



Plug-ins - a source of insecurity

Many software product manufacturers provide customer access into their products. There are many reasons to do this, including:

- allow for local customization;
- allow extra facilities to be added;
- support features not provided by the manufacturer.

Sometimes these points of access are called APIs, and sometimes they are called plug-ins.

What do they do?

Well they allow outsiders a degree of access to what is happening inside a manufacturer's product by making available information such as where some data is stored and how to manipulate that data. They expose the internal workings of the manufacturer's product to the outsider. They tell the outsider where data is found and how to interpret it.

But are they secure, and what would we mean by secure?

Ideally a plug-in would be secure by virtue of its own design, and adding it to an existing application would not introduce a new weakness, and the plug-in would not conflict with any other plug-ins used in the same application.

It seems that plug-ins sometimes conflict with each other. The following web site provides developers with a plug-in conflict detection tool, so the problem must be a genuine hazard - <http://www.sovanto.com/morrowind/tespcd.htm>

And, unfortunately, plug-ins, like any other computer programs, may contain errors that need to be corrected. By way of example if you follow the link <http://weblogtoolscollection.com/pluginblog/2005/07/20/plugin-security-update/> it provides some insight into this problem. So the solution is to update. But of course everyone has to implement the update, and we know just how difficult that is to achieve.

It can be strange to consider that IT departments frequently prohibit the installation of executable files by users by denying them administrator rights, "Because they may compromise the security of the PC," whilst at the same time living with the fact of users being able to download and install almost any plug-in they want without any knowledge of what the impact of that download will be. Perhaps the real issue here is that the IT department can deny administrator rights but can't stop plug-ins, so they do what they can. So the claim that the plug-in is safer than allowing an exe file may prove more a matter of what IT departments can achieve than actuality. A plug-in for example, may obtain the rights of the application it is plugged into, which may be very considerable indeed.

Of course plug-ins could be made secure, in the sense that by cryptography (digital signatures) the manufacturer can verify that plug-ins have been digitally signed before allowing the plug-in code to run (provided that the manufacturer evaluates and certifies all plug-in code before signing it so that every user may be certain that there can be no compromise to the application). But only the manufacturer can do that – nobody else. And anyway, what would that mean? Are we to assume that the manufacturer has the technical ability to certify the security and quality of every plug-in that is digitally signed and who is going to pay for that? It would create an immensely complex administration system, not to mention always having to have the manufacturer's product being fully up to date.

Of course this puts an enormous responsibility on the manufacturer to exercise high levels of due diligence if that strong control is to be exercised.



If that strong control is not exercised (we are not aware of any manufacturer doing this), then in reality the providers of plug-ins have a free-for-all as to what they actually do. And since they cannot know if they are the only plug-in running, and the nature and intent of any other plug-ins running at the same time as them, they have no ability to police the situation for themselves. In fact, just as they may expose the manufacturer's system, they may also expose each other's actions.

So let us say that a security plug-in is installed in a system. How will it protect itself against other plug-ins, or against the manufacturer? What will be its approach to verifying the environment it finds itself in? Some security vendors providing plug-ins to interoperate with other products have been unlucky, such as the [PGP Outlook plug-in vulnerability](#) reported at The Register where a security plug in weakness could compromise the system.

Of course you can always just place blind trust in the manufacturer and all the plug-in providers. I guess that's how we got into the virus, worm, spam and intruder mess. We just put blind trust in the providers of computer systems that there were no threats and risks and we did not need any controls.

Self-evidently we were wrong on that occasion. Major industries have since made fortunes out of the fact that we did not have any security, and now we have to rely heavily on their products to keep us safe. But the important conclusion we have to draw is that putting trust in the manufacturer to get it right and control the system perfectly every time isn't working.

It seems clear, as the spam wars develop, that manufacturers are changing from providing technical quality to threatening transgressors with litigation instead. The digital millennium copyright act is an excellent example of where industry forbids any examination of a claimed security mechanism and tries to make any research (never mind actual compromise) a criminal activity. As a result manufacturers are encouraged to make outrageous claims about their security safe in the knowledge that any attempt at criticism will be met with a criminal prosecution instead of a high quality system.

So plug-ins are not a guarantee of security, and, if used at all, should be used with great care and caution. It seems that, "One man's meat is another man's poison."