same vulnerability exists in the Daemon tool sidebar gadget (Vulnerability responsibly disclosed to vendor) distributed with the daemon tools software package and installed by default.

Perception of sidebar gadgets

Gadgets are not considered to be executables by most users (which they are), but rather as a cool addition to the desktop. The user's perception of gadgets is formed from using common gadgets such as clock, calendar, "rss feed reader" or "cpu usage". Unfortunately to an attacker this misconception is priceless.

Traditional methods of protection

Because of the techniques used by modern antivirus, they are not particularly effective in detecting malicious gadgets. The way antivirus software works is by detecting known patterns of malicious code. For vulnerable code, these elements usually arise from bugs in software allowing an attacker to inject code. In the case of malicious code these patterns are defined by looking for known fingerprints of assembly or patterns of execution. In contrast, gadgets are created using languages that are not commonly used for malware and executed through an interpreter such that they appear to be legitimate software, exactly like any other non-malicious piece of software.
The big difference is that by design a gadget can perform actions exactly like a traditional compiled executable but operate under a completely different scope within the sidebar process.
Simply put, a gadget can do all that an executable can, without being considered as executable by the antivirus software.

Recommendations

The issues we have identified are common to many application platforms. As a result, the recommended mitigations are relatively simple:

- If you are a user of gadgets, only install gadgets that come from known trusted sources.
- If you are a developer of gadgets, the best option is to stop developing using the gadget framework and move to the Windows 8 Metro platform. If you must continue to develop on the gadgets platform then follow well-known secure development best practices for writing applications that need to run over the Internet.
- If you do not use gadgets, remove the sidebar functionality using the instructions detailed by Microsoft: http://technet.microsoft.com/en-us/security/advisory/2719662

References

1. http://www.w3schools.com/browsers/browsers_os.asp - OS popularity statistics
2. http://msdn.microsoft.com/en-us/library/windows/desktop/aa965850(v=vs.85).aspx - Windows Sidebar
3. http://msdn.microsoft.com/en-us/library/windows/desktop/ff486358(v=vs.85).aspx - Gadgets for windows sidebar security
4. http://packetstormsecurity.org/files/cve/CVE-2007-3033 - CVE 2007-3033
5. http://www.securitytracker.com/id?1018566 - several vulnerabilities in windows gadgets
6. http://video.google.com/videoplay?docid=-542607127966826627 - Defcon 15 - The Inherent Insecurity of Widgets and Gadgets